Real-time Moving Vehicle Detection, Tracking, and Counting System Implemented with OpenCV

Da Li[1, 2] Bodong Liang[1, 3] Weigang Zhang[2]

1. School of Automotive and Transportation Engineering, Shenzhen Polytechnic, Shenzhen 518055, China

2. School of Information Engineering, Chang'an University, Xi'an 710064, China

3. Corresponding author, Email: liangbodong@szpt.edu.cn

***Abstract* – Moving vehicle detection, tracking, and counting are very critical for traffic flow monitoring, planning, and controlling. Video-based solution, comparing to other techniques, does not disturb traffic flow and is easily installed. By analyzing the traffic video sequence recorded from a video camera, this paper presents a video-based solution applied with adaptive subtracted background technology in combination with virtual detector and blob tracking technologies. Experimental results, implemented in Visual C++ code with OpenCV development kits, indicate that the proposed method can detect, track, and count moving vehicles accurately.**

***Index Terms* – vehicle detection and tracking, video analysis, background estimation, virtual detector, blob tracking**

I. INTRODUCTION

Expressways, highways and roads are becoming overcrowded with increasing of large number of vehicles. Intelligent transportation systems (ITS), applied to collect, cognize, and manage information about transportation flows from various sources, are emerging worldwide to make transportation more efficient, reliable, cleaner and safer. The requirement to detect, track, and count the moving vehicle is getting very important for traffic flow monitoring, planning, and controlling.

The vehicle detections can be traditionally achieved through inductive loop detector, infrared detector, radar detector or video-based solution. Compared to other techniques, the video-based solutions based on surveillance camera mounted outdoor are easily influenced by environments such as weather, illumination, shadow, etc. However, because video-based systems can offer several advantages over other methods such as traffic flow undisturbed, easily installed, conveniently modified, etc., they have drawn significant attention from researchers [1][2] in the past decade.

A traditional computer vision method for moving objet detection in video-based system is so-called "background subtraction", or computing the difference between a background model and current frame, which demands to estimate a robust background to deal with the changing object. For this reason, in the case of vehicle detection on road, an adaptive rather than static background is needed for real-time road situations [3].

Regarding to real-time vehicle tracking system, the crucial issue is initiating a track automatically. Here we describe two systems in which the problem is attacked quiet differently. First one is virtual detector constructed with a set of rectangular regions in each frame [4][3]. Because the camera is fixed, the virtual detector can be chosen to span each lane and the system then monitors changes in area of virtual detector that indicate the presence of a vehicle. The second one is blob tracking [5][6]. In this algorithm, a background model is generated for the scene. For each input image frame, the absolute difference between the input image and the background image is processed to extract foreground blobs corresponding to the vehicles on the road. Both above two approaches have difficulty tackling shadows, occlusions, and large vehicles (e.g., trucks, trailers), all of which cause multiple vehicles to appear as a single region.

To overcome the previous limitations in vehicle detection and tracking, we present an improved method in this paper to accurately separate the vehicle foreground from the adaptive background model through a combination of Otsu's thresholding method [7][8] and moving cast shadow detection method [9][10].

This paper is organized as the following. Section II describes in detail the adaptive categorizing of background and foreground algorithm bundling with Otsu's method and shadow detection method. Section III and IV present the method of virtual detector and blob tracking, respectively. Section V shows experimental results implemented with Intel's OpenCV (Open Source Computer Vision Library) [11][12] to illustrate the performance of our solution. Section VI gives conclusion.

II. ADAPTIVE CATEGORIZING OF BACKGROUND AND FOREGROUND ALGORITHM

The system software deals with image frames read from video sequences, so that we can make image binarization. The whole processing flowchart is shown in Fig. 1.

The work described in this paper was partially supported by grants from Shenzhen Polytechnic (No. 2210K3100042 and No. 2613K3100021).

978-1-4799-4808-6 /14/$31.00 ©2014 IEEE

A. *Adaptive background generation in gray scale*

Firstly, our algorithm reads image frames from video, decomposes the initial 50 image frames into RGB channels, weights average of the top 50 frames, and gets an original background picture. This original background could be erroneous if there are moving objects present in the field of view, but it will converge to a robust background after handling more frames thanks to the operation of "Background update".

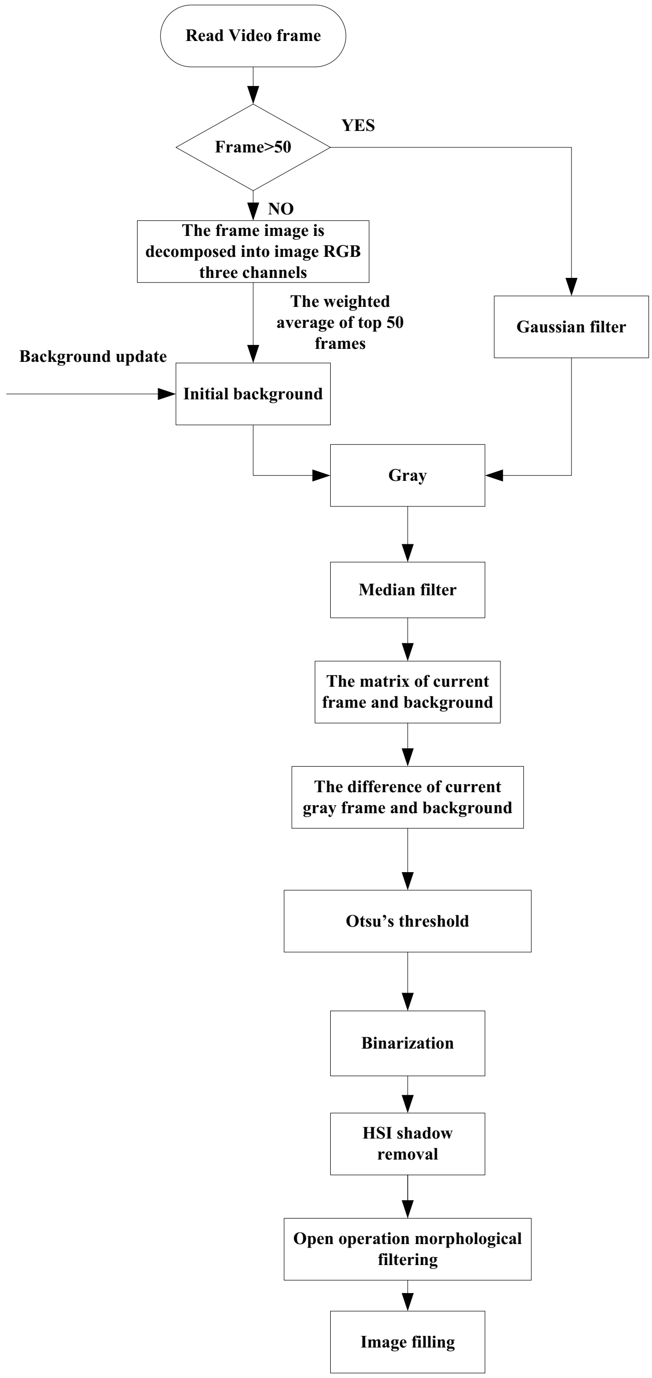


Fig. 1 Flowchart of algorithm processing.

After having original background, we apply Gaussian filters to smooth the following image frames and convert all RGB color images to gray ones. The background gray image and the current grayscale image are shown in Fig. 2(a) and Fig. 2(b), respectively.

Following software calculates the difference between the grayed image of background and the current image frame, that means we can extract the foreground presenting the moving objects shown in Fig. 2(c).

At the same time, the system updates in real-time the background image in the above difference calculation processing. The latest background image can be defined as a weighted sum of the current background and the extracted foreground.

B. *Binarization with shadow removal*

At following this software will reduce the gray level of extracted foreground to a binary image, in which we adopt the Otsu's algorithm to obtain the threshold value. Otsu's algorithm assumes that the image to be thresholded contains two classes of pixels or bi-modal histogram (e.g. foreground and background) then calculates the optimum threshold dividing those two classes so that their intra-class variance is minimal. Otsu's method is considered to be the best one to get threshold value in image segmentation algorithm, and is not affected by the image brightness and contrast, therefore it has been widely used in digital image processing.

After getting binary image by thresholding, we see some errors due to shadow appearance in the video. Therefore HSI shadow removal algorithm is used to segment the moving vehicle according to the characteristic of cast shadows in HSI color space. Our solution uses morphology filtering on image to remove the shadow, including steps in combination of corrosion and expansion. In real processing we firstly do corrosion and following by expansion, which is usually called open operation. It can eliminate small objects, separate boundary of the object in the fine points, smooth the boundary of the bigger objects but not significantly alter area. The binary image removing shadow is shown in Fig. 2(d).

C. *Filling in contour*

However, sometimes we find other problems like as over-segmentation, in which one vehicle is divided into two parts, and weak boundary, which is hard to detect. Therefore we also need to adopt algorithm to fill the current binary image.

The specific process is as follows: firstly making edge detection based on the extracted foreground gray level image, secondly adding the binary image removing shadow, and finally finding a closed contour and filling the region in contour. The three results are shown in Fig. 2(e), Fig. 2(f), and Fig. 2(g), respectively. It is clear that the binary image after filling the contour of moving objects has greatly improved over the original shadow removal binary image.

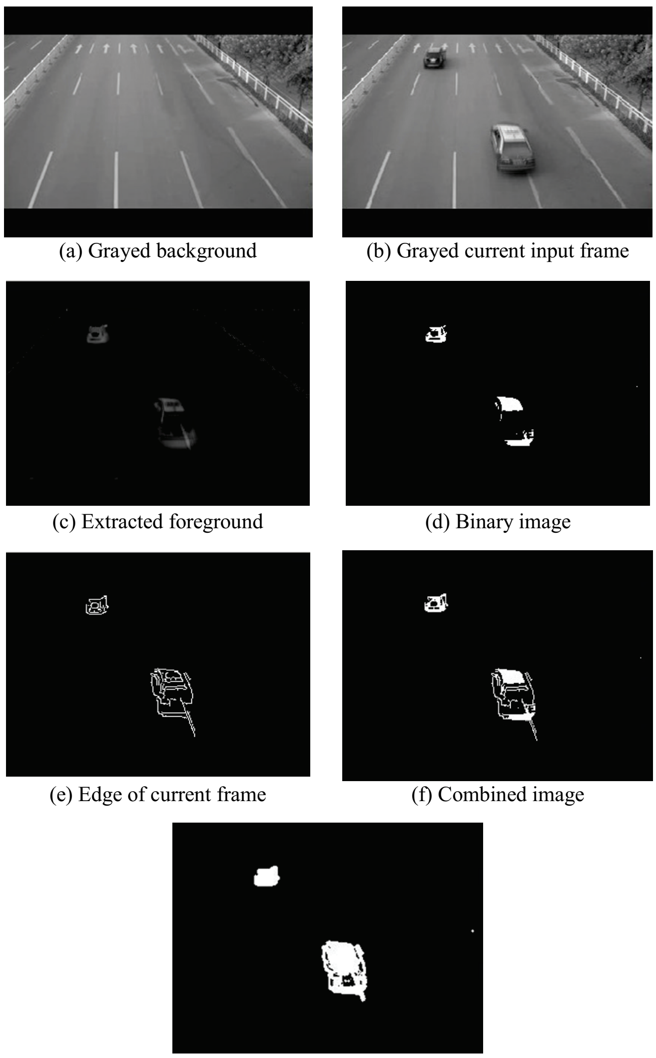
Then we will make process based on this filled image in later sections.

III. VIRTUAL DETECTOR

Here we shall present a technique called "Virtual Detector" to count the moving vehicle in each lane.

The traditional and popular technology to detect the vehicle on road is based on principle of electromagnetic induction. Its sensor is a real inductive loop being buried under the road. Once a vehicle enters, passes through, and leaves the lane area embedded inductive loop below, induced current will change and indicate the presence of vehicle. However, this technology in practical application will encounter some challenges such as difficult construction, expensive cost, short lifetime, etc.

In this paper, we propose a method in combination of vehicle detection and virtual detector to count the number of vehicles on road in real time. It draws a set of rectangular regions of interest (ROIs) in each lane as the virtual detector in image space, and determines the presence of vehicles by monitoring changes in area of virtual detector. The virtual detector in running system is highlighted with the thin red rectangular box and shown in Fig. 3(a). While vehicles pass through the virtual detector, pixel changes in histogram can be found as shown in Fig. 3(b).



(a) Grayed background (b) Grayed current input frame

(c) Extracted foreground (d) Binary image

(e) Edge of current frame (f) Combined image

(g) Binary image after filling

Fig. 2 Example of algorithm processing.

IV. BLOB TRACKING

To track a moving object is to detect the movement of the object, number the moving object, and obtain its trajectory. In our method we adopt the blob tracking technology to track and number the vehicles moving in the field of view from the recorded video camera.

The whole process of blob tracking can be divided into several modules at following: foreground detection module, new blob detection module, blob tracking module, path generation module, and trajectory of post-processing module. The processing flow is shown as Fig. 4 and all modules are described in detail at below:

a) *Foreground detection module*: To determine each pixel in image space belongs to foreground or background. This part of processing can be resulted from Section II;
b) *New blob detection module*: Using the latest foreground detected previously to check new blob corresponding to the vehicle just entering into the field of view on the road;
c) *Blob tracking module*: Monitoring the new blobs detected previously to initialize the tracking to all existing blobs;
d) *Path generation module*: To collect all trajectories of being tracked blobs and save the corresponding trajectory at the end of each trace (such as tracking lost, leaving scene, etc.);
e) *Trajectory of post-processing module*: Mainly to smooth the trajectory.

V. EXPERIMENTAL RESULTS BASED ON OPENCV

In this part, we shall apply above all steps from video sequences to detect, track, and count the moving vehicle on road. We shall compare the results from two solutions: virtual detector and blob tracking. All software programs are developed in Visual C++ code with OpenCV image development kits. OpenCV, standing for Open Source Computer Vision Library, is a library of programming

(a) (b)

Fig. 3 Virtual detector. (a) Be highlighted with red color of rectangular region in image space. (b) Pixels change in histogram while vehicle traversing

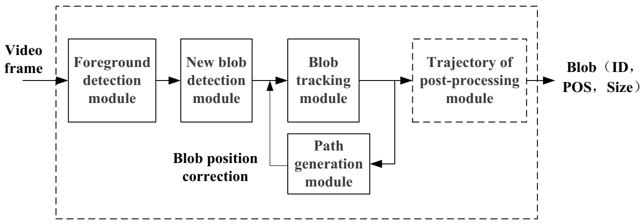


Fig. 4 Flowchart of blob tracking.

functions mainly aimed at real-time computer vision.

We mount a video camera on footbridge crossing a main street to monitor three lanes simultaneously and avoid erroneous counting caused by vehicle overlapping in image. The results of two different approaches and their respective accuracy rate are listed in Table 1. The accuracy rate reaches to 97.1% for virtual detector method, and 98.4% for blob tracking method. From the table, we observe that this system is able to detect, track, and count most vehicles successfully.

VI. CONCLUSION

Due to increasing demands in ITS, there is a huge amount of potential applications of detecting, tracking, and counting the moving vehicles on road in real time.

In this paper we presented unitized techniques to achieve the goal. In vehicle detection, we applied a method to accurately separate the vehicle foreground from the adaptive background model through a combination of Otsu's thresholding method and moving cast shadow detection method. In vehicle tracking and counting, we applied two methods (virtual detector and blob tracking) to double check the accuracy and maturity of our proposed method.

Experimental results, implemented with OpenCV, indicate that the proposed method is effective to detect, track, and count moving vehicles accurately.

REFERENCES

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

[2] Y. Liu, B. Tian, S. Chen, F. Zhu, and K. Wang, "A survey of vision-based vehicle detection and tracking techniques in ITS", *2013 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pp. 72-77.

[3] M. Lei, D. Lefloch, P. Gouton, and K. Madani, "A Video-Based Real-Time Vehicle Counting System Using Adaptive Background Method", *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pp. 523-528.

[4] G.D. Sullivan, K.D. Baker, A.D.Worrall, C.I. Attwood, and P.M. Remagnino, "Model-based vehicle detection and classification using orthographic approximations", *Image and Vision Computing*, vol. 15, no. 8, pp. 649-654, 1997.

[5] S. Gupte, O. Masoud, R.F.K. Martin, and N.P. Papanikolopoulos, "Detection and classification of vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37-47, 2002.

[6] S.C. Lee and R. Nevatia, "Speed Performance Improvement of Vehicle Blob Tracking System", in *Multimodal Technologies for Perception of Humans, Lecture Notes in Computer Science*, vol. 4625, pp: 197-202, 2008.

[7] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62-66, 1979.

[8] P.S. Liao, T.S. Chen, and P.C. Chung, "A Fast Algorithm for Multilevel Thresholding", *Journal of Information Science and Engineering*, vol. 17, no. 5, pp. 713-727, 2001.

[9] B. Sun, S. Li, "Moving Cast Shadow Detection of Vehicle Using Combined Color Models", *2010 Chinese Conference on Pattern Recognition (CCPR)*, pp. 1-5.

[10] A. Sanin, C. Sanderson, and B.C. Lovell, "Shadow detection: A survey and comparative evaluation of recent methods", *Pattern Recognition*, vol. 45, no. 4, pp. 1684-1695, April 2012.

[11] OpenCV User Site: http://opencv.org/

[12] OpenCV on SourceForge: http://sourceforge.net/projects/opencvlibrary/

TABLE 1 EXPERIMENTAL RESULTS BASED ON OPENCV

Number of input frames	Lane			Total number of vehicles in ground truth	Counting number		Accuracy rate	
	1	2	3		Virtual detector	Blob tracking	Virtual detector	Blob tracking
60	1	0	1	2	2	2	100%	100%
300	3	1	2	6	6	6	100%	100%
450	6	3	8	17	18	18	94.4%	100%
876	11	9	14	34	35	33	97.1%	94.2%
1401	15	15	18	48	50	49	96.0%	98.0%
1812	23	22	30	85	88	84	96.6%	95.4%
2512	41	31	50	122	126	124	96.0%	98.4%