Towards a Distraction-free Waze

Kevin Christensen
Carnegie Mellon University
kchrist1@andrew.cmu.edu

Christoph Mertz
Carnegie Mellon University
cmertz@andrew.cmu.edu

Padmanabhan Pillai
Intel Labs
padmanabhan.s.pillai@intel.com

Martial Hebert
Carnegie Mellon University
mhebert@andrew.cmu.edu

Mahadev Satyanarayanan
Carnegie Mellon University
satya@cs.cmu.edu

ABSTRACT

Real-time traffic monitoring has had widespread success via crowd-sourced GPS data. While drivers benefit from this low-level, low-latency road information, any high-level traffic data such as road closures and accidents currently have very high latency as such systems rely solely on human reporting. Increasing the detail and decreasing the latency of this information can have significant value. In this paper we explore this idea by using a camera along with an in-vehicle computer to run computer vision algorithms that continuously observe the road conditions in high-detail. Abnormalities are automatically reported via 4G LTE to a local server on the edge, which collects and stores the data, and relays updates to other vehicles inside its zone. In this paper we develop and test such a system, which we call *LiveMap*. We demonstrate its accuracy on detecting hazards and characterize the system latency achieved.

CCS CONCEPTS

• **Information systems** → **Mobile information processing systems**; *Sensor networks*; • **Networks** → **Application layer protocols**; *Mobile networks*; • **Computing methodologies** → **Object detection**; *Neural networks*.

KEYWORDS

Vehicular Systems; Automotive Systems; Edge Computing; Cloudlet; Cloud Computing; Cellular Wireless Networks; 4G LTE; 5G ; Distracted Driving; Crowd-sourcing; Neural Networks; Object Detection

ACM Reference Format:
Kevin Christensen, Christoph Mertz, Padmanabhan Pillai, Martial Hebert, and Mahadev Satyanarayanan. 2019. Towards a Distraction-free Waze. In *The 20th International Workshop on Mobile Computing Systems and Applications (HotMobile '19), February 27–28, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3301293.3302369

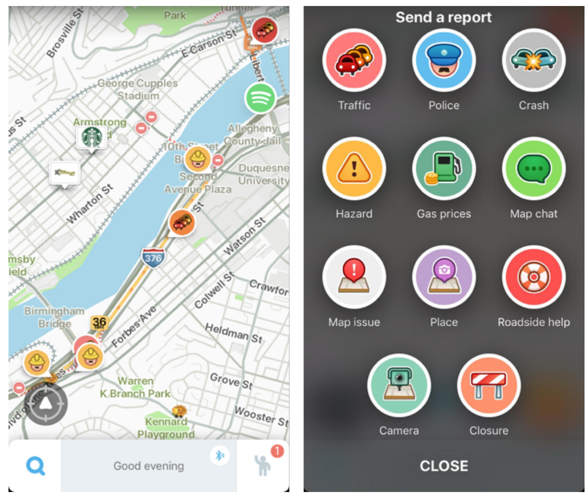

Figure 1: a) Left: A screenshot of Waze app with road events overlayed on GPS navigation map. b) Right: A screenshot of the Waze hazard reporting interface.

1 INTRODUCTION

Over 100-million active users benefit from Waze every month [17]. This crowd-sourced application allows drivers and passengers to report road events to a collecting entity that merges information and overlays it on GPS navigation maps via the mobile Waze app [4]. Figure 1 shows the Waze app interface.

In 2013 Google purchased Waze for 1.15 billion dollars [17], which is an indication of the perceived value for its service. Unfortunately the benefits of Waze come at the cost of user distraction, which is known to be a major source of traffic accidents [5][8]. Since the majority of vehicles have just the driver in the car with no passengers, Waze reports are typically made by a driver who incurs distraction in creating and submitting the Waze report. In short, the service is valuable but dangerous to not only the driver but also to nearby drivers, pedestrians, and bicyclists. With advances in computer vision and edge computing, we ask the question: "Can we have the benefits of Waze without user distraction?" Our approach to solve this problem utilizes an in-vehicle camera with an in-vehicle computer termed the Vehicle Cloudlet, to run computer vision algorithms to observe the road conditions. Abnormalities are then reported via 4G LTE to a local server on the edge, termed the Zone Cloudlet. The Zone Cloudlet synthesizes the data, stores it in a database, and notifies other Vehicle Cloudlets inside its zone of responsibility. The Zone Cloudlet is situated on the edge in order to improve bandwidth scalability and provide more localized control of user data and privacy, as well as for potential national security reasons to decentralize such information [15]. This is in contrast to Vehicle-to-Vehicle (V2V) Communication systems [16][21], which

HotMobile '19, February 27–28, 2019, Santa Cruz, CA, USA

ACM ISBN 978-1-4503-6273-3/19/02.
https://doi.org/10.1145/3301293.3302369

focus on addressing issues such as immediate collision detection and avoidance, as well as highway platooning. V2V communication exhibits additional security concerns, such as message accuracy and reliability, etc. The Zone Cloudlet in contrast can address such issues by vetting information before sharing it with other vehicles. Note that this pipeline would also enable autonomous vehicles to make reports as well since it does not require a human in the loop. In this paper we focus on the development of an architecture for such a system, and demonstrate it detecting hazards such as potholes, which are frequently found locally for testing.

In prior work [10], we have shown that a 4G LTE network is sufficient to support such a large-scale system with tens of thousands of participating vehicles, as demonstrated through simulations in SUMO [12] (Simulation of Urban Mobility). While that work focused on the scalability challenges of using LTE, in this work we focus on the architecture, implementation, and characterization of the automated hazard reporting system.

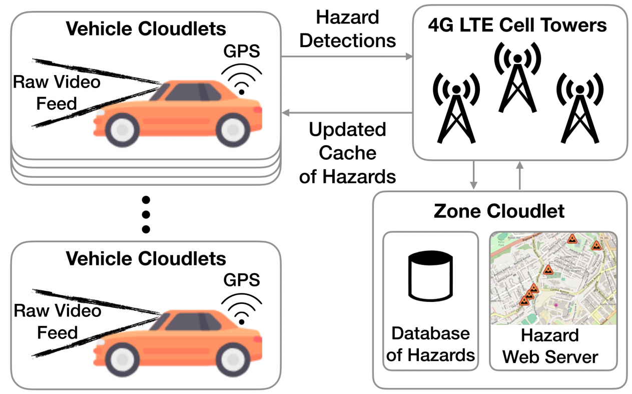

Figure 2: System Overview. Vehicle Cloudlets run hazard detection on live video feed, reporting any hazards to the Zone Cloudlet for dissemination to other vehicles and for display through a web interface.

2 BACKGROUND AND RELATED WORK

To the best of our knowledge, no previous work has attempted to create a Waze-like real-time data collection system for road monitoring without a human in the loop. From a broader perspective, there is a substantial amount of work relating to road condition monitoring [7][18][6][13]. However, none of these can handle the wide variations in hazard types and durations of events such as accidents, debris on the road, potholes, etc. Many of these systems focus on road infrastructure monitoring such as detecting potholes or general road health inspection.

A common approach for detecting potholes is to use an accelerometer with signal processing. The most notable of these, the Pothole Patrol [7], utilizes an accelerometer and on-board filtering to determine likely locations of a pothole. Such a system has several drawbacks, the most significant being that it requires the vehicle to physically run over the pothole for detection. This is harmful to both the vehicle and uncomfortable for the driver. For this reason, it is not uncommon for drivers to swerve in order to deliberately miss running over a pothole. Furthermore, it is limited to a very specific type of road monitoring — that is, whatever the car physically hits. A non-destructive alternative is to instead use image processing.

There has also been prior research in the field of vehicular distributed network system architectures. CarTel [11] has explored the development of such a system. They detail and prototype a system that utilizes vehicles to collect sensor data, store it locally, and prioritize the dissemination of sensor data to a local server. The main disadvantage of their system is that, as described, they have no significant on-board compute and cannot locally process data-rich media, such as images. Thus, their system is limited to selective transmission as the only approach to dealing with low-bandwidth situations. Furthermore, as it relies on opportunistically connecting to local Wi-Fi hotspots, it suffers from frequent loss of network connectivity. The main advantage of LiveMap is that it can process media rich sensor data such as video feed on-board the vehicle, and can greatly save bandwidth by transmitting only distilled, interesting data, thus enabling effective use of relatively low bandwidth, but ubiquitous 4G LTE networks.

A few companies offer commercial products that are relevant to our research. Waze [4] utilizes user input in order to add detailed input to their maps. As previously discussed, the main shortcoming of such a system is that it creates dangerous situations by distracting the driver, and tends to incur high latencies and unreliable updates due to reliance on human reporting. In contrast, LiveMap is safe, autotmated, and real-time. Another relevant company is Roadbotics [3]. Roadbotics uses cameras to capture video data of roads from the windshield, and uploads it to a server where machine learning algorithms score road conditions over 10-foot intervals. In comparison to Roadbotics, LiveMap operates in real-time, is not specific to detecting road quality, and can generalize to any hazards detectable through computer vision.

3 SYSTEM ARCHITECTURE

LiveMap is designed to scale to a large number of vehicles [10], and as such it is critical that our system scales with increased bandwidth such that we do not saturate the 4G LTE network. Sending video streams from every vehicle would quickly prove to be intractable, and moving the Zone Cloudlet to the cloud would also degrade the scalability of our system. Instead, we harness compute ability both in the vehicle and in the infrastructure on the edge to reduce network use.

The key idea behind LiveMap is that it performs the heavy video analytics computations on-board the vehicle. Each participating vehicle is required to have an on-board camera and GPS module, as well as a computer, called the Vehicle Cloudlet. As the vehicle drives around, it runs a Convolutional Neural Network (CNN) to detect various hazards. When a hazard is detected with a confidence level greater than a threshold, the Vehicle Cloudlet reports it to the Zone Cloudlet via a message containing the GPS coordinates, the image with bounding box(es) around the hazard(s), as well as the timestamp and other metadata pertaining to the drive. The Zone Cloudlet is responsible for handling the incoming data, storing it in a database, and notifying the vehicles in the vicinity of the hazard. See Figure 2 for a high-level overview of LiveMap.

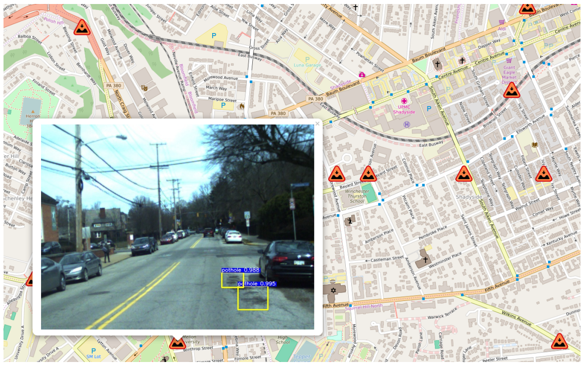

Figure 3: Map of Hazards. Icons for each hazard appear real-time. Clicking on each icon displays more hazard information.

There are several challenges associated with choosing when and how often to report hazards. One such challenge is in avoiding identical reports. Imagine a vehicle stuck in traffic behind an accident; until the accident clears the system will repeatedly detect the same accident. Now imagine there are many cars that view this accident. This single event may be reported thousands of times. Such careless report sending policies would quickly saturate the network with useless duplicate data.

To address this issue of duplicating data, we employ a policy that if a vehicle detects a hazard of the same type in close proximity to one that has previously been reported, it will not send the report. This achieves the desired result of saving bandwidth by avoiding duplicate reports, with the small cost of perhaps missing a few instances of the same class. We argue that this is indeed an acceptable tradeoff with the following example: Suppose several objects fall from a truck in one area on the highway, creating hazardous debris. A car driving by will detect multiple frames of debris, but will only send one report. In general, this single report for multiple related hazards should suffice. Another such issue that complicates this design decision is the issue of hazard removal. Different types of hazards may have various temporal lifespans. For instance, some hazards like car accidents are usually cleaned up and removed within a matter of hours, while potholes can exist for several months. At the Zone Cloudlet, detecting when a hazard is no longer present can be tricky. A naive solution would be to wait until reports of the hazard stop arriving. This idea is flawed due to our previous design decision to not report duplicate hazards. It is difficult to know if a hazard has "expired" or if the duplicate elimination policy is preventing further reports. It is possible the Zone Cloudlet may never hear again about a long-lasting hazard.

To address this second issue, we use a polling scheme. In this polling scheme, the Zone Cloudlet occasionally sends a message to vehicles near a previously-reported hazard asking them if they still see the hazard, and optionally whether or not to send an image. The Vehicle Cloudlets then send a "yes" or "no" reply back after validating the continued presence of the hazard, minimizing the bandwidth consumed for verifying the hazard presence. The rate at which such verification polls are sent would be inversely proportional to the expected duration of the hazard, based on the mean or median duration of the hazard class type. This can be further optimized as the system collects more data and can generate more accurate predictions of when best to poll while minimizing network usage.

4 IMPLEMENTATION

4.1 Zone Cloudlet

The Zone Cloudlet is situated on the edge, and in our experiments is implemented on a server on the Carnegie Mellon University (CMU) campus. It has a number of responsibilities, which can be classified into two main functionalities: 1) hazard message operations and 2) web server operations. Within the class of hazard message operations, the Zone Cloudlet must accept and handle incoming hazard reports from vehicles, and transmit update data to vehicles. See Figure 4.

When the Zone Cloudlet receives a hazard report, it queries its database for any matching hazards previously reported within the tolerance of the GPS module (typically around 3-5 meters). If no active matches are found (i.e., this is not a duplicate report), it adds the hazard to its database. It then sends a message to all vehicles in its area of responsibility with the GPS coordinates of the hazard, and an image of the hazard with a bounding box around it.

For our implementation, we opted to use the Message Queuing Telemetry Transport (MQTT) protocol, an ISO standard built on top of TCP/IP [1]. MQTT is a publisher-subscriber-based messaging protocol intended for the "Internet of Things," and is designed around the idea that machine-to-machine communication will have limited network bandwidth and will suffer from intermittent connectivity. This fits our application requirements, since moving vehicles will invariably be in a dead zone at one time or another, and the 4G LTE network has limited bandwidth. For a Vehicle Cloudlet to receive updates from the Zone Cloudlet, it simply has to look up its GPS coordinates, find the nearest Zone Cloudlet, and subscribe to the updates being published by the Zone Cloudlet. In order to avoid major safety concerns, we limit communication in LiveMap as follows: the Zone Cloudlet is the message broker, and the Vehicle Cloudlets are the clients; no vehicle-to-vehicle communication is performed, and all messages to the Vehicle Cloudlets must come from the Zone Cloudlet.

The second class of operations has to do with the presentation of hazard information, which is done via a web-based interface. The Zone Cloudlet doubles as a web server, and actively delivers hazard information on a map overlay to connected web clients. When a client, say a city official, connects to the web server, the Zone Cloudlet sends all present hazards to the client, which are displayed on the map as icons. The user can then click on the icon and have additional information displayed, such as GPS coordinates and the image with bounding box of the detected hazard. When a new hazard is added to the database, a notification is sent via web sockets to all connected clients in real-time. An example screenshot of the web-based hazard display is shown in Figure 3. We use Leaflet [2] as our map serving framework, and Node.js to dynamically deliver content. This web server display can effectively serve as a quality control measure. Since the details of all hazards can be displayed on the map as images with bounding boxes encapsulating the hazards, a city official can easily verify the accuracy of hazard detections with a click on each hazard icon. This provides an interface for human oversight of the system, letting an official reject any false positives before notifying the appropriate response teams, for example.

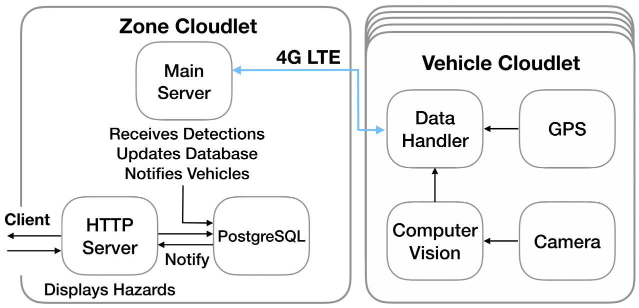

Figure 4: System Architecture. The Zone Cloudlet handles incoming hazard reports and notifies other vehicles. It also runs an HTTP server that displays a map of detected hazards.

4.2 Vehicle Cloudlet

The Vehicle Cloudlet performs image processing to find hazards. It utilizes a Convolutional Neural Network (CNN) to perform object detection to identify road abnormalities, and then sends a message to the Zone Cloudlet with accompanying data. When it detects a hazard, it checks its local database of current hazards for a nearby hazard of the same type. If it doesn't find any, it adds it to its database and sends a message to the Zone Cloudlet. In doing this it avoids repeatedly sending notifications of known hazards. When the Vehicle Cloudlet receives a hazard notification from the Zone Cloudlet, it adds it to its database. In previous work [10] we have shown detection of deer on the roadside (https://youtu.be/_GrP42359z8). We have also recently demonstrated the system detecting traffic cones (https://youtu.be/TToOb2rTNZU), which often signal lane closures. The list of objects that can be detected is extensible: new object classes can be added to the system by providing a classifier trained on the object data. Some other items that may be useful to detect include road closure signs, road debris, construction equipment, and accidents. New detectors can be incrementally added as they become available.

Not all types of hazards are of equal importance. Vehicle Cloudlets would contain a list of hazard types with a possibly dynamic importance ranking for each hazard type. Hazards that are ranked with high importance would be sent immediately, while those that are less serious can be deferred until the Vehicle Cloudlet is connected to WI-FI in a garage, for example. This feature would save bandwidth without sacrificing completeness.

The accuracy and recall capability of the sensing is a function of compute capability, which is a function of cost. We explore the trade-off between accuracy/recall and cost by experimenting with two different designs and implementations for the Vehicle Cloudlet. One configuration is a powerful server with state-of-the-art compute capability but can run reliably off of a car alternator. We call this the Big Vehicle Cloudlet (BVC). It can afford to use a more computationally expensive and memory intensive CNN architecture for detection, employing dual GPUs with high bandwidth and large memory. The second option uses a mobile phone as the Vehicle Cloudlet, which has significantly less compute capability and memory, but is an order of magnitude lower in cost. We term this the Small Vehicle Cloudlet (SVC). We outline both implementations below and highlight the key differences between them.

Table 1: Latency Measurements in ms

Config	Detection	Transmission	End-to-end
BVC	38.6 (3.1)	205.6 (50.2)	244.2 (50.3)
SVC+ZC	391.6 (67.1)		597.2 (83.7)

BVC: Big Vehicle Cloudlet, SVC: Small Vehicle Cloudlet
ZC: Zone Cloudlet, std. deviation in parentheses
Note that the latency for BVC is per GPU

4.3 Big Vehicle Cloudlet

The first system we test is a ruggedized server, configured with 2 Intel® Xeon® Processors, 2 Nvidia Tesla V100 GPUs, and a liquid cooling system. This system configuration can afford to run a large CNN model with a large number of weights, which is both memory and compute intensive. On this system, we run Faster R-CNN [14] as our object detector, which provides state-of-the-art accuracy, but is computationally demanding. The image processing is run on both GPUs independently in order to double the processing frame rate. Each GPU has a copy of the CNN weights and can run inference on an individual image independently. The output image is the original image overlaid with bounding boxes indicating where a hazard was detected. If a hazard is detected and is not a duplicate, the Vehicle Cloudlet prepares a message and sends it to the Zone Cloudlet. If the image is not interesting and no hazards were detected, the system simply discards the image.

This setup provides the best scalability, as we have moved all of the compute to the vehicle, and the aggregate compute capability will scale with number of vehicles. Furthermore, the BVC is well-positioned to address privacy concerns that arise from recording people in such video feeds. The BVC has enough compute to denature images as done in [19].

4.4 Small Vehicle Cloudlet

The second system we test uses a smartphone-class device as the Vehicle Cloudlet. As this platform is not capable of running the large CNN used for hazard detection, we employ the early discard method proposed by [20] that uses lightweight computations to selectively send only the interesting images to the Zone Cloudlet, which would then run the hazard detection algorithm. The small vehicle cloudlet is limited to making send-don't send decisions using a small and simple neural network model. The expensive hazard detection algorithm is then run on the Zone Cloudlet. This significantly reduces costs of the vehicles, but comes at the expense of scalability, since we move the hazard detection to the centralized Zone Cloudlet.

We implement the Small Vehicle Cloudlet using a Nexus 6 smartphone, and run MobileNet [9] as the image classifier. This has significantly lower computational requirements than Faster-RCNN, and can process roughly three frames per second on this platform.

5 EXPERIMENTAL RESULTS

There are three aspects of performance that we consider when evaluating LiveMap. The first is the end-to-end system latency for LiveMap given a new hazard. This is the time it takes from the point at which a vehicle detects a hazard, sends it to the Zone Cloudlet, and the Zone Cloudlet sends out the newly captured data to nearby

Figure 5: Examples of pothole detections with bounding boxes.

Table 2: Detection Results

Config	FPS	mAP	Event Recall	Avg. Mbps
BVC	30	52.7	92.3%	0.46
SVC+ZC	2.8	52.7*	84.6%	0.91

BVC: Big Vehicle Cloudlet, SVC: Small Vehicle Cloudlet, ZC: Zone Cloudlet, *Zone Cloudlet only

vehicles. The second evaluation criterion is the hazard detection accuracy and recall. Ideally we want high accuracy (quality detections with a high ratio of true positives among all detections), as well as high recall (most hazards are actually reported). Lastly, we need to quantify the bandwidth savings by moving compute into the vehicle.

System latency can be further broken down into two categories: 1) detection latency, or the time it takes to process a single image, and 2) message round-trip latency, or the time it takes to send a message to the Zone Cloudlet and receive an acknowledgement. The detection latency is dependent on the Vehicle Cloudlet server, while the transmission latency is the same for both configurations since they will both be using 4G LTE for transmission. Note that for the Small Vehicle Cloudlet, the detection latency includes both the local image classification, which gives a send-don't send result, as well as the actual hazard detection code, which places bounding boxes around hazards, running on the Zone Cloudlet. To test the detection latency, we simply record the time the system takes to process each frame over a one minute interval. See Table 1 for Transmission Latency and Detection Latency results. Note that the Big Vehicle Cloudlet has two GPUs, but Table 1 shows latency *per* GPU. On an end-to-end basis, using the Big Vehicle Cloudlet incurs less than half of the latency as using the Small Vehicle Cloudlet.

The camera frame rate is 30 fps, therefore the ideal processing latency is below 33 ms to achieve real-time performance. Per-frame processing times greater than 33 ms would imply that frames are dropped, or not processed. By utilizing 2 GPUs and alternating frames assigned to each, the Big Vehicle Cloudlet can avoid dropping any frames. While it is great to process every frame, it is often not necessary in order to detect hazards.

Our second metric attempts to quantify how well the system actually detects road hazards. We consider this in two different ways. For the Big Vehicle Cloudlet we measure the mean Average Precision (mAP), which is a common metric for evaluating bounding-box-based object detection algorithms. We use the standard mAP_{50} which defines a correct detection if the intersection over union of the detected and the ground truth bounding boxes is greater than 50%. Note that we use this metric to evaluate BVC's detection algorithm, which is also run on the Zone Cloudlet for the SVC configuration. We obtain this metric over all test images.

Frequently a hazard will be encountered more than in just one frame. In fact, we expect to encounter any given hazard in multiple frames. Even if detection failed in one frame, the system may still be able to identify the hazard in another one. To address this, we employ an event-level recall metric, which we define as the number of distinct hazards correctly identified over the total number of hazards. For the Big Vehicle Cloudlet we filter out duplicate detections based on GPS location. If we detect a hazard of the same class in the same location, we can filter out the message as it was likely the same instance of the hazard previously detected. Note that we cannot utilize this for the Small Vehicle Cloudlet, as it simply categorizes the image as interesting or not interesting, and furthermore runs at a much slower frame rate.

Deep Neural Networks require large amounts of annotated data for training. For our prototype, it was not feasible to collect a large training set of accidents or debris on the road. Rather, we focused our proof-of-concept on detecting hazards for which we could collect data, namely potholes. We annotated approximately 3,000 pothole images to train our detector from a set of set of driving videos we collected. Data collected consists of footage in the greater Pittsburgh region across various lighting and weather conditions, as well as from various viewpoints. We kept aside a portion of the data for testing. We show the metrics in Table 2 and example positive detections in Figure 5. Due to the limited processing rate of the SVC configuration, many frames are dropped. This reduces its consumed bandwidth, but also reduces its event recall.

Finally, the third metric is the average bandwidth saved by utilizing compute in-vehicle. We run both the Big and Small Vehicle Cloudlets on the same recorded driving video, and record the amount of data transferred over TCP. We compare these to a third baseline option, in which the compressed H.264 video is streamed to a central server for processing. Figure 6 summarizes our results. Here, the video stream rate of the baseline is plotted as a red line. The GPS-based duplicate suppression is heavily dependent on the speed of the vehicle, and the rate at which we encounter hazards. Therefore, we test the Big Vehicle Cloudlet with several different radii parameters for redundant hazard checking, ranging from 1 meter to 25 meters. We show the theoretical best detector as the green star in the bottom right for reference, which exhibits perfect accuracy and recall, as well as the lowest possible bandwidth (i.e., each unique hazard is reported exactly once). Note that the consumed bandwidth is inversely proportional to the detection accuracy.

We can see that the Big Vehicle Cloudlet performs the best in terms of recall with GPS filtering of radii 1m, 3m, and 5m. Using a larger GPS filtering radius decreases bandwidth consumed at the cost

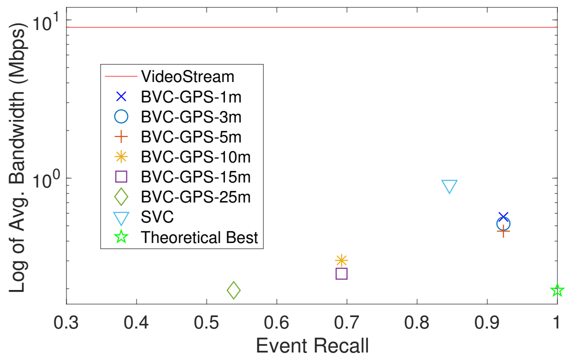

Figure 6: Bandwidth uplink in Mbps. Y-axis shown in log scale.

of hazard recall. The Small Vehicle Cloudlet provides a reasonable compromise between the two. The SVC requires more bandwidth since it uses a "filtering out" approach, and sends the image without knowing what the predicted class type is due to its limited memory and compute capabilities. We cannot use such spatial filter strategies because the SVC does not know what class of hazard was just detected. This is the main drawback of such a filtering out approach, and it therefore tends to send images more frequently on average. Additionally, since it operates at a low frame rate, it may be susceptible to missing hazards that are only visible briefly, a situation not reflected by these experiments. Overall, moving compute inside the vehicle reduces the average bandwidth consumed by around 95% compared to the baseline. Streaming the data costs nearly 9 Mbps uplink while our system used less than 0.5 Mbps with 5 meter GPS filtering. Extrapolating from this, if the average vehicle is driven for 1 hour a day, over the course of a month streaming video would require approximately 121 GB per vehicle, whereas our method would consume only 6.3 GB.

In Figure 6 we can see that even the theoretical "perfect detector," which exhibits perfect precision and recall, still sends a significant amount of data detecting potholes. Further restricting the report rate for non-urgent hazards would further reduce this number as well, as potholes in some cities may be categorized as non-urgent and can be transmitted when connected to WI-FI.

6 CONCLUSION

We have proposed a system architecture, LiveMap, that automates the detection and reporting of road hazard information utilizing in-vehicle compute and recent advances in computer vision. We have built and demonstrated a prototype system using both powerful and modest in-vehicle computers coupled with edge computing services. Both variants are able to detect and report potholes with no human involvement. Furthermore, we reduce the bandwidth consumed by such a system by over twenty-fold compared to video streaming to the cloud for processing. Future work includes expanding the types of hazards we can detect as well as developing reporting protocols to allow prioritization of hazard classes, and efficient dissemination of collected hazard information. In order to address privacy concerns, we plan on utilizing the compute in the BVC to selectively blur license plates and faces in images. We plan on adding such functionality to the first production release of LiveMap. Another future goal is to develop algorithms that are able to reliably fuse together reports of the same hazard from different vehicles that are off due to GPS inaccuracies and localization inaccuracies, and yet can determine that perhaps a line of traffic cones is signaling a road closure. Lastly, there is value in being able to detect street-view changes, not just hazards. Such examples include detecting new or removed buildings, road signs, and more. We plan on investigating these aspects in future versions of LiveMap.

ACKNOWLEDGMENTS

We thank Tom Eiszler and John Kozar for their technical assistance. We also thank our shepherd, Aruna Balasubramanian, and the anonymous reviewers for their guidance in strengthening this paper. This research was supported in part by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051 and by the National Science Foundation (NSF) under grant number CNS-1518865. Additional support was provided by Intel, Vodafone, Deutsche Telekom, Verizon, Crown Castle, NTT, and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view(s) of their employers or the above-mentioned funding sources.

REFERENCES

[1] Eclipse paho. https://www.eclipse.org/paho/.
[2] Leaflet. https://leafletjs.com/.
[3] Roadbotics. https://www.roadbotics.com/, Accessed: 2018-09-17.
[4] Waze. https://www.waze.com/, Accessed: 2018-09-17.
[5] N. E. Boudette. Biggest spike in traffic deaths in 50 years? blame apps.
[6] D. B. A. Chacra and J. S. Zelek. Fully automated road defect detection using street view images. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pages 353–360, May 2017.
[7] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, MobiSys '08, pages 29–39, New York, NY, USA, 2008. ACM.
[8] L. L. Firm. Is the waze driving app too distracting?
[9] 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. *CoRR*, abs/1704.04861, 2017.
[10] W. Hu, Z. Feng, Z. Chen, J. Harkes, P. Pillai, and M. Satyanarayanan. Live synthesis of vehicle-sourced data over 4G LTE. In *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '17, pages 161–170, New York, NY, USA, 2017. ACM.
[11] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *4th ACM SenSys*, Boulder, CO, November 2006.
[12] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, December 2012.
[13] M. G. N. Angelini, J. Brache and G. Shevlin. Gps coordinate pothole mapping. Technical report, Worchester Polytechni, 2006.
[14] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.
[15] M. Satyanarayanan. Edge computing for situational awareness. In *2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pages 1–6, June 2017.
[16] M. L. Sichitiu and M. Kihl. Inter-vehicle communication systems: a survey. *IEEE Communications Surveys & Tutorials*, 10(2), 2008.
[17] C. Smith. 14 interesting waze statistics and facts (june 2018).
[18] S. Varadharajan, S. Jose, K. Sharma, L. Wander, and C. Mertz. Vision for road inspection. In *IEEE Winter Conference on Applications of Computer Vision*, pages 115–122, March 2014.
[19] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan. A scalable and privacy-aware iot service for live video analytics. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, MMSys'17, pages 38–49, New York, NY, USA, 2017. ACM.
[20] J. Wang, Z. Feng, Z. Chen, S. George, M. Bala, P. Pillai, S.-W. Yang, and M. Satyanarayanan. Bandwidth-efficient live video analytics for drones via edge computing. In *SEC '18: Proceedings of the Third ACM/IEEE Symposium on Edge Computing*, New York, NY, USA, 2018. ACM.
[21] F. Ye, M. Adams, and S. Roy. V2v wireless communication protocol for rear-end collision avoidance on highways. In *ICC Workshops - 2008 IEEE International Conference on Communications Workshops*, pages 375–379, May 2008.