Amateur: Augmented Reality Based Vehicle Navigation System

CHU CAO, Nanyang Technological University, Singapore
ZHENJIANG LI, City University of Hong Kong, China
PENGFEI ZHOU, Tsinghua University, China
MO LI, Nanyang Technological University, Singapore

This paper presents Amateur, an augmented reality based vehicle navigation system using commodity smart phones. Amateur reads the navigation information from a digital map, matches it into live road condition video captured by smart phone, and directly annotates the navigation instructions on the video stream. The Amateur design entails two major challenges, including the lane identification and the intersection inference so as to correctly annotate navigation instructions for lane-changing and intersection-turning. In this paper, we propose a particle filter based design, assisted by inertial motion sensors and lane markers, to tolerate incomplete and even erroneous detection of road conditions. We further leverage traffic lights as land markers to estimate the position of each intersection to accurately annotate the navigation instructions. We develop a prototype system on Android mobile phones and test our system in a total number of more than $300km$ travel distance on different taxi cabs in a city. The evaluation results suggest that our system can timely provide correct instructions to navigate drivers. Our system can identify lanes in $2s$ with 92.7% accuracy and detect traffic lights with 95.29% accuracy. Overall, the accuracy of the navigation signs placement is less than 105 pixels on the screen throughout the experiments. The feedback from 50 taxi drivers indicates that Amateur provides an improved experience compared to traditional navigation systems.

CCS Concepts: • **Human-centered computing** → **Mixed / augmented reality**; **Smartphones**; *Ubiquitous and mobile computing design and evaluation methods*; User studies; Interface design prototyping; • **Computing methodologies** → Image representations; Matching;

Additional Key Words and Phrases: Augmented Reality, Navigation Service, Smartphone

ACM Reference Format:
Chu Cao, Zhenjiang Li, Pengfei Zhou, and Mo Li. 2018. Amateur: Augmented Reality Based Vehicle Navigation System. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 155 (December 2018), 24 pages. https://doi.org/10.1145/3287033

1 INTRODUCTION

Navigation service on vehicles or cars can be provided by vehicle-mounted navigators or smart phones. They display vehicle's instant GPS location and overlay the navigation information onto a digital map to assist the driving. The major limitation of existing navigation systems is that the navigation information is displayed on the digital map, while the driver's focus is on the traffic condition through the front window — the navigation

This work is supported by Singapore MOE AcRF Tier 1 grant 2017-T1-002-047, Tier 2 grant MOE2016-T2-2-023 and the CoE grant M4081879 from NTU. This work is also supported by the ECS grant from Research Grants Council of Hong Kong (Project No. CityU 21203516), and the GRF grant from Research Grants Council of Hong Kong (Project No. CityU 11217817). Corresponding authors: Zhenjiang Li and Mo Li.
Authors' addresses: Chu Cao, School of Computer Science and Engineering, Nanyang Technological University, Singapore, ccao001@e.ntu.edu.sg; Zhenjiang Li, Department of Computer Science, City University of Hong Kong, China, zhenjiang.li@cityu.edu.hk; Pengfei Zhou, Tsinghua University, Beijing, China, zhoupf05@tsinghua.edu.cn; Mo Li, School of Computer Science and Engineering, Nanyang Technological University, Singapore, limo@ntu.edu.sg.

2474-9567/2018/12-ART155 $15.00
https://doi.org/10.1145/3287033

information is displayed in an indirect and inconvenient way to the driver. The driver thus has to understand the navigation instructions presented on the device, and then map them to the front traffic view observed. In many complicated road conditions, *e.g.,* multiple entrances of nearby intersections in front, drivers may take a wrong turn due to the misunderstanding of displayed instructions or navigation errors. Concentration on translating the digital map and navigation instructions to the real world view may also cause potential safety issues, especially for the unfamiliar road conditions, long-term driving and new drivers.

The major motivation of this paper aims to improve the navigation service. To overcome above limitations, we propose Amateur, *i.e.,* **A**ug**m**ented re**a**li**t**y (AR) [4] based v**e**hic**u**la**r** navigation system, with the help of nowadays smart phones. Instead of displaying instructions on a digital map, the smart phone directly displays the front traffic conditions, where all navigation instructions are annotated as arrows on the live video stream to guide the driver, *e.g.,* to keep going straight, to change to the right/left lane, to turn right/left at an intersection, *etc.* Because the live video display is identical to the driver's front window view and navigation annotations are clearly marked on the video, such an augmented reality design minimizes the display-understanding gap for drivers, and enhances the navigation experience. To precisely place navigation instructions into the live video, the system, however, needs to continuously track the lane-level and navigation status at a high granularity to ensure navigation accuracy, *e.g.,* unlikely missing a turn or making a wrong turn, as well as to improve the navigation reliability with GPS errors. In particular, developing such a system in practice entails the following challenges:

1) Amateur needs to precisely identify the host lane, *i.e.,* the lane vehicle currently stays at. GPS cannot reliably locate vehicles at a lane level, (*e.g.,* 3 to 4 meters), due to high localization errors especially in urban areas. In the literature, some existing works have studied the host lane detection problem, but they mainly leverage extra sensors. For instance, the authors in [5] utilize the extra module to communicate with other vehicles. Jiang *et al.* [20] deploy hundreds of sensors on the vehicle to fulfill the design. High-definition map and dead-reckoning sensors are required to achieve lane-level map matching in [39]. In computer vision, there are also existing efforts made for the lane detection for the automatic driving. In [1], image processing techniques are used to detect the lane markers and roads. A comprehensive computer vision based algorithms are summarized in [18]. These solutions extract the complete lane information from video frames, but they rely on non-trivial deployment of spinning cameras or camera arrays around the vehicle, and incur high computation overhead which is not affordable by mobile devices directly.

2) To place navigation instructions for the turn at an intersection, the accurate position of the intersection needs to be identified in video frames. A straightforward solution is to use GPS locations of the vehicle to estimate the distance to the intersection, and then translate it to a proper position in the video frame. Such an approach is not a good choice due to GPS errors. Amateur tries to leverage traffic lights that are normally nearby intersections as indicators to place navigation instructions. Although some traffic light detection designs exist, they mainly rely on expensive computer vision techniques, which cannot be afforded by smart phones either, *e.g.,* the design in [14] applying machine learning algorithms, the approaches in [8] and [27] adopting expensive image processing techniques for detecting the traffic light's shape and edge respectively, *etc.*

We address above issues in designing Amateur. In summary, this paper makes the following contributions.

- We propose Amateur, an augmented reality based vehicle navigation system, which directly displays the navigation instructions on the live road condition video captured by the smart phone. Compared with traditional methods that mainly display the navigation information on a digital map, Amateur could provide a more convenient and user-friendly navigation service.
- We propose effective techniques to address the lane identification and intersection inference challenges. For the first challenge, our key insight is that identifying which lane the vehicle currently stays on is sufficient for the Amateur design, so that the original lane detection problem can be simplified to the host

lane identification problem. For the second challenge, we leverage traffic lights and a pin-hole model to estimate the position of the intersection for correctly placing the navigation instructions in the video.

- With above designs, Amateur is lightweight enough to smoothly execute on the mobile platforms, and we implement a prototype on Nexus 5X. We comprehensively evaluate its performance over road segments totaling over a 300km distance and also conduct a user study on 50 different taxi cabs in a city. The results demonstrate the efficacy of the Amateur design, which can improve the navigation service from four important aspects: ease of use, perceived distraction, navigation experience and user-friendliness.

The rest of this paper is organized as follows. We review the related works in Section 2. We detail the Amateur system design in Section 3 and evaluate its performance in Section 4. Design limitations are discussed in Section 5 before we conclude in Section 6.

2 RELATED WORK

Vehicle navigation. Many commercial navigation softwares have been developed for the smart phones and the most representative example is the Google Maps [16]. Although many of these products now can support the "audio message reporting" function, drivers still need to manually understand the navigation information. The authors in [23] find that the acoustic information is preferred by the driver compared with the text and map. However, the research in [30] find that the long-time audio messages may significantly distract drivers' attention, causing the potential safety issues, especially with long auditory messages [11]. The automobile manufactures propose their own AI-based navigation, such as TESLA [38], but they have powerful sensors embedded. Many companies give their conceptual products, like Navion [40], Hudify [19] and Exploride [9], while they do not have available devices currently.

Augmented reality for vehicle navigation designs. In the literature, Augmented Reality (AR) [4] serves as a promising technique to advance the vehicle navigation design. AR is a technology that overlays the digital information on objects or places in the real world for the purpose of enhancing the user experience [2]. The existing attempts to utilize AR in the vehicle navigation designs have been made in both academia and industry.

The study in [12] is a pioneer work to introduce the visualization technique to the vehicle navigation designs, in which the authors compare three navigation modes (audio only, audio plus map, and visualization) on a simulated platform and find that drivers watch less frequently on the navigation screen with the visualization. The authors in [13] further examine an AR based design on a simulated platform and achieve further improved user experience. The authors in [21] report a user study also in a simulated environment to compare the driver's behaviors in different environments. These prior studies suggest that using AR is a more effective and safer way to design a navigation system [32]. Following this trend, the authors in [41] build an in-vehicle navigation system using the AR idea. Vehicular AR is further proposed in [10] using a special camera. However, both of them cannot provide the lane-level navigation guidance. On the other hand, an LED array is introduced on the windshield of vehicles to provide affordable AR instructions [29], which again requires dedicated hardware to support. Some researchers combine the real world video with expensive computer graphical models to provide AR navigation information [33]. Different from these works, we propose a mobile-affordable AR navigation system, without installing dedicated hardware on the car, to provide a fine-grained lane-level navigation service.

There are also existing attempts made from the industry. Sygic [35] overlays the navigation information on a real or virtual environment, without the actual navigation provided. Other products like Navdy try to integrate incoming calls, messages and navigation together and project those information onto the windshield. Such AR services can only supplement the navigation design. On May 9 2018, Google started to provide the AR function for their own map application on the Google I/O [15] event in the latest Android system. The captured scenes by camera can be used to match the Google Street View so as to localize the user and provide proper navigation instructions. While with such a design, it still cannot provide the lane-level instructions due to the localization error. Meanwhile, if the driver's Google Street View is not the latest version, the matching error can be large or even the matching failure occurs. As far as we know, Amateur is not the first work that introduces AR in the navigation system design but it is the first system realizing this vision on commodity mobile devices by addressing unsolved design challenges.

Lane identification. Many approaches in the literature are proposed to extract lanes from the video streaming for automatic driving designs [18], which can be used to avoid the lane weaving or drifting [42], or detect whether a remote object can potentially collide with the vehicle. As these existing works aim to achieve a complete lane search, labeling entire lanes on the frame [37] by leveraging the lane edge detection [3] involves excessive computations [6]. These methods cannot be afforded by the mobile devices. Although the authors in [22] propose a down-sampling technique to reduce the number of frames to be processed, the computation overhead of image processing is still beyond the capacity of mobile devices. Different from these existing works above, we focus on the host lane identification problem in the Amateur design and explicitly address the incomplete and erroneous input issue, which is not been solved yet.

Traffic light detection. Traffic light detection is a key technique in automatic driving designs, which can be grouped into three categories [8], namely, color segmentation, shape detection and direct classifier. The former two types of techniques involve extensive computer vision computations. The study in [27] proposes to identify the traffic lights by detecting the edge of a circle and also the colors of the lights. Authors in [36] determine the position of traffic lights in the video by their colors and the central point. The extensive computer vision computations make these methods not suitable for Amateur. The direct classifier technique, on the other hand, requires a large volume of training data [24]. The researchers in [14] apply the machine learning algorithm to detect traffic lights. Such a design is also computationally expensive and requires a training phase. To avoid both the high computation complexity and training overhead, we observe that traffic lights consist of LED bulbs, which are powered by alternating current. We thus propose to utilize the flickering feature of LED bulbs to detect the position of traffic light on the frame, and further leverage the pin-hole model to estimate the position of the intersection for correctly placing navigation instructions in the video. Although there are also some existing efforts made to identify visible lights on images, *e.g.,* [25], they strive to the indoor positioning designs, which requires an off-line processing from servers to achieve a high-precision light location identification.

3 SYSTEM DESIGN

3.1 Design Overview

Fig. 1 illustrates the system architecture of Amateur. Suppose that Alice plans to drive from position A to position B, using Amateur. After Alice inputs the origin and the destination, Amateur first launches the underlying navigation services, *e.g.,* Google Maps [17] or OpenStreetMap [28], to obtain detailed lane-level[1] navigation information from A to B. Note that Amateur is not positioned to redesign the entire navigation stack. Instead, it reuses the existing navigation functions and focuses on automatically matching the navigation information to the live road condition video to achieve a much augmented user experience.

When Alice's car is moving, Amateur periodically reads GPS to acquire a rough location of the car for determining the road segment the car is currently on. According to the navigation plan provided by the underlying navigation service, Amateur then knows which lane should be taken and how many lanes on current road in total. On the other hand, Amateur executes the *Lane Identification* module in Fig. 1 to determine the host lane. If the host lane differs from the lane suggested by the navigation service, the *Instructional Sign Placement* module displays an annotation arrow on the live road condition video to inform drivers changing to the correct lane.

[1]The lane-level road information is already available for most cities in commercial navigation products, like Google Maps [17]. However, due to the limited resolution of GPS positioning, drivers cannot directly leverage such detailed information in existing systems throughout the navigation procedure.

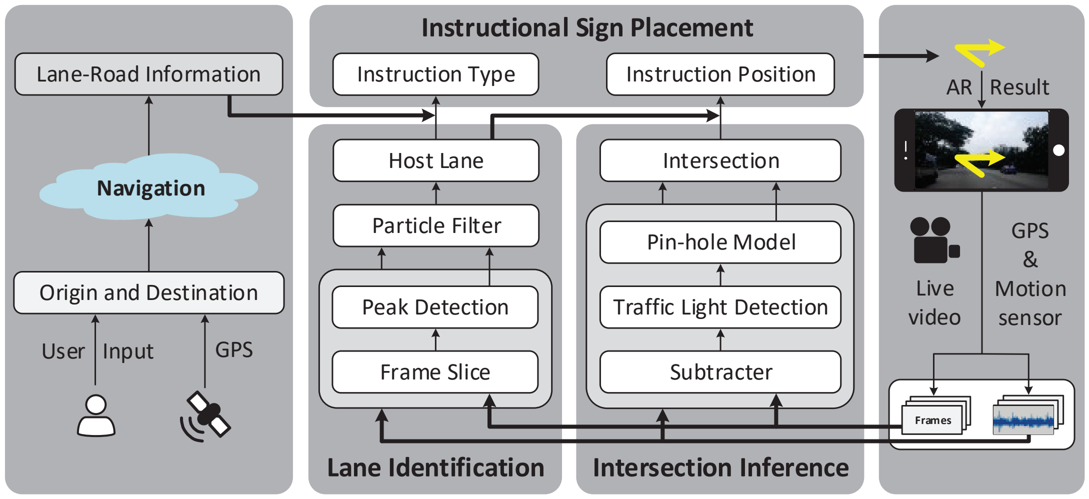

Fig. 1. System architecture of the Amateur design.

To fulfill above design, video frames first undergo a slicing operation to extract the interested points from each frame. Amateur then executes peak detection to obtain all lane markers (boundaries) on the road. The observed lane markers however can be incomplete or mixed with detection errors. The lane detection module further adopts a particle filter to statistically infer the most likely position of host lane through consecutive observations, by matching the observed lane marker patterns and accelerometer readings with their expected distribution.

When Alice needs to take a turn at one intersection, Amateur can specify the intersection's position in the video stream for displaying the corresponding turning instruction. To this end, the *Intersection Inference* module first reads the GPS data. If the car gets close to the intersection, *e.g.,* 50 meters in front, it conducts the frame subtraction operation to unveil the flicking feature of traffic lights to identify the position of traffic lights in video frames, based on which the module further applies a pin-hole model to estimate the position of the intersection, so that the turning instruction arrow can be correctly placed.

With above visualized navigation design, Alice could simply follow the displayed annotations to reach the destination. In the rest of this section, we detail the design of lane identification and intersection interference modules in Amateur.

3.2 Lane Identification

To determine the lane markers on the road, we have the following key observation: as lane markers are usually painted with bright colors, *e.g.,* white and yellow, after we transform the video frames to the gray scale, the pixels on the lane markers will exhibit large pixel values, *i.e.,* peaks. Fig. 2 illustrates the gray scale values of the pixels along a straight horizontal line extracted from the frame in Fig. 3(a). After manually checking the ground truth of the lane marker positions on the frame, we find that each highlighted peak in Fig. 2 indeed corresponds to the boundaries of all the lanes.

Although we propose a design, in Section 3.2.2, to precisely quantify and extract each individual peak from the converted gray scale pixel values, the obtained overall peak pattern is not always reliable due to two types of unavoidable errors:

- False alarm peaks: bright reflections, paintings and objects from the road surface can cause the detection of additional peaks, not corresponding the lane boundaries.

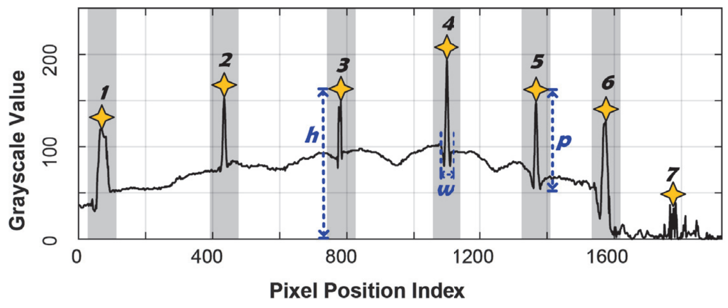

Fig. 2. Gray scale values of one row of pixels.

- Missing peaks: certain lane boundaries may also be miss-detected due to light conditions or blocking by nearby cars.

If we directly use such noisy or erroneous peak patterns, host lane cannot be reliably detected. In Amateur, we introduce a particle filter based host lane identification design to tackle this issue.

3.2.1 Particle Filter Based Host Lane Identification. The core idea for the host lane identification, *i.e.,* identifying the lane number of host lane on the road, is to conduct a template matching based on the observed gray scale value peaks as in Fig. 2.

To facilitate the discussion, we first introduce necessary notations used in the design. Fig. 3(a) depicts the live video from the smart phone and we convert the pixels from the highlighted narrow stripe to the gray scale values for the peak detection (detailed in Section 3.2.2). For all the detected peaks from one frame, we find that the two peaks that are closest to the center line of the video frame are usually from the two markers of the host lane (yet the lane number is still unknown), denoted as X_{mid}. This is because smart phone is normally placed close to the middle of the car. For the example in Fig. 3(a), the host lane is lane-3 (with respect to the left). The two markers of this lane in Fig. 3(b) indeed enclose the frame's center line. On the other hand, because drivers need to keep a distance to the car in front, we find that the two markers of the host lane can reliably be detected.

As a result, we can place an x-axis along the narrow stripe as in Fig. 3(b), where the origin is the center of these two lane markers, denoted as [0]. In addition, we can treat these two lane markers as a delimiter. The locations of other peaks detected on the left (and right) hand side of this delimiter, denoted as X_{obs}, can serve as an observation to infer the host lane's location. In Fig. 3(b), X_{obs} contains two peaks, whose coordinates along x-axis are -895 and -526 respectively, on the left and other two, 407 and 614, on the right. Viewing X_{obs} as the constraints, we can then guess the location of the host lane with the following design.

Particle filter framework. Suppose that the current road segment has 4 lanes with 5 lane markers (obtained from the underlying digital map) illustrated in Fig. 4(a). The host lane, *e.g.,* surrounded by the middle two lanes in X_{mid}, is one of these four. If the host lane's lane number is one, the expected positions of other three lane markers or peaks are depicted as X_1 in Fig. 4(b), and we call it a template[2]. Similarly, we can generate different templates for other three possible cases. In addition to the four templates, we also have a real observation $X_{obs} = [-895, -526, 407, 614]$ with one extra peak detected, as shown in Fig. 4(c). We thus want to know which template in Fig. 4(b) that X_{obs} matches best.

If each peak in the observation X_{obs} corresponds to one lane marker, *i.e.,* containing no errors, the matching is trivial. However, as aforementioned, X_{obs} may contain the peaks from other bright objects' reflection, *e.g.,* one

[2]We adopt a default width, *e.g.,* 316 pixels, between two consecutive peaks in each template. As all templates adopt the same width, even it is different from the actual width on the current road, it does not affect the matching result.

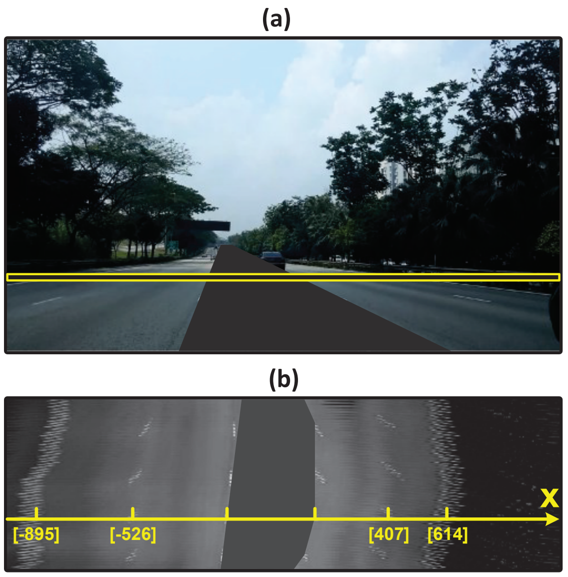

Fig. 3. Illustration of peak detections from video frames. (a) Selected horizontal pixels at the same position from one frame. (b) Peak positions along the x-axis. The ticks inside brackets are the positions of lane markers.

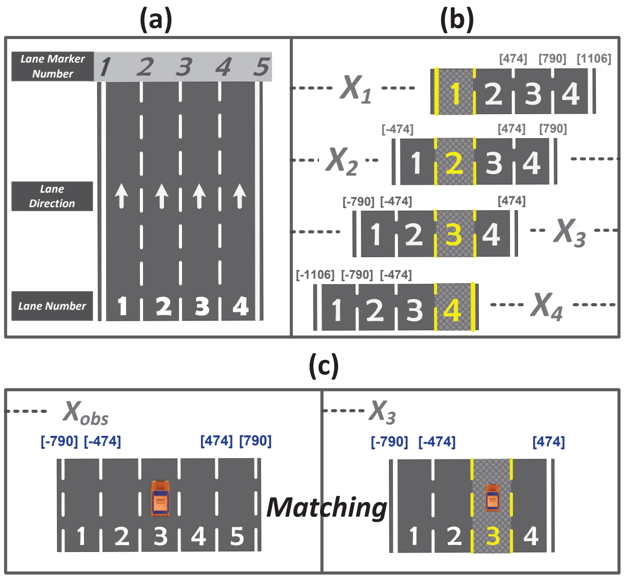

Fig. 4. Illustration of the particle filter design. (a) One road segment has four lanes with five lane markers. (b) Four templates by assuming the host lane is at each of these four lanes. (c) Template matching based on the observed peak pattern, *e.g.*, X_{obs} matches X_3 best.

more peak in X_{obs} in Fig. 4(c). X_{obs} may also miss "legal" peaks due to the blocking from surrounding cars. To address this issue, we leverage particle filter which can tolerate observation errors and statistically compute the possibility for each template. After a series of consecutive observations, the probabilistic belief on the real host lane could rapidly accumulate and lead to the highest confidence.

By this principle, we design a particle filter based method. Suppose there are M particles, *e.g.*, $M = 100$. We treat each lane on the current road (obtained from the digital map) as a bucket and then throw particles into these buckets. Initially, the probability or weight into each bucket is identical. Whenever one observation X_{obs} is generated from peak detection, we update weight w_l for particles into lane (*i.e.*, bucket) l. The update is based on the similarity between observed vector X_{obs} and each template. Due to the observation errors, the number of elements in X_{obs} may be different from each template. To measure their similarity with variant lengths, we adopt the dynamic time warping (DTW) technique [34]. Given two discrete sequences of different lengths, *e.g.*, X_{obs} and X_1, DTW searches for the best alignment between these two sequences by minimizing the mutual distance using dynamic programming. After the DTW computation for each template, the weight updating can be expressed as:

$$w_l = e^{-d_l/k}, \tag{1}$$

where d_l is the DTW distance of X_{obs} to the l-th lane template and k is a scaling factor. From each video frame i, we can calculate $w_l^{(i)}$ by Eq. (1) and further conduct a weighted average with the previous $w_l^{(i-1)}$ to tolerate peak pattern errors from individual observations:

$$w_l^{(i)} = \alpha \cdot w_l^{(i)} + (1-\alpha) \cdot w_l^{(i-1)}, \tag{2}$$

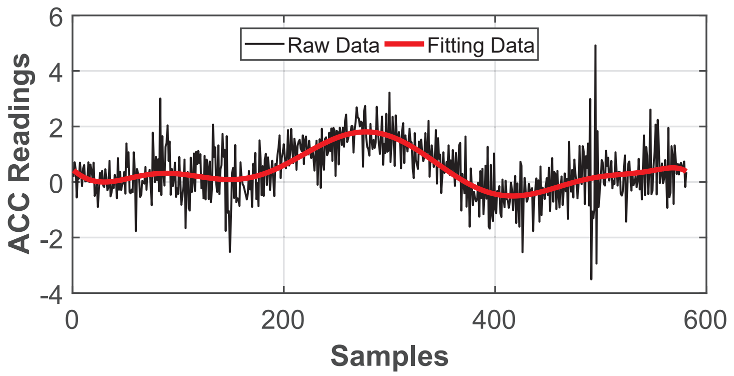

Fig. 5. Accelerometer readings when the car makes a right turn.

where α is set as 0.5 in our current implementation to balance the contributions from both the current and past observations for avoiding the sudden parameter changes caused by the observation errors. The weight average will restart from the scratch after the car makes a turn. After Eq. (2), we further normalize each $w_l^{(i)}$ so that their summation equals to 1, *i.e.,* $\sum_l w_l^{(i)} = 1$. After the weight update, all particles will be redistributed cross all possible lanes with a new distribution. After several rounds of updating, most particles will converge and we select the lane occupied by particles whose weight is larger than 0.5 as the host lane.

Enhanced particle filtering. With this primary design, in Amateur, we also propose an enhancing technique to further improve the host lane identification performance, by detecting and leveraging the event of car's lane-changing. Suppose that Amateur detects the car has changed to the right lane. In this case, the left-most lane on the road is unlikely to be the current host lane, as it contradicts to this lane-changing event. Therefore, the weights of particles in each bucket, after the awareness of such an event, can be further adapted to speedup the convergence and improve the reliability.

To leverage such an opportunity, we observe that if the car goes straight, the two markers in X_{mid}, enclosing the host lane, are on the different sides of the frame's center line, as shown in Fig. 3(b). When it starts to turn right, the right marker starts to move towards the center line. After the car turns to the right lane, this marker will move cross the center line. The same principle can be applied when the car turns to the left lane. Therefore, we can utilize such a center line crossing phenomenon to detect the lane-changing event. On the other hand, when a car changes its lane, it may change s lanes, where s can be greater than 1. For instance, if the car changes from lane 1 to lane 3, s equals to 2. With different s values, the weights of particles should adapted differently. However, when the car changes multiple lanes, its heading direction becomes not parallel to the lane's direction. We cannot determine the exact value of s based on the center line crossing phenomenon. To this end, we find that the IMU sensor readings from smart phone can facilitate to determine the value of s.

In Amateur, smart phone is tightly mounted on the car and its IMU sensor readings (along the direction that is perpendicular to car's moving direction) suffice to infer the car status, *e.g.,* moving straight or making a turn. Fig. 5 illustrates such accelerometer readings when the car makes a right turn. It clearly shows a convex bump. We conduct a moving average for accelerometer readings. Normally, the readings are near zero. However, when the car turns to a new lane, its readings exhibit an obvious change. The shape of such reading patterns is highly related to the lane-changing details.

More precisely, given the acceleration readings (perpendicular to car's moving direction) under the lane-changing event (detected by the center line crossing phenomenon in the video), *i.e.,* observation O, we aim to compute the posterior distribution $\mathcal{P}(\mathcal{L}_s|O)$, where $\mathcal{L}_s$ represents the lane-changing event with s lane(s) being changed. Therefore, the Bayesian inference can be expressed as:

$$\mathcal{P}(\mathcal{L}_s|O) \quad = \quad \mathcal{P}(O|\mathcal{L}_s) \cdot \mathcal{P}(\mathcal{L}_s)/\mathcal{P}(O). \tag{3}$$

The likelihood for the occurrence of the lane-changing event $\mathcal{P}(\mathcal{L}_s)$ and the observed sensor readings $\mathcal{P}(O)$ can be estimated from the off-line analysis of the collected live video and sensor data traces. But later we find that their exact values are not needed in Amateur, and we denote Eq. (3) as $\mathcal{P}(\mathcal{L}_s|O) = \mathcal{P}(O|\mathcal{L}_s) \cdot \beta$, where $\beta = \mathcal{P}(\mathcal{L}_s)/\mathcal{P}(O)$. To calculate $\mathcal{P}(O|\mathcal{L}_s)$, we record different IMU readings under the events $\mathcal{L}_s$ as templates in advance. Those templates can be utilized to match the actual sensor readings during the driving to compute $\mathcal{P}(O|\mathcal{L}_s)$. For this computation, we further apply DTW to measure the similarity between the observed O and each template, and compute $\mathcal{P}(O|\mathcal{L}_s)$ as $\mathcal{P}(O|\mathcal{L}_s) = e^{-d'_s/k'}$, when d'_s is the DTW distance of O to each template, and k' is a scaling factor. Then we can incorporate $\mathcal{P}(\mathcal{L}_s|O) = \beta \cdot \mathcal{P}(O|\mathcal{L}_s)$ to update particle filter's weights in Eq. (1) when the lane-changing event is detected:

$$\hat{w}_l = \sum_{s+i=l} \mathcal{P}(\mathcal{L}_s|O) \cdot w_i = \sum_{s+i=l} \beta \cdot \mathcal{P}(O|\mathcal{L}_s) \cdot w_i, \tag{4}$$

where $\mathcal{P}(\mathcal{L}_0|O)$ is defined as 1. The rationale of Eq. (4) is: when $s = 0$, w_i is the original weight w_l and only the peak observations contribute to the weight update in line 9 of Algorithm 1; Otherwise, the term $\mathcal{P}(\mathcal{L}_s|O)$ describes the likelihood that the car changes from the i-th lane to the l-th lane by changing s lanes in total, *i.e.,* $s + i = l$. In this case, the peak observations and accelerometer readings of the lane switching both contribute to the weight update in line 7 of Algorithm 1. In other words, in addition to the original weight w_l, Eq. (4) also considers the transitions from other lanes. After Eq. (4), we further normalize all $\hat{w}_l$, so that their summation equals to 1, and then apply the normalized $\hat{w}_l$ in Eq. (2) to throw particles to identify host lane. Note that the factor β will be canceled out during normalization and we do not need to know its exact value because given the same observation, they are equal in different lane-changing events.

A formal description of the weight updating is given in Algorithm 1. In particular, the algorithm maintains a set of particles and each particle corresponds to a possible host lane. Then, the algorithm considers to associate each pair of peak observation and accelerometer readings with every particle to calculate how likely (probability) that each lane could be the host lane (line 7). When there is no lane-switching event, the corresponding probability is calculated in line 9. After the probability contributed by new peak observations and accelerometer readings is calculated, the algorithm updates the weight of each particle (line 11) and normalizes their weights (line 15). The algorithm utilizes DTW function (line 15) to calculate the weight of each particle, which finds the best match between two time sequences (line 24–29). In DTW, the algorithm usually utilizes the Euclidean distance (line 26) as the cost function. Finally, the DTW function returns the cost value (Euclidean distance under best match) of two input time sequences (line 30), which is used in the weight updating in line 7 and 9.

Lane-changing arrow placement. As stated before, after the weight update, all particles will be redistributed cross all possible lanes with a new distribution. After several rounds of updating, most particles will converge and we select the lane occupied by particles whose weight is larger than 0.5 as the host lane. After the host lane is identified, if it differs from the suggested lane by the navigation service, a lane-changing annotation is displayed in the live road condition video, where the annotation arrow design is introduced in Section 3.3.

3.2.2 Peak Quality Control. Although the particle filter could automatically tolerate observation errors, as a matter of fact, many peak errors can be excluded before the particle filtering to speedup the host lane identification.

To this end, for any peak pattern as depicted in Fig. 2, peaks are empirically characterized by three features:

- Height h: the gray scale value of a peak.
- Width w: the width of the detected peak.

ALGORITHM 1: Weight update for particles in the host lane identification

```
Input: peak template $X_b$, peak observation $X_{obs}$, accelerometer template $\mathcal{L}_s$ and observation $O$;
Output: updated weights for all the particles;
for each template $X_b$ do
    for each template $\mathcal{L}_s$ do
        for each particle $p$ do
            if $\mathcal{L}_s$ event happens then
                $w_{l\pm s} = e^{-DTW[X_b, X_{obs}]/k - DTW(\mathcal{L}_s, O)/k'}$ ;    // Both peak patterns and accelerometer readings
                 contribute to the weights of the particles
            else
                $w_l = e^{-DTW[X_b, X_{obs}]/k}$ ;    // Only peak patterns contribute to the weights of the particles
            end
            Update the weight of particle $p$ following equation 2 ;    // Updating of the weights
        end
    end
end
Normalize the weights of all particles ;    // Normalization of the weights
Function DTW($\mathcal{T}$, $O$):
    for $i = 1$ to $lenth(\mathcal{T})$ do
        $DTW[i, 0] = \infty$ ;    // Initialization
    end
    for $j = 1$ to $lenth(O)$ do
        $DTW[0, j] = \infty$ ;    // Initialization
    end
    $DTW[0, 0] = 0$;
    for $i = 1$ to $lenth(\mathcal{T})$ do
        for $j = 1$ to $lenth(O)$ do
            $cost = d(\mathcal{T}[i], O[j])$ ;    // The cost is measured by the Euclidean distance
            $DTW[i, j] = cost + \min\{DTW[i-1, j], DTW[i, j-1], DTW[i-1, j-1]\}$ ;    // DTW calculation
        end
    end
    return $DTW[length(\mathcal{T}), length(O)]$ ;    // Return the final DTW value
    End Function
```

- Prominence p: the prominence of a peak within range w.

Suppose the width (pixel-wise) of one frame is N. Along a row extracted from one frame, the gray scale values of the N pixels are denoted as $G = \{g_i\}_{i=1}^{N}$. Through our extensive investigations, we find the following empirical criteria could well characterize the peaks corresponding to lane markers:

$$h \geq \frac{1}{2} \cdot max\{g_i\}_{i=1}^{N}, \tag{5}$$

$$p = h - min\{g_i\}_{i=g^k-\frac{w}{2}}^{g^k+\frac{w}{2}}, \tag{6}$$

$$\frac{1}{3} \cdot p \leq p - G_{10th}. \tag{7}$$

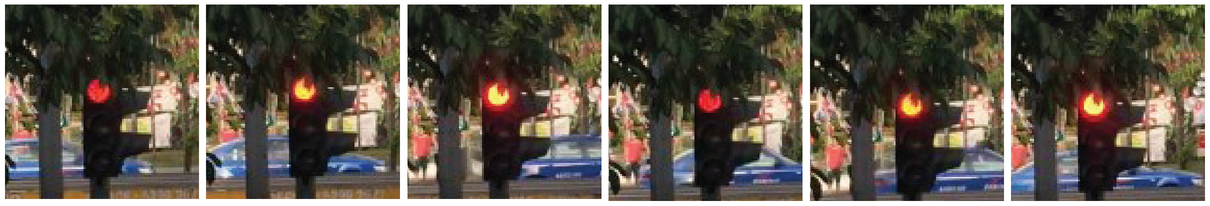

Fig. 6. Cropped parts of six continuous frames containing the traffic light from live video.

where G_{10th} is the 10th percentile in set $\{g_i\}_{i=g^k-\frac{w}{2}}^{g^k+\frac{w}{2}}$ and g^k is the position of the k-th peak. Eq. (5) ensures the detected peaks should be bright enough. Eq. (6) and (7) ensure that the detected peak should be sharp enough. The three features, *e.g.,* h, w and p, are carefully investigated in Section 4.

Due to the perspective effect, these parameter settings may vary if the straight horizontal line in Fig. 3(a) is selected differently. To collect consistent observations, we adopt a fixed offset in Amateur, *e.g.,* 230 pixels to the frame bottom on our current development platform Nexus 5X. The 230 pixels on Nexus 5X correspond to a horizontal line about 5 meters in front of the car. This distance is rarely blocked by the front car, as the driver needs to keep certain distance to the front car during the driving. Therefore, the pixels from this position could provide a reliable lane marker detection on our current implementation platform. In Section 5, we further discuss how to set this parameter on different smart phones. On the other hand, as the pixels adopted to detect lane markers correspond to a horizontal line that is not far away from the car, when a car moves along a curvy road, the observed bending of all the lanes is also similar to each other. As a result, after we use DTW to find the similarity between the observed peak pattern and the peak templates, the curve lanes have the same impact on each template. The calculated similarity can still update the weights of the participle filter as on a straight road.

3.3 Intersection Inference

With the host lane identification design in Section 3.2, the lane switching annotation arrow can be displayed in the live road condition video. To further display an annotation arrow for the intersection turning, the position of the intersection needs to be specified in the video frames and we leverage traffic lights as an anchor to estimate this position for the arrow placement in this section.

3.3.1 Traffic Lights Detection. To determine the position of traffic lights, we utilize the flickering feature of LED bulbs. As traffic lights are powered by alternating current with a time varying intensity, *i.e.,* of 50 or 60 Hz frequency, the light intensity varies accordingly, as shown in Fig. 6. Such a difference can be unveiled by subtracting consecutive frames. According to GPS, Amateur can detect whether the car gets close to an intersection, *e.g.,* 50 meters in front. If so, the intersection inference module conducts the frame subtraction to unveil the flicking feature of traffic lights to identify the position of traffic lights in video frames, based on which the module further applies a pin-hole model (Section 3.3.2) to estimate the position of the intersection, so that the turning instruction arrow can be correctly placed.

Fig. 7 illustrates the flow chart of the traffic lights detection. Frames are streamed into a subtracter. It outputs the subtracted results of two consecutive video frames. As the camera captures the real-time video with 30 frames per second (FPS) and the frequency of the traffic light's intensity varying is 50 or 60 Hz, the area with significantly changed pixel values likely indicates traffic lights due to the flickering feature of LED bulbs, while other areas will not exhibit such a high intensity varying frequency because they reflect the sun light, which can be viewed as a constant approximately within a short time span. Fig. 8 shows that the statistic difference of pixels representing traffic lights varies from 31 to 189 in the gray scale domain. In particular, 70% of the pixels at traffic

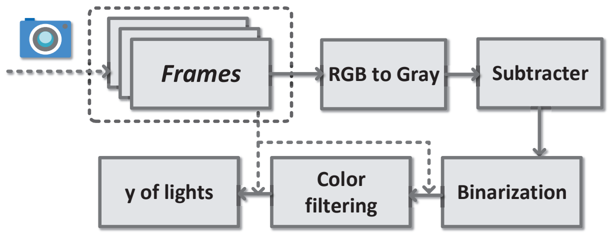

Fig. 7. The flowchart of traffic lights detection for the intersection inference in live video.

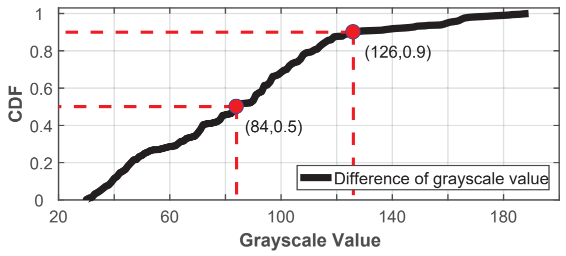

Fig. 8. Statistical results of differences of pixel at traffic lights' positions in gray scale domain.

lights' position is larger than 60 in the gray scale domain. On the contrary, the pixel changing in the gray scale domain for other areas is rarely larger than 50. Therefore, we use the threshold 50 to binarize the subtraction result in our current implementation.

After the binarization, the preserved area in the frame mainly indicates the traffic lights. We then apply the color filter [24] to extract the positions of pixels representing the traffic lights and use their averaged position to denote the traffic light. After the identification, the y coordinate of the traffic light, *i.e.,* its height, is passed to the pin-hole model, as illustrated in in Fig. 9.

3.3.2 Pin-hole Model. In Amateur, we utilize the position of traffic light y in the frame to infer the position of the intersection. We observe that the supporting point of a traffic light, *e.g.,* point A in Fig. 9, could serve as a good reference to indicate the nearby intersection because it approximately shares a similar depth as the intersection to the car in the video frame.

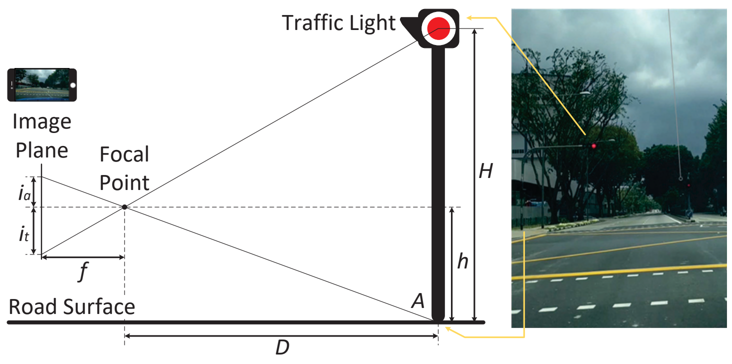

Fig. 9. Pin-hole model to calculate the position of an intersection in a video frame.

Therefore, we introduce a pin-hole model as shown in Fig. 9 to fulfill the calculation. We first determine the position of the traffic light in the frame in Section 3.3.1 and denote this position as i_t, *i.e.,* $i_t = y$ in Fig. 9. When there are multiple traffic lights are detected, Amateur selects the one with the largest y value (*i.e.,* the highest one on the screen) as the indicator. This is because when a car is approaching a crossroad, the traffic light that the driver needs to follow is in front of the car (maybe far away but the direction is almost upright). Therefore, according to the perspective relation, such traffic light has the highest y value on the screen. On the other hand, the physical height of traffic light is normally fixed in a city, which can be denoted as H. After the smart phone is

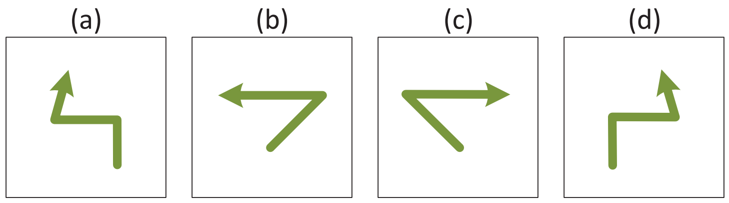

Fig. 10. Instructional annotation arrows designed in Amateur. (a) Left-lane changing. (b) Left turning. (c) Right turning. (d) Right-lane changing.

mounted to the car, we can also measure its distance to the ground h. With these three parameters, the position of the supporting point, denoted as i_a, in the video frame can be calculated by:

$$i_a = \frac{h}{H - h} * i_t. \tag{8}$$

After i_a is obtained, the upper edge of the annotation arrow is aligned with this height to indicate a turning at the intersection.

Annotation arrow design and display. The primary arrow set to navigate the driving consists of four basic elements. The first two are *turning left or right*, as shown in Fig. 10(b,c), which inform the driver to turn left or right at the crossroad. The other two arrows are *left- or right-lane changing*, as Fig. 10(a,d) depicts. These two arrows inform the driver how to change the lane, where the lane changing number also appears on the top of the arrow to facilitate the driver to recognize. We show concrete use cases of these annotation arrows in the evaluation Section 4. One more arrow might be included in Amateur, *i.e., going straight*. Because if there is no turning or lane-changing event, the default operation is going straight. Therefore, we skip this arrow to reduce the amount of arrows displayed that may distract the driver's attention.

To display the annotation arrows, in addition to the smart phone's screen, another possibility is to project them on the dashboard or windshield of the car. However, this may require the car release the permission to let third-party navigation system access its dashboard, which needs the car manufacture's cooperation and may also have security risk concerns. In addition, displaying information on the dashboard or windshield requires the transparent windshield to support as well, which is more expensive. Therefore, we provide a lightweight and easy-to-deploy solution which is transferable regardless of car models and windshield types by using the smart phone, due to its independence to car's internal system and the wide availability to most drivers.

4 IMPLEMENTATION AND EVALUATION

4.1 Experiment Methodology

We implement Amateur using the Nexus 5X smart phone, which is equipped with the IMU sensors, two cameras and a $1920 * 1080$ resolution screen. It has a 1.8 GHz hexa core 64-bit ARMv8-A processor and 2GB RAM. In the implementation, we invoke the OpenCV library for the image processing. The sampling rate of GPS is set to be $1hz$ during this experiment.

To evaluate the performance, we hire taxi cabs in a city and attach the mobile phone to the windshield with a mobile phone holder. The phone is plugged with the USB power and there is thus no energy concern to the usage of Amateur. Prior to the experiment on each taxi, we measure the distance (one time effort for each taxi) from the phone holder to the ground, *i.e., h* in the Eq. (8) used by the pin-hole model. The evaluation lasts several weeks and covers 5 experimental routes (depicted in Fig. 11) cross the city, with the total length of over 96 km, where the connected markers are the expressway and isolated markers are the highway in reality. These five routes contain both the highways and expressways, which also cover both the east-west and north-south directions in

Table 1. Detailed information of the 5 experimental routes in Fig. 11.

Route	A	B	C	D	E
Length (km)	12.7	17.4	36.8	12.5	16.8
Length of Expressway (km)	8.73	15.35	36	4.46	14.28
Length of Highway (km)	3.97	2.05	0.8	8.04	2.52
Number of Traffic Lights	15	19	5	18	17
Average Velocity (km/h)	49.7	51.2	56.1	45.4	53.7

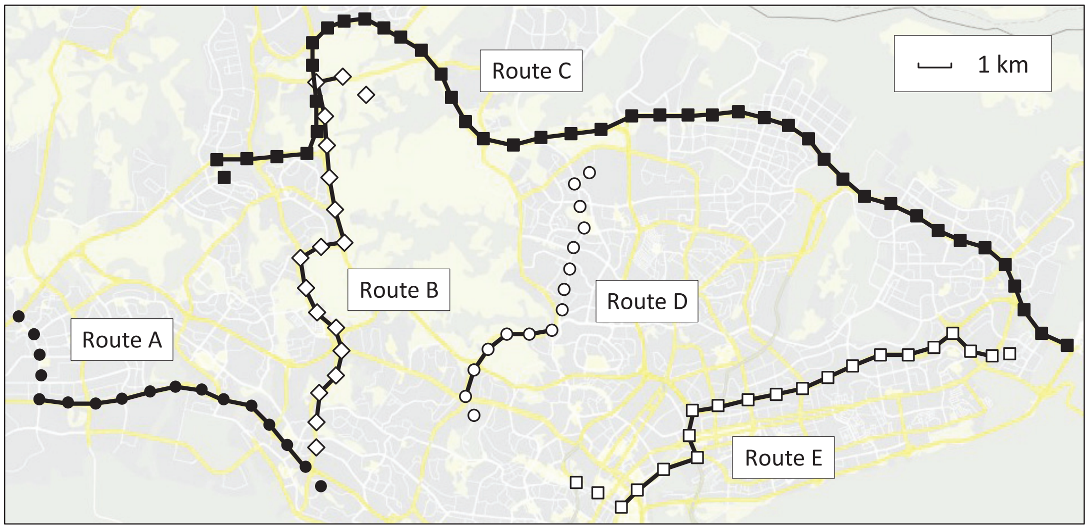

Fig. 11. Experimental routes in the evaluation and their total length is about 96km.

the city. Their more detailed information is summarized in Table 1. In particular, the longest testing route (Route C denoted as ■ in Fig. 11) lasts about 36.8 km and only 0.8 km of them is on the highway. The shortest route is Route D of 12.5km, in which the length of its highway lasts longer than its expressway.

Throughout the experiments, each route is evaluated multiple times, in 1) different times of a day, *e.g.,* in the daytime, at night, at sunset and at nightfall with different traffic patterns and light conditions on the road, and 2) also with different weather conditions, like sunny and rainy, to achieve a comprehensive performance evaluation. The total traveling distance is more than 300km in the evaluation, and we report Amateur's performance from the following four detailed aspects in this section.

As Amateur consists of two major modules, *i.e.,* host lane identification and intersection inference, in this section, we first evaluate the performance of these two modules individually (Sections 4.2 and 4.3) as follows.

Host lane identification. For the host lane identification, we adopt the accuracy rate to evaluate its performance, which is defined as the ratio between the correctly placed lane-changing arrows and the total number of such arrows displayed during the navigation. We check the accuracy of host lane identification under different

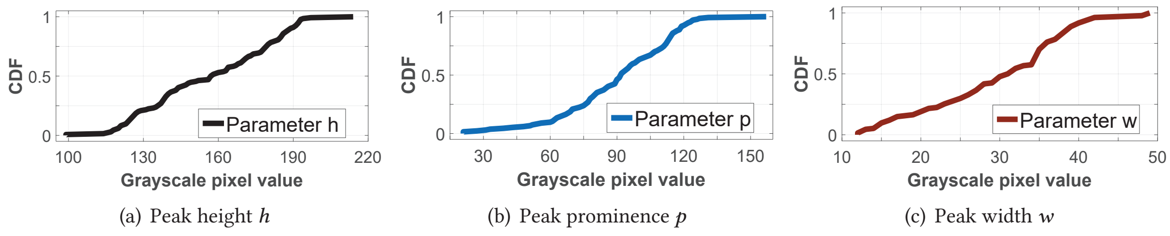

Fig. 12. Investigation of the three features, *i.e.,* *h*, *p* and *w* in Fig. 2, to detect the gray-scale pixel value peaks.

conditions. If host lanes are determined correctly, lane switching arrows can be properly displayed. In addition to the accuracy, we also investigate the parameter settings and the particle filter convergence speed for this module.

Intersection inference. For the intersection inference module, we also use the accuracy to quantify its performance. As we record all the navigation videos for obtaining the ground truth, we can manually check the traffic light detection accuracy for the evaluation. On the other hand, we also examine the position difference between the placed turning arrows at each intersection, which reflects the effectiveness of the pin-hole model. Intersection inference is evaluated on highways, since there are no traffic lights on expressways. On expressways, identifying the correct lane alone is sufficient to navigate the driving.

In addition to the investigation of these two individual modules, we further evaluate the overall instruction correctness performance of Amateur in Section 4.4 and also conduct a concrete user study in Section 4.5.

Instruction correctness. We then test whether the pop-up instructions from Amateur are correct or not, comparing with the ground truth obtained from our manual investigation of the live video off-line. For each trip in the experiment, all the annotation instructions from Amateur can logically form a vector, in which each element is one of the four annotation arrows depicted in Fig. 10.

User feedback. Finally, we conduct the survey from each taxi driver to get their feedback to Amateur compared with the traditional navigation service.

In the following, we report the evaluation results for each of the four aspects above.

4.2 Host Lane Identification

In the host lane identification module (Section 3.2.2), we utilize three features, *i.e.,* *h*, *p* and *w* in Fig. 2, to detect the gray-scale pixel value peaks, which correspond to the candidate lane boundaries. In this section, we first examine each of these features to verify the design efficacy. In Fig. 12(a), the height of each detected peak, *h*, varies from 100 to 218. The difference comes from the light intensities and we observe that the pixel value increases dramatically on the lane boundary markers. Fig. 12(b) further indicates that the prominence of a peak, *p*, is also sharp, *e.g.,* varying from 20 to 159. Finally, Fig. 12(c) shows that the width of the peak, *w*, is sufficiently wide enough, ranging from 12 to 49.

The host lane identification in Amateur is based on the particle filter. In Fig. 13, we first investigate the latency spent by this particle filter design. In particular, we test this latency performance for different types of roads (with two lanes up to five lanes). We record the time until the particle filter correctly identifies the host lane by comparing with the ground truth. From the result, we can see that the average latency varies from 0.74*s* to 3.83*s* when the number of lane increases from 2 to 5. Fig. 13 implies that the host lane identification in Amateur is reliable as long as enough time is given. However, in practice, it may not be possible to wait for a relatively long time, *e.g.,* up to 4.85*s* on the road of five lanes, to identify the host lane for only once. We thus consider a more stringent setting to evaluate the host lane identification next.

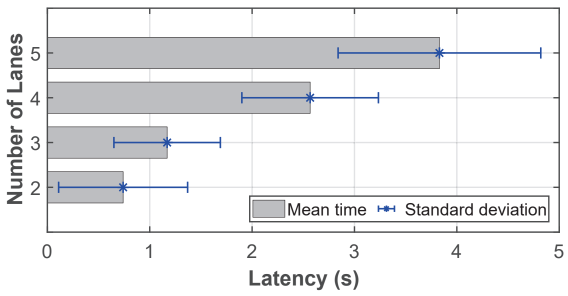

Fig. 13. Latency to correctly identify the host lane on the roads with different numbers of lanes.

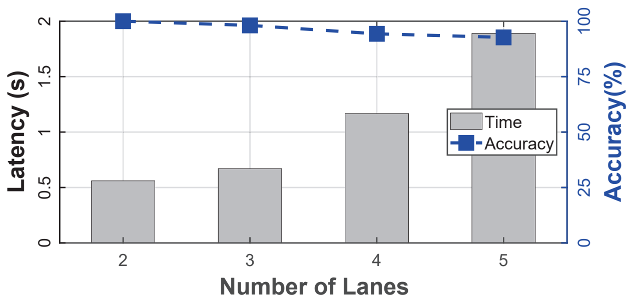

Fig. 14. Average delay to identify the host lane (left y-axis) and the identification accuracy (right y-axis) on the roads with different number of lanes.

In Fig. 14, we further examine the host lane identification performance within a short time window, *e.g.,* 2*s*. In particular, after the particle filter starts a host lane identification, if the weight is larger than 0.5 (according to the design in Section 3.2.1) on one lane within 2*s*, it is reported as the current host lane. If the 2*s* time window is reached but there is no lane accumulating a larger weight, the filter outputs the lane with the highest weight at this moment as the host lane. Then for different types of roads (with two lanes up to five lanes on the x-axis), we report the average latency for the host lane identification (left y-axis) and the identification accuracy (right y-axis) in Fig. 14. When the number of lanes is small, *e.g.,* 2, the particle filter converges in nearly 0.5*s* and achieves 100% accuracy. As the number of lanes increases, the accuracy decreases, *e.g.,* for a road of five lanes, the accuracy is 92.7%. The performance slightly degrades for more lanes because the distribution of particles is more dispersed and the peak amount increases as well. In this case, the confidence of each peak is reduced (spreading to other peaks). As a car normally stays on a road segment longer than 2 seconds, it explains the good performance achieved by our host lane identification design in practice.

4.3 Intersection Inference

For the intersection inference module, we first examine the accuracy of the traffic light detection. Since there are no traffic lights on the expressway, the results in Fig. 15 are all from the common roads. As the red light has the longest wave length than other two lights (green and yellow), it has a better penetration and Amateur mainly detects the red light on the screen. From the result, we can see that the detection rate for the red light is highly accurate, where the precision and recall are 99.72% and 99.28%, respectively. As a comparison, if we detect all other two lights at the same time, the precision and recall could slightly degrade to 99.61% and 95.29%, respectively. The reason is that the vehicles are static when detecting red lights. In fact, Amateur can tolerate certain miss-detected traffic lights. We only need to find out the positions of traffic lights on screen when vehicles are distant to lights. Those lights locating at the next intersection can be detected when cars are about 50 meters away from them, which is enough for drivers taking actions and for Amateur displaying instructional annotations on the screen.

After the traffic light is detected, Amateur applies the pin-hole model to determine the position on the screen to display the turning arrow. In our current implementation, the end of a turning arrow is displayed near the middle of the screen's center line and its upper part should ideally align with the intersection in front of the car. However, due to the measurement error, the turning arrow may exhibit an offset with respect to the actual intersection position in the screen, which is denoted as Δ in Fig. 16.

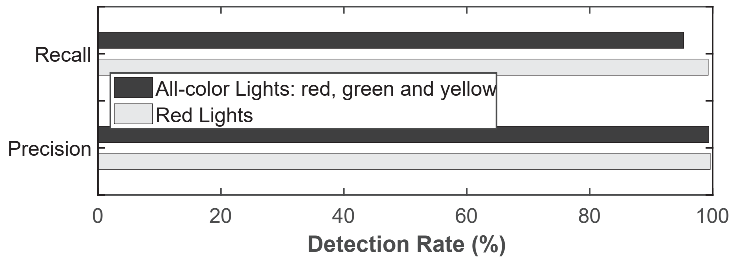

Fig. 15. Precision and recall of the traffic light detection. We compare the detection for red light only and the detection for all three lights together.

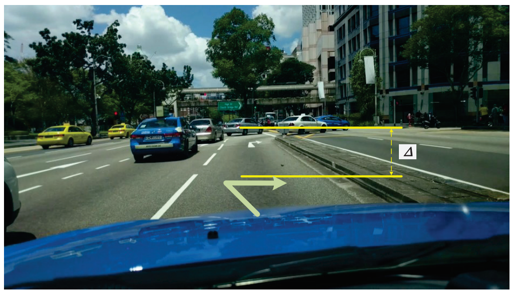

Fig. 16. Illustration of the turning arrow placement offset.

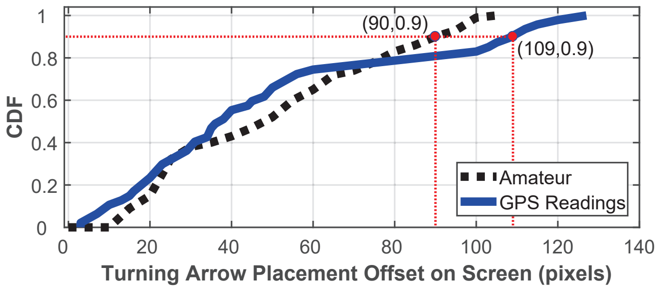

Fig. 17. CDF comparison of the offsets of turning arrow placement on the scree between Amateur and GPS readings in the unit of pixel.

In Fig. 17, we plot the CDF of Δ for all the turnings encountered in our experiment. In addition, we also compare with the performance achieved by the traditional navigation service based on GPS (its result is computed for the evaluation only, not displayed on the screen). From the result, we can see that the average and maximum placement offsets of GPS are 49 (12*m* in terms of distance) and 127 (41.2*m*) pixels, respectively. Amateur could reduce them to 44.7 (9.6*m*) and 105 (26.5*m*) pixels, respectively. After we analyze the captured video for each turning, we find that two methods perform comparably when the GPS signals are strong, while Amateur can leverage the pin-hole model to further improve the GPS's performance in the areas with more skyscrapers nearby.

In Fig. 18, we show a series of concrete Amateur's navigation cases in different scenarios. Compared with the traditional navigation service based on the digit map as in Fig. 18(a), Amateur is more convenient and user friendly. We also find that Amateur performs well even in the rainy and dim environments. In particular, in the dim conditions, such as in Fig. 18(b), (c) and (e), cameras automatically increase the exposure time to improve the frame quality. However, the increased exposure time may affect the detection of flickering of LED bulbs. To address this issue, we currently divide each second into two 0.5*s* parts — one for the flickering capture and the other part after a longer exposure time for getting clear frames. We can also utilize the latest smart phone like Samsung Galaxy S9, Huawei P20 Pro for better image quality in dimmer environment. Fig. 18(e)–(h) further show the lane switching arrows after the lane-changing event is detected. The number above the arrow indicates the number of lanes that the car should change.

4.4 Instruction Correctness

To evaluate the overall instruction correctness, we leverage the Levenshtein distance metric [26] to measure the difference between the ground truth vector and the generated vector by Amateur. It counts how many instructions

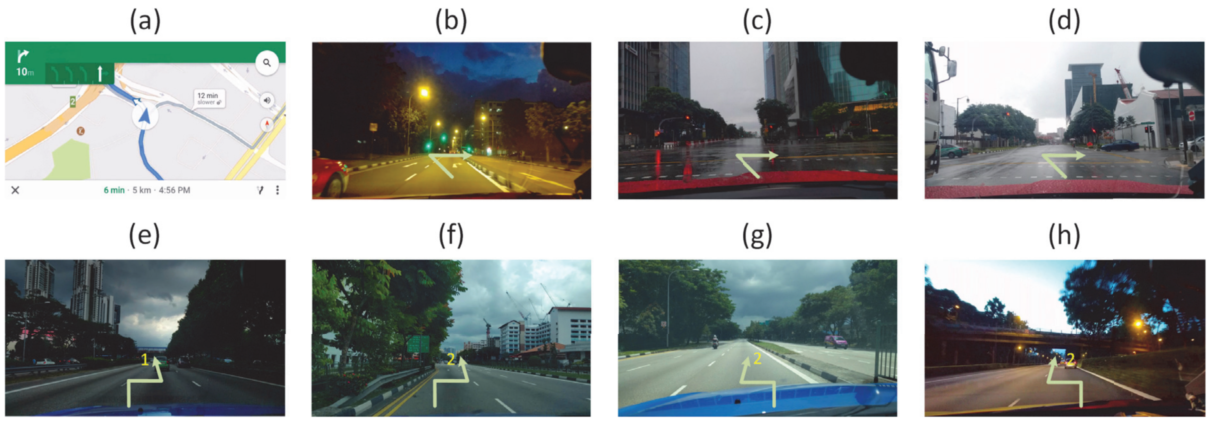

Fig. 18. Screen shots of Amateur and traditional navigation service. (a) Traditional navigation service; (b) Turning arrows at night; (c) Turning arrows at rainy sunset; (d) Turning arrow in rainy daytime; (e) Right 1-lane changing; (f) Right 2-lane changing; (g) Left 2-lane changing; (h) Left 2-lane changing at sunset.

in the generated vector by Amateur need to be modified, so that it becomes the same as the ground truth vector. Suppose the generated vector is $V_{gen} = \{LC2, L, RC1, R\}$ and ground truth vector is $V_{tru} = \{LC1, L, RC1, R\}$. In these two vectors, LC2 means "left changing by 2 lanes"; RC1 means "right changing by 1 lane"; L and R mean "turning left" and "turning right", respectively. The Levenshtein distance between V_{gen} and V_{tru} is 1, which means editing one element in V_{gen}, from $LC2$ to $LC1$, can make these two vectors equal. In general, the smaller the Levenshtein distance indicates higher instruction correctness.

Fig. 19 illustrates the statistical results of Levenshtein distance on each road and examine its performance under different time slots when there are different traffic patterns, *i.e.,* morning peak, evening peak and off traffic peak. From the result, we observe that Route C has the largest Levenshtein distance 9 when running Amateur during the evening peak, while the smallest one is on Route A without the traffic peak. In Fig. 19, compared with the total number of annotations made for each road, *i.e.,* 43, 74, 101, 67 and 86 for Routes A to E respectively, the instruction correctness of Amateur is high ranging from 88.4% to 97.3%.

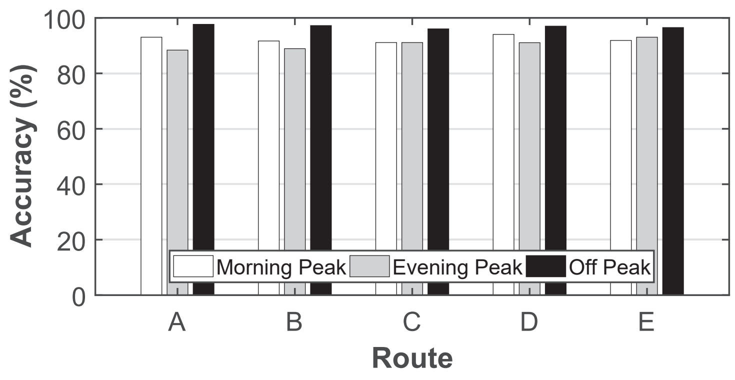

Fig. 19. Accuracy of arrow placement on the different routes with different traffics.

Even the accuracy may vary in different traffic conditions, we find that **none** of instruction errors lead to the wrong navigation destination. To further understand the reason of these annotation errors, we refer to the ground truth and observe that the errors are all caused by the lane-changing event. When the current host lane is detected wrongly, Amateur will recommend drivers to change the lane, which actually is not needed. However, Amateur keeps tracking the host lane by our particle filter design and this unnecessary lane-changing can be corrected by changing back to the real interested-lane, which is essentially error-tolerant. After we examine the ground truth, we find that the reason to cause the host lane detection error is mainly due to the shield of lane markers by nearby vehicles or some bright reflections on the road (*i.e.,* noise). Fortunately, nearby cars do not cover the lane markers at the same position for very long time. Their impact is thus temporary. Moreover, the error rate itself is not high, *e.g.,* less than 10%. Therefore, we observe that such errors did not cause the navigation to a wrong destination throughout our experiments. Please note that in case that Amateur made a wrong navigation, similar as the existing navigation systems, it will re-calculate the route and treat it as a new navigation from the current location to the destination. In summary, Amateur can provide convenient and reliable navigation experience in practice.

4.5 User Study

4.5.1 Study Design. Finally, we conduct a with-in subject user study with 50 taxi drivers to understand the user's experience of Amateur in practice and we also compare it with the traditional navigation method, *e.g.,* Google Maps. In the user study, taxi driver uses one of two navigation systems (Amateur or Google Maps) first on route D of Figure 11. After the taxi reaches the destination, we ask the driver to come back to the origin and drive again using another one. For the first 25 taxi drivers (Group A), they use Amateur first and then Google Maps. The other 25 taxi drivers (Group B) switch the order of the system usage. This is a standard technique named Latin Square [31] to counterbalance the order effect. After arriving the destination each time, the taxi driver rates the system just used for the navigation. The questionnaire is detailed in Table 2.

Table 2. Likert scale rating questions in our user study.

No.	Rating Question Statement
Q1	It was easy to navigate using this navigation service.
Q2	I need to pay extra attention on this navigation service when driving.
Q3	This navigation service provided user-friendly guidance.
Q4	This navigation service was useful in helping me navigate properly.
Q5	It was easy for me to learn how to use this navigation service.
Q6	I paid most of my attention on driving using this navigation service.
Q7	The guidance was user-friendly to interact with.
Q8	This navigation service provided me with effective guidance.

1) Question design. For the question design, we consider the following four key points that we believe are important to evaluate a navigation system.

- **Ease of use**: we propose Q1 and Q5 in Table 2.
- **Perceived distraction**: we propose Q2 and Q6 in Table 2.
- **Navigation experience**: we propose Q4 and Q8 in Table 2.
- **User-friendliness**: we propose Q3 and Q7 in Table 2.

In particular, "ease of use" and "navigation experience" have been adopted in a prior simulated AR-based navigation design [21]. Since they have different survey targets, we only utilize these two points instead of

Table 3. Detailed information of the participating drivers.

Property	Description of Group A	Description of Group B
Age (Year)	32-61 (Mean 40.3)	31-57 (Mean 42.8)
Driving Experience (Year)	2-27 (Mean 15.8)	1-30 (Mean 17.6)
Gender	21 Male (84%), 4 Female (16%)	19 Male (76%), 6 Female (24%)

their exact questions. In addition, we further propose "perceived distraction" and "user-frienliness" in our user study, which are closely related to our design. Based on the theory from MIS (Management Information System) research domain, two questions for each aspect could improve the reliability and validity of our questionnaire [7].

2) Participants. Table 3 summarizes the detailed information of the participanting taxi drivers. In Group A, there are 21 male and 4 female drivers. Their ages are from 32 and 61 years old (Mean: 40.3 years old) and driving experiences are within 2 to 27 years. In Group B, there are 19 male and 6 female drivers. Their ages are from 31 and 57 years old (Mean: 42.8 years old) and driving experiences are within 1 to 30 years.

3) Rating methodology. For each question, we adopt the standard 7-point Likert scale rating mechanism, where each question will be rated by a scaling score from 1 (strongly disagree) to 7 (strongly agree). The results from these 50 taxi drivers are summarized in Fig. 20.

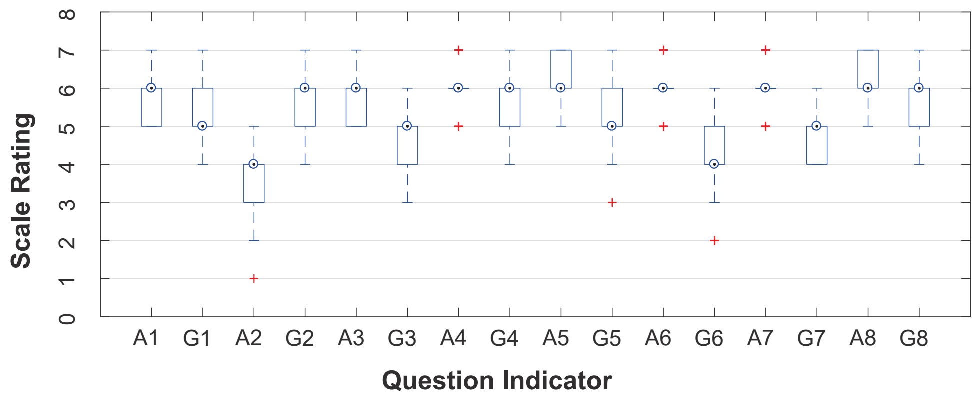

Fig. 20. Results of Likert scale rating (1: strongly disagree ~ 7: strongly agree). On x-axis, 'A' means Amateur; 'G' means Google Maps and the following number is the question number in Table 2. Median is shown as a ⊙. The ratings for A4, A6 and A7 have different means and standard deviations. However, due to the illustration, they look to be identical in the figure.

4) Numeric analysis. With the 7-point Likert scale rating mechanism and the results in the Fig. 20, we can then apply the standard method called Wilcoxon Signed Rank Tests to mathematically analyze the user study, where the significance level is set as $\alpha = 0.05$. The results of inferential statistics analysis and the observations are detailed in the following.

4.5.2 Results. With the 7-point Likert scale rating mechanism, we then apply the standard method to mathematically analyze the user study and obtain the insight for each question design considerations. In particular, the p-value results are summarized in Table 4 and we have the following observations.

- For the ease of use, the results show that there was a significant difference between the two systems ($p = 0.0016$) for Q1. Fig. 20 depicts that Amateur is easier to use comparing with Google Maps, where the mean of our system A5 (*i.e.,* $mean_{A1} = 5.78$) is larger than that of Google Maps G5 (*i.e.,* $mean_{G1} = 5.44$) . In

Table 4. The p-values for the 7-point Likert scale rating questions.

Question No.	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
$p - value$	0.0016	< 0.0001	< 0.0001	0.0003	< 0.0001	< 0.0001	< 0.0001	< 0.0001

addition, there was a significant difference between two systems ($p < 0.0001$) for Q5. The result indicates that Amateur is easier for drivers to learn how to use the navigation system than the traditional service Google Maps as in Fig. 20, where $mean_{A5} = 6.08$ and $mean_{G5} = 5.42$.

- For the perceived distraction, the result indicates a significant difference of two systems for both Q2 and Q6 ($p < 0.0001$ for both). The drivers pay more attention on the traditional service (*i.e.,* $mean_{G2} = 5.5$) than that of Amateur (*i.e.,* $mean_{A2} = 3.72$). The rating results also show that drivers focus more on driving when utilizing Amateur while they might focus more on the navigation display when utilizing the traditional service. The averaged ratings are $mean_{A6} = 5.9$ and $mean_{G6} = 4.26$. Therefore, Amateur can cause much less distractions than the traditional service.
- For the navigation experience, the results of Q4 and Q8 showed that there were significant differences between two systems, with p-value being $p = 0.0003$ and $p < 0.0001$ for Q4 and Q8 respectively. The drivers gave better ratings for Amateur ($mean_{A4} = 5.94$, $mean_{A8} = 6.2$) than the traditional service ($mean_{G4} = 5.44$, $mean_{G8} = 5.56$).
- For the user-friendliness, two systems also exhibit quite different results (Q3: $p < 0.0001$ and Q7: $p < 0.0001$). Compared to the traditional system ($mean_{G3} = 4.76$, $mean_{G7} = 4.92$), Amateur is more user-friendly in the rating results with $mean_{A3} = 5.76 > mean_{G3}$ and $mean_{A7} = 6 > mean_{G7}$. The main reason is that the AR display is much easier to understand for the drivers, which is also our original motivation.

In summary, the results indicate that Amateur can provide a better navigation service than the traditional method for the ease of use, perceived distraction, navigation experience and user-friendliness four aspects.

5 LIMITATIONS AND DISCUSSIONS

In this section, we discuss some limitations of our system and the potential future works.

1) Brightness requirement. The primary requirement of Amateur is that the environment cannot be completely dark. We have tested Amateur at different time periods of a day. The system works well in the daytime and dimmer environments. Since Amateur utilizes general camera, if the environment is completely dark, Amateur cannot extract information from the captured images. Nowadays, the camera advances rapidly on smart phones and one major effort of smart phone companies is to improve the picturing performance in the dark. In the near future, once camera can capture more meaningful contents in a dark environment, this limitation can be alleviated and the performance of our system can be improved accordingly.

2) System setup. To deploy a smart phone with Amateur on a car, we need the following two setups. First, the phone needs to be placed along the gravity direction, which is required by the pin-hole model. Fortunately, the motion sensor (more precisely the gyroscope) on the smart phone provides accurate measurement of the posture which we can leverage in the setup phase. The phone posture may not strictly align with the gravity direction even after the setup phase. Through our experiment, we find that the detected position of intersection on the screen could differ from its real position on the screen in tens of pixels, which is still acceptable. Second, we need to select the position of a narrow strip for Amateur to perform the lane boundary detection. In our implementation on Nexus 5X, the current setting of the 230 pixels corresponds to a horizontal line about 5 meters in front of the car. In the future, we plan to introduce a narrow strip appeared on the Amateur's UI and let the driver drag this strip on the screen and move to the appropriate position, which gives the driver freedom to configure this system parameter. We note that these setups are one-time overhead for installing Amateur.

3) Availability of traffic light information. In Amateur design, we leverage traffic lights as an indicator of the position of intersection using a pin-hole model. This model needs the traffic light height as one input. However we may lack the specification for this information in some cities. In some scenarios, *e.g.,* usually outside of the city, the traffic lights may even miss at intersections due to lack of the infrastructure. In these cases, GPS always serves as a fall back solution to provide the estimated distance to the front crossroad. Such a distance can also help determine the arrow's position in the video. As a matter of fact, our design incorporates with GPS (instead of being independent with GPS) and takes the GPS reading as one system input. However, GPS could become inaccurate in many urban areas, and the detection of the traffic light essentially provides an opportunity to leverage the pin-hole model to improve the distance estimation. To further improve the availability of the traffic light height information for more cities, such a database could be constructed through the crowdsourcing, which is one feasible solution to overcome this limitation in the near future.

4) Camera's field of view. For the general camera, we find that it can capture five lanes at the same time. If a road has more than five lanes, as long as Amateur has detected one of the left most or right most lanes initially (most likely since a car rarely starts at the center of one road), the particle filter can continuously track the host lane. In case that the host lane tracking is lost or inaccurate for a long time, the driver can explicitly change the lane to let Amateur detect one of the left most or right most lanes to track the host lane again.

5) Camera sampling rate. In our current design, we set the sampling rate of the camera sensor at 30 frames per second (FPS). We find that the difference between consecutive frames is acceptably small for the normal driving speed, *e.g.,* $< 30km/h$. A larger speed may cause larger difference in subtracted image, leading to longer time to detect the traffic light. Higher sampling rate (*e.g.,* 120 FPS or 240 FPS) can ensure a reliable detection even under a high speed, but it increases the computing overhead. This is a trade-off between the efficiency and the computing overhead. We believe that as smart phones have more powerful CPUs in near future, Amateur can support higher sampling rates, which can further enhance the performance of Amateur.

6 CONCLUSION

This paper presents an augmented reality based vehicle navigation system, called Amateur, using commodity smart phones. Amateur can automatically display navigation instructions as a series of annotative arrows on the live road condition video stream to assist the driving. The system design encounters host-lane identification and intersection determination challenges. We propose a set of effective techniques to address these issues and our solution can comfortably work with lightweight smart phone platforms. To demonstrate the efficacy of the Amateur design, we implement the system on Android phones. The experimental results indicate that Amateur can improve the navigation service from four important aspects: ease of use, perceived distraction, navigation experience and user-friendliness.

REFERENCES

[1] Mohamed Aly. 2008. Real time detection of lane markers in urban streets. In *Proceedings of IEEE IV*. 7–12.

[2] Donna R Berryman. 2012. Augmented reality: a review. *Taylor & Francis Medical reference services quarterly* 2 (2012), 212–218.

[3] John Canny. 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6 (1986), 679–698.

[4] Julie Carmigniani, Borko Furht, Marco Anisetti, Paolo Ceravolo, Ernesto Damiani, and Misa Ivkovic. 2011. Augmented reality technologies, systems and applications. *Springer Multimedia tools and applications* 1 (2011), 341–377.

[5] Thanh-Son Dao, Keith Yu Kit Leung, Christopher Michael Clark, and Jan Paul Huissoon. 2007. Markov-based lane positioning using intervehicle communication. *IEEE Transactions on Intelligent Transportation Systems* 4 (2007), 641–650.

[6] A David and Ponce Jean. 2002. Computer vision: a modern approach. (2002).

[7] Fred D Davis. 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly* (1989), 319–340.

[8] Moises Diaz, Pietro Cerri, Giuseppe Pirlo, Miguel A Ferrer, and Donato Impedovo. 2015. A survey on traffic light detection. In *Proceedings of Springer ICIAP*. 201–208.

[9] Exploride. 2018. Exploride website. (Jan. 2018). Retrieved May 4, 2018 from https://exploride.com/
[10] Eric Foxlin, Thomas Calloway, and Hongsheng Zhang. 2014. Improved registration for vehicular AR using auto-harmonization. In *Proceedings of IEEE ISMAR*. 105–112.
[11] Peter Froehlich, Raimund Schatz, Peter Leitner, Stephan Mantler, and Matthias Baldauf. 2010. Evaluating realistic visualizations for safety-related in-car information systems. In *Proceedings of ACM CHI*. 3847–3852.
[12] Peter Fröhlich, Matthias Baldauf, Marion Hagen, Stefan Suette, Dietmar Schabus, and Andrew L Kun. 2011. Investigating safety services on the motorway: the role of realistic visualization. In *Proceedings of ACM AutomotiveUI*. 143–150.
[13] Peter Fröhlich, Raimund Schatz, Peter Leitner, Matthias Baldauf, and Stephan Mantler. 2010. Augmenting the driver's view with realtime safety-related information. In *Proceedings of ACM AH*. 11.
[14] Jianwei Gong, Yanhua Jiang, Guangming Xiong, Chaohua Guan, Gang Tao, and Huiyan Chen. 2010. The recognition and tracking of traffic lights based on color segmentation and CAMSHIFT for intelligent vehicles.. In *Proceedings of IEEE IV*. 431–435.
[15] Google. 2018. Google I/O Event. (May 2018). Retrieved May 9, 2018 from https://events.google.com/io/
[16] Google. 2018. Google Maps. (Jan. 2018). Retrieved February 11, 2016 from https://itunes.apple.com/us/app/google-maps/id585027354?mt=8
[17] Google. 2018. Google Maps Direction API. (Jan. 2018). Retrieved February 1, 2016 from https://developers.google.com/maps/documentation/directions/intro?hl=en
[18] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. 2014. Recent progress in road and lane detection: a survey. *Springer Machine Vision and Applications* 3 (2014), 727–745.
[19] Hudify. 2018. WAYRAY Website. (Jan. 2018). Retrieved May 4, 2018 from http://www.gethudify.com
[20] Yurong Jiang, Hang Qiu, Matthew McCartney, Gaurav Sukhatme, Marco Gruteser, Fan Bai, Donald Grimm, and Ramesh Govindan. 2015. CARLOC: Precise Positioning of Automobiles. In *Proceedings of ACM SenSys*. 253–265.
[21] Richie Jose, Gun A Lee, and Mark Billinghurst. 2016. A comparative study of simulated augmented reality displays for vehicle navigation. In *Proceedings of ACM OzCHI*. 40–48.
[22] Soonhong Jung, Junsic Youn, and Sanghoon Sull. 2016. Efficient lane detection based on spatiotemporal images. *IEEE Transactions on Intelligent Transportation Systems* 1 (2016), 289–295.
[23] Clemens Kaufmann, Ralf Risser, Arjan Geven, and Reinhard Sefelin. 2008. Effects of simultaneous multi-modal warnings and traffic information on driver behaviour. In *Proceedings of European Conference on Human Centred Design for Intelligent Transport Systems*. 33–42.
[24] Emmanouil Koukoumidis, Li-Shiuan Peh, and Margaret Rose Martonosi. 2011. SignalGuru: leveraging mobile phones for collaborative traffic signal schedule advisory. In *Proceedings of ACM MobiSys*. 127–140.
[25] Ye-Sheng Kuo, Pat Pannuto, Ko-Jen Hsiao, and Prabal Dutta. 2014. Luxapose: Indoor positioning with mobile phones and visible light. In *Proceedings of ACM MobiCom*. 447–458.
[26] Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*. 707–710.
[27] Masako Omachi and Shinichiro Omachi. 2009. Traffic light detection with color and edge information. In *Proceedings of IEEE ICCSIT*. 284–287.
[28] OSM. 2018. Open Street Map. (Jan. 2018). Retrieved February 1, 2016 from https://www.openstreetmap.org
[29] Oskar Palinko, Andrew L Kun, Zachary Cook, Adam Downey, Aaron Lecomte, Meredith Swanson, and Tina Tomaszewski. 2013. Towards augmented reality navigation using affordable technology. In *Proceedings of ACM AutomotiveUI*. 238–241.
[30] Christian M Richard, Richard D Wright, Cheryl Ee, Steven L Prime, Yujiro Shimizu, and John Vavrik. 2002. Effect of a concurrent auditory task on visual search performance in a driving-related image-flicker task. *SAGE Human Factors: The Journal of the Human Factors and Ergonomics Society* 1 (2002), 108–119.
[31] John TE Richardson. 2018. The use of Latin-square designs in educational and psychological research. *Educational Research Review* (2018), 84–97.
[32] Michelle Krüger Silvéria Santos. 2013. Virtual windshields: merging reality and digital content to improve the driving experience. (2013).
[33] H Sawano and M Okada. 2006. Real-time video processing by a car-mounted camera and its application to car navigation systems by an augmented reality-based display method. *The Journal of the Society for Art and Science* 2 (2006), 57–68.
[34] Longfei Shangguan, Zheng Yang, Alex X Liu, Zimu Zhou, and Yunhao Liu. 2017. STPP: Spatial-temporal phase profiling-based method for relative RFID tag localization. *IEEE/ACM Transactions on Networking* 25, 1 (2017), 596–609.
[35] Sygic. 2018. Sygic Official Website. (Jan. 2018). Retrieved May 2, 2018 from https://www.sygic.com/gps-navigation
[36] Hwang Tae-Hyun, Joo In-Hak, and Cho Seong-Ik. 2006. Detection of traffics for vision-based car navigation system. In *Springer Advances in Image and Video Technology*. 682–691.
[37] Zhenning Tao, Philippe Bonnifait, Vincent Fremont, and Javier Ibanez-Guzman. 2013. Lane marking aided vehicle localization. In *Proceedings of IEEE ITSC*. 1509–1515.
[38] TESLA. 2018. Tesla official websit. (Jan. 2018). Retrieved March 2, 2016 from https://www.teslamotors.com/

[39] Rafael Toledo-Moreo, David Bétaille, and François Peyret. 2010. Lane-level integrity provision for navigation and map matching with GNSS, dead reckoning, and enhanced maps. *IEEE Transactions on Intelligent Transportation Systems* 1 (2010), 100–112.

[40] WAYRAY. 2018. WAYRAY Website. (Jan. 2018). Retrieved May 4, 2018 from https://wayray.com/navion

[41] Yoshihisa Yamaguchi, Takashi Nakagawa, Kengo Akaho, Mitsushi Honda, Hirokazu Kato, and Shogo Nishida. 2007. AR-Navi: An in-vehicle navigation system using video-based augmented reality technology. In *Springer Symposium on Human Interface and the Management of Information*. 1139–1147.

[42] Chuang-Wen You, Nicholas D Lane, Fanglin Chen, Rui Wang, Zhenyu Chen, Thomas J Bao, Martha Montes-de Oca, Yuting Cheng, Mu Lin, Lorenzo Torresani, et al. 2013. Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones. In *Proceedings of ACM MobiSys*. 13–26.

Received May 2018; revised August 2018; accepted December 2018