An Innovative Smartphone-based Solution for Traffic Rule Violation Detection

Waleed Alasmary
Computer Engineering Department
College of Computer and Information Systems
Umm Al-Qura University
Saudi Arabia

***Abstract*—This paper introduces a novel smartphone-based solution to detect different traffic rule violations using a variety of computer vision and networking technologies. We propose the use of smartphones as participatory sensors via their cameras to detect the moving and stationary objects (e.g., cars and lane markers) and understand the resulting driving and traffic violation of each object. We propose novel framework which uses a fast in-mobile traffic violation detector for rapid detection of traffic rule violation. After that, the smartphone transmits the data to the cloud where more powerful computer vision and machine learning operations are used to detect the traffic violation with a higher accuracy. We show that the proposed framework detection is very accurate by combining a) a Haar-like feature cascade detector at the in-mobile level, and b) a deep learning-based classifier, and support-vector machine-based classifiers in the cloud. The accuracy of the deep convolutional network is about 92% for true positive and 95% for true negative. The proposed framework demonstrates a potential for mobile-based traffic violation detection by especially by combining the information of accurate relative position and relative speed. Finally, we propose a real-time scheduling scheme in order to optimize the use of battery and real-time bandwidth of the users given partially known navigation information among the different users in the network, which us the real case. We show that the navigation information is very important in order to better utilize the battery and bandwidth for each user for a small number of users compared to the navigation trajectory length. That is, the utilization of the resources is directly related to the number of available participants, and the accuracy of navigation information.**

***Keywords*—Participatory sensing; traffic violation detection; automatic detection; applied computer vision; resources optimization**

I. Introduction

There are major challenges in transportation that require immediate innovative solutions [1]. The first one is road congestion. The second but most important one is accidents and fatalities. These are worldwide issues, where major cities and suburban areas become more congested, and car accidents with fatalities continue to occur. For example, in Saudi Arabia, there were 7000 fatalities and $45,000$ accidents in January 2015 [2]. It is clear that the disobeying traffic rules is among the top causes of fatalities, which motivated the government of the Kingdom to use smart stationary cameras equipped with sensors to detect the car speed on the highway, and also other cameras that could detect cars that crosses red light signal at intersections. However this solution is considered expensive. Using a system of *stationary* cameras requires manpower to operate. Therefore, we propose a smartphone-based system to detect traffic rule violation. To the best of our knowledge, this is the first paper that proposes such a system.

Current intelligent transportation solutions assume an on-board device within the car [3]. This device can monitor the location and speed of the car and then the *data* can be used to evaluate the traffic violation. A well-known product is provided by MobilEye that requires installing multiple devices and sensors into the car [4]. In [5], smartphones are used to detect the car turning or speeding, and it is intended to collect the data of the car that has the smartphone in it. In this paper, we propose a solution to detect the traffic rule violation of the other cars on the road, without the need to install any device in the *monitored* car, rather, we propose the use of smartphones as *monitoring* devices. We pursue this research with the mindset that technology can help in improving road congestion, and traffic accidents by combining driving traffic violation detection with participatory sensing.

Our proposed system automatically detects a number of traffic rule violations on the roads. Specifically, we propose that each driver willing to participate can download our application on his/her smartphone. The major sensor used is the camera in the device. This is different from the currently used technologies where an on-board unit monitor the vehicle in which it is installed. Our proposed system would make each car act as a monitor for the other vehicles on the roads.

The system has two main components, namely, an in-mobile component, and an in-cloud component. The in-mobile component first detects the traffic rule violation on the roads using based on multiple subsequent time frames. This component is fast and had limited resources, and hence limited accuracy. After that, if a traffic rule violation is detected, the snapshot of the video-frame is sent to the in-cloud component which is more intensive in resources and has higher accuracy.

We evaluate our proposed system by validation with video traces captured by cameras on the cars in the Middle East and USA. Our system shows a very high accuracy for multiple traffic scenario detection, which hypothetically represent traffic violation detection. We used multiple data sets for training and testing [6]–[21]. We used some images as is, and we created some data from video traces as will be explained in Section IV-A. Our results strongly supports the feasibility of the proposed scheme. Furthermore, we propose a model to optimize the use of participatory sensors selection under the limited resources scenario. The main challenge of this

optimization problem is to simultaneously optimize multiple resources within a dynamic mobile environment. We show that the proposed model can achieve near-optimal resource utilization results compared to the fully known mobility dynamic.

The contributions of this paper are as follows:

- We propose a novel smartphone-based participatory sensing system for traffic rule violation detection that is accommodating to new computer vision, sensing, and networking technologies.
- We propose the relative positioning and relative speeding concept in out system design in order to infer the traffic violation from the computer vision algorithms and the smartphone sensors readings. These two metrics allow us to understand the driving context from which the traffic rule violation can be detected.
- We evaluate the proposed system using multiple data sets, and videos that are taken by smartphones (or smartphone like cameras).
- We propose a participatory sensor optimization framework to enhance battery and bandwidth utilization while guaranteeing smartphone sensing.

In the following section, we discuss the related works to our proposed framework. Then, in Section II, we explain the system model and the traffic detection framework, followed by explanation of the participatory sensor optimization scheme in Section III. After that Section IV demonstrates the experiment setup, the data set used for evaluation, and the evaluation results. We discuss the recent related works in the areas of applied computer vision and intelligent transportation systems in Section V. Finally, Section VI concludes the paper and outlines the position of the proposed framework within current and future technologies including autonomous vehicles.

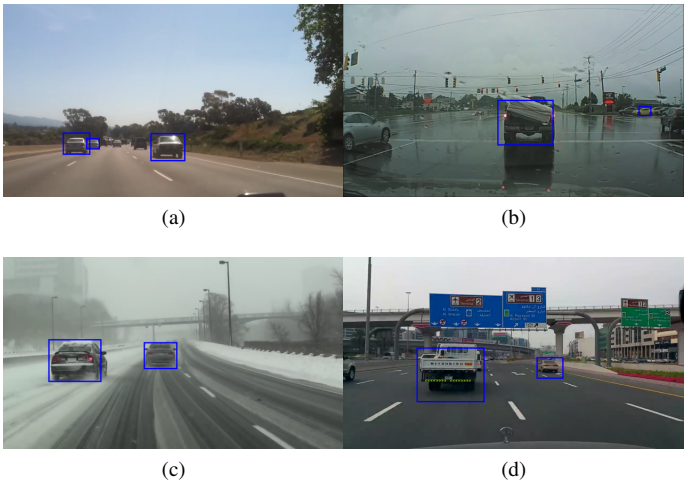
II. System Model

This section introduces an the overview of the proposed system, which is composed of three phases. The involved phases are as follows: 1) A fast in-mobile traffic rule violation detection phase, 2) A high-accuracy and cloud-based traffic rule violation detection phase, and 3) An in-mobile enhancement of the traffic rule violation detection phase. Each of the above mentioned phases is composed of a number of steps. Each of the following subsection describes the processes involved in each stage.

A. Fast in-mobile Traffic Violation Detection

This phase utilizes the captured video streams to detect the vehicles, the background, and the lane-markers. Then the said detected objects are used to track driving-behavior of all surrounding vehicles. This phase contains the following steps:

- Vehicles detection and classification.
- Lane marker detection.
- Motion detection.



(a) (b) (c) (d)

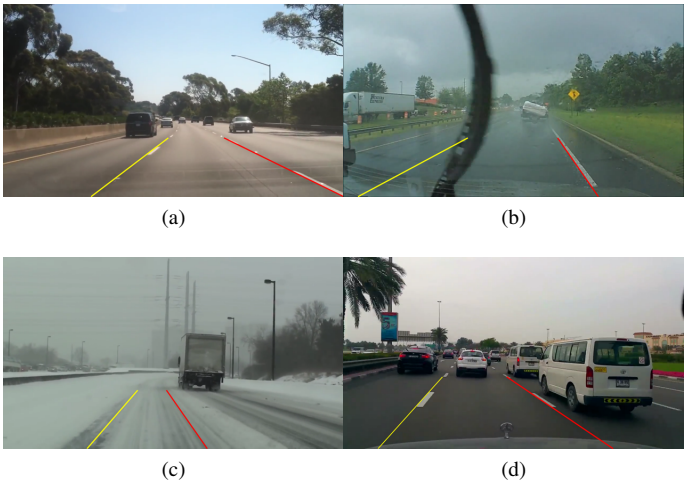
Fig. 1. Output samples of vehicles detection and classification using the in-mobile detector.

1) In-mobile: Vehicles Detection and Classification: We propose the use of the Haar-like feature-based cascade classifier driven by the AdaBoost algorithm as in [22] [23]. One reason is that this detector is very fast due to the use of integral images. It is also very accurate given a large number of training images, which is doable for such a crowd sourcing system. The main reason though is that optimized implementations are available for the algorithm (mainly designed for face detection), and can be integrated to the smartphone without much loss of speed. There are multiple variants and optimized detectors that also uses Haar-like features. We discuss them in Section V. A fast RCNN is the other candidate, but it is much slower than the Haar feature-based cascade classifier. It is shown in [24] that the processing time in CPU takes about 2 seconds, and the current smartphones are not powerful as our desktops. An RCNN can be used whenever the hardware allows it. This is the reason that deep learning is usually used in the server-side, for faster processing. Fig. 1 shows some samples of the outputs of this step. Each successfully detected vehicles is surrounded by a blue bonding box. [1]

2) Lane-markers detection: We use Hough transform to detect the two lane-markers on the street. We can also detect the two outer lane-markers on a three-lane, but we only focused on the two-lane-markers ahead of the car. This fast lane-detector was also used in [22], [23]. The lane-detector nominates lanes, and we only select the lanes if they are repeated in multiple frames. The repetition threshold can be adjusted as desired. We found that setting the repetition threshold to 4 or 5 is sufficient to detect lanes with high accuracy. ‘ Fig. 2 depicts some sample of lanes markers detection, the detected left and right lane markers are shown in yellow and red color, respectively.

3) Motion Detection: Motion detection can be performed using various methods. We chose our method to be aligned with the characteristics of the fast in-mobile detector. The main principle is as follows. Given an object that is detected around the same area in two consecutive frames in time, we map

[1] In Section IV-A, we explain which data set we used and the parameters used for training and testing of the classifier.



(a) (b)

(c) (d)

Fig. 2. Output samples of lane markers detection of the in-mobile phase.

the features of the same object between the two frames. If all the mapped features move towards the top of the image, the object is assumed to be moving forward, and vice versa. Similarly, the object that is detected in the left side of the lane in one frame and mapped to an object that is detected in the right side of the same lane in a subsequent frame describes a motion from left to right, and vice versa. In this setting, we can detect multiple vehicle movements with respect to the available lanes. These motions can represent some traffic rule violations in some contexts. We use the SURF algorithm [25] for fast detection. We follow the feature matching with the variant of the random sample consensus (RANSAC), which is referred to as MSAC [26] and considered robust.

In order to make the motion detection procedure more accurate, we set a number of heuristic intuitive rules. First, we divide the image into three areas given the detected lane-markers. In this way, a detected vehicle bounding box must have its centroid somehow within in the correct position with respect to the lane-markers. Moreover, we also condition the far edges of the bottom points of the bounding box to be under the vanishing points of the image. These two procedures eliminate a good number of false positives. We can think of this detector as a motion inference tool. Combined with the information about the smartphone, it can understand the driving violation. For example, assume that the speed of the monitoring car is X. If the monitored car that is moving in front is speeding, the monitoring car can detect the speeding action speeding because the speed is $> X$. Another example is relating the camera parameters with my speed to detect the stalled vehicles. The detector also can infer extra information for approaching traffic light or a stop sign with high speed.

The false detection of the SURF algorithm are very minor. Moreover, when SURF is followed by MSAC, it provides matching points unless there is a very close match between the two different vehicles. Based on our experimental evaluation, we have not seen noticeable false detection. There are a few car detections on the background where there are objects similar to car structure. As the car is driving on the road, the background always moves backward. However, object motion is detected if all the matched points between the two consecutive frames are in the same direction, which is a strong condition for motion detection.

B. In-cloud Detector Traffic Violation Detection

The in-cloud detector is a component that is executed in the cloud. At the server side, we propose the use of a more powerful object detection and classification algorithms. We propose the use of the the deformable part-based models, and Discriminative learning with latent support vector machines (SVM) [27] [28] as an accurate detector. This detector is one of the most popular ones before the start of deep learning success in object detection. The detector has a very high accuracy and low recall rates, and can work on a regular CPU. We refer to this object detection method as Discriminatively Trained Part Based Models (DTDPM). This in-cloud detector has a processing delay that can take up to 1 second. *We performed multiple experiments on the laptop, and the processing delay is pretty consistent, but correlated with the image size*, but is very accurate in detecting vehicles, as we will demonstrate in the evaluation section. We also use the same lane-detection as in the fast detector. We also use convolutional neural networks (CNNs) to reduce the false positive detections. CNNs have several implementations and has a very high accuracy.

We also add an additional layer of verification using a *deep convolutional neural network* (CNN). Currently, there is a fast implementation of CNN [29] on the smartphone. However, it cannot perform object detection, rather, just classification. Therefore, our proposed in-mobile and in-cloud car detector extracts the possible vehicles and use a pre-trained CNN to remove false detection and affirm correct detections. We will demonstrate through the experiments that the accuracy of the CNN classifier is excellent. The CNN used in our model is based on [30], and has two classes, vehicles and backgrounds.

The lane-markers detection and motion detection is the same in both the fast in-mobile and in-cloud traffic rule violation detectors.

C. Data Collection and Dissemination Communication Models

For data communication, we define two data streams, data collection and data dissemination. The main focus in this paper is data collection. We assume that the data dissemination follows the current state-of-art technology. Data collection is mandatory, but can be performed in delay-sensitive or delay-tolerant methods. Sensors can opt in for delay-sensitive data collection, or they can use delay-tolerant option.

We assume two modes for data collection in our system model. First, we assume that the smartphone camera is used for data collection. In this mode, there is no communication overhead for data collection. Second, we assume that there are third party cameras that are used for data collection. These cameras can communicate with the smartphone via WiFi/Bluetooth. We assume that pairing is set up once, and repeated automatically every time the user enters the vehicle. In the sequel, we do not distinguish the two methods. We assume that the camera captured the data, and then the data is available at the smartphone for the next stage of processing.

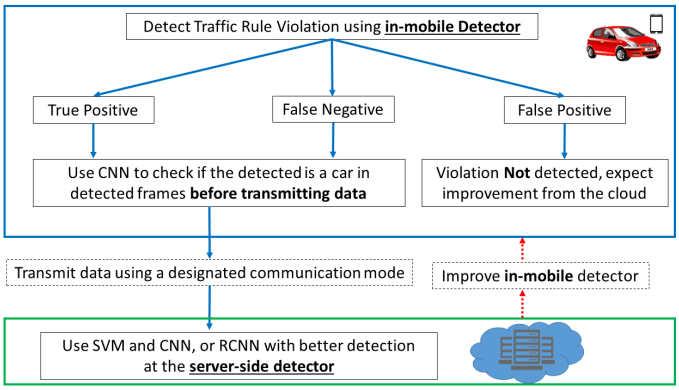


Fig. 3. Illustration of the Interconnection between the in-phone and in-cloud detection components.

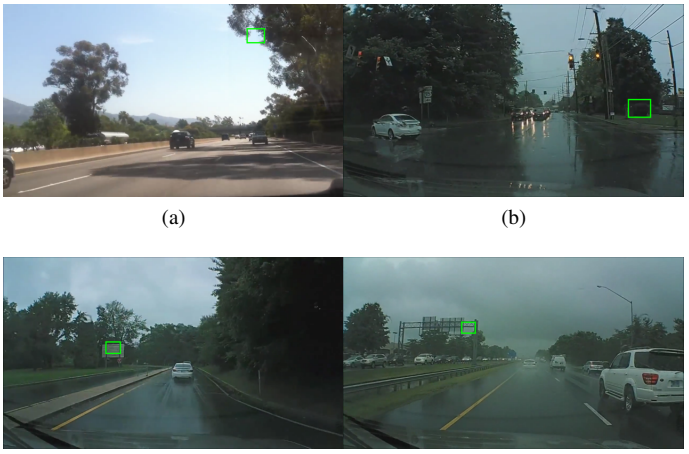


Fig. 4. Samples of the correct identification of the false positive detections using the CNN classifier. That is, incorrectly detecting backgrounds as vehicles. Those detections are then fed into the in-mobile detector.

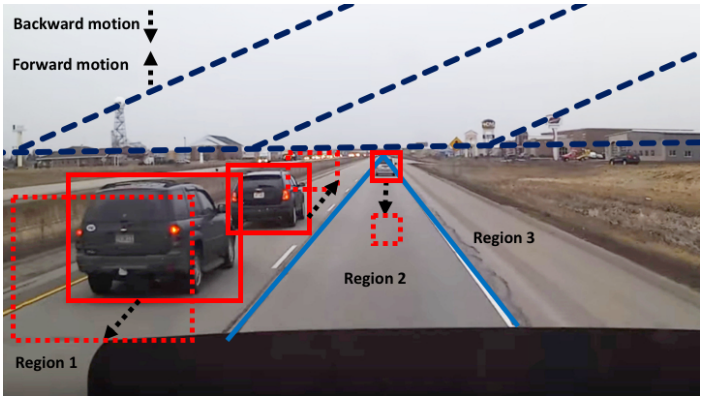


Fig. 5. The information detected by the driving activity detection model.

Dissemination of collected data is according to the user preference and the available communication interface. We assume that our model can collect the data, and the dissemination is available according to the device communication capability. A user with DSRC communication device attached to his/her smartphone (e.g., mobile accessory by Arada systems DSRC-enabled devices [31]) can disseminate messages using vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), or vehicle-to-any (V2X) communication modes. A user who has a data subscription on the SIM card can transmit the data to the data centre, or disseminate warning messages using device-to-device (D2D) communication. The most convenient and low-cost option is to transmit data using WiFi offloading by treating the data in delay-tolerant communication mode.

D. Interconnection between the In-Mobile and In-cloud Components

The accuracy of different components of the system are previously known in certain applications. In this context, we know the limited computation capability of the smartphone compared to the cloud. Hence, we use the optimized detection at the cloud in order to improve the in-phone detector results. This can be performed in two ways. First, the negative results of the in-phone component can be neglected once the result is corrected at the cloud. Second, the neglected information can be used to improve the in-phone by retraining it. This also can be performed for improving the true positive detections as well. The interconnection between the two components is shown in Fig. 3. Fig. 4 shows sample outputs of correct classifications of false outputs detects by the CNN classifier using in-cloud detector. Those results are then fed for training in the in-mobile detector for better detection.

E. Driving Activity Detection

In this section, we aim at understanding the driving activity of each detected car on the road. The detected information is the respective position with every other detected object in the image. This information provides an understanding of each car on the road, the location with respect to the road lanes, and the relative motion with respect to the capturing phone. The general information we aim to collect in this section is illustrated in Fig. 5. The figure demonstrates how a car the road is divided into multiple lanes via the lane markers. The classifiers detect each car and then in subsequent frames detect their motion. We also show the regions of elimination of false positives.

III. Participatory Sensing Optimization Problem

In this section, we assume that the smartphone parameters including battery and cellular infrastructure transmission are accessible by the our system. The user can also specify a quota that can be used by the application. The objective of the proposed optimization problem is to utilize the available resources to provide the best coverage of the roads to avoid missing any traffic rule violation while at the same time minimizing the battery usage and the data usage. We approach this problem in two stages in the following sections.

A. Sensor Selection Problem

Battery consumption is a critical parameter in participatory sensing. Many users might opt in to participate in improving the travel quality on the roads, reducing accident rates and

fatalities, and capturing the traffic rule violators. However, the participants might refrain from using the proposed system because of the associated battery consumption with the application. Hence, we provide a custom solution based on each user preference.

It might be argued that an interested participant would use a charger in his/her car. That could be true. However, the fact that a monitoring app–even if optimized for battery consumption, would still use the battery that can be used for social networking or other entertaining apps is a critical point for discourage some users. Therefore, we optimize our model for battery consumption and participant efficiency. A participant is efficient in this context if his/her smartphone covers the intended areas of the transportation networks [2]. Therefore, a participant's efficiency in the sequel refers to the effective coverage of area.

It is critical to understand that scheduling of the smartphones based on their location is a well studied problem in the literature. The challenge in this context is that we only know the locations at the current position, and the car mobility can change in the future. That is, scheduling the participants have randomness in the future trajectories. Scheduling cars without taking into consideration the trajectory information of their future movement might result in lower spatial and temporal coverage, and under-utilization of the available battery and transmission bandwidth resources. Hence, we introduce the concept of an estimated trajectory-based scheduling. Therefore, the resources are not optimally utilized, but a near-optimal utilization can be achieved, which results in a practical solution.

We assume that based on the current location of the car, and its current speed and heading information, the future locations of the vehicles can be estimated within a short window of time. The estimation is not perfect, hence a slight degradation in the utilization of resources would be expected. Whenever the mobility information is not available anymore, the only solution that can be used is a non-optimal solution.

B. Battery Optimization Problem Formulation

Assume that some users do not want to use their app continuously. Therefore, we define B_i^Q as the battery quota that is allowed by user i for the crowdsourcing process. Suppose we have N participants. Let L be the number of participants where B_i^Q is unlimited. On the other hand, let M be the set of participants where B_i^Q is limited. M are the focus of our resource allocator. Note that if N can provide the required spatial and temporal coverage of roads, then we do not have to use the M participants. However, we do not consider this case in our study. We consider the case where we are forced use the M resource limited participants. We assume that the location of all participants in the system are known at the scheduling time. Let $a_i[t]$ denote the activity of fast detector on the smartphone i at time t. That is

$$a_i[t] = \begin{cases} 1 & \text{if fast detector is activated} \\ 0 & \text{otherwise} \end{cases}.$$

[2]Accuracy is mainly related to the computer vision algorithms discussed in the previous sections, and is not considered as a metric here.

Then selection among the M participants to run the fast detector is performed via the following optimization problem in a centralized fashion[3]:

$$\underset{a_i[t]}{\text{Minimize}} \quad \sum_{i=1}^{M}\sum_{t=1}^{T} a_i[t] \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{M} a_i[t] \odot b_i[t] = 1,\ \forall t \in \{1,\cdots,T\}, \tag{2}$$

$$\sum_{t=1}^{T} a_i[t] \leq B_i^Q,\ \ \forall i \in \{1,\cdots,M\}, \tag{3}$$

$$a_i[t] \in \{0,1\}, b_i[t] \in \{0,1\}. \tag{4}$$

where $b_i[t]$ is an indicator that is set to 1 if the mobility is estimated in the near future; otherwise is set to 0[4]. Moreover, $B_i^Q < B_i$ is the amount of battery that is allocated for the fast detector at one scheduling epochs. In other words, the user specify $B_i = kB_i^Q$, where k is a predetermined parameter that demonstrates the length of acceptable predictability of mobility information. The above optimization problem allocates only one participant to cover an uncovered area for each time instant (Constraint 2). Moreover, it schedules sensors such as the battery consumed over time is related to the allocated battery consumption by the participant (constraint 3). Furthermore, it allocates participants according to their predicted availability in the uncovered locations using the variable $b_i[t]$ (constraint 2).

C. Realtime Transmission Problem Formulation

For the offloading communication option, all participating participants can collect data and transmit them using WiFi, whenever there is an available connection. However, when delay-sensitive data transmission is limited by the user, we optimize the transmission according to the provided quota set by each user. We also provide a single transmission for each captured event among all users. Let

$$T_{x_i[t]} = \begin{cases} 1 & \text{for delay-sensitive transmission} \\ 0 & \text{otherwise} \end{cases},$$

and D_i^Q, the normalized number of transmission times of captured traffic violations using the fast detector. Normally, the fast detector is used, then the transmission is established. Therefore, the delay-sensitive transmission scheduling is performed by the following scheduler:

$$\underset{a_i[t]}{\text{Minimize}} \quad \sum_{i=1}^{L}\sum_{t=1}^{T} T_{x_i[t]} \tag{5}$$

$$\text{subject to} \quad \sum_{i=1}^{L} T_{x_i[t]} b_i[t] = 1,\ \forall t \in \{1,\cdots,T\}, \tag{6}$$

$$\sum_{t=1}^{T} T_{x_i[t]} \leq D_i^Q,\ \forall i \in \{1,\cdots,M\}, \tag{7}$$

$$T_{x_i[t]} \in \{0,1\}, b_i[t] \in \{0,1\}. \tag{8}$$

[3]The central controller can be the fusion center or a cluster-head node within a group of nodes.

[4]Whenever $b_i[t] = 0$, the scheduler loses location information and would results in under-utilization of the resources.

This optimization problem would results in transmitting only one detected traffic violation in the same area (Constraint 6). The transmission quota of each participant is satisfied in Constraint 7. Note that we do not perform any battery constraint in the transmission. However, adding a battery constraint for transmission is doable. It should add a parameter that will constraint the number of transmissions. We assume that D_i^Q includes that information. In other words, the user enters the number of transmission time quota, and we calculate D_i^Q accordingly.

D. Battery and Transmission Optimization

It might be argued that problems (1)-(4) and (5)-(8) can be performed as a single optimization problem. However, the concept in our crowd sourcing system is that each user might chose to fully utilize activation, but not transmission, or vice versa. In other words, capturing an event does not necessarily require a delay-sensitive transmission. In fact, it might be expected that most users of such a crowd sourcing system will perform sensing without any delay-sensitive transmission as long as the transmission over the cellular network is not free. The incentive model that we use in the sequel does not require the users to pay extra money, unless they are willing to. Therefore, covering the traffic violation of the roads is the main objective of this work whether a realtime transmission occurs or not (realtime transmission of vehicular information as a separate topic that is widely researched). In addition to that, transmission might affect the battery consumption especially over cellular infrastructure. This directly reflects in our problem where we actually excluded the group of workers who are offering limited battery consumption. Adding that group of workers would result in chaining the parameters in problem (5)-(7), but not the scheduler design.

Therefore, we also add a third optimization problem, where each activated camera is activating the fast detector and transmitting given the fact it has a battery limitation and transmission budget. Let $c_i[t] = a_i[t] \odot T_{xi}[t]$, where $\odot$ is the binary product. Then the combined battery and transmission optimization problem becomes

$$\underset{c_i[t]}{\text{Minimize}} \quad \sum_{i=1}^{M}\sum_{t=1}^{T} c_i[t] \tag{9}$$

$$\text{subject to} \quad \sum_{i=1}^{M} c_i[t] \odot b_i[t] = 1,\ \forall t \in \{1,\cdots,T\}, \tag{10}$$

$$\sum_{t=1}^{T} c_i[t] \leq B_i^Q,\ \ \forall i \in \{1,\cdots,M\}, \tag{11}$$

$$\sum_{t=1}^{T} c_i[t] \leq D_i^Q,\ \ \forall i \in \{1,\cdots,M\}, \tag{12}$$

$$c_i[t] \in \{0,1\}, b_i[t] \in \{0,1\}. \tag{13}$$

This optimization problem actually combines the two set of constraints (2)-(4) and (6)-(8), which results in combining the fast detector activation and transmission according to the availability of battery power and transmission quota.

TABLE I. LIST OF DATA SETS USED FOR TRAINING THE CLASSIFIERS AND VEHICLE DETECTORS.

System Component	Training Data	Size
Fast detector (Positive)	Caltech car data set [6], [7]	1,182
Fast detector (Negative)	Caltech background data set [8]	1,599
CNN detector (Positive)	Extracted cars from [14], [15]	1,186
CNN detector (Negative)	Caltech background data set [8]	1,599

IV. PERFORMANCE EVALUATION

A. Experimental Setup

The proposed framework can be tested on a smartphone on the roads. However, such an experiment would consume extra time and effort for collecting the data and testing. Therefore, we chose to setup our evaluation procedure as follows. First, we downloaded several vehicles and backgrounds data sets from internet for training and some testing [6]–[13]. Second, we downloaded different videos from YouTube that are taken by smartphones and are used to capture the road [14]–[21]. The experiment we performed uses two separate training and testing sets for each part of the system. In other words, we used different training sets for different parts of the system, and the testing data is different from those data sets. A detailed information on how we trained the data is shown in Table I. For the positive training of the fast detector, we use the extended data that can be found at [7]. For the negative samples of the fast detector, we used Caltech background data set and added some negative images from [9]. For the CNN detector, we used extracted positive training samples from two videos using the car detector in [27], [28], and then removed images that have large background potions. For the negative samples, we removed any background images in [8] that contains cars in them. For the CNN detector, we used transfer learning of the pre-trained network [30].

For testing the CNN detector, we used the positive samples that contains cars in data set in [10]. We removed the background, and just used the bounding box of the car. We used the negative background samples in [11] for background detection. Note that the testing data were not used in the training of the fast detector nor the CNN detector at all. The CNN network has two outputs indicting whether the image is a car or not. For the testing of the system framework, we use the videos in [16]–[21]. Neither the videos nor images extracted from the videos were used for training at all. The videos represent different cities, different driving conditions including sunny, during and after sunset, night time, raining time, and snowing time.

The design of out experiment is based on the currently available data sets and videos. We believe that the accuracy of our proposed model can be enhanced given larger number of sets, which can be though a large-scale crowd sourcing or deployment of the system.

B. Performance Metrics

To evaluate the proposed system, define two categories of metrics, namely, accuracy metrics, and efficiency metrics. The

accuracy metrics are meant to evaluate the accuracy of the components of the driver detector. The efficiency are meant to evaluate the efficiency of the components of the framework including the driving detectors, and the resource scheduler.

Accuracy Metrics:

- *Accuracy*: The accuracy of the traffic violation detection is directly related to the accuracy of the computer vision algorithms. We define the accuracy as the percentage of correct classification using the machine learning algorithm.
- *Number of detected objects*: We use the number of identified objects as an indicator for the machine learning algorithm. It provides an intuition on the accuracy of the algorithm when we compare different algorithms with the same data set.
- *Above Horizon*: The horizon in our experiment is the vanishing point intersecting the two detected lanes. It is a measure of false positive vehicle detection.

Efficiency Metrics:

- *Processing time*: The time it takes to complete a specific computer vision operation.
- *Battery usage metric*: We define battery usage metric as the number of times the battery is used within the assigned budget by the participant, divided by the scheduling time. The lower this metric reflects a better use of the battery due to a better scheduling of the workers. Having a smaller battery usage metric during a scheduling epoch prolongs the use time of the each participant, and hence tends to provide more coverage on the roads.
- *Bandwidth usage metric*: Similar to the battery usage metric, the bandwidth usage metric is the number of times a transmission occurs given the assigned transmission budget, divided by the scheduling time. A lower bandwidth usage metric indicate better scheduling of the workers in favour of transmission budget.

Traffic Violation Metrics:

- *Detected motion*: This is the motion of the monitored vehicle. It can be detected as moving forward or moving backward between subsequent frames.
- *Relative position with respect to lanes*: This is the relative position of the detected cars with respect to each of the detected lanes. In other words, we know if the car is to the right of a lane or to the left.

C. Experimental Results

1) CNN Classifier: The CNN classifier reached very small incorrect classifications based on our training and testing. In order to show that, we test on labeled data of cars and backgrounds, and we show the results in Table II. We have tested the data over $16,186$ car images from [10], and the accuracy is more than 92%. We have also testing the data over $1,156$ backgrounds from [11], and the accuracy is more than 95%. We believe that the training data could be improved in order to generalize the system to get a better accuracy. We also notice in the same figure that the processing time is large due to the fact of using CNN on Matlab, and Matlab is known to be slow, and the implementation of the CNN is currently not optimized for speed. However, for realtime implantation, SDKs such as [29] can be used. A recently optimized implementation for speed has been published achieves an average precision of at most 88.7% on smartphones [32].

TABLE II. THE ACCURACY AND PROCESSING TIME FOR THE CNN CLASSIFIER.

Classification	Accuracy	Processing Time
Car	92%	1.3799 seconds (on CPU)
Background	95%	1.3763 seconds (on CPU)

2) Single Video Resolution for Training and Testing: We now demonstrate the accuracy metrics of the proposed framework on average. We use the video from YouTube [16]. We run the algorithm over the videos and plot the performance metrics in Fig. 7(a), (b), and (c). We use one video to show the relative performance between the fast and in-cloud detectors. In Fig. 7(a), we the number of frames fed to each detector, the number of detected cars, and the results of post processing after detection. We can see the number of detected cars is somehow close between the two detectors. However, after pre-processing, more than $2/3rd$ of the vehicles that were detected by the fast detector are eliminated due to false detections, detection of elements above the horizon, or the bounding box is not properly aligned between the detected lane markers. We noticed that if we relax the lane markers constraint, then the number of vehicles after post-processing becomes closer to the in-cloud detector. It is also expected that the in-cloud detector is more accurate especially in eliminating the false negatives. However, to minimize the number of false detections in the fast detector, we used the CNN detector and post-processing.

In Fig. 7(b), we show the number of detected motions. That is, out of the multiple detected cars, in which a small number of the detections might be false, the number of relative motions is much smaller than the number of vehicles. This is due to the fact that a detected motion has to be between two images of the same vehicles, and the feature matching-based motion detection is very strict in favour of accuracy. The accuracy arises from the fact that we only detect a vehicle motion if all matched feature points move in the same direction. This procedure eliminates the noise in the background (i.e., another vehicle detected within the bounding box, or the sidewalks). We can see that the number of matching in in-cloud detector is lower than the fast detector. We checked multiple frames in the in-cloud detector, and we found that the detector usually provides a bounding box that contains a noticeable portion of the background. Features matched to the background will reduce the accuracy of the motion detector. The same figure shows the number of detected lane marker is the same for both detectors, which is normal for a Hough transform-based lane-marker detector. However, the number of detected relative positions with respect to the lane marker is much lower than the number of the detected cars using the either fast or in-cloud detectors. This is due to the fact that we do not consider a lane-marker until it gets repeated for multiple consecutive frames, and that we require the centroid and the edges of the bounding

box to be in the same relative position to the lane-marker. Finally, the figure shows that we detected 543 false positive detections above the horizon. This is how post-processing eliminates such false detections. The in-cloud detector did not results in any cars that are positioned above the horizon.

We show the processing times for each detector including the lane marker detector in Fig. 7(c). We can see that the fast detector is very fast while the in-cloud detector takes about 2 seconds (0.511×4 as the results are scaled in the figure for the first column). The lane marker detection is very fast, too.

3) Different Video Resolution for Training and Testing: What we want to achieve in this section is detection of the traffic violation with multiple video characteristics, each has a different resolution, frame rate, etc. We will shortly demonstrate the results for different resolutions of a video, and discuss the detected traffic violations.

We want to mention that we did not change the parameters of the fast or lane detectors at all, although the detector could be optimized for different resolutions. These parameters are set in away that is suitable for a resolution that is either 360p and 720p. However, we don't use any parameter tuning, and we test the robustness of the detector across different frame resolutions. For example, the search space of the lane marker detection area is the same for all resolutions. Hence, it performs poorly for the low resolution (144p) because it takes the most of the image in the search space, and for the high resolution (720p) because the search area is too small and it becomes hard to detect lane markers. The fix for this issue is to set the search area for the lane marker as a percentage of the image size. We found that this hard thresholding works very well, but we don't put the result due to the space limitations of the paper.

Another example for parameter tuning is the merge threshold of the fast detector. The merge threshold used to generate all the results is the same. However, it could be related to the size of the input image. We found that the detector struggles to find cars within the 144p video. Actually the images were not clear at all. Increasing the merge threshold could help in this case as multiple weak detections that are not accurate accumulate to select an object within the input image. Similarly, In the higher resolution, the merge search window size for the fast detector could be enlarged in order to capture cars with bigger sizes. These parameters can be optimally selected given the device type, the camera resolution, or the application settings.

An interesting test was performed when testing the fast detector using multiple resolutions of the same video [16], namely, 144p, 180p, 360p, and 720p. We generated these videos from the same YouTube link. We would like to mention that we trained the classifier with images that has different resolution. In Fig. 8(a), we can see that the number of fames is the same for all resolutions except for 144p, where some frame are lost during conversion. The figure shows that there is a pattern in the number of detections that follows the increase in the resolution. However, Fig. 8(b) shows that these numbers might be deceiving. For example, most of the detections do not result in the same pattern when detecting the motion, the lane markers, or relative positions. Hence, the fast detector should be trained for several resolutions in order to perform similarly regardless of the input video resolution. We did not use this criterion for the generation of data, but it is an interesting observation. Finally, we see a clear trend between the resolution and the processing time of the fast detector, and the lane-marker detector in Fig. 8(c). As the resolution increases, the processing time increases, and vice versa.

4) Battery and Bandwidth Optimization: For the evaluation of the optimization problem, we set the the optimization epoch time to $T = 10$. We then restrict the battery and bandwidth for each participant, by choosing a batter or bandwidth that are uniformly distributed in the range $[1, f(T, \delta)]$, where δ is an integer. We chose δ to enable smaller for the bandwidth constraints in the experiment (i.e., the bandwidth constraint are more stringent because some participants might not have unlimited data plans. However, the simulations can be performed in different ways as we show next. We set $B_i^Q = \lfloor \frac{T}{\delta} \rfloor$ for $\delta = 3$, and $D_i^Q = \lfloor \frac{T}{\delta} \rfloor$ for $\delta = 2$, and we plot the results in Fig. 6 (a). The figure shows the batter and bandwidth metrics vs the number of participants.

First, we observe that knowing the navigation information (dashed lines in the figure) has a lower use of the battery and bandwidth compared to the estimated navigation information (solid lines in the figure) regardless of the used metric or the optimization problem. The dashed lines are mostly higher than the solid lines, which means the scheme used more resources either in terms of batter or bandwidth. We also observe for this setting there is not a significant difference for performing the joint optimization versus a metric specific optimization (at least for the estimated navigation information case, i.e., solid lines). Hence, if the resources are scarce compared to the optimization length. It can also be inferred from the figure that as the resources increase, the resources utilization improves for the solid lines. However, when the navigation information is known, the combined battery and bandwidth optimization provides significant improvement over separate optimization, especially for the lower number of participants case.

We change the $B_i^Q = T - \delta$ for $\delta = 2$, and $D_i^Q = T - \delta$ for $\delta = 1$, and we plot the results in Fig. 6 (b). In this case, the bandwidth constrains should very close to the battery constraints as the difference in δ is small. As expected, the solid lines became closer to each others, which means that the solving either of the problems would not make a significant difference. Interestingly, losing the navigation information would also result in a significant gap between the solid lines and dashed lines, which means that using accurate navigation information (i.e., $b_i[t]$) would improve the utilization of resources for a small number of participants. However, if the number of participants is large, then, the navigation information cannot provide significant improvement.

V. Related Work

Car mobility and traffic violation detection has been studied from different perspectives. One part studies the computer vision interpretation of the car mobility. Another aspect is the communication methodology. Third, the different specific applications in the context. We provide the necessary review of the current works.

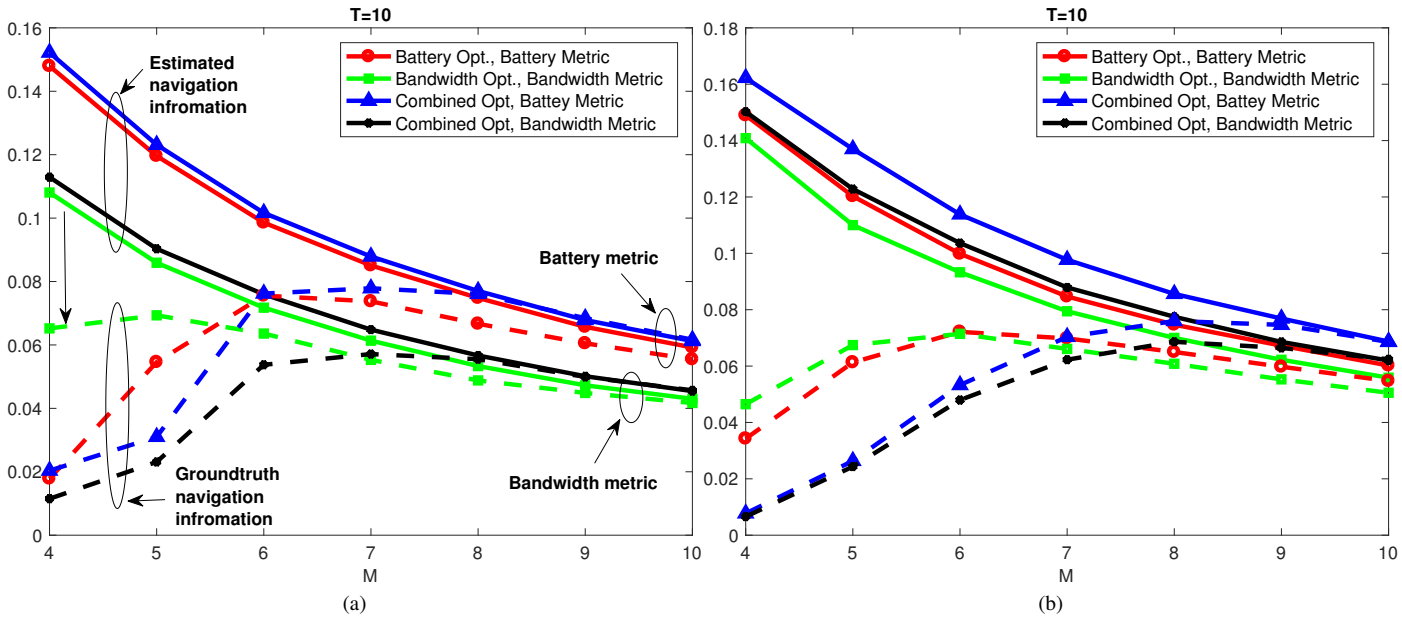


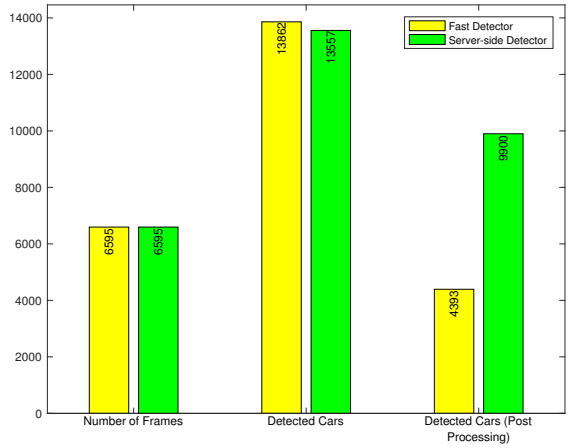
Fig. 6. Evaluation of the battery and bandwidth utilization metrics by solving the battery optimization problem (1)-(3), the bandwidth optimization problem (5)-(7), and the combined battery and bandwidth optimization problem (9)-(12).

A. *Visual Processing of Traffic Information*

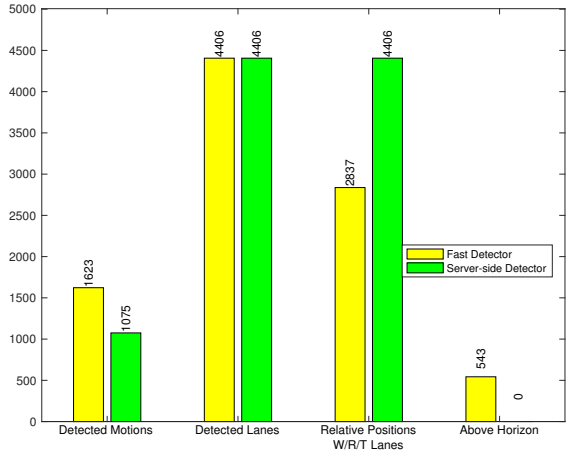
Haar-Like Cascade detector with AdaBoosting [33] is considered one of the fast detectors with a rate that can reach 50 frames per second because it deals with the integral image, and the learning procedure is performed through AdaBoosting which chooses a small number of features. In [34], a fast detector is used to classify the car from the background. In [23], the Cascade classifier is used similarly to alarm the driver if the car is closely tailgating the car in front of it [22]. The authors suggested some pre-processing of the data such as alignments of the cars in training set, and providing different version of the image in the original data set. Different other algorithms can be used for classifications and detection of cars i images. For example, in [35], a method combining temporal difference with and edge detector is used to detect cars. Such methods is widely proposed in the literature, and considered fast in terms of processing. However, these methods do not work properly with cars of different sizes (e.g., buses and trucks), and cause low accuracy of detection. Deep learning and neural networks resulted into exceptional performance in terms of localizing and detecting objects using traditional and Region-based convolutional neural networks (CNN and R-CNN) [30] [36] [37].

Lane detection has been a very active research topic in visual sensing of intelligent vehicles. The top-view or sometimes referred to as "bird's view" is a transformation method that is widely used in the literature for detecting the lanes on the road as in [38] [22] [39]. This method is formally referred to as, the inverse perspective mapping (IPM). In [22], the authors used the IPM procedure followed by a Hough transform. In [39], a robust and fast lane detector is proposed based on IPM, removing the outliers and reach excellent spline fitting using the well-known RANSAC algorithm. In [38], the authors used a combined IPM transformation and Hough transform line detector to detect the lanes. In [35], a modified Hough transform with a hardcoded search area is used for lane detection. A robust and fast line detector is proposed in [40] shows a potential to detect lines on roads.

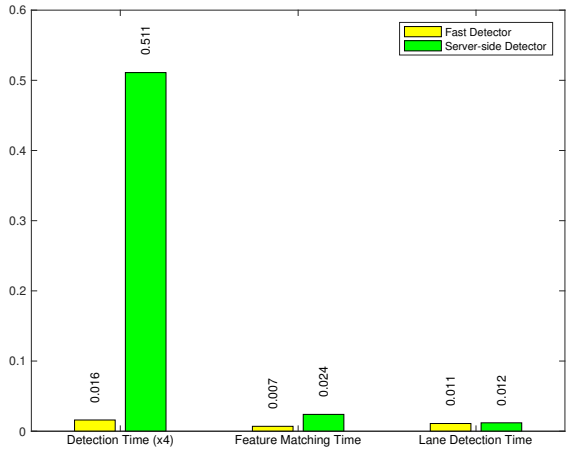
The distance between the vehicle ahead of the camera (or time to contact) has been studied in the literature. In [41], the authors proposed a robust, fast, and accurate estimation of time of contact between the car that has the camera within and the car in-front of it. In [22], the authors used the pinhole model to estimate the distance from the car ahead. In [42], a modified Harris corner detector is used to track vehicles motion. The famous feature detector used in the literature is Scale-Invariant Feature Transform (SIFT) [43]. A faster implementation version of SIFT is the Speeded-up Robust Features algorithm (SURF) [25]. It is also worth mentioning that combining other sensors with visual sensing could result in better understanding of the environment. A good source of information for visual processing on the roads is in [44]. The paper discusses many references and how they detect objects, lanes, motion using different machine learning algorithm, probabilistic methods, and using monocular and stereo-vision. A general architecture for video surveillance systems, namely, hierarchical and networked vehicle surveillance, is proposed where different used techniques in the literature are discussed [45]. The general architectural system is overviewed, but no performance evaluation is presented. A comparison between roadside (pole-mounted, stationary) and in-vehicle (mobile platforms) systems are presented in [46], but the authors focus on camera-based roadside monitoring systems, with special attention to omnidirectional setups. Another good source of information for safety analysis and the traffic behaviour at intersections with focus of visual sensing technology is in [47]. Different performance metrics are introduced, and are used to evaluate analyze some of the algorithms in the literature.



(a)

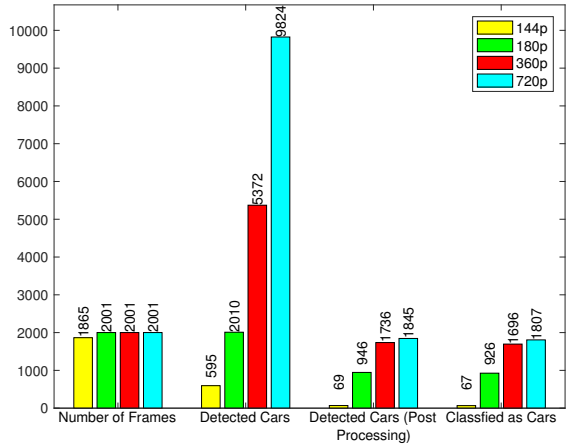


(b)

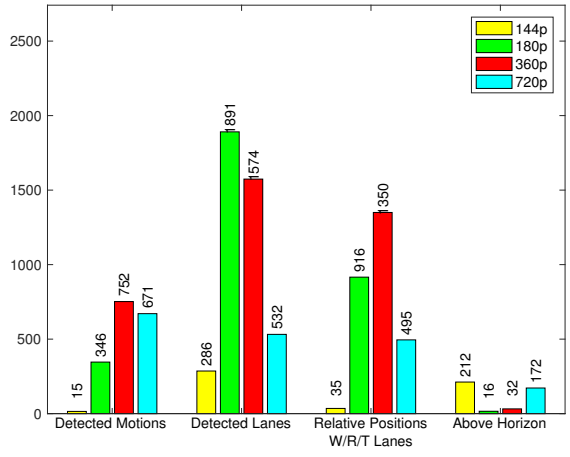


(c)

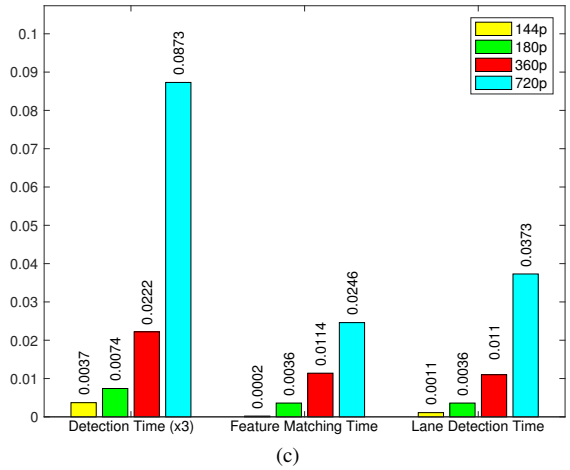
Fig. 7. Traffic violation metrics of the fast in-mobile and in-cloud detector using the testing video [16]. a) We compare the number of detected objects in the smartphone and the server, b) we illustrate the detected traffic violation within the smartphone and the server, and c) we compare the processing time for the different components of the system.



(a)



(b)



(c)

Fig. 8. Traffic violation metrics of the fast in-mobile and in-cloud detector using the testing video [16] with multiple resolutions. a) We compare the number of detected objects in the smartphone and the server, b) we illustrate the detected traffic violation within the smartphone and the server, and c) we compare the processing time for the different components of the system.

B. Communication Modes in Transportation Networks

Vehicles can communicate over the *dedicated short range communications* (DSRC) spectrum or the *cellular communication* infrastructure. The communication can occur in a regular cellular infrastructure mode, where the vehicles relay the information to the backend as a routing stage for the destination vehicle. Another option is that vehicles can directly

communicate in a device-to-device mode using 5G or over the DSRC spectrum. A more delay-tolerant transmission can occur via WiFi offloading. Communication between vehicles has been widely studied in the recent literature [48], [49], [50]. In [51], the authors proposed a method for optimizing the collection of visual data crowd sourced by vehicular networks. In [52], a solution to mitigate the communication channel congestion is proposed using compressive sampling of packets. Reliability of transmission and delay of packets reception are still considered open research areas in vehicular communications.

C. Applications of Connected Visual Processing in Transportation

Normal cars on the roads can be used as active or passive participants for crowd sourcing. They can capture images or videos [53], [51], or provide the traffic state on the road. Moreover, visual understanding of the driving scene is becoming a crucial part of autonomous vehicles [54]. Connected vehicles can be used to optimize vehicles routing and fuel consumption [55], or optimize traffic light operation [56]. Visual information of the roads can feed the algorithms with additional information of the road state, and the number of vehicles.

Safety is the top issue in driving, and visual processing can be an asset to complement the communication capability where the missed information can be crucial. Unlike most of the work that is currently focused on autonomous driving, a costly equipment that can be given to specific users such as transportation officers, we propose a unique solution that is low-cost, can be used for a variety of driving detection, and is pervasive in nature.

VI. Conclusion

This paper propose a novel solution that provides accurate and low-cost traffic violation detection. We design a framework that uses the current cameras in smartphones, and implemented a fast detector that works on the smartphones. The fast detector works with good accuracy. Therefore, we propose the use the deep CNN classifier to distinguish cars from backgrounds, which has a very high accuracy. After that, we process the videos on the servers-side using a more accurate detector that tolerate-delay. After detecting cars in different frames, we use feature matching following by an outlier detector algorithm to match the positions of the cars within the frames. The motion detector can classify the motion of the vehicle in different directions and relative to the lane markers. The sensory information of the phone such as GPS, gyroscope, compass, and accelerometer are used to detect the relative behaviour to the monitoring car. This can be an over- or under-speeding vehicle, a vehicle driving over the lane marker or switching lanes, etc. We demonstrated that the accuracy of the proposed system is directly related to the accuracy of the used computer vision algorithms. Moreover, the more data are used to train object detectors and classifier, the better becomes the accuracy. We also proposed a method to utilize the resources of the participants according to the limited battery and bandwidth of each participant, and the availability of the navigation information. We show that the navigation information is important for a lower number of participants, but loses importance as the number of participants increases. We also demonstrated that battery and bandwidth optimization can be combined, but that does not provide significant improvement unless the amount of resources have a large variance (i.e., the battery and bandwidth are not close in value on average). Finally, we conclude that our solution can be pervasive and can be a low-cost asset to any driving detection facility, and that we do not compete with self-driving vehicles, rather, the proposed solution can be integrated to them.

Acknowledgment

This author would like to thank the Deanship of Scientific Research at Umm Al-Qura University for the financial support of the project #17-COM-1-02-0002.

References

[1] S. Kaewunruen, J. M. Sussman, and A. Matsumoto, "Grand challenges in transportation and transit systems," *Frontiers in Built Environment*, vol. 2, p. 4, 2016. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fbuil.2016.00004

[2] "Traffic accidents occur every minute in ksa," http://www.arabnews.com/saudi-arabia/news/717596, accessed: 2019-03-15.

[3] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE communications surveys & tutorials*, vol. 13, no. 4, pp. 584–616, 2011.

[4] "Mobileye," http://www.mobileye.com/en-us/, accessed: 2019-03-01.

[5] "everdrive," https://www.everquote.com/everdrive/, accessed: 2019-05-01.

[6] "Caltech car dataset," http://www.vision.caltech.edu/html-files/archive.html, accessed: 2019-11-01.

[7] "Caltech car dataset – extended," http://www.robots.ox.ac.uk/~vgg/data/data-cats.html, accessed: 2019-11-01.

[8] "Caltech background dataset – extended," http://www.robots.ox.ac.uk/~vgg/data/data-cats.html, accessed: 2019-11-01.

[9] "Caltech road dataset," http://vasc.ri.cmu.edu/idb/images/road/, accessed: 2019-11-01.

[10] "Cars dataset," http://ai.stanford.edu/~jkrause/cars/car_dataset.html, accessed: 2019-11-01.

[11] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[12] "Karlsruhe dataset: Labeled objects (cars + pedestrians)," http://www.cvlibs.net/datasets/karlsruhe_objects/, accessed: 2019-11-01.

[13] A. Geiger, C. Wojek, and R. Urtasun, "Joint 3d estimation of objects and scene layout," in *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[14] "Youtube," https://www.youtube.com/watch?v=znlKpb2TDe8, accessed: 2019-11-01.

[15] "Youtube," https://www.youtube.com/watch?v=J_AHVG7mutA, accessed: 2019-11-01.

[16] "Youtube," https://www.youtube.com/watch?v=D5WxoZJsZpM, accessed: 2019-11-01.

[17] "Youtube," https://www.youtube.com/watch?v=9G4dGTy5Cyw, accessed: 2019-11-01.

[18] "Youtube," https://www.youtube.com/watch?v=p5T8RzYE60M, accessed: 2019-11-01.

[19] "Youtube," https://www.youtube.com/watch?v=hHfoLnCN2no, accessed: 2019-11-01.

[20] "Youtube," https://www.youtube.com/watch?v=xkvKV5wsYUE, accessed: 2019-11-01.

[21] "Youtube," https://www.youtube.com/watch?v=GceaIO5OmiU, accessed: 2019-11-01.

[22] C.-W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de Oca, Y. Cheng, M. Lin, L. Torresani, and A. T. Campbell, "Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones," in *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '13. New York, NY, USA: ACM, 2013, pp. 13–26. [Online]. Available: http://doi.acm.org/10.1145/2462456.2465428

[23] T. Wang, G. Cardone, A. Corradi, L. Torresani, and A. T. Campbell, "Walksafe: A pedestrian safety app for mobile phone users who walk and talk while crossing roads," in *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, ser. HotMobile '12. New York, NY, USA: ACM, 2012, pp. 5:1–5:6. [Online]. Available: http://doi.acm.org/10.1145/2162081.2162089

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[25] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[26] P. H. Torr and A. Zisserman, "Mlesac: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[27] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively trained deformable part models, release 5," http://people.cs.uchicago.edu/ rbg/latent-release5/.

[28] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[29] "clarifai," https://clarifai.com, accessed: 2017-03-01.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[31] "Arada systems," http://www.aradasystems.com, accessed: 2017-04-01.

[32] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[33] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[34] K. Chang, B. H. Oh, and K. S. Hong, "An implementation of smartphone-based driver assistance system using front and rear camera," in *2014 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2014, pp. 280–281.

[35] S. J. W. Tang, K. Y. Ng, B. H. Khoo, and J. Parkkinen, "Real-time lane detection and rear-end collision warning system on a mobile computing platform," in *2015 IEEE 39th Annual Computer Software and Applications Conference*, vol. 2, July 2015, pp. 563–568.

[36] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[37] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[38] C.-Y. Fang, W.-H. Hsu, C.-W. Ma, and S.-W. Chen, "A vision-based safety driver assistance system for motorcycles on a smartphone," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct 2014, pp. 328–333.

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

[40] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "On straight line segment detection," *Journal of Mathematical Imaging and Vision*, vol. 32, no. 3, p. 313, 2008. [Online]. Available: http://dx.doi.org/10.1007/s10851-008-0102-5

[41] B. K. P. Horn, Y. Fang, and I. Masaki, "Time to contact relative to a planar surface," in *2007 IEEE Intelligent Vehicles Symposium*, June 2007, pp. 68–74.

[42] Z. Di and D. He, "Forward collision warning system based on vehicle detection and tracking," in *2016 International Conference on Optoelectronics and Image Processing (ICOIP)*, June 2016, pp. 10–14.

[43] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. Ieee, 1999, pp. 1150–1157.

[44] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, Dec 2013.

[45] B. Tian, B. T. Morris, M. Tang, Y. Liu, Y. Yao, C. Gou, D. Shen, and S. Tang, "Hierarchical and networked vehicle surveillance in its: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 557–580, April 2015.

[46] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A survey of vision-based traffic monitoring of road intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2681–2698, Oct 2016.

[47] M. S. Shirazi and B. T. Morris, "Looking at intersections: A survey of intersection monitoring, behavior and safety analysis of recent studies," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 1, pp. 4–24, Jan 2017.

[48] W. Chen, *Vehicular communications and networks: Architectures, protocols, operation and deployment*. Elsevier, 2015.

[49] C. Campolo and A. Molinaro, "Multichannel communications in vehicular ad hoc networks: a survey," *IEEE Communications Magazine*, vol. 51, no. 5, pp. 158–169, 2013.

[50] W. Viriyasitavat, M. Boban, H.-M. Tsai, and A. Vasilakos, "Vehicular communications: Survey and challenges of channel and propagation models," *IEEE Vehicular Technology Magazine*, vol. 10, no. 2, pp. 55–66, 2015.

[51] W. Alasmary, H. Sadeghi, and S. Valaee, "Crowdsensing in vehicular sensor networks with limited channel capacity," in *2013 IEEE International Conference on Communications (ICC)*, June 2013, pp. 1833–1838.

[52] W. Alasmary and S. Valaee, "Compressive sensing based vehicle information recovery in vehicular networks," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, July 2013, pp. 700–705.

[53] M. Gerla, J.-T. Weng, and G. Pau, "Pics-on-wheels: Photo surveillance in the vehicular cloud," in *Computing, Networking and Communications (ICNC), 2013 International Conference on*. IEEE, 2013, pp. 1123–1127.

[54] S. Alletto, A. Palazzi, F. Solera, S. Calderara, and R. Cucchiara, "Dr(eye)ve: A dataset for attention-based tasks with applications to autonomous and assisted driving," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2016, pp. 54–60.

[55] M. Alsabaan, W. Alasmary, A. Albasir, and K. Naik, "Vehicular networks for a greener environment: A survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1372–1388, Third 2013.

[56] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): Methodology and large-scale application on downtown toronto," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1140–1150, Sept 2013.