Traffic Flow Statistic Based on Computer Vision

Liu Qiushi, Ray C. C. Cheung, Fu Ziyi, and Sun Yihang

***Abstract*—With the development of economy, transportation is becoming more and more important, so a new method to control traffic is needed. Intelligent traffic system emerges as the time require. In this intelligent system, it is the basic that acquire traffic information instant. We used lots of way to get that information, such as induction coil, ultrasonic devices, microwave devices etc,. These tradition ways performed not as except. The cars on the road may block each other in different lane, so the result from the tradition devices may incorrect. Improving the tradition ways, we choose to use computer vision to statistic traffic flow. We use Hough Transform and Gaussian Mixture Model to detect road and cars. The whole system can be achieved on OMAP4460. We test the whole system on a real road, we captured all the cars and according to the test the road is detected accurately. The system performance is as we except.**

***Index Terms*—Accurate, hough, real-time.**

I. Introduction

There has been some method to statistic the traffic flow based on computer vision, such as background subtraction or optical flow. But the result is not accurate, because they cannot filter the interference from the opposite lane and the light changes will influence the background so that the systems bring up the wrong result. Background subtraction is a simple way to find the foreground and background. We always save an image of the background and subtract the new picture we captured, utilize the gray value, we can find out the cars. But it will be influenced by the weather, light changes or even the road ageing. The optical flow is so complex. We need too much hardware resources, so it is not practical.

In a long time consideration, we generate a new system to conquer these problems. At first it is important to filter all the possible interference outside the road, because the car from the opposite or the pedestrians on the road will bring lots of wrong result. In our test, we found that the pedestrians are easily recognized as a moving car. All the cars captured in the screen will be detected no matter which lane it is. So all these interference should be filtered firstly.

We use Gaussian Mixture Model (GMM) to detect foreground. With the analysis of probability density functions, it can be approximated using Gaussian mixture model. We can improve the result by changing parameters. Compared with other models, GMM can improve the problem which is brought by light changes and shadow.

Our system is generated on SimLink/MATLAB. The system has been tested on different real roads. It shows its accuracy and instantaneity. It runs on a DSP + ARM embedded development board with a web camera, Logitec C410. During the whole test, no one car missed even motor bikes. In this work, we will present how we build the whole system and the entire details. We will also show all the test result.

Manuscript received November 18, 2012; revised January 28, 2013.

Liu Qiushi, Fu Ziyi, and Sun Yihang are with the Beijing Information Science & Technology University, Beijing, 100020 China (e-mail: lqs1990@ gmail.com, 465625854@qq.com, syh199011140 @qq.com).

Ray C. C. Cheung is with the City University of Hong Kong, Hong Kong (e-mail: cheung@ieee.org).

II. Method

Nowadays, we need to take a consideration of all the interference in the traffic flow system. There are pedestrians, cars from different lanes, bicycles even wild animals. We also need to think about all these interference in different conditions, so it is a very complicated problem. It can be expressed by a math equation. As Eq. (1):

$$Info = \sum_{m,k=0}^{n} b_m + c_m + i_{m,n} \tag{1}$$

In Eq. (1), $Info$ means all the information in the video stream we captured. b_m means the back ground in different conditions. c_m means the cars we need to detect in different conditions. $i_{m,n}$ means the different interference in different conditions. So it is hardly processed.

But actually, we just need the information on the road. We do not need to worry about the inference from outside and there are too less interference on the road, so we can simplify Eq. (1) into Eq. (2):

$$Info = \sum_{m=0}^{n} b_m + c_m \tag{2}$$

We can see that there are only background and cars in this equation. The car is what we need, so we just need to handle the background issue. No matter what conditions the cars are in, all we need to do is capture them and due to the limit range of background, we only need to consider the light changes. That makes this issue more easily. In Eq. (3), we can see a very simple model.

$$Info = c + \sum_{m=0}^{8} b_m \tag{3}$$

In Eq. (3) we can see only 8 types background. Because the light changes very little and gradually, we can divide the

DOI: 10.7763/IJCCE.2013.V2.198

whole day into 8 or less light conditions. Also, we could change the parameter of GMM to adapt the new background. The model can be easily achieved. We can refresh the background periodically.

We simplify a very complicated issue into a simple question by limiting the range of whole images. There are two figures which can show how it changes. Both these figures are captured in a real road, and Fig. 1 is captured in the road detection system which is a crucial part in the whole system.

Fig. 1. Whole range of an image.

Fig. 2. After limit the range of image.

Compared with the two figures, we can see in Fig. 2 we just need to process only a little part of the entire image. More importantly, we can filter some complicated interference.

Fig. 3. Interference.

In Fig. 3, we highlight some interference outside the road. They are complicated and hard to define in a math model. So it is hardly filtered in a traditional traffic flow system. We can avoid wrong statistic by using limit the range of image. Furthermore, it will save lots of hardware resource.

In conclusion, we improve the accuracy of traffic flow statistic by simplifying a complex issue properly while keeping the real-time character and saving more hardware source.

III. Road Detection

As we present, road detection is a crucial part of this system. The road detection system can be divided into three steps, edge detection, Hough line and logic calculation. The result come from Hough Transform Block is coordinates of four points, but in Fig. 1 and Fig. 2 we can see that the shadow area is a polygon, so we still need to do a coordinates transform. All these steps are all achieved in MATLAB; therefore, it is easy to read.

A. Edge Detection

Edge detection is needed for Hough Transform. In other words, it influences the Hough Transform directly. As we know, Hough Transform can help us find out the line with common characters easily, but in an image, there are so many lines, so we need to do edge detection first to find the edge of the road we need.

We choose to utilize the different Chroma of a picture to find the road edge. At first, we need to transform RGB to HSV. S is saturation; it means that the ratio between the selected color purity and the largest purity color. So we need to set a threshold value to find the road part and non-road part. In that case, it is easy to find the edge. The values of S are between 0 and 1. After times test, we finally set this threshold value as 0.25. We only output the pixel which the saturation value is greater than 0.25. That will reduce most noise so that create a better environment for Hough Transform. After close and erode operation, the figure will be in a good condition to do Hough Transform to find the road edge.

In Fig. 4 and Fig. 5 we can the difference.

Fig. 4. Saturation.

Fig. 5. Saturation value greater than 0.25.

B. Hough Transform

Hough Transform was come up by Paul Hough in 1962. It achieves mapping relations from image space to parameter space. Due to some obviously advantages, it is concerned by scholars and engineers. Actually Hough Transform uses parameter estimation based on voting principle. It utilizes the duality relation between point and line and transforms the

detection issue in image space to parameter space [1]. By accumulating and statistic, it is easily to find a peak value, and the coordinate point of the peak value in parameter space corresponding with an only line in image space. There is a fantastic effect in two dimension space [2].

In our project, we use Hough Transform to draw a line on the edge of the road. Then, we can calculate the four coordinate points of the road endpoint [3]. Because the road edge may intersect with left and right borders of the screen, the road area will be a hexagon or pentagon. It is necessary for us to do a logic judgment to refresh the coordinates [4]. In Fig. 5, the right boundary of the road beyond the screen size, in that case, we can set the coordinates of the right corner point an experienced value.

Fig. 6. Fill the road.

We can see in Fig. 6, the road is filled into white. There is a little error, because of the shooting angle. But we have filtered most interference outside the road. At present, we still operate the image in RGB space. For the following step, it is necessary for us to transform this image into intensity space. Furthermore, if we want to obtain the result like Fig. 2, we will need to fill the outside road part black. We can achieve that by set a threshold value in an intensity space. In terms of image color space, we only need the white part in Figure6, so we set the threshold 1. Only when the value of pixel is greater than 1 or equal with 1, it will be displayed.

Currently, we get a black-and-white image, the road part should be white and the outside part is black. Now, we need to connect this image with the whole video stream. In our system, we multiply the black-and-white image with every frame of the video stream. It makes an image processing issue into a math issue. In intensity space, white is 1, and black is 0. An image can be presented as a matrix. Because the frame and the black-and-white image has a same size, they can do a multiply operation. Or a logical 'AND' can achieve this too. After multiply, we will get a picture like Fig. 2.

We finish the road detection part now. There exist a little error, but we already filter the most interference outside the road. There are too many types of the road edge, such as the example we give. The dam-board influences the Hough Transform, but we still find the road, and the test result shows that even in this condition, the system still performs as excepted.

We applied Hough transform into road detection. Compared with traditional way, Hough transform can avoid the side effect caused by light change. Even more, Hough transform can fix the defect of the road in a flexible range.

IV. Car Capture

This is the last step of the whole system. We choose to use Gaussian Mixture Model (GMM) and Blob Analysis to find cars [5].

Gaussian Mixture Model can adapt itself for the background change. Because we build this system on MATLAB, it will be easy to use GMM. There are only a few parameter needed to set. 'video. Foreground Detector' can be used directly. Given a series of either grayscale or color video frames, the object computes the foreground mask using GMM [6], [7].

```
hfdet = vision.ForegroundDetector(...
    'NumTrainingFrames', 30, ...
    'LearningRate',0.001,...
    'MinimumBackgroundRatio',0.8,...
    'NumGaussians',3,...
    'InitialVariance', 30^2);
```

Fig. 7. Parameters set.

In our system we store 30 frames at first to detect background, so we set 'NumTrainFrames' 30. Because we have already filtered most interference, we do not need so much Gaussians, we set 'NumGaussians' 3. 'InitialVariance' is determined by the input types. For unit8 a typical value is 30^2. 'LearningRate' and 'MinimumBackgroundRatio' are determined by test and experience.

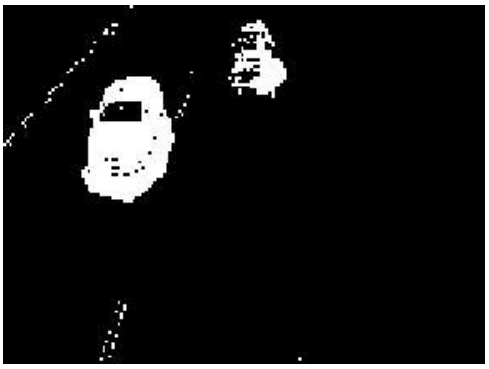

Fig. 8. GMM result.

In Fig. 8, the cars are detected by GMM. We can see on the front windshield position, due to the light issue, it does not connect with the car body. That goes against the Blog Analysis. So a close operation is needed. We can see the different between Fig. 8 and Fig. 9.

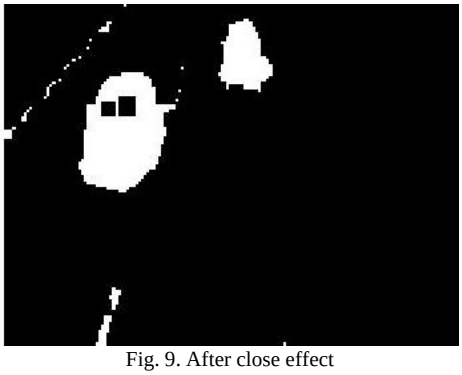

Fig. 9. After close effect

Blob Analysis analyses the connected domain of an image, and the connected domain area is called Blob. That the reason why we need to do a close operation. Blob analysis can provide location, shape, direction, numbers information to computer vision system. But Blob Analysis can only be used on a high intensive picture. SimuLink provides a Blob Analysis block; it can finish a Blob Analysis easily and feed back position information to us. We can draw a polygon with this position information.

Until now, we finish the whole system. We can do an optimizing for this system. We need to run the traffic flow system on an embedded development board, so we should keep the system efficiency.

V. Optimize

For saving hardware resource, and keep the real-time character. We can resize every frame firstly. When the system read video from web camera, the original size is 640 × 480, it is too large for the hardware to store 30 frames to detect the background. So we resize the video to 25%. As we tested, if we keep the original size of the picture, it will cause 'Memory allocate' issue. Before output the result, we still need to regain the size. The coordinates point information is also need to gain 4 times. If we keep all the data proportional, the result is still correct.

For using the hardware resource more efficiently, we can divide the Hough Transform. At first, we need to divide Fig. 10 into 2 parts, left part and right part. Then, do Hough Transform alternately for every part. If we do Hough Transform to a whole picture, the calculated quantity is huge and will bring more error. The crossing of two line will bring in image space will bring more lines in parameter space.

It is unnecessary for us to detect the road all the time. We can detect road regularly, for example, we can detect the road 3 hours one time. That will make the whole system more stable and the hardware resource is saved greatly. In our system, we do road detection for every frame of the video stream. But the road is very stable; it is no need for us to detect the road all the time. We can cut the road detection regularly. All the optimization can be achieved by using a state-flow to restrain every block in this system.

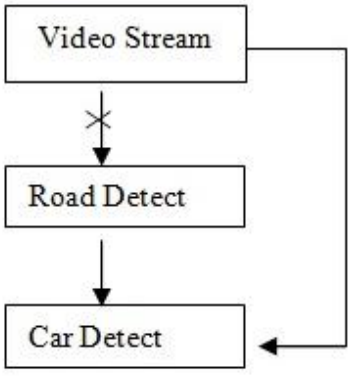

Fig. 10. Optimize the hardware resource.

According to our test, it is shown occasionally delay in this system without any optimization so that we cannot keep the real-time character. But after optimizing the system, all the programs run smoothly.

VI. Result and Conclusion

We captured 11 video stream files, nearly20 minutes. Table I shows the test result.

TABLE I: Test Result

Video No.	Test Result	Actually Result	Error
1	55	53	2
2	24	24	0
3	27	26	1
4	25	25	0
5	42	41	1
6	46	44	2
7	32	31	1
8	17	17	0
9	40	39	1
10	12	12	0
11	23	22	1
Total	343	334	9

The main reason for the errors is that the car light is recognized as cars. Because of shadow of cars and the near position, two cars might be connected by the close operation. But the result is satisfied for us. We can modify the Gaussian parameters and change threshold value to improve this issue.

Compared with the traditional way to statistic traffic flow, the new method we present is more efficient and economical. More importantly, this system can be updated easily. We utilize Hough transform to detect the road and filter all the interference outside the road. By using this way, it can reduce the complex of algorithm for filtering interference. People always color segment to detect road, but it will be influenced by sun light. Our method can avoid this matter effectively.

Further work includes optimize the whole system to integrate more camera on one chip.

Acknowledgment

Thanks for the supporting from Beijing Information Science& Technology University. The 501 lab provides us experiment equipments and software. Thanks for Professor Wu who works in the 501 lab gave us practical advice.

References

[1] J. Illingworth and J. Kittler, "A survey of the hough transform," *Computer vision, Graphics, and Image Processing*, vol. 44, pp. 87-116.
[2] S. R. Deans, "Hough Transform from the Radon Transform," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. PAMI-3, no. 2, March 1981.
[3] R. O. Duda and P. E. Hart, "Use of the hough transform to detect lines and curves in pictures," *Commun. Ass. Comput*, vol. 15, pp. 11-15, Jan. 1972.
[4] N. Guil, J. Villalba, and E. L. Zapata, "A fast hough transform for segment detection," *IEEE Trans. On Image Processing*, vol. 4, no. 11, November, 1995.
[5] L. Zhao "Probability density function approximation using mixture model," *Radio Communications Technolog Y*, vol. 2, no. 33, pp. 20-22, 2001.
[6] G. Yan, G. D. Cui, and M. Yu, "Player detection algorithm based on gaussian mixture models background modeling," *Computer Simulation*, vol. 27, no. 9, pp. 259-262, 2009.
[7] H. H. Chen and T. S. Huang, "Matching 3-D line segments with applications to multiple object motion estimation," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 12, no. 9, pp. 1002-1008.

Liu Qiushi was born in Beijing in October 10th 1990. He is in Beijing Information Science & Technology University for a Bachelor degree and majoring in Electronic Information Engineering. His research interest is hardware design and digital signal processing. He is professional in ISE, MATLAB and Quartus II. Mr. Liu received the second prize scholarship for one time and the third prize scholarship for two times, respectively in 2011, 2011 and 2012.

Ray C. C. Cheung received the B.Eng(Hons) and M.Phil. degrees in computer engineering and computer science & engineering from the Chinese University of Hong Kong in 1991 and 2001 respectively, and the DIC and Ph.D. degree in computing from Imperial College London (IC) in 2007. He joined the Department of Computer Science & Engineering at the Chinese University of Hong Kong in 2002 as an Instructor. Before that, he worked as a system administrator in a parallel cluster computing company, Cluster Technology Limited for one year. Two years later in 2004, he received the Hong Kong Croucher Foundation Scholarship and moved to London where he spent three years for his Ph.D. study. He is currently an Assistant Professor in the Department of Electronic Engineering at City University of Hong Kong, and with Digital System Lab. His current research interests include cryptographic hardware design and design exploration of System-on-Chip designs and embedded system designs. Dr. Ray is the newsletter and web editor of the SIGDA UK chapter. He is the author of 10 journal papers and over 20 conference papers.

Fu Ziyi was born in Beijing in September 23th 1990. He is a student in Beijing Information Science & Technology University majoring in Electronic Information Engineering. He is professional in MATLAB and CCS, especially the coordination between different software. His research interest is digital image and video processing and data mining. Mr. Fu received scholarship from Beijing Information Science & Technology University in 2012.

Sun Yihang was born in Beijing in November 14th 1990. He is studying in Beijing Information Science & Technology University and majoring in Electronic Information Engineering. He is professional in MATLAB、CCS and EDIUS. Most of his projects focus on wireless communications and the technology of digital signal processing, especially in the realm of digital image and video.Mr. Sun received scholarship from Beijing Information Science & Technology University in 2011 and 2012.