IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 18, 2018, accepted August 24, 2018, date of publication September 5, 2018, date of current version September 28, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2868610

Real-Time Android Application for Traffic Density Estimation

FATMA KEROUH AND DJEMEL ZIOU
Faculté d'Informatique, Université de Sherbrooke, Sherbrooke, QC J1K 2R1, Canada
Corresponding author: Fatma Kerouh (fatma.kerouh@usherbrooke.ca)

ABSTRACT This paper deals with automatic traffic density estimation using new technologies that are cost-effective, quick to deploy, and easy to adapt. Therefore, a real-time video processing-based android application (*app*) is developed. It can be used online by exploiting the smartphone's camera placed at the roadside or offline and by using pre-recorded vehicles flow videos and an Android emulator. The system autonomously computes the vehicle flow density in real time and saves all the relevant information in the smartphone memory. The developed *app* is validated under real conditions in Sherbrooke and Montreal cities, by considering different meteorological conditions and varying intrinsic and extrinsic camera parameters. Obtained results allow being optimistic about the effectiveness and the applicability of the proposed *app*.

INDEX TERMS Traffic management, vehicles density estimation, Android application, real-time, video processing, line of interest.

I. INTRODUCTION

Traffic density estimation is one of the key elements for traffic management, infrastructure, transportation planning and policy, pollution reduction, and forecasting [1], [2]. Thus, considerable efforts are provided to develop technologies in order to understand and improve the issues. There exist different commercial solutions for traffic density estimation as Doppler radar, infrared, and inductive loop-based sensors [1]. However, they present some limitations due to complexity, human effort, maintenance, portability, and monetary cost. To overcome these limitations, image/video processing-based algorithms have been proposed [6]–[17]. The solution is low cost because only a camera is required. Nevertheless, the task is challenging due mainly to the processing runtime and the video quality commonly affected by noise, blur, and illumination variation. The present paper focuses on the real-time video processing-based systems.

Developing such system requires a camera, an operating system, a processor, and a memory. Therefore a centralized IP camera, a computer, or a smartphone can be used. Because we aim to develop a cost effective, simple using, and quick deploying solution, the smartphone device is more suitable. Nonetheless, using a smartphone imposes some limitations concerning the processing complexity, used memory, and energy consumption [18]. To overcome these limitations, a simple video processing algorithm with minimal energy consumption is implemented. The idea turns on analysing a line of interest (LOI) instead of the whole frame. The LOI is considered as a line crossing the road. Observations over this line are sufficient for traffic density estimation. Herein, we especially consider an Android smartphone equipped with a rear camera and a global positioning system (GPS). The use of an Android device offers the possibility to exploit other free Android *apps* to manage the proposed one as the *AirDroid app* which allows access to an Android device from a different location using a computer.

According to the state of the art, two papers concerning Android application related to traffic density estimation exist. The first one, proposed by Toth *et al.* [3], presents a Tablet-based application for off-line count of vehicles. The application allows the user to count vehicles by touching. While playing the video, the user has to touch each vehicle crossing a predefined area and accordingly a counter is incremented. The authors claim that their system allows to reduce the fully manual counting errors. Later, an application is developed by Bhatt *et al.* [4]. Authors describe an automatic vehicle counting *app*. Time consuming video processing techniques are used for vehicle detection as background subtraction, contour detection, and morphological operators. No experimentations nor system evaluation is discussed in the paper. From the presented related works, the existing *apps* are limited against the autonomy, time consuming, and validation process. These factors motivate us to go further by proposing a new application that is real time, simple to

use, automatic, and validated on long-lasting videos from different real scenarios, including, different smartphones, video duration, geographical position, intrinsic and extrinsic camera characteristics, and weather conditions. The developed *app* can be used by many public agencies, private sectors, governments for traffic flow analysis or for personal use, for example, to gather information about road traffic in the neighborhood or to present a quantitative evaluation of the traffic flow when selling a residence. To the best of our knowledge, such application does not exist. In the next section, we review the proposed system. Section 3 details the used video processing algorithm. Tests and validation are exhibited in section 4.

II. SYSTEM OVERVIEW

Recall that we are going to estimate the vehicle flow density. In this context, density stands for the total count of vehicles over a time unit. The estimator requires to be low cost, simple to use, real time, automatic, and run on a smartphone. To fulfill these requirements, we implement two strategies. First, we use the video flow and other data acquired by the smartphone. Let us consider an Android smartphone with a processor, a memory space, a GPS, a rear camera, in addition to a network to access data when required. The smartphone camera records the video which is used as an input to the vehicle density estimator and then, a video processing algorithm automatically estimates the density of vehicles. Second, a line of interest (LOI) is used instead of processing the entire frames of the video. The grey level observations over the LOI form time series that are analysed to performing the counting task. The implementation of both strategies requires the data acquisition, reference LOI selection, vehicles density estimation, and relevant information saving. For the data acquisition, the smartphone's camera is placed in a high position facing the vehicle flow, both the GPS position, date, time, and a reference frame are acquired before starting recording the video. Concerning the reference LOI, it is selected according to the following rules. First, by using the reference frame, the user selects a line crossing each lane of the road. One can choose one LOI crossing all the road lanes. However, it may make it difficult the vehicles detection task, especially when more than one vehicle crosses the LOI with different speeds. Second, the selected LOI must not be collinear to avoid the double counting, especially, when a vehicle passes between two collinear LOI. Figure 1 shows an example of the selected LOI. In this case, the smartphone's camera is placed on a bridge facing the vehicle traffic flow. As the road is composed of four lanes, four LOI are selected. For vehicle counting, a reference LOI is seen as a list of pixels in the reference frame on which the LOI is placed. When a vehicle crosses the LOI, a temporal change in the gray level is observed. The temporal change informs about the presence of a vehicle within a lane. Indeed, detecting vehicles is equivalent to making a decision about the change of gray level on the reference LOI. Figure 2 points of an example of the temporal change of grey levels over a lane. When no

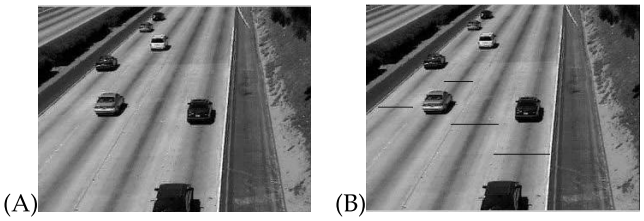

FIGURE 1. (A) Reference frame, (B) Selected LOI.

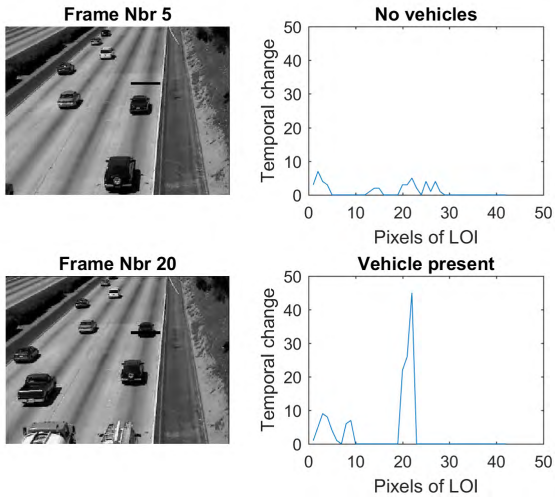

FIGURE 2. Gray level change on a LOI.

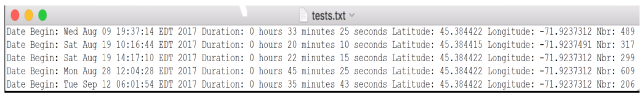

FIGURE 3. A sample of a saved text file.

vehicle crosses the LOI, small variations, due mainly to noise, are observed. After few frames, the vehicle crosses the LOI and the temporal change reaches higher values. The detection of high values is the basis of the estimator. More details will be tackled in the next section. Let us explain the output of the system. Because of the limited memory space and energy, only the relevant information including the traffic density, GPS position, date, and the recording duration is stored in a text file. An example is shown in Figure 3. This file contains the history of five experiments in the form of the date, starting time, GPS position, recording duration, and total count. For energy management, an SMS is sent to a predefined phone number when the battery level is less than a predefined value. As shown in Figure 4, the message specifies the battery state and the GPS position of the smartphone.

III. VEHICLES DENSITY ESTIMATION ALGORITHM

Before detailing the proposed approach, we review in Table 1, some existing video processing-based algorithms for vehicles counting. The existing works are described by considering five criteria; the used video preprocessing tools, counting

FIGURE 4. A sample of low battery alert message.

TABLE 1. Summary settings of presented related works.

Ref	Preprocessing	Counting	R-T	Data	Accuracy
[6]	Background subtraction Image segmentation	Learning Mixture model	Yes	180 frames	98.66%
[7]	Background estimation Foreground segmentation	Bounding box	Yes	70 min	87.82%
[8]	Background subtraction Image segmentation	Mixture model	No	5 sec	91%
[9]	Corner detector Centroid detection	Threshold	Yes	8 min	88%
[10]	Background subtraction Prewitt Edge detector Morphological operator	Threshold	No	Youtube video!	88%
[11]	Background subtraction Kalman filter	LDA classifier Use database	No	3400 frame	97.37%
[12]	Background modelling	Gaussian mixture	No	N-M	93%
[13]	Background subtraction Segmentation SIFT transform	SIFT features matching	No	2 frames	Good!
[14]	Background subtraction morphological operators	Threshold on a rectangular region of interest	Yes	N-M	Effective!
[15]	Segmentaion (GMM+EM) and restoration	Threshold applied on a virtua loop	No	720 vehicles	95%
[16]	Foreground and background segmentation	Threshold	Yes	200 min	89.2%
[17]	Background substraction and vehicles tracking	Threshold	Yes	2800 frame	89.4%

approach, whether the algorithm is real-time or not (R-T), validation data size, and system accuracy. According to Table 1, a commonly used strategy is vehicles segmentation using different techniques as filtering and background registration and subtraction using morphological operators followed by a counting approach based on classification using mixtures and linear discriminant analysis (LDA) [8], [10], [13]–[15] or an empirical threshold. Accordingly, the presented works use the whole video pixels and heavy preprocessing approaches to achieve the counting task that's explain why most algorithms are not real time [10], [13]. Furthermore, notice that the size of the used validation data is often insufficient [6], [8], [9], [13], [17] or not mentioned (N-M) [12], [14]. Therefore, it is difficult to judge the generalization of these approaches. According to the accuracy column, some papers provide a qualitative instead of quantitative evaluation [13], [14] that make it difficult to compare quantitatively the performance. All these limitations motivate us to propose a new approach which does not require any tracking or preprocessing tools. To count vehicles, only the time series variation on a line of interest (LOI) is studied. That makes the algorithm faster, requiring less memory space, minimizing the energy consumption, and real time on a smartphone. Let us detail the proposed counting approach. As explained before, the vehicle density is computed by only using the selected LOI. Motion estimated from grey level frames is enough for vehicles detection and counting. Note that for the example of Figure 1, the selected LOI represent 0.097% of the total size of the frame (224 pixels over 230400). Let assume that the reference lines for all lanes have been recorded from the first frame. To detect vehicles, the total of the absolute difference between the LOI of each frame and the reference one is computed. To be independent from the LOI length, the total absolute difference is normalized. The output time series are used to detect and count vehicles by using an automatic decision rule. To achieve that, the following strategy is implemented. Two thresholds are used, the first one Th_1 to detect the vehicle and the second Th_2 ($Th_2 > Th_1$) to count it. Hence, when the vehicle started crossing the LOI, it is detected by the threshold Th_1 and itself is counted using Th_2 when leaving the LOI. The use of two thresholds makes it possible to implement a fast and low memory consumption rules to detect and count vehicles. The challenge, however, is to define rules for calculating these two thresholds.

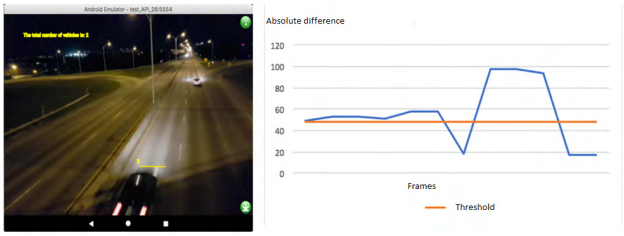

FIGURE 5. Example of time series in the night time.

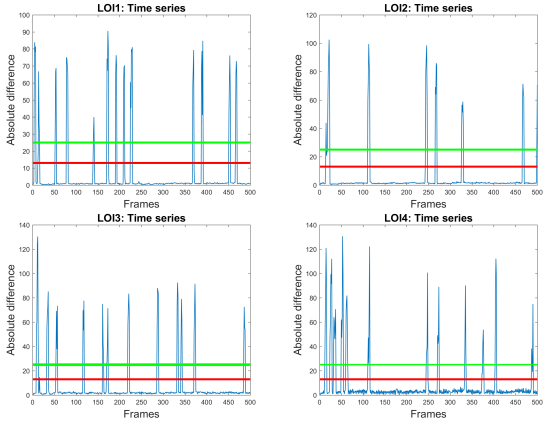

FIGURE 6. Illustration of the selected thresholds on time series, green Th_1 and red Th_2.

The estimation of the two thresholds is based on the following observations. First, the amplitude of the time series depends on the illumination condition during the acquisition as it is shown in Figure 5. Indeed, during the night time, two time series are noticed for each vehicle, the first due to its lights and the second caused by the vehicle itself. Second, Th_1 and Th_2, which is a function of Th_1, must be adapted to noise amount since time series is noisy (see Figures 2 and 6). Third, because the ratio of the peak due to noise and the peak due to vehicles is low, as illustrated in Figure 6, the frame rate per second (Fps) of a video can be used as a rough noise estimator. Indeed, we expect that the noise is

high when the Fps is, for the off-the-shelf cameras. In other words, Th_1 depends also on the intrinsic characteristics of the used camera. More precisely, experimentations show that the following estimation rules of the thresholds are useful:

$$Th_1 = K \ F_{ps} \tag{1}$$

$$Th_2 = \frac{Th_1}{2} \tag{2}$$

The parameter K is a positive real number where the value depends on the illumination. Tests show that the parameter K holds the value of the golden ratio $k_1 = 1.618$ in the daytime and $k_2 = 2\ k_1$ in the night time.

The selection of k_1 or k_2, i.e., the day or the night time, is automatically achieved according to the LOI average. Indeed, when no vehicle is present, the LOI can be seen as an indicator of the illumination. Hence, we use the LOI average as a cue to discriminate day and nighttime. Furthermore, because of the change in weather conditions or natural phenomena, we propose to recompute the LOI average frequently to reduce the counting errors relying on the selection of k_1 or k_2. To conclude, the proposed decision rule is simple, fast, and consider the noise and the illumination variation. Figure 6 presents time series of the four lines of Figure 1 and the two considered thresholds. We can see that Th_1 must be higher than Th_2 to avoid considering noise and Th_2 is small to ensure that the vehicle leaves the LOI.

IV. EXPERIMENTATION

Before detailing the achieved experiments, we provide first some details about the *app* implementation. It is developed using the JAVA software under Android studio. We choose a sequential architecture to implement three interfaces as pointed out in Figure 7. In the first interface, the user captures the reference frame and selects the considered LOI. As illustrated in Figure 8 (A), the interface allows to choose the autofocus mode to focus on the vehicle flow and avoid out of focus blur. In the second interface, the user selects the LOI. An example is depicted in Figure 8 (B). Note that a vehicles counter initialized to zero is associated to each selected LOI. Moreover, the interface offers the possibility to delete and reselect the LOI. The validation of the LOI is required before moving to the last interface where the counter will be automatically incremented when a vehicle crosses the LOI as depicted in Figure 8 (C). Furthermore, in the smartphone, the proposed *app* requires 1.45 Mo memory, it consumes about 84mAh battery energy (more than 35 hours

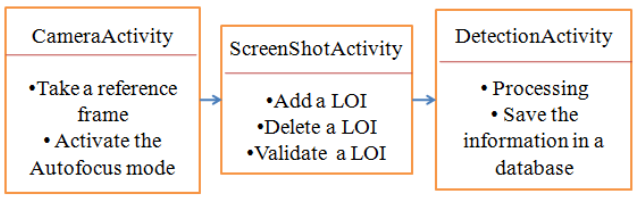

FIGURE 7. Architecture of the proposed *app*.

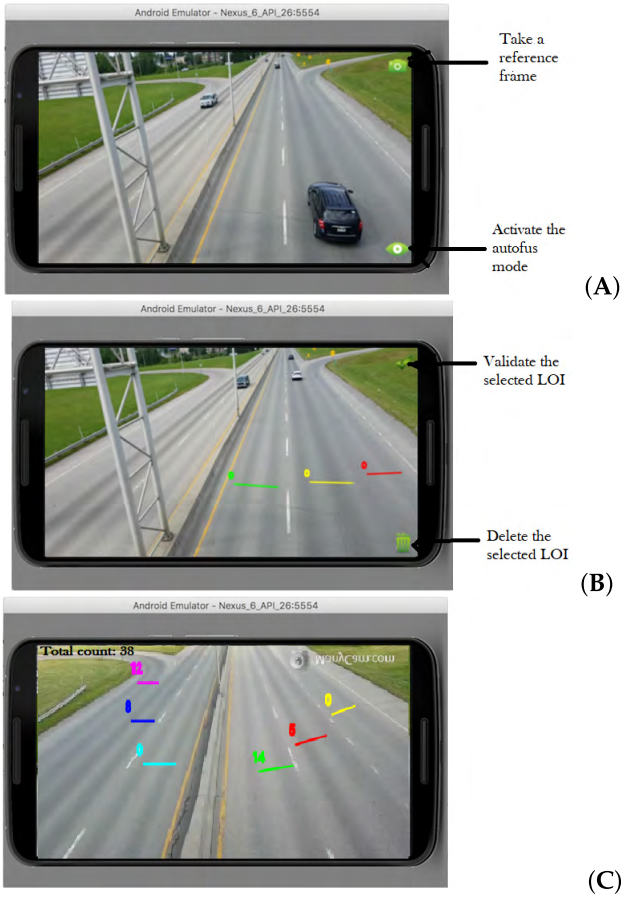

FIGURE 8. Illustration of the three interfaces.

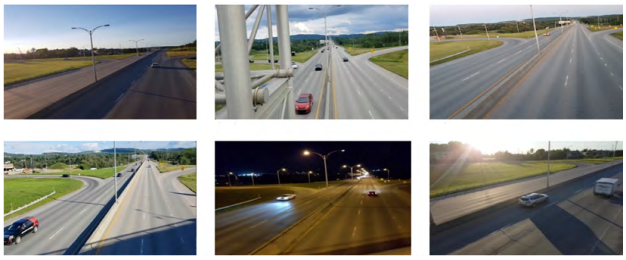

FIGURE 9. Example of considered weather conditions.

autonomy on galaxy alpha smartphone of 3000 mAh battery), and the vehicle density is estimated in real time (about 0.06 seconds per frame). Let us now test the performance of the proposed *app*. Several tests on three different road sites in Sherbrooke and Montreal cities are conducted. Tests were achieved online and offline under different visibility conditions (sunny, cloudy, rainy, night, daytime, sunrise, sunset) including bike paths, at three different altitudes (low: 3 feet, medium: 8 feet, high: 15 feet), and at different camera position (in front and beside the vehicle flow). Figure 9 and 10 illustrates examples of shooting. A total of almost 5 hour video was captured using three different Android smartphones that are: Nexus 4: Resolution 768 × 1280,

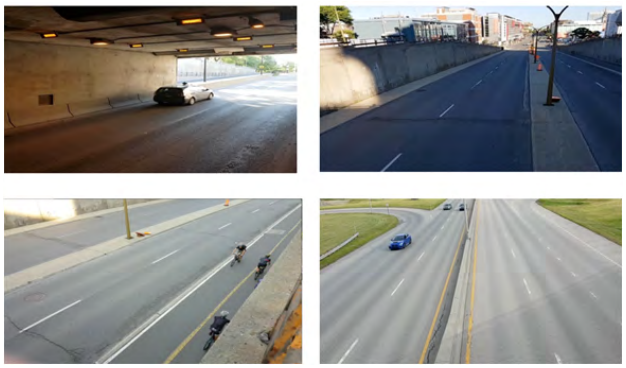

FIGURE 10. Example of frames taken at different altitudes.

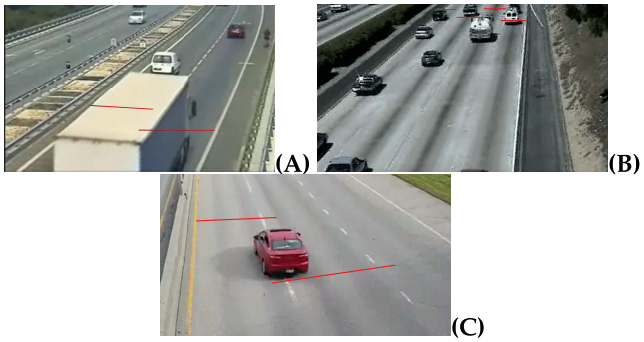

FIGURE 11. Impact of the LOI position.

Camera 8 megapixels, Frame rate 15/s, Processor 1.5 GHz, RAM 2G; Samsung Galaxy alpha: Resolution 720 × 1280, Camera 12 megapixels, Frame rate 25/s, Processor 1.2 GHz, Ram 1G; Samsung S7: Resolution 1080 × 1920, Camera 13 megapixels, Frame rate 30/s, Processor 1.9 GHz, Ram 2G.

We discuss first the effect of the LOI position on the *app* accuracy. The LOI position must be selected carefully to maximize the performance of the *app* . The experimentation show that the best location is in the middle of the road within the lane to avoid the problem of vehicle occlusions and recounting. Indeed, when the LOI is too close or too far from the camera, an occlusion problem may occur as explained by Figure 11 (A) and (B). Similarly, when the LOI is larger than the lane, one vehicle may be counted twice as depicted on Figure 11 (C). To conclude, the choice of the LOI position is important to achieve good performance. Once the LOI carefully selected, the *app* accuracy is measured by considering three criteria: the false positive (FP), true negative (TN), and the accuracy computed as follows:

$$Accuracy = 100 * (1 - \frac{FP + TN}{TotalCount}) \quad (3)$$

In Table 2, we report the *app* performance on different video duration captured by three different smartphones under different weather conditions. Accordingly, high accuracy scores are obtained in all cases, over 92%. Note that the highest FP value is obtained at night time. This is mainly due to the double counting problem discussed previously.

TABLE 2. Counting evaluation in different weather conditions.

Weather Conditions	Duration (min)	FP	TN	Accuracy%
Sunny	40	5	7	96.7
Cloudy	27	0	5	97.8
Night	22	9	0	92.8
Rainy	27	2	4	96.57
Sunrise	22	6	2	94.02
Sunset	39	5	3	93.49
Bike path	30	0	2	97.22
Average	30	3.85	3.28	95.51

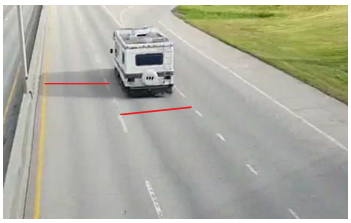

FIGURE 12. Illustration of extreme shadow effect.

TABLE 3. Counting evaluation at different altitude.

Altitudes	Duration (min)	FP	TN	Accuracy%
Low	24	3	3	97.53
Average	32	2	7	94.82
High	21	6	5	95.09
Average	25.66	4	5	95.81

In sunny, sunset and sunrise conditions, the FP is mainly due to the extreme shadow as pointed out in Figure 12. The TN cases are less than FP. They occur, especially when a vehicle avoids the lane and passes between two LOI. Similarly, tests carried out on the bike path were conclusives and the error rate is less than 3%. Averagely, the accuracy rate is over 95% which is a satisfying score compared to the state of the art (see Table 1). To conclude, the proposed *app* provides an encouraging results independently from the used smartphone, video duration, and weather conditions. The effect of altitude and camera position on the accuracy of the application is tabulated in Table 3. Note that four different camera positions have been considered (see Figure 10). According to Table 3, high performance are obtained (over 94%). The highest accuracy rate is obtained in the low altitude. This can be explained by the fact that the video test is captured inside a covered bridge where the effect of shadow and illumination variation is minimal. Overall, there is neither impact of the altitude nor the camera position on the accuracy results.

Let us now compare the performance of the proposed approach against a recent threshold based counting approach [17]. In this work, authors propose a real time vehicles counting algorithm using background subtraction model with low-rank decomposition (see Table 1). The comparison is performed by considering the same two videos used in [17] named respectively highway (600 frame) and highwayII (500 frame). Note that an example frame of highway is depicted in Figure 1 and highwayII in Figure 11A. Theses videos contain the following particularities; sunny day,

TABLE 4. Comparative study.

	#vehicles	Counted	Accuracy%
Highway [17]	16	14	87.5
Highway proposed	16	15	93.75
HighwayII [17]	91	84	92.3
HighwayII proposed	91	91	100

shadows, and waving trees for highway and crowed scene and occlusion for highwayII. The comparison is achieved in terms of the total number of vehicles (#*vehicles*), number of correctly counted vehicles (Counted), and accuracy percentage. Obtained results tabulated in Table 4 show that the proposed system outperform the existing one in terms of accuracy.

V. CONCLUSION

In this work, a novel real time Android application for automatic traffic density estimation is developed. It exploits the smartphone camera and a simple video processing algorithm. We have shown that a simple decision rule is sufficient to detect and count vehicles under real conditions. The achieved evaluation in different weather conditions, altitudes, camera position, intrinsic and extrinsic smartphone parameters proves its consistency. The developed *app* has the advantage of being real time, simple to deploy, portable, low cost, and accurate. Future works will involve improving the performance against the strong shadow and adding other functionality as vehicles classification and speed estimation.

REFERENCES

[1] S. Ezell, "Intelligent transportation systems," Inf. Technol. Innov. Found., Washington, DC, USA, Tech. Rep., Jan. 2010.

[2] M. Kumar and S. Albert, "A summary of rural intelligent transportation systems (ITS) benefits as applied to ODOT region 1," Oregon Dept. Transp. Region 1, Portland, OR, USA, Tech. Rep., Apr. 2005.

[3] C. Toth *et al.*, "Tablet-based traffic counting application designed to minimize human error," *Transp. Res. Rec., J. Transp. Res.*, vol. 2339, pp. 39–46, Dec. 2013.

[4] R. Bhatt, M. Lala, A. Deshmukh, S. Lodha, and P. Patil, "Real time vehicle counting and mapping on Android app," *Int. J. Res. Emerg. Sci. Technol.*, vol. 2, no. 4, pp. 59–62, Apr. 2015.

[5] O. Prakash, M. Aggarwal, A. Vishvesha, and B. Kumar, "Traffic detection system using Android," *J. Adv. Comput. Commun. Technol.*, vol. 3, no. 3, pp. 56–60, Jun. 2015.

[6] P. M. Daigavan and P. R. Bajaj, "Real time vehicle detection and counting method unsupervised traffic video on highways," *Int. J. Comput. Sci. Netw. Secur.*, vol. 10, no. 8, pp. 112–117, Aug. 2010.

[7] M. Liang, X. Huang, C.-H. Chen, X. Chen, and A. Tokuta, "Counting and classification of highway vehicles by regression analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2878–2888, Oct. 2015.

[8] S. A. Meshram and A. V. Malviya, "Traffic surveillance by counting and classification of vehicles from video using image processing," *Int. J. Adv. Res. Comput. Sci. Manage. Stud.*, vol. 1, no. 6, pp. 169–175, Nov. 2013.

[9] N. C. Acheruvu and V. Muthukumar, "Video based vehicle detection and its application in intelligent transportation systems," *J. Transp. Technol.*, vol. 2, no. 4, pp. 305–314, Oct. 2012.

[10] R. Javadzadeh, E. Banihashemi, and J. Hamidzadeh, "Fast vehicle detection and counting using background subtraction technique and prewitt edge detection," *Int. J. Comput. Sci. Telecommun.*, vol. 6, no. 10, pp. 8–12, Nov. 2015.

[11] H. Rabiu, "Vehicle detection and classification for cluttered urban intersection," *Int. J. Comput. Sci. Eng. Appl.*, vol. 3, no. 1, pp. 37–47, Feb. 2013.

[12] C. Salvadori, M. Petracca, S. Bocchino, R. Pelliccia, and P. Pagano, "A low-cost vehicle counter for next-generation ITS," *J. Real-Time Image Process.*, vol. 10, no. 4, pp. 741–757, 2015.

[13] M. C. Narhe and M. S. Nagmode, "Vehicle counting using video image processing," *Int. J. Comput. Technol.*, vol. 1, no. 1, pp. 358–362, Aug. 2015.

[14] K. Srijongkon, R. Duangsoithong, N. Jindapetch, M. Ikura, and S. Chumpol, "SDSoC based development of vehicle counting system using adaptive background method," in *Proc. IEEE Regional Symp. Micro Nanoelectron. (RSM)*, 2017, pp. 235–238.

[15] Y. Xia, X. Shi, G. Song, Q. Geng, and Y. Liu, "Towards improving quality of video-based vehicle counting method for traffic flow estimation," *Signal Process.*, vol. 120, pp. 672–681, Mar. 2016.

[16] M. T. Yang, R. K. Jhang, and J. S. Hou "Traffic flow estimation and vehicle-type classification using vision-based spatial-temporal profile analysis," *IET Comput. Vis.*, vol. 7, no. 5, pp. 394–404, Oct. 2013.

[17] H. Yang and S. Qu, "Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition," *IET Intell. Transp. Syst.*, vol. 12, no. 1, pp. 75–85, 2018

[18] J. A. Montenegro, M. Pinto, and L. Fuentes, "What do software developers need to know to build secure energy-efficient Android applications?" *IEEE Access*, pp. 1–23, 2017.

[19] J. A. Montenegro, M. Pinto, and L. Fuentes, "Low-cost IP camera for traffic monitoring," *Int. J. Comput. Appl.*, vol. 109, no. 7, pp. 30–35, 2015.

FATMA KEROUH received the B.Sc. degree in electronic engineering from Jijel University in 2008 and the M.Sc. and Ph.D. degrees in computer science from the University of Science and Technology Houari Boumediene, Algiers, Algeria, in 2010 and 2014, respectively. She was an Assistant Professor at the M'Hamed Bouguerra University of Boumerdés from 2014 to 2016. She is currently a Post-Doctoral Researcher with the Computer Sciences Department, Université de Sherbrooke, Sherbrooke, QC, Canada. Her research interests span the areas of signal, image, and video processing, computer vision, and pattern recognition.

DJEMEL ZIOU received the B.Eng. degree in computer science from Badji Mokhtar University, Algeria, in 1984, and the Ph.D. degree in computer science from the Institut National Polytechnique de Lorraine, Lorraine, France, in 1991. From 1987 to 1993, he was as a lecturer at several universities in France. During the same time period, he was a Researcher with the Centre de Recherche en Informatique de Nancy, Nancy, France, and the Institut National de Recherche en Informatique et Automatique (INRIA), France. He has been a Visiting Professor at the Chinese Academy of Sciences, Shenzhen, Université Paris-Sorbonne, Université de Monastir, Université de Lorraine, INRIA Lorraine, and Sidi Mohamed Ben Abdellah University, Fes. He is currently a Full Professor with the Department of Computer Science, Université de Sherbrooke, Sherbrooke, QC, Canada. He is the Head of the Laboratory MOIVRE and the consortium CoRIMedia, which he founded. His research interests include image processing, information retrieval, computer vision, and pattern recognition. He has served on numerous conference committees and panels as a member or a chair. He was a recipient of the Prestigious NSERC Research Chair.