ALT: Towards Automating Driver License Testing using Smartphones

Akshay Uttama Nambi, Ishit Mehta, Anurag Ghosh, Vijay Lingam, Venkata N. Padmanabhan
Microsoft Research India
https://aka.ms/hams

ABSTRACT

Can a smartphone administer a driver license test? We ask this question because of the inadequacy of manual testing and the expense of outfitting an automated testing track with sensors such as cameras, leading to less-than-thorough testing and ultimately compromising road safety. We present ALT, a low-cost smartphone-based system for automating key aspects of the driver license test. A windshield-mounted smartphone serves as the sole sensing platform, with the front camera being used to monitor driver's gaze, and the rear camera, together with inertial sensors, being used to evaluate driving maneuvers such as parallel parking. The sensors are also used in tandem, for instance, to check that the driver scanned their mirror during a lane change.

The key challenges in ALT arise from the variation in the subject (driver) and the environment (vehicle geometry, camera orientation, etc.), little or no infrastructure support to keep costs low, and also the limitations of the smartphone (low-end GPU). The main contributions of this paper are: (a) robust detection of driver's gaze by combining head pose and eye gaze information, and performing auto-calibration to accommodate environmental variation, (b) a hybrid visual SLAM technique that combines visual features and a sparse set of planar markers, placed optimally in the environment, to derive accurate trajectory information, and (c) an efficient realization on smartphones using both CPU and GPU resources. We perform extensive experiments, both in controlled settings and on an actual driving test track, to validate the efficacy of ALT.

CCS CONCEPTS

• **Human-centered computing** → **Mobile computing**; • **Computing methodologies** → *Computer vision*;

KEYWORDS

Automated license testing, gaze tracking, SLAM, Road safety

ACM Reference Format:
Akshay Uttama Nambi, Ishit Mehta, Anurag Ghosh, Vijay Lingam, Venkata N. Padmanabhan. 2019. ALT: Towards Automating Driver License Testing using Smartphones. In *The 17th ACM Conference on Embedded Networked Sensor Systems (SenSys '19), November 10–13, 2019, New York, NY, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3356250.3360037

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SenSys '19, November 10–13, 2019, New York, NY, USA
© 2019 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery.
ACM ISBN 978-1-4503-6950-3/19/11…$15.00
https://doi.org/10.1145/3356250.3360037

1 INTRODUCTION

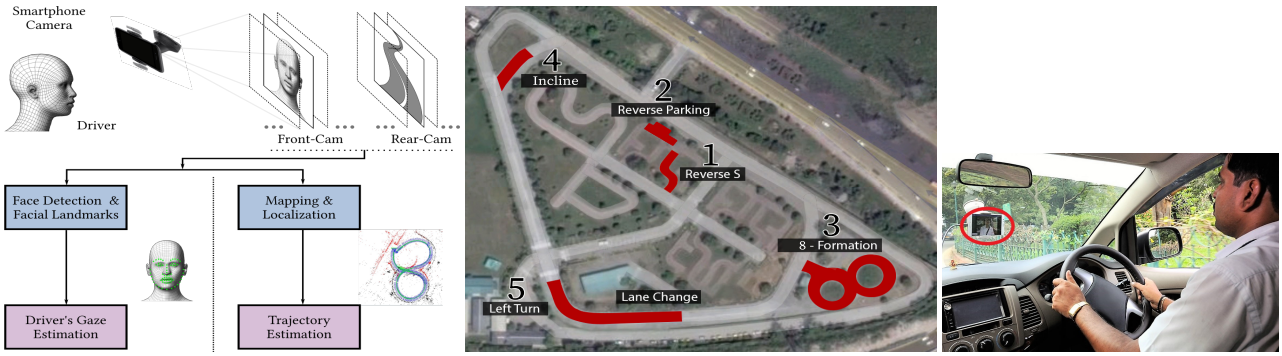
Road safety is a major public health issue, with road accidents causing an estimated 1.25 million fatalities and many more injuries each year [7], placing it among the top ten causes of death. Many studies have found that the primary factors responsible for road accidents centre on the driver [8]. A significant contributor is that drivers are often not trained or tested adequately, e.g., an estimated 59% of licenses in India were issued without conducting a test [10].

The reasons for inadequate or improper testing of drivers are many. It is tedious for a human inspector to thoroughly monitor and evaluate all aspects of a license applicant's driving. Indeed, it is quite challenging for an inspector, who is seated beside the driver, to turn and monitor the driver's gaze, say to know whether the driver is scanning the mirrors. The human involvement also creates incentives for shortcuts and corruption. Efforts at automation, centered around instrumenting driving tracks, have been stymied by poor test parameters coverage and high cost (e.g., a system with 15 pole-mounted cameras, for monitoring just the vehicle trajectory during driving maneuvers on a track, costs about USD 100K [11].)

Our goal is to have a smartphone administer the driver license test in an automated manner. We present ALT, a step towards **A**utomating **L**icense **T**esting using a smartphone. Dependence on the smartphone and little else makes ALT low-cost, scalable, and hence amenable to broad deployment. Figure 1a shows the overview of ALT. The smartphone in ALT is mounted on the windshield or dashboard, such that its front-facing camera has a view of the driver's face and monitors the driver's gaze, while the rear-facing camera has a view of the road for evaluating driving maneuvers.

Driver license testing entails two broad components: (1) how well and safely a driver performs in regular driving scenarios (e.g., scanning mirrors before performing lane changes or turns), and (2) how much control the driver exhibits during special maneuvers (e.g., driving in a figure of 8 or parallel parking). Testing a driver on the former entails monitoring vehicle maneuvers (e.g., lane change) and driver actions (e.g., mirror scanning) in tandem, while testing on the latter focuses largely on the maneuvers but requires them to be tracked with precision (e.g., whether the vehicle was parallel parked within the designated box or not). We leverage prior work for detecting speeding, lane changes, turns, etc. [20, 47, 54]. In ALT, we focus on the novel technical challenges pertaining to driver testing and our contributions are:

1. Driver gaze monitoring: During maneuvers such as lane change or turn, it is important to know whether the driver scanned the mirrors *before* executing the maneuver. Even during regular driving, it is important to know that the driver is scanning the mirrors regularly to maintain situational awareness. It is quite



(a) System overview **(b) Layout of the driving test track** **(c) Smartphone mounting in ALT**

Figure 1: Overview of ALT.

tedious and hence challenging for a human inspector, who sits beside the driver, to monitor the driver's gaze.

In principle, the front camera of the smartphone in ALT could be used to monitor the driver's gaze. However, challenges arise because of variation in the environment (e.g., phone orientation, vehicle geometry) and across drivers (e.g. seating position). To address these challenges, ALT*combines head pose and eye tracking information* to determine the driver's gaze. Furthermore, ALT employs *auto-calibration*, where we cluster observations of gaze direction over time to learn the positions of the mirrors relative to the driver automatically, without requiring any manual per-driver training (Section 3). This is more robust than past work, which has used fixed thresholds or focused on the head pose in isolation [32, 54].

2. Fine-grained vehicle tracking: Fine-grained tracking of a vehicle's trajectory and movement are key to evaluate driving maneuvers. For instance, it is necessary to know whether the vehicle was parked within a designated box during parallel parking, or that it did not roll back while climbing up an incline (see Section 2.2 for a description of these driving tests). Both GPS positioning and inertial tracking fall short of the requirement of fine-grained resolution.

In ALT, we develop a novel *hybrid visual SLAM technique*, aided by a minimal deployment of fiducial markers (just printed on sheets of paper, as shown in Figure 5(c)). The hybrid visual SLAM approach tracks vehicle trajectory using just a smartphone camera and reduces trajectory error to 20 cm across various maneuvers. This improves on prior work, yielding better accuracy than approaches based just on visual features [45]. It uses fewer fiducial markers and avoids the need for prior mapping, in contrast to approaches based just on fiducial markers [27] (Section 4).

3. Efficient smartphone implementation: For efficiency, our implementation of ALT takes full advantage of the typical low-end, non-Nvidia GPUs available on these platforms. We perform several optimizations towards efficient facial-landmarks tracking for driver's gaze estimation and fuse inertial and GPS data to speed up execution of visual localization on smartphones (Section 8).

We have evaluated ALT's efficacy in automating aspects of driver license testing, with multiple drivers in actual driving test tracks. We performed an extensive evaluation of driver's gaze and driving maneuvers. We obtained an accuracy (F_1 score) of 91% for mirror scan detection and the trajectory derived using the proposed hybrid approach outperforms the state-of-the-art visual SLAM (simultaneous localization and mapping) techniques. Our implementation of ALT as an Android app can support processing at up to 10 fps (frames per second) on a low-end smartphone.

Besides the above, there are also other aspects of ALT that help address the inadequacies of manual testing today and improve the integrity of the testing process (see [28] for more details). For instance, continuous face recognition helps ensure that the person taking the test is the same as the one to whom the license is being issued. However, in the interest of focusing on the novel research elements, we do not discuss such additional aspects in this paper.

2 SYSTEM DESIGN

2.1 Driving test overview

In ALT, we focus on a driving test conducted on a testing track like that pictured in Figure 1b. It comprises several segments, each designed to test the driver's skills in the context of a specific maneuver. In each maneuver, various parameters are monitored to evaluate the driving. These parameters are generally derived based on the rules for the issuance of driving licenses per the regulations of the respective countries [1, 2]. Some of these parameters according to Indian Central Motor Vehicles Rules [1] include the driver's ability to maneuver the vehicle in reverse, obey traffic signal lights, the driver's gaze for situational awareness, maintaining smooth speed profile, and so on. Based on extensive conversations with personnel involved in driver training and license testing across multiple cities, we arrived at the following set of tests to focus on automating in ALT (although the specifics of the tests may vary, we believe that these tests are representative of those in other countries too [3]):

(1) **Reverse S maneuver:** The driver reverses through an S-shaped track and is expected to do so without changing direction (i.e., rolling forward) more than twice.
(2) **Reverse Parking (RP):** The driver reverses into the designated parking spot. The driver is expected to execute this maneuver without changing direction (i.e., rolling forward) more than twice and park their vehicle within the designated box.
(3) **8-formation:** The driver is expected to drive through an 8-shaped section with proper speed control and without changing direction (i.e., stopping and reversing). The driver is expected to enter and exit the 8 section at the designated points.
(4) **Incline start:** The driver is asked to bring the vehicle to a stop on an inclined road segment and then asked to start moving forward again, without rolling back more than 12 inches or stalling the engine.

(5) **Turns and Lane change:** The driver drives through a left/right turn and/or lane change, where the driver's gaze is monitored to check if they scanned the mirrors correctly before executing the turn or lane change.

Current driving tests are typically conducted manually, with an inspector seated beside the driver. This is prone to shortcuts and provides inadequate coverage as the human inspector may not be able to monitor all aspects of the candidate's driving behavior such as mirror scanning. This motivates the need for automating driving license tests. In an automated setup, the system has to objectively evaluate various parameters in each maneuver and determine whether the driver passed the driving test or not. The accuracy requirement for evaluating these parameters are relatively high. For example, the system should be able to accurately detect a mirror scan even if the driver performs one quick glance during a lane change. Similarly, for trajectory estimation, the positioning error should be under 15-20 cm, so that the system can evaluate various parameters accurately, e.g., whether the vehicle is parked inside the designated area or how much it rolled back during an incline start. Furthermore, the system should be able to process the videos and evaluate all the parameters in near real-time so that soon after the test, the candidate knows the result and the mistakes, if any, committed during the test.

2.2 ALT Design Choices and Setup

For reasons of cost and ease of setup, the automation system should ensure there is little or no instrumentation of the infrastructure. Furthermore, the license test is typically conducted on the driver's own vehicle, so there is little scope for extensive instrumentation of the vehicle.

Accordingly, ALT uses a minimal setup comprising a windshield-mounted smartphone as the sole sensing platform. This setup is used to gather multiple sensor streams, including imagery captured simultaneously from the front (driver-facing) and rear (road-facing) cameras at 1080p and 30 fps, along with the inertial sensor and GPS data. Based on extensive discussion and evaluation with the drivers in our deployments, we have found the optimal placement of the smartphone in the vehicle to be just below the center mirror (as shown in Figure 1c), as it does not obstruct driver's view.

It is clear that the tests noted in Section 2.1 require precise and continuous monitoring of the trajectory and movement of the vehicle, along with driver's gaze. To do these with just a smartphone is challenging. ALT makes novel contributions in addressing the following technical challenges.

2.2.1 ***Robust monitoring of driver's gaze.*** Past work on driver gaze detection has focused on just tracking the driver's head pose [32, 38, 54]. We argue that this is insufficient; indeed, as our results in Section 6.2 show, a pure head pose based approach tends to fail in real-world settings in which a driver often scans the surrounding environment with quick glances, without turning their head much. Therefore, to robustly track the driver's gaze, it is imperative to track the eyes as well as the head pose of the driver. To address this, we train a deep neural network (DNN) to *combine the head pose information and eye patch image to obtain the direction of the driver's gaze* (Section 3.2).

2.2.2 ***Accurate vehicle trajectory tracking:*** The maneuvers noted above require accurate, sub-meter tracking using a single (i.e., monocular) smartphone camera. One possible approach is to use visual SLAM to derive trajectories, wherein visual features, or keypoints, detected in successive frames are used to obtain correspondences and thereby estimate camera pose. However, the feature points obtained tend to be fragile, resulting in noisy matching across frames. Also, this approach lacks knowledge of the true scale of the 3D scene, as detailed in Section 4.1. To overcome these issues, recent work [27] has used an exhaustive deployment of fiducial markers of known size in the environment to enable accurate trajectory estimation. However, this involves plastering the environment with markers to ensure that at least few are within view of the camera at all times and moreover requires recording the ground truth location of each marker, both of which are tedious.

We present a *novel hybrid-visual-SLAM technique that combines visual features and sparse fiducial markers to obtain the accurate trajectory of the vehicle.* Specifically, we introduce fiducial markers, in the form of April Tags [48], in a scene. Each marker provides four correspondence points (corners of the marker), that can be localized with pixel-level precision. The proposed SLAM technique detects these markers and uses the corresponding features to compute correct camera pose information, while adjusting scale based on the known fiducial marker size (see Section 4.2). Furthermore, by also leveraging visual features in the image scene, the need for fiducial markers in the environment is reduced and the need for knowing their exact positions and mapping the environment is also avoided.

2.2.3 ***Efficient realization of ALT on smartphones.*** Since a smartphone is used as the sole sensing and computation platform, *we perform several optimizations to run ALT efficiently on smartphones.* To derive the driver's gaze, we use a standard facial landmark model [35], which yields 68 landmarks, as the starting point but then identify a subset of 22 landmarks that yields a significant speedup up to 1.5×, while trading off some accuracy (Section 8.1). Furthermore, we leverage the GPU available on a low-end smartphone to run our DNN model for driver gaze estimation, yielding over 2× speedup as compared to running on the CPU. Finally, while the hybrid SLAM approach, by default, uses all the frames to estimate the vehicle's trajectory, the fact is that in between successive frames, there is very little or no displacement and hence the visual features obtained will be almost the same. Therefore, we discard all frames with little or no displacement to trade off some accuracy for speed of execution.

3 ROBUST DRIVERS GAZE MONITORING

We now present how ALT monitors the driver's gaze information robustly, focusing on our novel contributions of combining the head pose and eye gaze (Section 3.2), and performing auto-calibration to accommodate variations in camera position, driver seating, and vehicle configuration (Section 3.3).

3.1 Facial Landmarks Detection

We present some background and our optimizations on facial landmarks detection, which is fundamental to estimating a driver's gaze. The first step is face detection, which yields a bounding box around the driver's face [25]. We then detect facial landmarks, corresponding to facial features such as eye corners, mouth corners, nose tip, etc., as shown in Figure 2(a).

(a) (b)

Figure 2: Facial landmarks (a) 68-landmarks derived using CLNF model (b) 22 landmarks used in ALT.

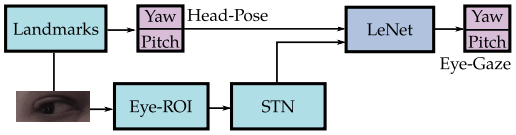


Figure 3: Overview of driver gaze detection.

To robustly obtain landmarks in a driving setting, we extend state-of-the-art computer vision algorithms, *viz.,* the CLNF (Constrained Local Neural Fields) [17] facial landmark model, by training on new datasets, which include multiple subjects, with extreme head poses, and diverse lighting conditions, mimicking real-world conditions. We use Menpo [55] and Multi-Pie [29] datasets that include over 30,000 face images with annotations and exhibiting a wide range in pose, expression, gender, and environmental conditions, to obtain 68 facial landmarks. We chose to build on top of CLNF models because: (i) these models are relatively computationally inexpensive and can run on smartphones, and (ii) they are robust against adverse lighting conditions. While CLNF models are computationally less expensive, we still need to further optimize their operation to run efficiently on smartphones. We now present two optimizations:

3.1.1 Detect and localize a subset of 68 landmarks. Based on our experimentation and prior work, we picked out landmarks corresponding to the nose, chin, eye and mouth regions, to determine the accurate head pose. This yields a subset of 22 landmarks, as shown in Figure 2(b), which is 1.5× faster to detect than the set of all 68 landmarks (see Section 8.1 for detailed results.

3.1.2 Landmarks tracking in subsequent frames. To minimize the cost incurred on landmark detection, ALT performs tracking of the identified landmarks, where possible, instead of a fresh detection in consecutive frames. Since the driver's face position does not change drastically across successive frames, tracking previously obtained landmarks is robust and efficient (as it narrows the region of interest for the landmark predictor). The landmarks are re-initialized (i.e., a fresh detection is made) only when there is no face detected or the landmarks in successive frames do not overlap sufficiently, indicating that the head pose has changed significantly.

3.2 Robust Driver Gaze Detection

We design a novel framework that combines both head pose and eye information to derive a driver's gaze. Figure 3 shows an overview of driver gaze detection. The input to the neural network is an eye-ROI (Region of Interest) image and the head-pose angles – yaw and pitch. The network combines these and outputs a vector with the overall yaw and pitch of the driver's gaze direction, as elaborated on below.

Head pose computation: Obtaining head pose angles from landmarks is a well-studied problem [17]. We use the PnP (Perspective n Point) and Random sample consensus (RANSAC) algorithms [56] to obtain the head pose, specifically yaw and pitch angles with respect to the camera.

Eye-ROI extraction: Previous efforts, centered on identifying iris and eye corners to determine the gaze direction [24, 33], are prone to errors due to adverse lighting conditions and inadequate (smartphone) camera resolution. To avoid these difficulties, ALT just extracts an eye-ROI (region of interest) that bounds the eyes of the driver. Such an eye-ROI can be extracted more robustly than specific landmarks in the eye region. The eye-ROI and the head pose are both fed as inputs to a convolutional neural network (CNN).

Combining headpose and eye-ROI: We use a neural network that takes as inputs the yaw and pitch angles estimated based on the head pose and the eye-ROI, and outputs the combined yaw and pitch angles. Our network is based on LeNet [37], which is a shallow CNN and can be ported onto a smartphone with a minimal computation overhead (see Section 8.3). We make two modifications to the neural network:

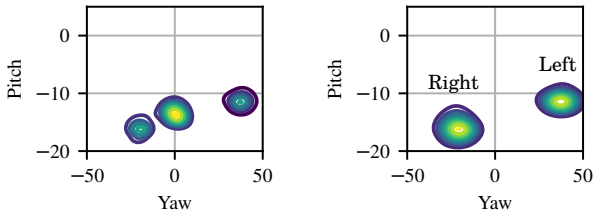
First, we use a synthetically generated dataset, Unity Eyes[51], to train the network since, to our knowledge, there are no large-scale datasets available with real images due to the difficulty in annotating gaze direction. To aid the model in generalizing, the training data is augmented using random scaling and rotation along with a random shift in brightness, gamma, and hue values, to account for variation in skin tone and illumination. We use 60,000 synthetically generated images, split into training (90%), validation (5%), and test (5%).

Second, we add a spatial-transformer (STN) module [31], which corrects misaligned eye-ROIs originating from occasional incorrect landmark localization. For example, if the eye-ROI image is slightly displaced or rotated, STN learns a suitable affine transformation to "fix" the eye-ROI.

3.3 Auto-Calibration for Mirror Scanning

Mirror scanning helps drivers maintain situational awareness of their surroundings. The view obtained by the camera in respect of the driver's mirror scanning behaviour could vary depending on the mounting and orientation of the camera, the seating position of the driver, and the geometry of the vehicle. For example, a driver who sits close to the steering column would tend to turn their head more to look at the mirrors than one who sits further back. While previous work has used fixed yaw thresholds [54] to determine when the driver scans the left/right mirrors, in ALT, we develop a novel *auto-calibration* approach, which is employed at the beginning of a drive and *helps calibrate for the particular setting automatically, without any human input or any explicit per-driver model training.*

In the auto-calibration phase, ALT analyzes the gaze distribution during the initial part of the drive, to identify the dominant directions of the driver's gaze. Although the driver's gaze is not restricted in an uncontrolled setting, there are three frequently recurring states: (1) *Left*-mirror scan, (2) *Right*-mirror scan, and (3) *Straight* gaze (driver focusing on the road straight ahead). This is evident in Figure 4(a), which shows the gaze distribution for the frames in the auto-calibration phase (the yaw and pitch angles are the combined ones from Section 3.2). There are three high density



(a) Initial gaze distribution. **(b) Frames in left/right state.**

Figure 4: Gaze distribution. In our right-hand drive vehicles, the driver sits much closer to the right mirror, so scanning the right mirror involves a smaller yaw magnitude (looking to the right) but a larger pitch magnitude (looking down) compared to scanning the left mirror.

regions corresponding to the three aforementioned recurring states. Empirically, the high-density region closest to the mean yaw value corresponds to "Straight". To delineate "Left" and "Right" more clearly, we remove the "Straight" region by removing the frames with $|y(t) - \mu_y| \leq \sigma_y$, where μ_y is the mean yaw and σ_y the standard deviation. We then fit two Gaussian kernels to the remaining two distributions, as shown in Figure 4(b). These probability distributions corresponding to the 3 states (left, right and straight), allow us to perform classification. We use Naïve Bayes algorithm to identify the three clusters using the above probability distributions. Thus, *auto-calibration enables* ALT *to track driver's mirror scanning behavior despite variations.*

4 ACCURATE TRAJECTORY TRACKING

We now present how ALT derives accurate vehicle trajectory using just a smartphone camera, focusing on our novel contributions of augmenting visual SLAM techniques with sparse fiducial markers (Section 4.2) and placing fiducial markers to maximize the trajectory accuracy (Section 4.3).

4.1 Background on Visual SLAM

Visual SLAM (Simultaneous Localization and Mapping) comprises two parts: (i) mapping: building a map of the environment using visual feature detection and matching across successive frames, and (ii) localization: using the map generated to determine the camera's position and orientation. The camera's pose information is then used to derive the trajectory.

An alternative to visual SLAM is to use of built-in inertial sensors in the smartphone to derive vehicle trajectory. Previous research efforts have utilized smartphone inertial sensors to identify various driving events such as lane changes, left/right turns, and speeding [20, 50]. Such use of inertial sensors from the smartphone is valid only for short maneuvers, as inertial sensors suffer from integration drift over longer time scales and poor sampling frequency (typically <50Hz sampling) leading to noisy signals. Further, inertial sensor based trajectory estimation has error in the range of 90cm [36], which is large compared to the requirement of 15-20cm in ALT. In Section 9, we present opportunities for fusing inertial data with visual SLAM to enable accurate trajectory estimation.

Since the majority of the smartphone cameras are monocular, i.e., a camera with a single sensor, we restrict our focus to monocular SLAM techniques. While recent smartphones have multiple rear cameras, the short baseline (a separation of just a few centimeters between the cameras) severely limits the accuracy of distance, and hence trajectory, estimation.

(a) ORB Features (b) Fiducial Markers Setup (c) Planar fiducial marker

Figure 5: (a) visual features obtained using ORB, (b) fiducial markers in an outdoor setup, and (c) fiducial marker.

4.1.1 ***Visual SLAM with features.*** A popular, state-of-the-art monocular visual feature-based SLAM technique is ORB SLAM2 [45]. Figure 5(a) shows the ORB features in an image used to derive camera pose. While deriving accurate trajectory using monocular SLAM is an active area of research in robotics [16, 53], this approach generally suffers from two problems:

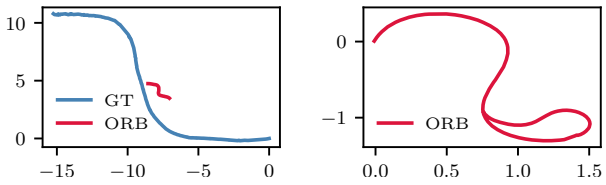
(1) **Fragility of visual features:** Visual feature detection and matching is often fragile, especially when there is a significant scene change (quite common in outdoor environments) and/or varying illumination, leading to poor trajectory estimation.
(2) **Absence of scale information:** Monocular SLAM techniques do not have depth information during localization and mapping. Hence, the trajectory derived is on an arbitrary scale, unconnected with the true scale in the real world [9, 15]. Further, due to error in correspondence points, the estimated scale could change over time ("scale drift"), resulting in the trajectory length being either expanded or contracted over time.

4.1.2 ***Fiducial marker based SLAM:.*** To avoid the problems of using the visual features in a scene, an alternative approach for mapping and localization is placing fiducial markers of a known size in the environment [39]. Recent works have used square planar fiducial markers such as ArUco [27] or April Tags [48] for camera pose estimation and localization. Each marker comprises an external black border and an inner binary matrix used to uniquely identify the marker (e.g., Figure 5(c)). The main advantage of using such fiducial markers is that each marker provides four correspondence points (its four corners), which can be detected robustly even in varying illumination and are sufficient for estimating camera pose [27]. Furthermore, the known size of the markers (e.g., 26.7 cm on a side when printed on an A3 sheet) helps establish scale. However, this approach suffers from the following drawbacks:

(1) **Need for a priori map creation:** To derive camera pose and hence trajectory using fiducial markers, we have to first create a map of the actual position and orientation of each marker in the environment. Further, this process has to be repeated whenever there is a change in marker placement, which makes it infeasible for real deployments.
(2) **Infrastructure support:** Fiducial markers need to be placed extensively in the environment, such that at least one but ideally more are within the camera's view at all times. The redundancy helps combat the ambiguity in pose estimation when there is only one planar marker in the scene [27] (see Figure 5(b)).

4.2 Hybrid Visual SLAM in ALT

Can we get the accuracy of SLAM using fiducial markers while avoiding, or at least minimizing, the need for an extensive deployment or calibration of such markers?



(a) Scale issue (b) Loop closure

Figure 6: Scale and loop closure issues in ORB SLAM (x-axis and y-axis in meters).

To this end, we present a novel hybrid visual SLAM technique that augments visual features from the scene with a sparse set of fiducial markers placed optimally in the environment. The idea is to place fiducial markers to act as "bridges" between frames at just the places where there is are insufficient matches of visual features across frames, say because there is a sharp curve in the trajectory that causes a significant shift in the pose of the vehicle-mounted smartphone camera. Since these markers are used just to serve as bridges, there is no need to map their locations a priori.

ALT's hybrid approach extends ORB SLAM [45] by integrating fiducial markers in the environment. We will briefly describe how ORB SLAM works and then present our hybrid approach in detail.

4.2.1 ***ORB SLAM Overview.*** ORB SLAM uses visual features to localize and map the environment. We now enumerate the key steps involved; more details can be found in [45]:

1. Feature extraction. For every frame in the video, the system begins by extracting ORB feature points and descriptors. These feature points are usually points of local extrema, such as corner points, edges, etc., which are more likely to be detected in successive frames of the same scene.

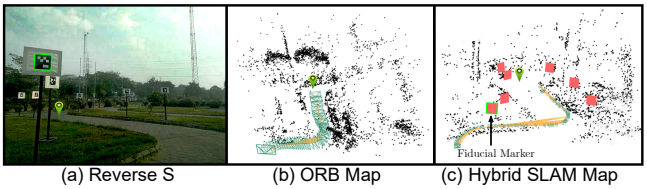
2. Map Initialization. Map initialization computes the relative camera pose between two frames to triangulate an initial set of map points. Map points are ORB feature points that are matched in successive frames. A good map initialization requires two frames with displacement along with good feature correspondences between the frames.

3. Pose Estimation. The ORB features from the previous frame are matched with the ORB features in the current frame to get correspondences. The correspondences are then projected onto a local map, which contains a set of keyframes K_1 that share correspondences with the current frame and a set of keyframes K_2 with neighbors to the keyframes K_1. Finally, the decision to add a frame as a keyframe in the map is based on the extent of the scene change.

4. Pose Optimization. As we keep updating the map, the camera pose is optimized using the current keyframe and all the keyframes connected to it based on correspondences. The algorithm takes as input, poses from all keyframes and map points to obtain optimized poses and 3D points by performing non-linear least squares optimization [40].

5. Covisibility graph. The map details is stored as a covisibility graph – an undirected, weighted graph, where each node represents a keyframe and edges between keyframes are created only if they share a minimum number of correspondences. Further, the weight of the edge represents the number of correspondence points.

4.2.2 ***Integration of fiducial markers into ORB SLAM.*** We use April tags [48] as our planar fiducial markers. These enable robust detection and derivation of scale information. We now discuss how these fiducial markers are leveraged to improve trajectories obtained from SLAM.



(a) Reverse S (b) ORB Map (c) Hybrid SLAM Map

Figure 7: Map and trajectory information (in yellow) using ORB SLAM and hybrid SLAM with fiducial markers.

1. Scale recovery using markers. Figure 6(a) shows the ORB SLAM trajectory obtained (without scale) for the reverse S maneuver (in red) compared to the ground truth trajectory (in blue). The mismatch is because the visual features used by ORB SLAM do not provide any sense of scale. To address this, in our hybrid approach, we leverage the known fixed size of fiducial markers (e.g., S_{marker} =26.7cm, when printed on an A3 sheet) to derive the scale factor. The idea, in a nutshell, is that when comparing two different camera poses (e.g., the smartphone camera at two different points along the driving track, in the context of ALT), the view of fiducial markers of known size allows the translation between the poses (which defines the initial baseline) to be estimated in terms of real-world scale. We have extended ORB SLAM to detect a fiducial marker in a scene. The 2D coordinates of the marker corners are then translated to the 3D map of ORB SLAM. Further, when we integrate the marker into the map, we ensure the marker remains planar and its size is translated based on the real-world scale. We then trigger a global adjustment to optimize the map points. This results in an initial map that is at the correct scale. We defer a more detailed discussion to Appendix - 1.

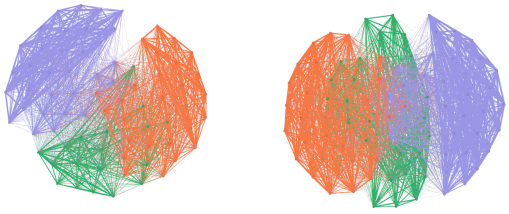
2. Accurate pose estimation. For every new frame, we derive pose information using both points extracted by ORB SLAM and markers in the scene. Previously, ORB SLAM pose optimization used only ORB map points to optimize the pose. We extend the optimizer to also include fiducial marker map points for pose optimization. The inclusion of markers in the pose optimization helps improve the accuracy, as described in Section 7. We defer a more detailed discussion to Appendix - 1.

Figure 7(a) depicts the setting of the reverse S maneuver. Figure 7(b) shows the map and trajectory derived using ORB SLAM. We can clearly see that the trajectory derived is erroneous both in shape and scale. Figure 7(c) shows the map and trajectory derived using hybrid SLAM. We can see that the markers in the real-world are correctly integrated into the map and further, the trajectory derived is accurate in both shape and scale.

4.3 Optimal Marker Placement

As described in the previous section, our hybrid approach, which combines visual SLAM with (a few) fiducial markers, is useful in both reducing the error in pose information and in recovering scale reliably. A key question, however, is how markers should be placed in the environment to maximize the benefit derived in terms of trajectory accuracy while minimizing the number of markers.

To address this, we present a generic approach to optimally place fiducial markers in the environment using the covisibility graph defined by ORB SLAM (as described in Section 4.2.1). A



(a) Without markers (b) With 6 markers

Figure 8: Covisibility graph- Reverse S with(out) markers.

significant scene change between two keyframes will result in a weak (or no) edge between the corresponding nodes. To strengthen the overall connectivity of the graph, we identify loosely connected "communities" and their corresponding connections. We then place markers at the peripheral points corresponding to the community boundaries, which helps increase the connectivity of the covisibility graph. In more detail:

(i) Determine the number of communities: We partition the covisibility graph G into C communities based on the modularity measure [19]. Modularity measures the strength of communities. Communities with high modularity have dense connections between the nodes within the community but sparse connections with nodes in a different community.

(ii) Determine peripheral nodes of a community: In order to determine the set of peripheral nodes in a community, we define a metric based on the weighted out/in degree of nodes. The intuition is that nodes with high weighted out-degree (to other communities) compared to weighted in-degree (to nodes in the same community) are peripheral nodes of a community. So, for each node i, we compute m_i:

$$m_i = \frac{(\frac{w_i^{out}}{k_i^{out}})}{(\frac{w_i^{in}}{k_i^{in}})} = \frac{w_i^{out}}{w_i^{in}} \frac{k_i^{in}}{k_i^{out}}, \quad (1)$$

where w_i^{out} (or w_i^{in}) is the sum of weights of edges from node i to all other nodes in different communities (or same community) and k_i^{out} (or k_i^{in}) is the degree of node i to all the nodes in different communities (or same community). The number of correspondences between the keyframes indicates the weight of the edge. Intuitively, m_i would be large for the peripheral nodes of a community; we designate the nodes with $m_i \geq \lambda$ as the set of peripheral nodes, where λ is a threshold.

(iii) Determine the placement for fiducial marker: Finally, we determine the node (or keyframe) that has the highest weighted out/in degree for each community. This keyframe will have the lowest feature correspondence with all other keyframes from the same community, making it likely that the computed pose at this keyframe would be inaccurate. The selected keyframe for each community represents the location in the environment to place the fiducial markers such that the correspondence between communities increases the most. This leads to overall improvement in graph connectivity and hence pose estimation accuracy, while maintaining the sparsity of marker placement. Figure 8 shows the covisibility graph for the reverse S maneuver (a) without any markers, and (b) with six markers, placed based on the proposed approach. We can clearly see the increase in graph connectivity, leading to better correspondences across frames (see Section 5 for results).

4.4 Performance Enhancement using Inertial and GPS Data

The current visual SLAM techniques analyze all the frames in a video to identify a small number of keyframes, which are the only frames used for mapping and localization.

In order to run our system on smartphones efficiently, we present a technique to discard non-keyframes without expensive computing. Intuitively, it is the frames with a significant change in the scene relative to the previous frame, due to a large displacement or rotation of the camera, that are likely to be keyframes. So, we use inertial and GPS measurements to filter out frames that correspond to small displacements and so are unlikely to be keyframes.

Latitude and longitude readings from a GPS receiver can be used to directly estimate displacement of the vehicle. However, the GPS modules in mobiles are not accurate and their sampling frequency is low. An accelerometer provides measurements at a higher frequency but this comes with higher noise. We draw on the advantages of inertial sensors and GPS, using a Kalman Filter [20] to combine the two measurement streams, to obtain a reliable estimate of position and velocity of the vehicle. We formulate the state of the vehicle as: $s_t = [x_t, y_t, \dot{x}_t, \dot{y}_t, \ddot{x}_t, \ddot{y}_t]^T$ Here, x_t and y_t are the position coordinates in the world reference frame (North and East axes), and $\dot{x}_t, \dot{y}_t$, and $\ddot{x}_t, \ddot{y}_t$ denote velocity and acceleration, respectively, in the same reference frame. We use the orientation matrix obtained using the Android SDK, to transform the accelerometer readings from local to world reference frame.

At each time step, the state is predicted with: $\boldsymbol{s}_t = A\boldsymbol{s}_{t-1}$ and subsequently updated using $s_t = \boldsymbol{s}_t + K \cdot (z_t - H\boldsymbol{s}_t)$, where A is the state transition matrix, H is the observation matrix, K is the Kalman gain, z_t is a vector of position coordinates from GPS and acceleration readings from the accelerometer. Thus, a frame is discarded only if the corresponding displacement of the camera, and hence of the vehicle it is mounted on, is less than d—the mean displacement between two consecutive measurements.

5 EVALUATION SETUP

In this section, we describe our deployment setup and data collection procedure to evaluate our two core contributions —driver gaze detection and vehicle trajectory tracking, both of which are key to automating driver license testing.

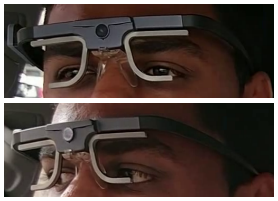
5.1 Deployment Setup

We perform extensive experiments to evaluate ALT in both controlled settings and on driving test tracks.

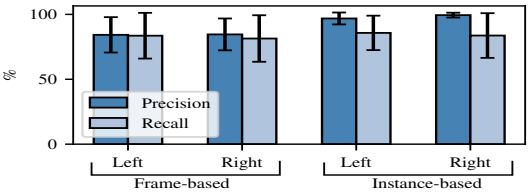
Driving test track: We deploy ALT in multiple large driver training institutes, which trains several thousand drivers each year. The institute conducts driver training and testing with a similar range of maneuvers and requirements as required by the license issuing authority. We use this setup to evaluate ALT's ability to automate the driver's license test. Figure 1b depicts the driving track along with the maneuvers being tested.

Controlled real-world experiments: In addition to an actual driving test track, we also evaluate ALT's ability in controlled outdoor settings. In such controlled settings, we can add instrumentation (e.g., an eye tracker worn by the driver) to collect accurate ground truth. This allows us to test the efficacy of our gaze detector and trajectory estimator extensively.

(a) Ground truth using Tobii.



(b) Occlusion issue with Tobii.



(c) Precision and Recall for mirror scan evaluation.

Figure 9: Mirror scan evaluation in ALT.

5.2 Data Collection

We use a windshield-mounted Android smartphone (Lenovo ZUK Z2 model running Android 7) for data acquisition. We record multiple sensor streams, including images captured simultaneously from the front (driver-facing) and rear (road-facing) cameras, motion data captured from inertial sensors, and location data captured from a GPS sensor. The videos are captured at 1080p and 30 fps, the data from inertial sensors (accelerometer, gyroscope) is sampled at 20 Hz and the data from GPS is sampled at 1 Hz. We collect data from the driving test track for the maneuvers described in Section 2.2 — reverse S, reverse parking, 8 maneuver, incline start, and lane change/turns. We collect data (camera and sensor) for 20 instances of each maneuver for each of 10 drivers.

6 EVALUATING MIRROR SCANNING

We present ALT's accuracy in detecting left and right mirror scans, both in controlled outdoor experiments, where we obtain the ground truth by having the driver on a wearable sensor for accurate gaze tracking, and also on an actual driving test track, where ground truth was obtained by annotating the recorded videos manually.

We use Precision (P), Recall (R), and F_1 score to evaluate the accuracy of mirror scanning in ALT. The F_1 score is defined as the harmonic mean of precision and recall: $F_1 = 2 \cdot \frac{P \cdot R}{P+R}$. Mirror scanning evaluation is performed at two levels:

Frame-level: How many individual frames are correctly classified as left or right mirror scans?

Instance-level: How many instances of left or right mirror scans are correctly identified? An instance is a temporal window of 10 frames, which we label according to the class that the majority of the frames correspond to.

6.1 Controlled Setting with Tobii Eye Tracker

We are not aware of any public datasets with annotated mirror scanning events, so we use a wearable eye tracker to accurately annotate the driver's gaze in controlled outdoor settings (with the driver driving on a pre-determined track). Specifically, we use the Tobii Pro Glasses 2 [6], which includes four eye cameras, a wide-angle HD camera, and an inertial sensor. Tobii's proprietary 3D eye model combines data from these sensors to provide accurate gaze information. The gaze sampling frequency of the tracker is 100 Hz and the data is stored on an onboard SD card. The images in Figure 9a show a driver wearing the Tobii eye tracker (captured using the ALT windshield-mounted smartphone camera). The inset images show the front view from the Tobii, with the red dot marking Tobii's estimate of where exactly the driver is looking.

The high cost of the Tobii device (over USD 15,000) means that we only perform limited experiments with a loaner device. We collect over an hour of actual driving data using the Tobii tracker, along with our windshield-mounted smartphone setup. Using the software provided by Tobii, we define Areas of Interest (AoIs) corresponding to the left and right mirrors in the vehicle, thereby obtaining a filtered subset of frames corresponding to just when the driver was looking at the left or the right mirror. The ground truth data thus obtained includes 62 and 58 instances respectively, of left and right mirror scans. ALT identifies these left/right mirror scan instances with a precision of 85% and recall of 82% (F_1 score: 83%). Most of the misclassifications can be attributed to the intrusive nature of the eye tracker, as in several frames it occludes the eyes of the driver from the view of the windshield-mounted smartphone (as shown in Figure 9b).

Expt		Fixed-Yaw threshold				ALT			
		Left		Right		Left		Right	
		P	R	P	R	P	R	P	R
Exp1	D1	0.93	0.65	1	0.08	1	1	1	0.69
	D2	1	0.2	nan	0	1	0.95	1	1
Exp2	C1	1	0.65	1	0.08	1	1	1	0.69
	C2	1	0.31	nan	0	1	1	1	0.89
Exp3	O1	1	0.46	0.4	0.15	0.93	1	1	1
	O2	1	0.85	0.75	0.16	1	0.9	0.91	0.68

Table 1: Precision (P), Recall (R) for left/right mirror scan instances based on fixed thresholds vs. ALT, with different drivers (D1 & D2), cars (C1 (SUV) & C2 (hatchback)), cam. orientation (O1 & O2).

6.1.1 ***Ground truth with manual annotations.*** To overcome the occlusion problem, we manually annotate a subset of videos from our dataset corresponding to around 120,000 frames (over an hour of video) across drivers. The annotations include 220 and 276 instances, respectively, of left and right mirror scans. Figure 9c shows the Precision and Recall values for identifying both left/right frames and instances. Note that ALT identifies individual left/right frames with a precision of nearly 84% and recall of 82%. Furthermore, it has a precision of over 98% and recall of over 85% in identifying left/right mirror scanning *instances* (F_1 score: 91%).

Summary: *ALT has an F_1 score of 91% for identifying left/right mirror scans at frame and instance levels, respectively.*

6.2 ALT's Auto-calibration vs Fixed thresholds

In Section. 3.2 we argued that to track mirror scanning, auto-configuration is needed to ensure robustness to variations in camera orientation, driver seating, and vehicle geometry.

We perform multiple experiments in the context of mirror scanning detection to show the efficacy of ALT's auto-calibration compared to using fixed yaw angle thresholds (+/-15 deg, as in [54]). In each experiment (lasting for about an hour), we vary one of the driver seating position, the vehicle type (which typically means that the geometry of the mirrors relative to the driver also changes), or the camera orientation (which could arise simply because of variation in how the smartphone is mounted).

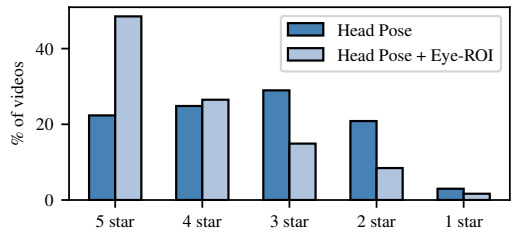


Figure 10: Ratings from driving instructors for video snippets picked out by ALT from driver training sessions.

Exp1: Two drivers, a short (D1) and a tall (D2) driver, with different seating positions, in the same type of car.

Exp2: Same driver, different cars (SUV vs hatchback).

Exp3: Same driver, same car but with slightly different camera orientations.

Table 1 summarizes the results. In general, ALT has higher precision and recall than an approach based on just head pose with fixed yaw thresholds. It is instructive to also delve into the numbers for the fixed yaw threshold case.

In **Exp1**, Driver 1 (short and hence seated more forward) tended to turn his head more to scan the left mirror compared to driver 2 (tall), which accounts for the higher recall (0.65 vs 0.2) but at the expense of precision (since the fixed threshold is likely to be exceeded more often in the case of driver 1).

In **Exp2**, the driver in an SUV tended to turn his head more to scan his mirrors compared to the same driver driving a hatchback, accounting for higher recall (0.65 vs 0.31). Since these were right-hand drive vehicles, looking at the left mirror involved turning the head to a greater extent than looking at the right mirror, so the recall tends to be higher for the left mirror.

In contrast to the fixed threshold based approach, the gaze distribution in the initial part of the drive enables auto-calibration in ALT to identify clusters corresponding to left and right mirrors scans *in the particular setting* of driver seating, vehicle geometry, and camera orientation. This contributes to the robustness of mirror scanning detection in ALT.

Summary: *ALT's auto-calibration has higher precision and recall than an approach based on just head pose with fixed yaw thresholds. This is so across drivers, vehicles, and camera orientations.*

6.3 Mirror scanning evaluation on testing track

We deployed ALT in 10 vehicles to monitor trainee drivers [1], with the specific objective of evaluating their mirror scanning behavior. The trainee drives around the test track shown in Figure 1b. ALT generates an event whenever the trainee did not scan either of the mirrors for at least 8 seconds and records a short video clip documenting the episode. At the conclusion of a training session, these clips were rated by an instructor based on whether these contained an event of interest.

ALT was deployed for a total of 200 sessions, each lasting about an hour, across 140 trainees and 12 instructors over a month. It selected about 2200 video clips with events of interest, out of which 1200 were rated by the instructors to indicate the appropriateness of the clip: from 1-star (least) to 5-star (most). A 1-star rating is provided when the event generated is a false positive or otherwise not interesting, and a 5-star rating is provided when ALT is able to correctly identify an instance of the trainee not scanning the mirrors, which the instructor had also missed during the actual test. We ran two variants of the mirror scanning detection algorithm, one (i) with only head pose, and the other (ii) with both head pose and eye gaze. Figure 10 shows the percentage of clips rated from 5-star to 1-star. When only head pose information is considered, 47% of the clips receive a 4-star or 5-star rating, whereas this number goes up to 75% when both head pose and eye gaze information is used. This shows the importance of combining both head pose and eye gaze in ALT to detect mirror scanning robustly.

Summary: *The evaluation with trainees in a driver training school shows the efficacy of* ALT *even in the hands of an external party who are unconnected with us, the researchers.*

[1] Our project has been vetted and approved by our IRB

Markers	Method	0-5 m	5+ m
11 (dense)	Our method (Hybrid SLAM)	0.04	0.05
	Prior work [44]	0.04	0.1
5 (sparse)	Our method (Hybrid SLAM)	0.05	0.05
	Prior work [44]	0.09	0.13

Table 2: Average error (in metres) in camera localization at various distances using fiducial markers and hybrid SLAM.

7 ACCURACY OF HYBRID SLAM

In this section, we evaluate the accuracy of trajectory tracking in ALT using our hybrid SLAM approach. We present results from both a controlled environment and a driving test track.

7.1 Controlled Experiments

We place 11 markers (April tags) in an unstructured configuration and capture a few images from different viewpoints, to create an initial map. Subsequently, we move the camera along a straight line and capture images every meter. The camera positions estimated from these images are then compared with the ground truth. The configuration of the markers along with the trajectory of the camera is shown in Figure 7.

To evaluate our method, we capture a video by moving the camera on the same line while pausing for a few seconds at every meter mark to enable the ground truth to be recorded. From the trajectory of the camera estimated using hybrid SLAM, we extract the estimated position of the camera whenever it was static. We compare these estimated positions with ground truth, to estimate the error in camera localization.

We compare our method against a state-of-the-art, fiducial markers based camera localization method by Munoz-Salinas *et al.* [44]. Their method comprises of two stages: mapping and localization. Images from various viewpoints, with at least two markers visible in every image, are captured to create an initial map. The map so obtained is subsequently used for localizing the camera in the scene through pose estimation.

As shown in Table 2, the average error in camera localization with hybrid SLAM is similar to the error obtained using the state-of-the-art method from [44]. However, mapping and localization are decoupled in [44], whereas our approach achieves them simultaneously without requiring any offline map creation.

We also evaluate the accuracy of both the approaches, with fewer markers placed in the scene. As shown in Table 2, the accuracy

Figure 11: Fiducial marker locations for 8, Reverse S, incline start and reverse parallel parking. Colors represent clusters in covisibility graph.

of [44] degrades as the number of markers in the scene decreases. This is evident from the increase in overall error, both with fewer markers in the scene and also with distance to the markers (i.e., error increases with fewer markers being visible at larger distances). In contrast, with our hybrid SLAM approach, the camera localization error with 5 markers is almost as good as it is with 11 markers, underscoring the effectiveness of hybrid SLAM even with a sparse deployment of fiducial markers.

Summary: *Hybrid SLAM performs comparably with a state-of-the-art localization approach with a dense set of markers, and outperforms it when a sparse set of markers is used.*

7.2 Experiments on Driving Test Track

To evaluate the performance of hybrid SLAM on the test track, we need to place a small number of fiducial markers in strategic locations. We first describe our method for placing these markers and then turn to the accuracy of trajectory estimation.

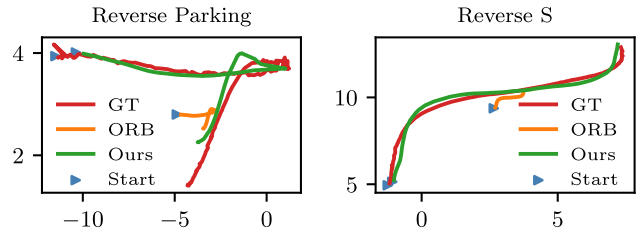
7.2.1 ***Fiducial Marker Placement.*** Recall from Section 4.3 that our hybrid SLAM approach uses the covisibility graph to identify the locations where visual SLAM by itself would suffer from errors (e.g., places where there is a drastic scene change, say due to a curve in the trajectory) and hence the introduction of markers would be most beneficial.

Figure 11 shows the marker locations identified by our approach for the reverse S and 8 maneuvers. The coloured lines, overlaid on the track, show the vehicle trajectories obtained using our approach. Each colored line segment corresponds to a community in the covisibility graph (see Section 4.3), with reverse S having 3 communities and the 8 maneuver having 6 communities. The boundaries between these communities are the places where the scene changes significantly and where we place markers, as shown in Figure 11.

By adding markers in these locations, we help increase the connectivity among nodes, i.e., keyframes in the neighbouring communities in the covisibility graph. We quantify this increase in connectivity in Table 3 using two metrics: 1) Density ($\mathcal{D}$), and 2) Average Weighted Degree ($\mathcal{W}$). $\mathcal{D}$ represents the average number of edges per node in the graph and $\mathcal{W}$ represents the average edge weight (see Section. 4.3). Increase in both the metrics corresponds to increase in feature correspondence between nodes. Table 3 shows a significant increase in both density ($\mathcal{D}$) and average weighted degree ($\mathcal{W}$) from the introduction of markers, across each of 3 maneuvers.

Maneuver	Reverse S	8 maneuver	Reverse parking
% increase in $\mathcal{D}$	7.1	6.2	30.5
% increase in $\mathcal{W}$	8.8	33.8	23.2

Table 3: % increase in density ($\mathcal{D}$) and average weighted degree ($\mathcal{W}$) of the graph for three maneuvers.



(a) **(b)**

Figure 12: Comparison of ground truth (GT) trajectory with Hybrid SLAM (Ours) and ORB-SLAM, where x-axis and y-axis is in meters.

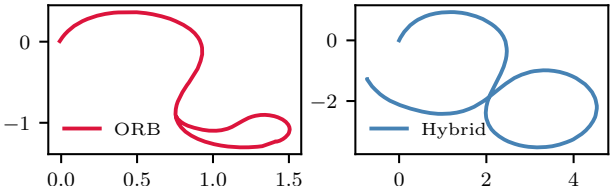


Figure 13: Trajectory for 8 maneuver using ORB-SLAM & hybrid. For illustration, loop closure in hybrid SLAM is used for right loop only.

7.2.2 ***Deriving accurate trajectory.*** We evaluate the accuracy of the trajectory derived using our hybrid approach and with ORB-SLAM [45], which is a pure visual SLAM based approach. Note that unlike with the controlled experiments in Section 7.1, we are unable to compare with [44], since the number of markers needed for this approach would be prohibitive.

Hybrid SLAM improves trajectory estimation over ORB-SLAM by performing pose adjustment, scale recovery, and loop closure using the fiducial markers (Section 4.2). We obtain the ground truth using the pole-mounted static cameras placed along the track.

Figure 12 shows the ground truth (GT) trajectory from the pole-mounted cameras, and the trajectories obtained from ORB-SLAM and our hybrid SLAM approach, for the reverse S and reverse parallel parking maneuvers. The trajectory estimated by hybrid SLAM matches the ground truth closely, due to scale recovery and pose adjustment using fiducial markers (Figure 12(a)). To measure the accuracy of the trajectory, we computer the root mean square error (RMSE) in meters of each estimated trajectory relative to the ground truth. For the reverse S maneuver, the RMSE between our hybrid SLAM and GT is 0.15 m, or 15 cm. However, as mentioned earlier, ORB-SLAM derives trajectory with an arbitrary scale, which leads to a poor trajectory estimate. It is non-trivial to derive and fix the scale of ORB SLAM without fine-tuning other components of the SLAM technique. Hence, we compute RMSE with respect to ground truth only. Furthermore, Figure 12(b) show the trajectory obtained for reverse parallel parking using ORB-SLAM and our approach. The RMSE between GT and our hybrid SLAM is 0.25 m.

Figure 13 shows the trajectory for 8-maneuver. It can be seen that hybrid localization was able to derive the trajectory with accurate

# of landmarks	Mean Yaw error	Mean Pitch error	Detection time (ms)	Tracking time (ms)
68	2.7^o	3.3^o	89	25
22	5.3^o	8.2^o	58	10

Table 4: Impact of 68 vs. 22 landmark configurations

shape and size (even in presence of loops). However, ORB-SLAM was not able to detect the loop and suffers from scale drift, resulting in inaccurate pose, and hence trajectory, estimation.

Summary: *Hybrid SLAM's RMSE for trajectory estimation on an actual driving track is about 20 cm, which is quite small relative to the dimensions (length, width) of both the vehicle and the track. This level of accuracy suffices for automating license testing.*

8 MOBILE IMPLEMENTATION

The key objectives of our mobile implementation are efficiency and effective operation on affordable smartphones with modest specifications. Therefore, we benchmark the performance of ALT running on low-to-mid end Android phones such as the Lenovo Zuk2 (USD 155) and OnePlus 3 (USD 350), both based on the Qualcomm Snapdragon 820 with Quad-core CPU and Adreno 530 GPU. ALT has two main components, (i) driver gaze monitoring, and (ii) fine-grained trajectory estimation using hybrid SLAM.

8.1 Landmarks reduction and tracking

To improve computational efficiency, we pare down the number of landmarks from 68 to 22 (see Figure 2(b) and Section 3.1) to derive head pose information. Table 4 shows the error in yaw and pitch angles with the 68 and 22 landmark configurations along with the time taken on a smartphone CPU to detect landmarks. Selecting the chosen subset of 22 landmarks out of 68, yields a 35% speedup (58ms vs 89ms) with only a marginal increase in error. For instance, while the yaw error goes up from 2.7^o to 5.3^o, this error is small relative to the -30^o to $+20^o$ yaw range for a left/right mirror scan.

Even with the speedup, the detection time of 58 ms per frame is substantial. To further reduce the computational cost, we *track* landmarks across consecutive frames instead of *detecting* these afresh in each frame. Tracking 22 landmarks across successive frames only takes 10 ms on a smartphone CPU, as compared to 58 ms for detection, yielding an 80% speedup (see Table 4). Therefore, the system persists with tracking unless a face is not detected or the landmarks in successive frames do not overlap much.

8.2 Leveraging GPU for Gaze Tracking

Certain operations in ALT are based on DNNs, hence we seek to run these efficiently on a smartphone. Specifically, in ALT a LeNet model (see Figure 3) is used to for gaze detection. We implemented this in TensorFlow and saved the model as a protocol buffer (PB) file. Currently, the TensorFlow models saved as PB files can run only on Nvidia GPUs. However, most of the low-to-mid end smartphones, only have the Qualcomm Adreno GPUs. While there has been recent work on custom accelerators to enable DNN execution on mobile GPUs (e.g., RSTensorFlow [14]), this is applicable only to a small set of GPUs, typically *not* the low-end GPUs. We seek a cleaner approach to leverage mobile GPUs, in particular, the commonly-used Qualcomm Adreno GPUs. To this end, we employ Qualcomm's Snapdragon Neural Processing Engine (SNPE) [12], which is a unified SDK for running neural network models on all Snapdragon platforms. SNPE takes the PB file and converts it into a Qualcomm-specific DLC file (Deep Learning Container). The DLC file can be used to run on either CPU or GPU of the smartphone. The LeNet model on the phone is 4.3 MB in size and in the next section we report the time taken to run it on the smartphone.

8.3 Benchmarking ALT on Android

We benchmark the two components of ALT running on phones:

Driver gaze monitoring: Gaze estimation takes 45 ms per frame on the CPU but only 16 ms (a speedup of over 2.5X), on the low-end Adreno 530 GPU on the smartphone.

Hybrid SLAM: The SLAM system takes 80 ms per frame to detect features and localize and runs only on the CPU. Finally, ALT supports 8-10 fps to run both driver gaze and hybrid SLAM together on CPU and GPU of the smartphone.

8.4 Performance Enhancement of Hybrid SLAM by fusing Inertial and GPS data

In this work, to improve the speed of hybrid SLAM, we discard consecutive frames with little or no displacement. This allows us to only process the relevant frames, without trading off accuracy.

By fusing the inertial and GPS data, as described in Section 4.4, we compute the reduction in number of frames that need to be processed. The reduction for reverse S, 8, incline and reverse parallel parking is 34.4%, 31.1%, 28.3% and 29.4%, respectively. Since around 30% of the frames can be discarded across various maneuvers, there is a corresponding speedup in the execution of hybrid SLAM. Furthermore, the error (RMSE), relative to the ground truth, of the trajectory obtained using all frames versus fewer frames, is similar. Thus, discarding frames with little displacement has minimal affect on the accuracy of the estimated trajectory.

Summary: ALT *supports 8-10 fps to run both driver gaze detection and vehicle trajectory estimation on a smartphone.*

9 DISCUSSION

ALT is currently deployed for driver license testing at a site in India. Based on an extensive evaluation, we find that the results based on ALT match with the manually-scored results in around 95% of the cases. In many cases, ALT was able to catch aspects of the test overlooked by the inspector during manual evaluation, e.g., the driver scanning their mirrors before performing a lane change, the number of forward and backward movements during a reversing maneuver, etc. While ALT is a promising first step towards comprehensively automating license testing, we now discuss some of its limitations and opportunities for improvement.

(i) Generalization to other tracks: Based on extensive conversations with experts in license testing, we believe the core components of ALT (driver's gaze, trajectory estimation) would be applicable to driver testing and tracks in general.

(ii) On-the-road testing: While the portable, smartphone-based design of ALT lends itself to going beyond confined tracks and testing on actual roads, our current trajectory estimation still relies on having a sparse set of markers in the environment. We believe that this requirement can be eliminated by leveraging recent advancements in low-cost Lidar [13] and/or fusing inertial sensor data with visual SLAM, as discussed next.

(iii) Leverage inertial sensor data to improve accuracy and performance: In this paper we showed the usage of inertial sensor

data to eliminate frames with no displacement, consequently improving the efficiency of our hybrid SLAM (see Section 4.4). Recent work [49] has shown that using inertial sensor data can increase the accuracy of visual SLAM techniques, but this is done by relying on high sampling rate (200Hz) inertial sensor data, which is typically not supported by low-end smartphones. An interesting direction to improve the accuracy of SLAM is to fuse low-rate inertial sensor samples from smartphones with visual SLAM techniques that uses sparse fiducial markers such as hybrid SLAM or markerSfM [26].

10 RELATED WORK

We discuss the relevant literature and the unique aspects of ALT.

Smartphone-based system for driver gaze monitoring: Some of the past work has used smartphones and their sensors to provide ADAS-like (Advance Driver Assistance System) capabilities [21, 46, 52, 54]. Much of the work has focused on driving-related issues like speeding, sharp braking, lane departure, etc. [18, 20, 30, 42, 50], however, our focus is on *driver* monitoring (i.e., gaze) and extending the driving monitoring for specific maneuvers using just a smartphone. Recent work [22, 34] has focused on driver gaze detection to determine situational awareness. However, these techniques depend upon (i) reliable pupil detection, and (ii) individual driver's calibration for gaze.

One work that is particularly relevant to ALT is CarSafe [54], which uses a smartphone to detect distracted driving. However, ALT differs from CarSafe in significant ways. CarSafe determines the driver's gaze solely based on head pose and also uses fixed thresholds (+/-15^o relative to when the driver is looking straight), to determine when the driver is looking to the left or right. As shown in Section 6.2, such an approach has shortcomings, and hence we design ALT to *combine both head pose and eye gaze*, and perform *auto-calibration for robustness*.

Fine-grained vehicle trajectory tracking: One approach to derive trajectory is using sensors such as LIDAR [5, 41], which are accurate but also have a high cost that stymies adoption. Another approach is to use inertial sensors on the smartphone to derive trajectory information. However, sensor drift, integrated over time, results in inaccurate trajectory estimation [20]. Monocular camera-based visual SLAM, e.g., ORB SLAM [45], has also been employed. However, the reliance solely on visual features tends to increase error and does not yield the real-world scale. On the other hand, approaches based on fiducial markers [43, 44] yield greater accuracy but suffer from the need for an extensive deployment of markers.

While ALT also relies on a monocular camera, it employs a *novel hybrid SLAM approach* that combines the accuracy of a fiducial markers based approach with the minimal infrastructure dependence of visual SLAM. ALT also incorporates a number of elements for *efficient yet accurate processing on smartphones*.

11 CONCLUSION

In this paper, we presented ALT, a low-cost smartphone-based system towards the goal of automating the driver's license test. The windshield-mounted smartphone acts as the sole sensing and computation platform in ALT, and monitors both the driver's gaze and maneuvers to assess the driver's performance. Our controlled experiments and deployments in real-world settings show ALT's effectiveness in automating many aspects of driver license testing.

APPENDIX - 1

Scale recovery and pose optimization using fiducial markers

We now describe the approach to recover scale using fiducial markers. Inspired by the initialization procedure described in [43], we find all possible relative poses from the common markers present in the initial keyframe and the current keyframe. This is computed using the corner points of each marker to find two possible poses of a frame [23]. Say, for a marker m (all of a known side length s) seen in both the frames, the two sets of poses would be $\{P_1, \hat{P}_1\}$ and $\{P_2, \hat{P}_2\}$ and the set of relative poses would be $C_m = \{P_2(P_1)^{-1}, P_2(\hat{P}_1)^{-1}, \hat{P}_2(P_1)^{-1}, \hat{P}_2(\hat{P}_1)^{-1}\}$. Let the set of all relative poses generated from every marker common in both frames be denoted by $C_{1,2}$.

We select the relative pose from this set such that it minimizes the *reprojection error* of all the corners of all the markers in all the frames (i.e. we triangulate the corners to the map using the computed poses and then re-project the corners back to each frame and compute the squared difference, denoted by $E_{i,m}$ where subscripts refer to the frame and marker respectively). That is,

$$P_{opt} = \underset{P \in C_{1,2}}{\arg\min} \sum_{m \in M_1 \cap M_2} min(E_{2,m}(PP_{1,m}), E_{2,m}(P\hat{P}_{(1,m)}) + min(E_{1,m}(P^{-1}P_{2,m}), E_{1,m}(P^{-1}\hat{P}_{2,m}))$$

Instead of using the correspondences found by ORB-SLAM to compute the pose, we use the aforementioned pose to establish the global reference system and initialize the map. We also ensure the two frames satisfy all map initialization conditions imposed by ORB-SLAM. Moreover, we add the markers found in our frames to our map structure and save their pose *wrt* the global reference system established (details in [43]). Lastly, we trigger a global bundle adjustment to optimize the computed poses and map points. This results in an initial map that's at the correct scale. However, the scale may drift as the SLAM progresses, the next step ensures that we minimize the drift and computed pose is accurate.

For every new frame we derive pose information using both points extracted by ORB SLAM and markers in the scene. For every frame we then detect if there are fiducial markers present in the scene. If there's a new marker that's detected in the frame, we add the new marker to the map structure and make the current frame a keyframe. To incorporate the contribution of markers to the estimation of pose, we weigh by $w_{i,m}$ and add the reprojection error of the markers to the optimization that optimizes pose with the correspondences only. The cost function then becomes,

$$C = \sum_{i,j} \rho_h(e_{i,j}^T \Omega_{i,j}^{-1} e_{i,j}) + \sum_{i,m} w_{i,m} E_{i,m}$$

where i denotes the keyframe, $e_{i,j}$ denotes the error of the map point j in keyframe i, $\Omega_{i,j}$ denotes the covariance matrix related to pyramid scale of map point detection and ρ_h denotes huber cost function. As in ORB-SLAM, the same cost function is optimized when performing pose optimization (all points fixed and just the camera pose as parameter), local bundle adjustment (local points are optimized while the keyframe subset is kept fixed) and global bundle adjustment (where all points and keyframes are optimized).

ACKNOWLEDGMENTS

ALT is part of the HAMS project [4] at Microsoft Research India and we thank our colleague, Satish Sangameswaran, and former interns, Shruthi Bannur and Harshvardhan Kalra, for their contributions to the broader HAMS effort. We thank the team at Maruti-Suzuki Institute of Driving and Traffic Research (IDTR), including specifically Aashish Mathur, Vasu Patel, Mahesh Rajoria, N. Seetharaman, and Ashish Shukla, for their partnership and for facilitating the deployment and testing of ALT at their faclities in New Delhi and Dehradun. We are grateful to have had the support of Abhay Damle (Joint Secretary, Ministry of Road Transport and Highways) and Shailesh Bagauli (Transport Secretary, Uttarakhand) in enabling us to take this research from the lab to the field. We also thank Pavan Kumar and Nipun Net Solution for helping with implementation in the field. We owe a debt of gratitude to the many drivers, instructors, and other members of staff at Microsoft Research India and at IDTR for humouring us with patience as we iterated through many versions of our system. Finally, we thank our anonymous shepherd and the reviewers for their valuable comments.

REFERENCES

[1] 1989. The Central Motor Vehicles Rules, 1989. http://admis.hp.nic.in/transport/pdf_forms/TheCentralMotorVehiclesRules,1989.pdf. (1989).
[2] 1990. National Traffic and Motor Vehicle Safety Act. https://uslaw.link/citation/us-law/public/89/563. (1990).
[3] 2015. DMV rules for road test. (2015). https://driving-tests.org/dmv-handbook-drivers-manual/.
[4] 2015. Harnessing Automobiles for Safety, Microsoft Research India. (2015). https://aka.ms/hams/.
[5] 2015. Low-cost LIDAR – A Key Technology to Enable Autonomous Driving in Urban Environments. (2015). http://www.osram-os.com/osram_os/en/press/press-releases/ir-devices-and-laser-diodes/2015/low-cost-lidar–a-key-technology-to-enable-autonomous-driving-in-urban-environments/.
[6] 2015. TOBII Pro Glasses 2. (2015). https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/.
[7] 2015. WHO Global Health Observatory (GHO) data . http://www.who.int/gho/road_safety/mortality/en/. (2015).
[8] 2015. WHO Road traffic injuries . (2015). http://www.who.int/mediacentre/factsheets/fs358/en/.
[9] 2016. Scale in Simultaneous Localisation and Mapping. (2016). https://www.kudan.eu/kudan-news/scale-simultaneous-localisation-mapping/.
[10] 2017. Road Safety in India Public Perception Survey. http://savelifefoundation.org/wp-content/uploads/2017/07/Road-Safety-in-India_Public-Perception-Survey_SLF.pdf. (2017).
[11] 2018. Personal communication with driving test track operator, October, 2018. (2018). .
[12] 2018. Qualcomm Neural Processing SDK for AI (SNPE). (2018). https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk.
[13] 2019. Lidar Lite v3. https://www.thingbits.net/products/lidar-lite-v3. (2019).
[14] Moustafa Alzantot, Yingnan Wang, Zhengshuang Ren, and Mani B Srivastava. 2017. Rstensorflow: Gpu enabled tensorflow for deep learning on commodity android devices. In *Proceedings of the 1st International Workshop on Deep Learning for Mobile Systems and Applications*. ACM, 7–12.
[15] Andrew J. Davison. 2003. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings Ninth IEEE International Conference on Computer Vision*. 1403–1410 vol.2. https://doi.org/10.1109/ICCV.2003.1238654
[16] Mohammad OA Aqel, Mohammad H Marhaban, M Iqbal Saripan, and Napsiah Bt Ismail. 2016. Review of visual odometry: types, approaches, challenges, and applications. *SpringerPlus* 5, 1 (2016), 1897.
[17] Tadas Baltrusaitis, Peter Robinson, and Louis-Philippe Morency. 2013. Constrained Local Neural Fields for Robust Facial Landmark Detection in the Wild. *2013 IEEE International Conference on Computer Vision Workshops* (2013), 354–361.
[18] Ravi Bhandari, Akshay Uttama Nambi, Venkata N Padmanabhan, and Bhaskaran Raman. 2018. DeepLane: camera-assisted GPS for driving lane detection. In *Proceedings of the 5th Conference on Systems for Built Environments*. ACM, 73–82.
[19] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 10 (2008), P10008.
[20] Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G Shin. 2015. Invisible sensing of vehicle steering with smartphones. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 1–13.
[21] Hon Chu, Vijay Raman, Jeffrey Shen, Aman Kansal, Victor Bahl, and Romit Roy Choudhury. 2014. I am a smartphone and I know my user is driving. In *COMSNETS*.
[22] Meng-Che Chuang, Raja Bala, Edgar A Bernal, Peter Paul, and Aaron Burry. 2014. Estimating gaze direction of vehicle drivers using a smartphone camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 165–170.
[23] Toby Collins and Adrien Bartoli. 2014. Infinitesimal plane-based pose estimation. *International Journal of Computer Vision* 109, 3 (2014), 252–286.
[24] N. H. Cuong and H. T. Hoang. 2010. Eye-gaze detection with a single WebCAM based on geometry features extraction. In *2010 11th International Conference on Control Automation Robotics Vision*. 2507–2512. https://doi.org/10.1109/ICARCV.2010.5707319
[25] Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, Vol. 1. IEEE, 886–893.
[26] Joseph DeGol, Timothy Bretl, and Derek Hoiem. 2018. Improved structure from motion using fiducial marker matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 273–288.
[27] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47, 6 (2014), 2280–2292.
[28] Anurag Ghosh, Vijay Lingam, Ishit Mehta, Akshay Uttama Nambi, Venkata N. Padmanabhan, and Satish Sangameswaran. 2019. Demo: Smartphone-based Driver License Testing. In *17th ACM Conference on Embedded Networked Sensor Systems, SenSys (ACM SenSys '19)*.
[29] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. 2008. Multi-PIE. In *2008 8th IEEE International Conference on Automatic Face Gesture Recognition*. 1–8. https://doi.org/10.1109/AFGR.2008.4813399
[30] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. 2006. CarTel: a distributed mobile sensor computing system. In *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, 125–138.
[31] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. 2015. Spatial Transformer Networks. In *NIPS*.
[32] S. Jha and C. Busso. 2016. Analyzing the relationship between head pose and gaze to model driver visual attention. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 2157–2162. https://doi.org/10.1109/ITSC.2016.7795905
[33] L. Jianfeng and L. Shigang. 2014. Eye-Model-Based Gaze Estimation by RGB-D Camera. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 606–610. https://doi.org/10.1109/CVPRW.2014.93
[34] Sinan Kaplan, Mehmet Amac Guvensan, Ali Gokhan Yavuz, and Yasin Karalurt. 2015. Driver behavior analysis for safe driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* 16, 6 (2015), 3017–3032.
[35] Vahid Kazemi and Josephine Sullivan. 2014. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1867–1874.
[36] M. Kok, J. D. Hol, and T. B. SchÃűn. 2017. *Using Inertial Sensors for Position and Orientation Estimation*. now. https://ieeexplore.ieee.org/document/8187588
[37] Yann LeCun. 1998. Gradient-based Learning Applied to Document Recognition.
[38] N. Li and C. Busso. 2016. Detecting Drivers' Mirror-Checking Actions and Its Application to Maneuver and Secondary Task Recognition. *IEEE Transactions on Intelligent Transportation Systems* 17, 4 (April 2016), 980–992. https://doi.org/10.1109/TITS.2015.2493451
[39] Hyon Lim and Young Sam Lee. 2009. Real-time single camera SLAM using fiducial markers. In *ICCAS-SICE, 2009*. IEEE, 177–182.
[40] Manolis IA Lourakis and Antonis A Argyros. 2009. SBA: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)* 36, 1 (2009), 2.
[41] Will Maddern, Geoffrey Pascoe, and Paul Newman. 2015. Leveraging experience for large-scale LIDAR localisation in changing cities. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 1684–1691.
[42] Prashanth Mohan, Venkata N. Padmanabhan, and Ramachandran Ramjee. 2008. Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones. In *ACM SenSys*.
[43] Rafael Munoz-Salinas, Manuel J Marin-Jimenez, and R Medina-Carnicer. 2019. SPM-SLAM: Simultaneous localization and mapping with squared planar markers. *Pattern Recognition* 86 (2019), 156–171.
[44] Rafael Muñoz-Salinas, Manuel J Marín-Jimenez, Enrique Yeguas-Bolivar, and Rafael Medina-Carnicer. 2018. Mapping and localization from planar markers. *Pattern Recognition* 73 (2018), 158–171.
[45] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics* 31, 5 (2015), 1147–1163.

[46] Akshay Uttama Nambi, Shruthi Bannur, Ishit Mehta, Harshvardhan Kalra, Aditya Virmani, Venkata N Padmanabhan, Ravi Bhandari, and Bhaskaran Raman. 2018. HAMS: Driver and Driving Monitoring using a Smartphone. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 840–842.

[47] Akshay Uttama Nambi, Adtiya Virmani, and Venkata N. Padmanabhan. 2018. FarSight: A Smartphone-based Vehicle Ranging System. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 181 (Dec. 2018), 22 pages. https://doi.org/10.1145/3287059

[48] Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 3400–3407.

[49] Tong Qin, Peiliang Li, and Shaojie Shen. 2018. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics* 34, 4 (2018), 1004–1020.

[50] Johan Wahlstrom, Isaac Skog, and Peter Handel. 2017. Smartphone-Based Vehicle Telematics: A Ten-Year Anniversary. *IEEE Transactions on Intelligent Transportation Systems* 18, 10 (Oct 2017), 2802–2825.

[51] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. 2016. Learning an Appearance-Based Gaze Estimator from One Million Synthesised Images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. 131–138.

[52] Jie Yang, Simon Sidhom, Gayathri Chandrasekaran, Tam Vu, Hongbo Liu, Nicolae Cecan, Yingying Chen, Marco Gruteser, and Richard P. Martin. 2011. Detecting Driver Phone Use Leveraging Car Speakers. In *ACM Mobicom*.

[53] Nan Yang, Rui Wang, Xiang Gao, and Daniel Cremers. 2018. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect. *IEEE Robotics and Automation Letters* 3, 4 (2018), 2878–2885.

[54] 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, and Andrew T. Campbell. 2013. CarSafe App: Alerting Drowsy and Distracted Drivers using Dual Cameras on Smartphones. In *ACM MobiSys*.

[55] Stefanos Zafeiriou, George Trigeorgis, Grigorios Chrysos, Jiankang Deng, and Jie Shen. 2017. The Menpo Facial Landmark Localisation Challenge: A Step Towards the Solution. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2017), 2116–2125.

[56] Xiangxin Zhu and Deva Ramanan. 2012. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2879–2886.