Real-time Traffic Management Model using GPU-enabled Edge Devices

M. Mazhar Rathore[1], Yaser Jararweh[2], Hojae Son[3], Anand Paul*, [3]
[1]*Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Doha Qatar*
[2]*Duquesne University, PA, USA*
[3]*The School of Computer Science and Engineering, Kyungpook National University, Daegu, Korea*
rathoremazhar@gmail.com, jararwehy@duq.edu, sonhj07@gmail.com, *paul.editor@gmail.com

***Abstract*—Auto management and controlling road traffic while identifying abnormal driving behavior is one of the key challenges faced by the traffic authorities. In most of the cities, the traffic violations are detected manually by placing sergeants at various regions on the road. Placing sergeants is not economical and does not cover all the metropolitan area. Only in modern countries, traffic authorities have developed systems that use static road cameras to monitor real-time city traffic for identification of major traffic violations. However, these cameras just cover limited areas of the cities, such as, intersections, signals, roundabouts, and main streets. Therefore, in this paper, we have proposed a real-time traffic violation detection model by using vehicular camera along with the edge device in order to control and manage the road traffic. The edge device is equipped with the graphics processing unit (GPU), deployed inside the vehicle, and directly attached to the vehicle camera. The camera monitors every vehicle ahead, whereas, the edge device identifies the suspected driving violation. As a use case, we have tested our model by considering a wrong U-turn as a traffic violation. We designed a wrong U-turn detection algorithm and deployed it on the GPU-enabled edge device. In order to evaluate the feasibility of the system, we considered the efficiency measurements corresponding to the video generation rate and data size. The results show that the system is able to identify violations far faster than the video generation time.**

Keywords—Smart Transportation; IoT; Smart City, Edge Computing

I. INTRODUCTION

The metropolitan population is growing day by day. People prefer to move from villages to cities due to the obvious advantages provided by the government in cities. One of the estimation says that in 2050, 70% share of the overall population will belong to cities [1]. Eventually, this growing city population has a major effect on the city traffic system and introduces many challenges to the authorities to control and manage road traffic. Thus, this makes impossible and uneconomical for authorities to control the traffic manually by placing sergeants everywhere in the cities. Also, due to the human error factor, it increases the possibility of mishandling by the police to the inhabitant while identifying the traffic violations. Therefore, to avoid human intervention, modern cities started the use of computer-based automatic system to detect violations by deploying cameras across the roads. This way they made their cities smarter and intelligent by deploying decision-making systems in the transportation sector.

Auto detection of traffic violations through cyber physical system (CPS) has a major positive impact on the overall smart city functions, especially in controlling and managing the road traffic. CPS infrastructure also used in many other application domains to improve the services of smart city, including air-pollution control, city noise reduction, and surveillance system [2]. However, among all these domains, the smart transportation is the key application that typically used by the professionals when they talk about smart city. Smart transportation not only manages the city traffic intelligently, but also provides the foundation for reduction in pollution, citizens' facilitation, rapid economic development, and much more. Many research outcomes have been accomplished in the field of smart transportation, as a project of smart city. Ondřej Přibyl [3] developed an objective function that is based on entropy measurement for the smart and intelligent management of traffic. Similarly, to establish a smart city, Fenghua Zhu et al., [4] designed a system called parallel transportation management and control systems (PTMS). They introduced social signal and traffic, intelligent transport system cloud services, agent-based traffic control, and transportation knowledge automation. Also, Ding, Yang, Chi, and Guo considered the spatial data of pedestrians, cyclists, and vehicles, which is taken from GPS devices and introduced a parallel spatio-temporal database model [5] to enable smart transportation.

All the aforementioned approaches do not deal with the real-time aspects of the smart traffic monitoring for the detection of traffic violations. The entire city roads cannot be monitored by placing sergeants or road cameras everywhere. The only possibility to observe each of the vehicle's activity on the road is to use other vehicle as an authority's spy. This can only be done through the use of vehicle camera to keep looking other neighboring vehicles for traffic violations. However, transmitting the overwhelming amount of video data from thousands of vehicles (as, each vehicle is continuously producing a video) to the traffic authority (or servers) for violation detection consumes a large portion of the bandwidth. Also, it is a challenging task to manage and process such big amount of video data by a central server [1, 6, 7]. Therefore, it is a need of the traffic authority to have an auto-violation detection system that uses vehicle camera as a sergeant's eye and intelligent fast device as his mind and avoid transmitting large amount of video data to the server.

In this decade, the GPU gain its popularity as a powerful co-processor, because of its parallel and high-speed processing capability for both video and general purpose computing. It has massive parallel computing ability than CPU in terms of number of operations performed per second [8]. Also, the CUDA and

978-1-7281-1796-6/19/$31.00 ©2019 IEEE

Brook+ programing frameworks created by NVidia and Stanford University respectively, have greatly improved its programmability for general purpose computing. The GPU is an expedient co-processor for parallel processing, especially when working with videos for decision-making. Thus, to deal with the above-mentioned processing challenges, the GPU is a good candidate.

Keeping in mind the traffic authority's need of auto-detection of traffic violations, the above-mentioned challenges, and the potential of GPUs, in this paper, we proposed a real-time traffic violation detection model by using vehicular camera along with the edge device in order to control and manage the road traffic. The edge device is equipped with the graphics processing unit (GPU), deployed inside the vehicle, and directly attached to the vehicle camera. The camera continuously monitors every vehicle ahead and sends the video to the edge device, which identifies the suspected driving violation. If any suspected activity is detected, the edge device transmits the reference video to the cloud (can be traffic authority server). The cloud server performs the deep analysis on the video and generates an alert to the authority in case the violation is committed. In such scenario, the device only sends the video data when any violation is suspected, thus, the bandwidth is saved. As a use case, we have tested our model by considering a wrong U-turn as a traffic violation. We also designed a wrong U-turn detection algorithm and deployed it on the GPU-enabled edge device. In order to evaluate the feasibility of the system, we considered the efficiency measurements corresponding to the video generation rate and data size. The results show that the system is able to identify violation far faster than the video generation time.

The rest of the paper is structured as follows. Section II, described the worked done in the related fields, Section III presented the proposed system model. The system implementation details given and the evaluation is performed in section IV. Whereas, Section V concluded the article.

II. Literature and Related Work

The vehicle detection is one of the key operations required by the proposed system. The need for performing accurate and real-time vehicle detection has sparked significant research. Even though several researches have already been done, but still it is a highly significant problem. To detect vehicles from pneumatic appearance, a dynamic Bayesian-network with color modal has been fashioned by cheng et al. [9] that gained dramatic achievements on momentous datasets. Many features are used by researchers [10-11] for vehicle detection; for example, Histogram of Oriented Gradients (HOG), Haar-like Features, and Local Binary Pattern, that revealed optimistic outcomes. An effective feature-selection to produce the vehicle's structural-feature has been employed by Kembhavi et al. in [12, 13,14]. They used the approaches based on Haar-like features and HoG features with Ada-Boost classifier for vehicle detection. Therefore, the handcraft features separate the vehicles and background even in the intricate atmosphere. Chen et al. introduced a method for vehicles detection by combining sparse representation and super pixel segmentation in [15]. Similarly, the segmentation is also performed by Fitzgerald and Wills on high-resolution images in the multicore environment [16]. Recently the methods based on machine learning are gaining keen attention for vehicle detection. An approach has been introduced by wang et al. [17] is based on sparse representation and multi-order features descriptor. By using convolution neural network, the satellite images of vehicles has been detected in [18] to achieve the excellent result in vehicle detection in aerial images. The approaches had been developed from shallow learning based features with discrimination.

Likewise, a bunch of publications are there for lane detection. A paper [19] presents a survey of many different kinds of lane detection systems in detail. There are many others efficient, excellent, and ideal methods were proposed and also implemented on many platforms [20-23]. For example, a single camera is used for detection of lanes on roads in [21]. This system detects lanes and boundary by using IPM, segmentation of colors and RANSAC fitting. In [22], for image transformation, researchers did the integration of IPM, for removal of outlier, they used RANSAC, and for prediction and tracking, they conducted Kalman filtering while detecting lanes and generating ground truths. In [23], for different rounded or curved lane marking, they predicted the vanishing point by linking V-disparity image. In [24], a method was proposed a few years ago based on single camera for autonomous vehicles.

Static roadside cameras have been used by scholars for traffic analysis while detecting vehicles on roads. A group of researchers applied the Optical Flow-based Transfer Learning (OFTL) for the detection of Traffic Rule Violation in Traffic Video Surveillance [25]. The same group devised a system for efficient handling of highway and urban traffic [26]. However, this system faced two significant disadvantages; 1) the camera model tuning was complex and manual, 2) fails to handle sudden variations in the background. These shortcomings were successfully addressed by an independent group of researchers through the application of predefined violation and trajectory feature for the detection of anomalies [27]. Similarly, another anomaly detection system was proposed by Lili Cui et al. [28]. In this system, three extracted trajectories were used to perform the behavior modelling. In addition, abnormal patterns are detected by Benezeth et al., [29]. He used co-occurrence matrix for two motion labels, as a prospective process while working with the Markov random field (MRF) model. Despite its broad applications, this model faces some disadvantages; 1) the trajectory extraction is very complicated, 2) it fails to describe a stopped or partially moving vehicle. A multi-feature library script was constructed by another group of researchers for normal event reconstruction [30]. The irrelevant scenes were all marked as random incidents. These resulted in memory space complexity and high computational time. Collins et al. [31] were able to categorize the detected objects as human, a group of human and vehicle. Song and Lee [32] used Optical Flow Analysis by a static road cameras for the detection of illegal U-turns. However, this system is just specific for one location where the camera is installed.

Even though there are systems that detect vehicles, road lanes, and traffic behaviors at a limited level through cameras

installed on specific location. But, these systems are limited to one region where the camera is installed. In such situation, traffic monitoring is not possible from other parts of the city. On the other hand, the desire for automatic city traffic control using video surveillance is also growing among authorities. The key purpose of automated system is to discover any traffic rule violations, e.g., taking U-turns at prohibited points, driving while being drunk, and driving on a yellow divider line. In such system, the main concern is the public safety for either a driver or a pedestrian. An accident survey [33] shows accidents results in Sweden occurred at various road intersections. Among all accidents on junctions, thirty percent are of high severity, whereas, twenty person lies under fatal class [34]. Existing traffic monitoring systems mainly rely on humans or static cameras for the supervision of road traffic. Conversely, with static cameras, we cannot monitor all the places in the city. Thus, we proposed an automatic traffic control system while identifying illegal driving behavior by using mobile vehicle's cameras. In subsequent sections, we will explain the system characteristics and its working in detail.

III. Proposed Real-Time Traffic Management Model

Here, in this section, we are giving the details of the proposed traffic violation detection model for the smart city. Initially, we described the designed video processing architecture that starts from the data harvesting to decision-making. Next, the short description is given to explain "how GPU processes the traffic video with CUDA". Afterwards, we presented the details of the main server and its video processing mechanism using Hadoop with GPUs. As, the vehicle and lane detection is also a part of the model, so we also presented the details of the used vehicle and lane detection methods. Finally, in this section, we took a wrong U-turn as a traffic violation usecase and developed its algorithm to deploy it on edge device to test the feasibility of the proposed system.

A. Proposed Architecture for Traffic Violation Detection using vehicle camera

As discussed earlier, it is not convenient and economical to place traffic sergeants everywhere in order to monitor all roads for traffic violations. Thus, it is needed to automate the system using cameras that monitor roads for traffic violations. The current camera monitoring models use static road/street cameras, which could not cover each and every place of all the roads in the city. Now a day, each modern vehicle comes with a camera to capture the front/back scene for the black box in case of an accident. Thus, the only way to continuously monitor each and every place on the road is to use the vehicle camera.

The proposed model assumes that every vehicle has a camera and an IoT-based edge device (such as, Raspberry Pi, Intel IoT board, or any other) that is equipped with GPU. The camera is directly attached to the edge device and continuously monitors the vehicles moving ahead, as shown in Figure 1. The camera

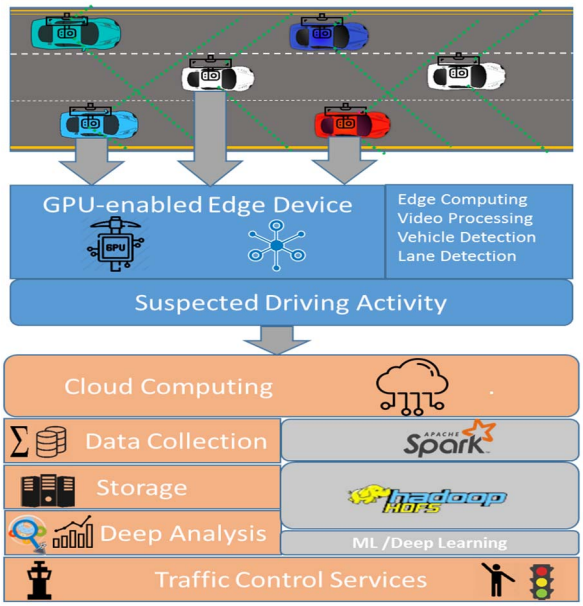

Figure 1. Proposed video processing architecture for the detection of traffic violations

continuously transmits the captured scenes to the edge device, which is configured with the vehicle detection, lane detection, and traffic violation detection algorithms such as wrong U-turn. The vehicle detection algorithm recognizes the vehicle running ahead, whereas, the lane detection algorithm marked the lanes on the road. The GPU makes the edge device extremely fast by performing parallel processing on video frames. This makes the edge device powerful enough to process the incoming videos far faster than the video duration. Once the edge device detected a suspected violation (illegal U-turn, wrong-side overtake, wrong parking, or others) by its configured algorithms, the device communicates the activity video to the cloud server (that can be a central traffic authority server). As, all the vehicles in the city only communicate the video to the cloud server when a suspected driving activity is detected, thus, the system saves the usage of the overall bandwidth.

The responsibility of the cloud server is to perform the deep analysis on the received suspected activity video and confirms the violation. At the start, the cloud server collects videos from edge devices at its collection unit and store them for later use. Then, the server performs the deep analysis on the received suspected activity video and confirms the violation. It generates an alert to the traffic authority and other concerning entities that notifies the violation. Since, the central server receives videos of suspected activities from thousands of vehicles moving around the city, accordingly, it needs efficient and fast tools to perform deep analysis for violation confirmation. This is achieved through the use of multilevel parallelism by the GPU-enabled distributed Hadoop nodes [35] along with the Spark [36]. The Hadoop has the ability to process data in parallel because of its distributed environment. However, its default MapReduce programing paradigm is not suitable for real-time processing. Thus, we used Apache Spark over the Hadoop parallel environment to make it more efficient for real-time processing.

The Spark comes with two major components i.e., 1) Spark Streaming that captures real-time data in chunks, and 2) Spark Engine that processes each incoming chunk from Spark streaming. It works with the GPU to achieve multilevel parallelism, which makes the overall process extremely faster due to the parallel computation on thousands of multiprocessors of GPU. The detail description of the video processing on central server using Hadoop and GPU is presented later in this section

B. GPU Processing with CUDA

The edge device speeds up the video processing by the GPU. The GPU is basically a combination of thousands of Multicore streaming processors (SMs) that forms a grid. A big task is divided into a multiple subtasks that can be executed in parallel on each SM. Nvidia created a parallel programing model called CUDA [37] to work with GPU. The GPU instructions are given by CUDA to compute the parameters for each of the video frames. The GPU processes each frame using its SMs grid. The CUDA provides the interfaces to logically decompose the grid into blocks and each block into threads, as shown in Figure 2. It is possible to create 1024 blocks on GPU that can be processed in parallel by giving a code through GPU_Kernal interface of CUDA. This dramatically reduces overall processing time. Each block has a small amount of memory that is shared among its threads. There is also a main GPU memory that is shared among all blocks. The blocks store their intermediate results in the main GPU memory. For any task given to the GPU, the identification of subtasks are important. A single subtask (or a job) should be identified that can be processed in parallel on SMs without the need of data from other SMs. Also, due to the limitations of GPU memory size, it cannot process large amount of data at once. Thus, we need distributed computing powers like Hadoop data nodes along with the GPU for large set of video data.

C. Video processing using Hadoop with GPU

Since, the standalone GPU is not suitable a large amount of video data processing, thus, at cloud server we used Hadoop along with the GPU to handle such overwhelming amount of videos coming from thousands of vehicles in the city. Figure 3 depicts the working principle of Hadoop nodes equipped with the GPU and Spark API. The video data coming from edge devices is captured and then divided into chunks by *Spark Streaming*. The Hadoop has a master node and various HDFS data nodes. The master node also works as a load balancer for HDFS nodes and each of the HDFS nodes is equipped with a GPU. Multilevel parallelism is achieved by processing the video on multiple HDFS nodes and then on thousands of SMs at each node. The *Spark Engine* executes the sequential instructions that cannot be accomplished by GPU, such as, combining the results of each of the image block. Whereas, the parallel instructions, that are independent threads, are completed on GPUs by implementing them using GPU kernel function of CUDA. Each video frame is divided into blocks by *Spark Engine* and each block is sent to GPU for parameter calculation on each block. Later, the results from GPU for each of the blocks is compiled by the *Spark Engine.* The Hadoop master node also implemented a global Reducer algorithm using *Spark Engine,* which combines the results from all HDFS nodes. The algorithm 1 presents the pseudocode of the parameters computation on GPU against each of the incoming block from *Spark Engine*. Table 1 gives the details of all the symbols used in Algorithm 1 and 2.

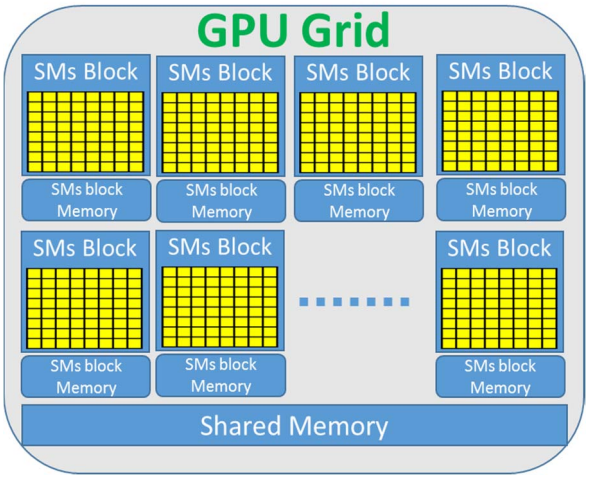

Figure 2. Image processing with GPU

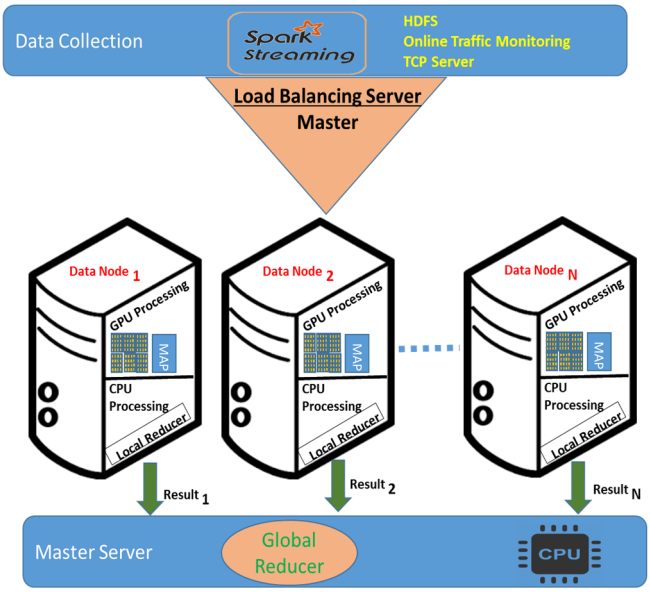

Figure 3. Data processing in distributed environment using GPU

Table 1. Symbols used in algorithms.

Symbols	Description	Symbols	Description
Blk_Array	Array of image blocks	Curr_val	Current Pixel Value
blkID	Block ID	Strat_val	Start Pixel in the block
blkSize	Block size	MaxVal	Maximun pixcel value
blkVal	Block value	CPU-GPU	Transmit from CPU to GPU
imgFrm	Video frame	GPU-CPU	Transmit from GPU to CPU
MinVal	Minimum pixel value	SD	Standard deviation

Algorithm 1: Parameter computation for each video frame on GPU for vehicle and lane detection

INPUT: Image/Video Frame
OUTPUT: Block with Calculated Parameters
STEPS:
1. Blk_Array [blkID, blkVal] := Devide_Image(ImgFrm, blkSize)

```
2.  CPU-GPU(Blk_Array)
3.  FUNCTION GPU_KERNAL(INPUT Blk_Array, OUTPUT result,
    OTHER_PARAMs blkSize)
4.     START
5.     ForEach idx_id=1 to blkSize)
6.     Do.
7.        COMPUTE sum, sum^2, MEAN, SD, MaxVal, MinVal,
8.        && other params
9.     END  ForEach LOOP
10. GPU-CPU(blkID, Params)
11. END GPU_KERNAL FUNCTION
```

D. Vehicle and Road Lane Detection

An auto-detection of traffic violation by camera can only be possible if the edge device is able to identify the vehicle at the early stage. For vehicle detection, we deployed the Caraffi et al. [38] algorithm at the edge device that uses WaldBoost [39] machine learning classifier with a sliding window framework for vehicle detection. Firstly, WaldBoost performs weak classification and then executes sequential probability ratio test (SPRT) to discard negative sample that does not have any vehicle. The SPRT raises the speed of the vehicle detection process, as most of portion of the image does not have any vehicle. The classification method used by the Wald is iterative in nature that initially starts with a sequential classifier building. At first iteration, it performs AdaBoost learning search as a weak classification and then it identifies the threshold value for SPRT to discard negative samples by considering a large size datasets. At the end, it performs pruning on the pool, and extra non-objects are collected. Figure 4 shows the detected vehicles by Caraffi's approach [39] using an edge device placed inside the vehicle.

For the road lanes detection, we deployed the three-feature automatic lane detection algorithm (TFALDA) that is basically developed for autonomous vehicles [40]. This lane detection methods uses three basic features including the initial point, the lane direction, and its gray level intensity. At start, many lane boundaries are identified including the false positives that are reduced by using a weight distance measure. Finally, the evolutionary algorithm is applied to reduce the false positives by considering the best values of the mentioned three-features.

E. Illegal Transportation Behavior Detection

As a use case, we considered a wrong U-turn as a traffic violation to test our model and designed an algorithm to detect this violation, as described by the algorithm 2. The algorithm is deployed on the edge device to test the feasibility of the system. In the beginning, vehicles and road lanes that are presented in

Figure 4. Vehicles detected by the edge device using a vehicle camera

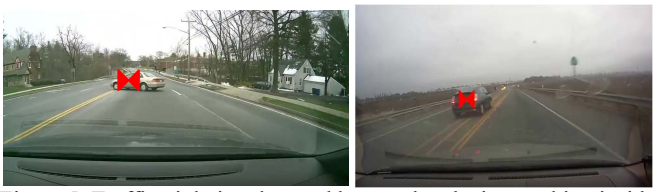

Figure 5. Traffic violation detected by an edge device working inside the following vehicle

the image are marked. Next, the wrong U-turn is identified by identifying the vehicle that is intersecting the central strong lane. Later, the algorithm considers the direction of the vehicle to reduce the false positive. Thus, the vehicle that is intersecting the strong yellow lane with the direction from right to left is considered as traffic violation. The vehicle direction is recognized by measuring the vehicle's width and it height. The sample wrong U-turns detected by the edge device is shown in Figure 9. Also, it is considered as a traffic violation if the vehicle is just driving on a central divider lane, as shown in Figure 9.

Algorithm 2: Detecting wrong U-turn as traffic voilation.

INPUT

1. Li: Lane boundaries (lines) detected by lane detection
2. V_j: Vehicle detected by vehicle detection mechanism

Where i: 0 → Total detected lanes.
j: 0 → Total detected vehicles

OUTPUT

1. IV: Illegal Driving vehicle

STEPS:

```
1. ForEach Vehicle Vj  Do
2.    ForEach Lane Li  Do
3.       IF (Li=YELLOW && Li==STRONG)    THEN
4.          IF (Li INTERSECT  Vj) THEN
5.             IV:=Vj
6.          ANNOUNCE IV
7.          END IF
8.       END IF
9.    END ForEach
10.  END ForEach
```

IV. System Implementation and Evaluation

A. Implementation environment and datasets

The functionality of the edge device is implemented on a machine with GTX 750 Ti GPU Engine, the machine has 640CUDA cores processors with 1020 base clock (MHz), 1085 boost clock (MHz), 5.4 Gbps memory clock, and 2GB standard RAM and GDDR-5 Interface. Whereas, the functionality of the main traffic server is implemented on an Intel(R) Core(TM) I5-6600 machine with 3.30GHz processors and 16 GB RAM. The machine is configured with a 2.7.2 Hadoop single node and Apache Spark 2.0.1 setup

For evaluation and testing purpose, we have considered various offline and online videos from various sources, such as YouTube, Earth Cam [41] and Arlingtonva.us [42].

B. System Evaluation

We know that the edge device has limited processing resources. Thus, our main concern is the real-time processing of traffic videos while detecting traffic violations. Therefore, we evaluated our system with respect to processing time and throughput corresponding to the increasing video duration, datasets size, no. of frames, and no. of files while implementing the wrong U-turn detection on the edge device. We have noticed that the processing time taken by the edge device for wrong U-turn detection is far less than the video duration, as analyzed in Figure 6. Also, it is observed that with the increasing video duration, the rise in the processing time is comparatively very short. With the video duration of 1400 seconds, the device just took 160 seconds to identify all wrong U-turns taken by vehicles. Similarly, we have analyzed the throughput corresponding to the increasing data size and number of processed frames. The results are shown in Figure 7 and Figure 8 respectively. The increasing data does not affect the throughput of the system. Generally, the best video uses generation rate of 30 frames per second. Accordingly, the real-time processing must handles 30 frames per second. Our system processed more than 200 frames per second even with the increasing dataset size. This result shows the ability of the proposed model for real-time processing of the traffic videos captured by the vehicle cameras to detect traffic violations. Thus, the proposed model can process 7-8 live videos at real-time. In addition, we measured the time consumed by the edge device while detecting the traffic violation corresponding to the increasing number of videos files of fixed size. Figure 9 shows the processing time measurements corresponding to the increasing number of fixed size video files in which it is obvious that the increasing number of video files have a constant impact on the processing time. Furthermore, the proposed model takes less than 5 milliseconds to process one video frame, as depicted by a graph in Figure 10. It means it can process a one second video in less than 150 milliseconds.

With these results it is obvious that the proposed model is feasible to deploy in the real-world, as it is able to process traffic videos at real-time while detecting traffic violations.

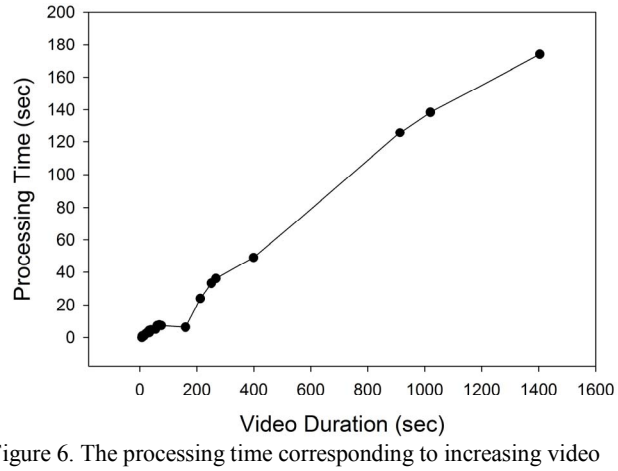

Figure 6. The processing time corresponding to increasing video duration while identifying the illegal U-turn.

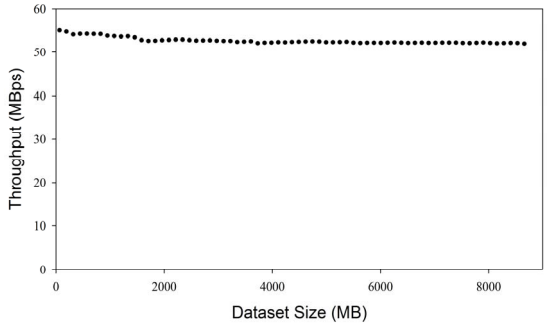

Figure 7. The throughput corresponding to increasing dataset size while identifying the illegal U-turn

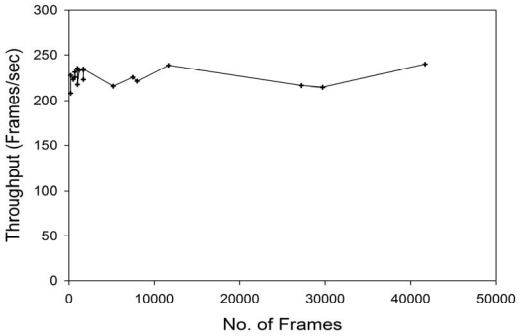

Figure 8. The throughput corresponding to increasing no. of video frames while identifying the illegal U-turn

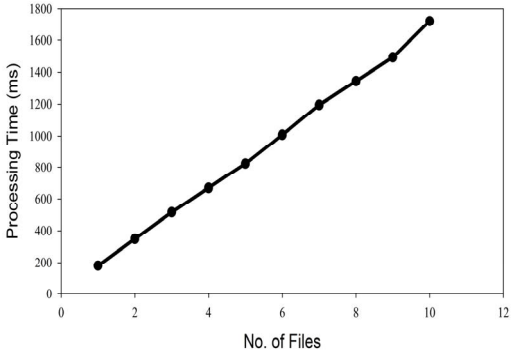

Figure 9. The processing time corresponding to the increasing no. of video files while identifying the illegal U-turn

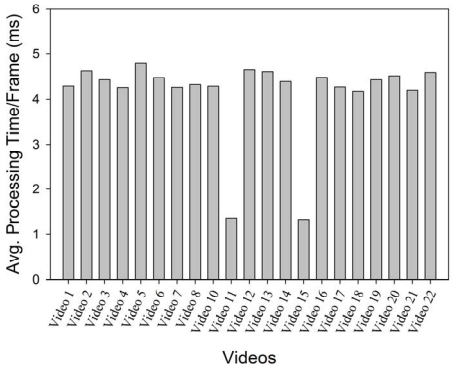

Figure 10. The average processing time corresponding various videos files while identifying the illegal U-turn

V. CONCLUSION

Managing the city transportation system has a key role in the advancement of a country. This paper focuses on the development of smart traffic control system using vehicle cameras. In this paper, we proposed a real-time traffic violation detection model by using vehicular camera along with the edge device in order to control and manage the road traffic. The edge device is equipped with the graphics processing unit (GPU), deployed inside the vehicle, and directly attached to the vehicle camera. The camera continuously monitors every vehicle ahead and sends the video to the edge device, which identifies the suspected driving violation. If any suspected activity is detected, the edge device transmits the reference video to the cloud (can be traffic authority server). The cloud server performs the deep analysis on the video and generates an alert to the authority in case the violation is committed. As a use case, we have tested our model by considering a wrong U-turn as a traffic violation. We also designed a wrong U-turn detection algorithm and deployed it on the GPU-enabled edge device. In order to evaluate the feasibility of the system, we considered the efficiency measurements corresponding to the video generation rate and data size. The results show that the system is able to identify violation far faster than the video generation time.

REFERENCES

[1] Jin, J., Gubbi, J., Marusic, S., & Palaniswami, M. (2014). An information framework for creating a smart city through internet of things. IEEE Internet of Things Journal, 1(2), 112-121.

[2] Rathore, M. M., Ahmad, A., Paul, A., & Rho, S. (2016). Urban planning and building smart cities based on the Internet of Things using Big Data analytics. Computer Networks, 101, 63-80.

[3] Přibyl, O. (2015, June). Transportation, intelligent or smart? On the usage of entropy as an objective function. In Smart Cities Symposium Prague (SCSP), 2015 (pp. 1-5).

[4] F. Zhu, Z. Li, S. Chen and G. Xiong. (JUNE, 2016). Parallel Transportation Management and Control System and Its Applications in Building Smart Cities. In IEEE Transactions on Intelligent Transportation Systems. vol. 17, no. 6, pp. 1576-1585, doi: 10.1109/TITS.2015.2506156

[5] Z. Ding, B. Yang, Y. Chi and L. Guo. (May, 2016). Enabling Smart Transportation Systems: A Parallel Spatio-Temporal Database Approach. In IEEE Transactions on Computers. vol. 65, no. 5, pp. 1377-1391, doi: 10.1109/TC.2015.2479596.

[6] Ahmad, A., Paul, A., Rathore, M. M., & Chang, H. (2016). Smart cyber society: Integration of capillary devices with high usability based on Cyber–Physical System. Future Generation Computer Systems, 56, 493-503.

[7] Rathore, M. M., Ahmad, A., Paul, A., & Jeon, G. (2015, November). Efficient Graph-Oriented Smart Transportation using Internet of Things generated Big Data. In 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS) (pp. 512-519).

[8] Ailamaki, N.K. Govindaraju, S. Harizopoulos, and D. Manocha, "Query Co-Processing on Commodity Processors," Proc. 32nd Int'l Conf. Very Large Data Bases (VLDB), 2006.

[9] H. Y. Cheng, C. Weng, and Y. Y. Chen, Vehicle detection in aerial surveillance using dynamic Bayesian networks, IEEE Transactions on Image Processing, vol. 21, no. 4, pp. 2152-2159, 2012.

[10] W. Shao, W. Yang, G. Liu, and J. Liu, Car detection from high-resolution aerial imagery using multiple features, In Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International Conference, pp. 4379-4382, 2012.

[11] S. Kluckner, G. Pacher, H. Grabner, H. Bischof, and J. Bauer, A 3D teacher for car detection in aerial images, in pro. IEEE 11th International Conference on Computer Vision (ICCV), pp. 1-8, 2007.

[12] A. Kembhavi, D. Harwood, and L.S. Davis, Vehicle detection using partial least squares, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 6, pp. 1250-1265, 2011.

[13] J. Leitloff, D. Rosenbaum, F. Kurz, O. Meynberg, and P. Reinartz, An operational system for estimating road traffic information from aerial images, Remote Sensing, vol. 6, no. 11, pp. 11315-11341, 2014.

[14] S. Tuermer, F. Kurz, P. Reinartz, and U. Stilla, Airborne vehicle detection in dense urban areas using HoG features and disparity maps IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol 6, no. 6, pp. 2327-2337, 2013.

[15] Z. Chen, C. Wang, C. Wen, X. Teng, Y. Chen, H. Guan, .. and J. Li, Vehicle detection in high-resolution aerial images via sparse representation and superpixels, IEEE Transactions on Geoscience and Remote Sensing, vol. 54, no. 1, pp. 103-116, 2016.

[16] DF. Fitzgerald, DS. Wills, and LM. Wills, Real-time, parallel segmentation of high-resolution images on multi-core platforms, Journal of Real-Time Image Processing, vol. 13, no. 4, pp. 685-702, 2017.

[17] Z. Chen, C. Wang, H. Luo, H. Wang, Y. Chen, C. Wen, ... , and J. Li, Vehicle detection in high-resolution aerial images based on fast sparse representation classification and multiorder feature, IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 8, pp. 2296-2309, 2016.

[18] X. Chen, S. Xiang, C. Liu, and C.H. Pan, Vehicle detection in satellite images by hybrid deep convolutional neural networks, IEEE Geoscience and remote sensing letters, vol. 11, no. 10, pp. 1797-1801, 2014.

[19] A.B. Hillel, R. Lerner, D. Levi, and G. Raz, Recent progress in road and lane detection: a survey, Machine vision and applications, vol. 25, no. 3, pp. 727-745, 2014.

[20] M. Aly, Real time detection of lane markers in urban streets, in IEEE Intelligent Vehicles Symposium, pp. 7-12, (2008).

[21] M. Haloi and D.B. Jayagopi, A robust lane detection and departure warning system, in proc. IEEE Intelligent Vehicles Symposium (IV), pp. 126-131, 2015.

[22] A. Borkar, M. Hayes, and M. T. Smith, A novel lane detection system with efficient ground truth generation, IEEE Transactions on Intelligent Transportation Systems, vol. 13, no. 1, pp. 365-374, 2012.

[23] U. Ozgunalp, R. Fan, X. Ai, and N. Dahnoun, Multiple Lane Detection Algorithm Based on Novel Dense Vanishing Point Estimation, IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 3, pp. 621-632, 2017.

[24] M. Fu, X. Wang, H. Ma, Y. Yang, and M. Wang, Multi-lanes detection based on panoramic camera, in proc. 11th IEEE International Conference on Control & Automation (ICCA), pp. 655-660, 2014.

[25] P. M. A. Kumar, V. Sathya, and V. Vaidehi, Traffic Rule Violation Detection in Traffic Video Surveillance, International Journal of Computer Science and Electronics Engineering (IJCSEE), vol. 3, no. 4, 2015.

[26] P. Kumar, S. Ranganath, H. Weimin, and K. Sengupta, Framework for real-time behavior interpretation from traffic video, IEEE Transactions on Intelligent Transportation Systems, vol. 6, no. 1, pp. 43-53, 2005.

[27] J. A. Vijverberg, N. A. Koning, J. Han, P. H. de With, and D. Cornelissen, High-level traffic-violation detection for embedded traffic analysis, in proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 2, pp. II-793, 2007.

[28] L. Cui, K. Li, J. Chen, and Z. Li, Abnormal event detection in traffic video surveillance based on local features, in proc. 2011 4th International Congress on Image and Signal Processing (CISP), vol. 1, pp. 362-366, 2011.

[29] Y. Benezeth, P. M. Jodoin, V. Saligrama, and C. Rosenberger, Abnormal events detection based on spatio-temporal co-occurences, in proc. , IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2458-2465, 2009.

[30] O. Boiman and M. Irani, Detecting irregularities in images and in video, International journal of computer vision, vol. 74, no. 1, pp. 17-31. 2007.

[31] R. T. Collins, A. J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, ..., and L. Wixson, A system for video surveillance and monitoring, VSAM final report, pp. 1-68. 2000.

[32] C. H. Song and J. Lee, Detection of illegal u-turn vehicles by optical flow analysis, The Journal of Korean Institute of Communications and Information Sciences, vol. 39, no. 10, pp. 948-956, 2014.

[33] J. Cano, J. Kovaceva, M. Lindman, M. Brannstrom, M. L. Cano, J. Kovaceva, and M. Brannstrom, Automatic incident detection and classification at intersections, Volvo Car Corporation, pp 09-0245, 2009.
[34] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, A vision-based approach to collision prediction at traffic intersections, IEEE Transactions on Intelligent Transportation Systems, vol. 6, no. 4, pp. 416-423, 2005.
[35] Apache Hadoop (2016).Welcome to Apache™ Hadoop®!. http://hadoop.apache.org/. Accessed on November 01, 2016.
[36] Apache SPARK (2016). Apache Spark™. http://spark.apache.org/. Accessed on November 01, 2016.
[37] NVIDIA ACCELERATED COMPUTING. (2016). CUDA Toolkit 8.0. https://developer.nvidia.com/cuda-downloads. Accessed on November 01, 2016.
[38] Caraffi, C., Vojíř, T., Trefný, J., Šochman, J., & Matas, J. (2012, September). A system for real-time detection and tracking of vehicles from a single car-mounted camera. In 2012 15th International IEEE Conference on Intelligent Transportation Systems (pp. 975-982).
[39] Sochman, J., & Matas, J. (2005, June). Waldboost-learning for time constrained sequential detection. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 2. pp. 150-156.
[40] Yim, Y. U., & Oh, S. Y. (2003). Three-feature based automatic lane detection algorithm (TFALDA) for autonomous driving. IEEE Transactions on Intelligent Transportation Systems, 4(4), 219-225.
[41] Earth Cam(2016). LIVE Webcam Network. http://www.earthcam.com/. Accessed on November 01, 2016.
[42] Arlingtonva.us (2016) .Live Traffic Cameras. https://transportation.arlingtonva.us/live-traffic-cameras/. Accessed on November 01, 2016.