ESTIMATING THE QUEUE LENGTH TO OPTIMIZE THE GREEN TIME FOR AN URBAN TRAFFIC CONTROL SYSTEM

Juan D'Amato[1], Pablo Negri[1] and Pablo Lotito[1]

[1]*PLADEMA, Universidad del Centro de la Provincia de Buenos Aires, Tandil, Argentina* , {*jdamato,pnegri,plotito*} *(a) exa.unicen.edu.ar*

Abstract: We describe a queue length estimation module that is part of an Urban Traffic Control System (UTCS). The main objective of this module is to estimate the traffic queue length at an intersection using image processing algorithms. This measurement feeds a traffic green time optimizer that chooses adequate green periods, based on the traffic load, and reduces the global time spent by drivers.

We compare the results of three algorithms that estimate the queue length based on Gaussian Mixture Models and Level Lines. A hybrid method, which combines both approaches, is better at minimizing the system errors. In addition, this framework makes it possible to test different camera positions to improve results.

Keywords: *Video processing, traffic flow, estimating*
2000 AMS Subject Classification: 21A54 - 55P54

1 INTRODUCTION

This paper presents and analyzes a methodology used in the congestion reduction module of a UTCS. This methodology incorporates recent advances in image processing, simulation models and control and optimization methods. The main objective of this module is to adapt the computation of adequate green times to demand variations (traffic load), and to reduce the global time spent by drivers accordingly. In order to achieve this goal, the full state of the traffic network should be known in each step [4].

Historically, inductive loops have been the most widely used sensors to measure the queue length. The vehicles entering and exiting the road section are detected, and the number of vehicles on the local road section is calculated by difference [3]. Many algorithms, which depend on the data provided by the loop detectors (number of vehicles, individual speed, etc.), still have some problems to be solved. As the algorithms need accurate data, the loss of one loop dramatically disturbs the system, and the length of the queue is under-estimated. Additionally,the number of loop detectors increase rapidly according to the number of roads at the inspected crossing and the number of lanes on each road. In contrast, the cost of one camera covering all the lanes of a road is equal to the cost of one loop in a lane.

Our application uses a computer vision system to estimate traffic metrics, such as the queue length. Several tests used to design the queue measurement system through virtual video captures simulating strategically placed cameras are presented. The advantage of 3D virtual reconstructions is that they yield the complete state of the system and make it possible to analyze and compare different configurations in order to reduce measurement errors. An efficient algorithm to estimate the queue length, which is advantageous as it does not increase the overall computation time, is also shown.

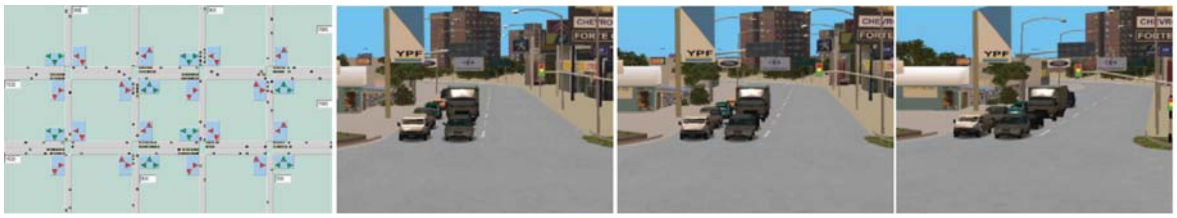

Figure 1: Screen-shot of the UTCS at PLADEMA showing three views of the same intersection generated by the 3D visualization module.

2 Measuring System Using Computer Vision

The 3D virtual reconstructions are generated by a UTCS module developed by this group. The visualization module recreates the vehicle movement in a realistic 3D scenario and generates synthetic videos subsequently used in our study. This virtual tool makes it possible to analyze scenarios with structural variations (e.g., changing the camera position on a corner) and provides instantaneous information, such as vehicle speed and position used for automatic validation. Figure 1 shows different points of view of the same state of the simulator, located in three places where cameras could be potentially installed. This methodology, compared to video campaigns, leads to a significant reduction in the time and cost of video acquisition, at the same time the analysis task is significantly reduced.

To start our study, we must first define a polygonal section for each view. This polygon is separated into two lines and then into four cells each, representing on the image the place where the vehicles are queuing up. The first cell of each line is the head of the queue. As we are modeling a traffic light cycle, the vehicle movement discriminates both cycles (green and red). We assume that a vehicle queue exists if there is detection in a cell without motion (red cycle). The queue length is defined as the amount of occupied cells without motion if the first one (the head) is stationary.

We analize three algorithms to estimate occupancy and motion on the images. These strategies model the background to detect differences between consecutive frames in a video. The first one uses Gaussian Mixture Models (GMMs) [6], and the second one uses Level Lines [1] but as both have certain limitations, we propose a new strategy that keeps their advantages and enhances the global performance.

2.1 Gaussians Mixture Models

The detection method proposed for Stauffer [6] considers each pixel as an independent statistical process where the n values over time of each pixel can be modeled by K Gaussians. The background is made up of those Gaussians which are stables in a sequence. One pixel is considered as a foreground (movement) if their probability of belonging to one of the K Gaussians background models is lower than a threshold. A pixel $\mathbf{p}$ belonging to the foreground switches on the binary detection matrix: $D(\mathbf{p}) = 1$.

To determine if a cell is occupied in our model, we consider all pixels $\mathbf{p}$ of the foreground ($D(\mathbf{p}) = 1$) contained in it. Using a similar approach to [7], the number of detected points must be divided by the cell area to get a standarized measure. If this value is greater than a threshold, the cell is assumed to be occupied. Formally, for each cell C_i we obtain ratio d_i:

$$d_i = \frac{N(D_i)}{A(C_i)} \geq T_i^d$$

where $N(D_i)$ is the number of detections in cell i and $A(C_i)$ is the area of cell i.

The motion detection method computes the difference of two successive frames in the sequence. This operation is performed only on the foreground pixels $\mathbf{p}$ with $D(\mathbf{p}) = 1$. We get motion matrix M:

$$M(\mathbf{p}) = \{\mathbf{p} | D(\mathbf{p}) = 1 \wedge |F_t(\mathbf{p}) - F_{t+1}(\mathbf{p})|\} \tag{1}$$

To estimate the movement inside a cell, we accumulate the values of motion pixels $\mathbf{m}$ and divide them by the number of detected pixels, obtaining a new ratio m_i for cell i:

$$m_i = \frac{S(M_i)}{N(D_i)} \geq T_i^m$$

If m_i is greater than threshold T_i^m, we consider that there are a moving vehicles in cell i.

2.2 Levels Lines

Other effective detection method, using objects contours as primitives, is the Level Lines representation [1]. The Level Lines are defined as follows. Let be $I(x, y)$ be the intensity value on image I at coordinates (x, y), the level set of I is the set of pixels $\mathbf{p}(x, y)$ whose intensity is greater than or equal to λ:

$$\Xi_\lambda = \{\mathbf{p}/I(\mathbf{p}) \geq \lambda$$

The boundaries of a level set Ξ_λ are called Level Lines λ ($LL\lambda$) and, for the same image, we can obtain a set of $LL\lambda$ choosing different thresholds $\lambda = \{0, 1, 2, ..., 255\}$. Each $LL\lambda$ also contains information about the direction of the level line in the pixel. The direction of the pixel $\mathbf{p}(x, y) \in LL\lambda$ is calculated as the slope of the straight line created by the neighbors of $\mathbf{p}$, using the least square method. Then, this value is quantified in η values (for our application $\eta = 12$). A binary function $f_t(\mathbf{p}, \theta_k)$ signals the presence of a direction θ_k on the pixel $\mathbf{p}$ at the time t, and $1 \leq k \leq \eta$. Pixel $\mathbf{p}$ belonging to the background model have a stable $f_t(\mathbf{p}, \theta_k)$ for all the directions, during an interval of time T.

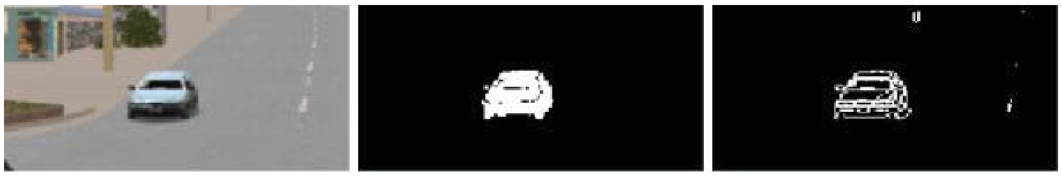
Figure 2: Detection matrix D for the GMM and the Level Lines algorithms.(left) Capture (center) GMM detection (right) LL detection

One pixel $\mathbf{p}(x, y) \in LL\lambda$ will be considered as a detection if its directions θ_k do not match the direction of the reference at the (x,y) location. Then, the detection matrix is switched on at this pixel: $D(\mathbf{p}) = 1$ (see Fig. 2).The vehicle detection follows the same procedure as that of the GMM algorithm, counting the number of detected lines in the cell.

In the motion detection, we match the detections in the previous frame $D_{t-1}(\mathbf{p})$ and the present frame $D_t(\mathbf{p})$. We define a new matrix $H(\mathbf{p})$ where each point $\mathbf{p}$ has a distance to the nearest detection $D(\mathbf{p}')$. For each $\mathbf{p}$ where $D_{t-1}(\mathbf{p}) = 1$ (a detection), we look for the most similar pixel $\mathbf{p}'$ in D_t in a NxN centered window at $\mathbf{p}$, using an intensity difference criterion $I_{t-1}(\mathbf{p}) - I_t(\mathbf{p}')$, the number of differences in the directions between $f_{t-1}(\mathbf{p}, \theta_k)$ and $f_t(\mathbf{p}', \theta_k)$, and comparing $H_t(\mathbf{p})$ and $H_{t-1}(\mathbf{p}')$.

Each pair of matched points $\mathbf{p}$ and $\mathbf{p}'$ generates a velocity vector $\vec{\mathbf{v}}_i = \overrightarrow{\mathbf{p}_i\mathbf{p}'}_i$. Inside the cell, we add up all the generated velocity vectors and the modulus is divided by the number of detected pixels to obtain a motion ratio. If this ratio is greater than a threshold, we conclude that a moving vehicle is inside the cell.

2.3 Hybrid Algorithm

We propose a new hybrid algorithm that improves the system performance while preserving the advantages of the previous methods. On the one hand, the GMM method detects vehicles more slowly but more accurately than the LL method (as we will see in the results). On the other hand, the LL method provides a velocity vector, which can represent the speed of a vehicle in the cell. It is impossible to obtain these vectors through the motion detection algorithm of the GMM method.

To compute vehicle detection, the LL algorithm computes the movement in the frame very quickly. The resulting curves D_t^{LL} are projected on a horizontal profile. The GMM algorithm is applied only to the areas of the image where the profile has a value that is greater than a threshold. Then, we obtain the detection matrix D_t^{GMM}, speeding up the vehicle detection.

As it is very expensive to compute the motion vectors over all the pixels of the level lines, we use motion matrix M (see eq. 1) where $D_t^{GMM}(\mathbf{p}) = 1$ instead. Those pixels where intensity difference is greater than a threshold generate a binary mask of the level lines in the frame D_{t-1}^{LL}. The velocity vectors are then generated by matching the pixels in the mask with the corresponding ones in I_t.

3 Results

The test sequence is made up of a virtual scene with more than 50 queues. Among other advantages already mentioned, the virtual simulator provides detailed information, such as vehicle position and speed at any time. In this regard, the user does not need to label the images used in learning and testing, nor even estimate the vehicle speed, which is a rather complex task. The motion T_i^m and detection T_i^d thresholds

Table 1: Table of results for the detection algorithms.

Views	Level Lines			Gaussians Mixture Models			Hybrid Method		
	FP (%)	FN (%)	Mean er.	FP (%)	FN (%)	Mean er.	FP (%)	FN (%)	Mean er.
1	0.78	20.43	0.65	1.31	5.20	0.92	4.34	2.64	0.85
2	1.09	23.08	0.55	0.50	9.87	0.59	3.44	4.43	0.55
3	0.67	13.05	0.62	1.22	7.71	0.57	2.59	2.61	0.61

of each detection method were adjusted to minimizing the errors. To analyze the results of the algorithms, like in [2], we define the false detections rate (false positives (FPs) over the total number of queues), the non-detection rate (false negatives (FNs) over the total number of queues), and the queue length mean error (in cells) with respect to detected queues. Table 3 shows the results obtained on assessed sequences.

The LL method yields a very large number of FNs, especially in the presence of dark vehicles, as detected lines are not enough to trigger the detector cell occupancy. The GMM and the Hybrid methods have fewer FNs and the latter has the best results for this evaluation metric. In fact, this is the most important metric because an FN error means that the UTCS required the queue length and the detection system estimated that the vehicles were not queuing up. The worst queue length error is observed when there is a truck in the line. Because of its height, the truck detection fills the cell behind it, increasing the error.

The proposed Hybrid algorithm shows an intermediate behavior between the GMMs and Level Lines, reducing the FN percentage and, at the same time, providing reliable information about the vehicle speed. As regards the view configuration, the best one can be found between views 2 and 3. The problem of view 1 is that vehicles in the same row are concealed, as can be seen in Figure 1. This effect is reduced in view 2 and 3, concluding that the installation position should be somewhere between them. For view 3, it is also possible to use algorithms like those proposed in [5] to reduce the occlusions.

4 Conclusions

A framework for the evaluation of video analysis algorithms to detect the length of a queue in front of a traffic light has been presented. We used three algorithms to count the number of vehicles waiting at the red light. Two of them were based on classical methods: Gaussian Mixture Models and Level Lines. The third one is an efficient combination of the other two, and it obtains better results minimizing the system errors. This framework also allows the user to test different camera positions to improve results.

As an extension of our method, we plan to incorporate a line length estimator, which tracks incoming vehicles in the sequence video before they queue up. Thus, we would be able to handle cases of undetected vehicles hidden by others.

References

[1] D. Aubert, F. Guichard, S. Bouchafa, *Time-scale change detection applied to real-time abnormal stationarity monitoring*, Real-Time Imaging, Vol. 10, 2004, pp 9-14.

[2] D. Aubert, F. Boillot,*Automatic measurement of traffic variables by image processing application to urban traffic control*, Recherche - Transports - Securite, Vol. 62, pp. 7-21, 1999.

[3] L.A. Klein, M.K. Mills and D.R.P. Gipson, ”Traffic Detector Handbook: Third Edition”, Technical Report, Vol. I & II, 2006.

[4] Mayorano, F., Rubiales, A., Lotito, P. A., *Optimal Control Based Heuristics for Congestion Reduction in Traffic Networks*, In: Proceedings of EngOpt, Brazil, 2008.

[5] C.C. Pang, W.W. Lam, N.H. Yung, *A method for vehicle count in the presence of multiple-vehicle occlusions in traffic images*, Intelligent Transportation Systems, Vo. 8, No. 3, pp. 441-459, 2007.

[6] C. Stauffer, W. Grimson,*Adaptive background mixture models for real-time tracking*, In: Proceedings of CVPR Vol. 2, 1999.

[7] M. Zanin, S. Messelodi, CM. Modena, *An Efficient Vehicle Queue Detection System Based on Image Processing*, In: Proceedings of ICIAP, pp. 232-237, 2003.