Online Vehicle Detection For Estimating Traffic Status

Ranch Y.Q. Lai

Department of Computer Science

Hong Kong Baptist University

Email: yqlai@comp.hkbu.edu.hk

Abstract—We propose a traffic congestion estimation system based on unsupervised on-line learning algorithm. The system does not rely on background extraction or motion detection. It extracts local features inside detection regions of variable size which are drawn on lanes in advance. The extracted features are then clustered into two classes using K-means and Gaussian Mixture Models(GMM). A Bayes classifier is used to detect vehicles according to the previous cluster information which keeps updated whenever system is running by on-line EM algorithm. Experimental result shows that our system can be adapted to various traffic scenes for estimating traffic status.

I. INTRODUCTION

As vision-based traffic monitoring systems become more and more prevalent due to their low-cost and easy-to-deploy aspects, research on application of computer vision to traffic state measurement attracts more interest than before. Although the history of traffic monitoring system can be at least traced back to 1991[1] and new approaches are emerging[2], various problems are still to be tackled. Situation is much worse in urban areas, where congestion most likely occurs, especially at the intersections of downtowns. Monitoring the urban scenes and highways has been extensively studied. Most of the approaches are based on background extraction (or motion detection) and vehicle tracking[3][4][5][6]. These approaches are common and prevalent because 1) background can be easily extracted because of light traffic condition. 2) Tracking is straightforward as targets are readily segmented. 3) Trajectories will be available for even analysis such as retrogradation, aberrance, clash, etc.

However, research on highways and suburban areas cannot necessarily apply to urban areas, in which there are two main problems need to be solved if we rely on vehicle tracking. One is that background rarely reveals in most situations of urban environment. For example, at the intersections, the whole lanes may be dominated by vehicles in waiting queue for a very long time as vehicles go and come. The other reason is that, tracking is difficult (if not impossible) because of the difficulty in vehicle segmentation when traffic jam happens, even though the background information may be still pure. In this case, vehicles are so heavily (partial or total) occluded that most of the tracking strategies fail to work because of difficulty in segmentation[7]. Blob tracker[8] is likely to link two or more vehicles together either by region or by silhouette[8]. Point tracker[3][9] may falsely group features from different targets. Instead of taking these approaches which rely on background
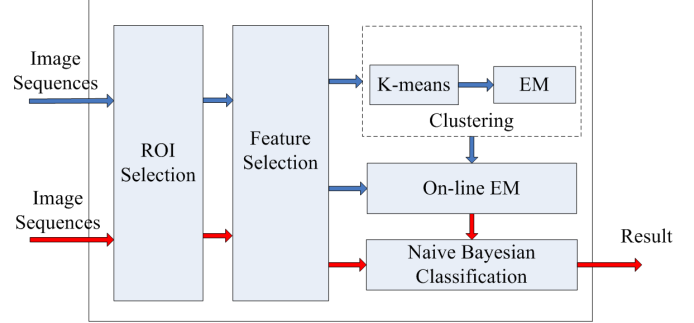

Fig. 1. Block diagram of our traffic state monitoring system.The blue arrows show the learning path whereas the red arrows show the classification path

and segmentation, we propose a novel approach to measure traffic state at the congested intersections.

Our system does not involve background extraction and vehicle segmentation. Instead, we classify vehicles from lanes directly through an online learning scheme and Bayes classifier. The block diagram of our system is showed in figure 1. In the ROI selection phase, lanes are divided into blocks. Features are extracted from these blocks and fed into a learning model, which consists of an unsupervised clustering phase in the initialization stage and an on-line expectation-maximum (EM) learning phase that runs after the initialization. Considering that the initialization stage should be short in duration and the on-line learning rate is relatively much lower, we do not merge them into one, although it is technically possible to do that.

The rest of this paper is organized as follows: Section II is a brief introduction to the related work. Section III discusses ROI section. Section IV focuses on Feature selection. In Section V we will talk about Clustering and On-line learning. Section VI is Bayesian classification. The experimental result will given in section VII followed by a conclusion section.

II. RELATED WORK

Zanin[10] proposed a vehicle queue detection scheme based on background subtraction and movement analysis. The need for background information made it difficult to handle usually congested scenes like intersection. Porikli[11] Proposed traffic congestion estimation scheme using Gaussian Mixture Hidden Markov models. They extracted features from MPEG compressed domain and trained a set of GM-HMM to estimate
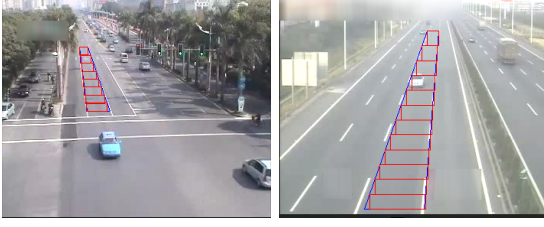
Fig. 2. ROI demonstration. Rectangles of Various size are used to approximate trapezoidal lane. Each Rectangle is treated as an independent sub-image in later processing.

traffic state. Kato[12] used HMM-based segmentation method to classify objects in traffic scenes as shadows, foreground or background objects. Our approach do not rely on background extraction and motion analysis, which means that it can work well even if the vehicle queue is completely stop for a long duration. Another difference is that we divide lanes into blocks, and detect vehicles inside the blocks. This will eliminate the necessity for vehicle segmentation and the measurement for the length of vehicle queue is as easy as counting the blocks that contain vehicles.

III. ROI SELECTION

The ROI selection scheme is pretty simple and straightforward here. As the length of the waiting queue is the most important parameter at the intersection, we need to detect the existence of vehicles both near the pavement and at the far end. Due to the position of traffic cameras and perspective projection, the far end of lanes is narrower than the front end. Therefore, we draw the outline of lanes which is similar to trapezoid. We then divide the trapezoid into several rectangles; each of these rectangles is called Region of interest (ROI). All the features are extracted within these ROI. See figure 2 for an example.

IV. FEATURE SELECTION

For a classier to work in really applications, multiple features are usually needed. AS the complexity of the classifiers depends on the dimension of the feature vector, we shall select those features that are most discriminant. What's more, the feature selection algorithms should be as simple as possible for maximally reducing the system overall complexity. During our experiments, we found that basic image features including local Entropy, edge and moment are best choice for our system to work in real-time. Since we are using Naive Bayes classifier, the ease of normalization of each feature has been taken into account when we choose these features.

A. Maximum Local Entropy

Entropy is a good measurement for texture and common used in object detection[4] [13]. It is defined as $E = -\sum_{i=1}^{L} p(i)\log_2 p(i)$, where $\sum_{i=1}^{L} p(i) = 1, p(i) >= 0, i = 1, ..., L$. The probability is naturally represented by histogram, i.e., $p(i) = \frac{1}{wh}\sum_{x=1}^{w}\sum_{y=1}^{h} 1\{I(x,y) = i\}$ where w and h are
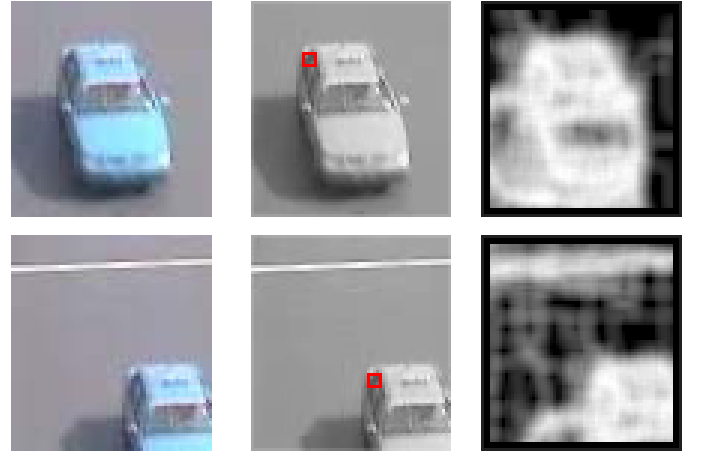

Fig. 3. Local entropy measurement of two images. The first column shows two different images containing the same vehicle fully and partially. The second column shows the location of maximum local entropy measurement, denoted by two red squares. Note that the two red squares are in the same position relative to the vehicle. The last column shows local measurement of the two images as gray scale images

width and height of the image patch respectively. and $1\{\cdot\}$ is the indicator function.

It is easy to find that the maximum entropy is achieved to $\log_2(L)$ if and only if $p(i) = \frac{1}{L}$ for $i = 1, ..., L$, which means that scattered gray-levels lead to large entropy while a clean background has a zero entropy. It is therefore discriminative to use entropy as a feature,

While entropy is a good measure of texture, applying entropy directly to a whole image will obscure its details. Two different images may have the same entropy value, while two images both contain the same object may have two different entropy values. For example, in figure 3, images of the first column both contain the same vehicle (although the first image has a clear view and the second one has only a partial view). The entropy of the first image is 5.5953 while the entropy of the second one is 6.2018. In order to best describe the similarity of these two images, we induce a local entropy measurement,

$$E' = \max_{x,y}\{-\sum_{i=1}^{L} p(i,x,y)\log_2 p(i,x,y)\} \quad (1)$$

where $x = 1, ..., m, y = 1, ..., n$ and

$$p(i,x,y) = \frac{1}{(2w'+1)(2h'+1)}\sum_{j=x-w'}^{x+w'}\sum_{k=y-h'}^{y+h'} 1\{I(j,k) = i\} \quad (2)$$

$2w' + 1$ and $2h' + 1$ are the window width and height respectively. w' and h' vary from 1 to 3 in different scenes and resolutions.The local entropy measurement is illustrated in Figure 3.

B. Edge

Unlike those tackle complete vehicle detection, we take edge as another measurement of texture here. Four different types of edge detection kernel are defined as follows

$$h_1 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, h_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix},$$

$$h_3 = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}, h_4 = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

h_1 and h_2 are the well-known Sobel detectors, which detect horizontal edges respectively; h_3 and h_4 detect left diagonal edges and right diagonal edges respectively. After convolving the above kernels with a gray-scale image, we obtain an edge image I_{h_t}. The edge image is further processed by the operation :

$$\Gamma(h_t) = \frac{1}{wh} \sum_{x=1}^{w} \sum_{y=1}^{h} 1\{I_{h_t}(x,y) > \varphi\} \tag{3}$$

C. Other measurements

We further define three measurements. The first one is rate of non-zero histogram bins, or B, which is defined as $B = \frac{1}{L} \sum_{i=1}^{L} 1\{p(i) > 0\}$. Another feature is normalized first moment, which is defined as $M_1 = \frac{1}{L} \sum_{i=1}^{L} i \cdot p(i)$. M_1 is simply the mean of image intensity. The last feature is the second moment of image:

$$M_2 = \sum_{i=1}^{L} \left(i^2 p(i) \right) - M_1^2 \tag{4}$$

To normalize M_2, we need to find its maximum value. Another definition of M_2 is $M_2 = \frac{1}{p} \sum_{i=1}^{n} x_i^2 - (\frac{1}{p} \sum_{i=1}^{n} x_i)^2 = \mathbf{x}^t(\frac{I}{p} - \frac{\mathbf{b}\mathbf{b}^t}{p^2})\mathbf{x}$ where $\mathbf{b} = [1,...,1]^t$. We treat an image as a $n \times 1$ vector and $\mathbf{x}$ is x_i its ith component. To find the maximum value of (11) is a quadratic programming subjected to $1 \le x_i \le L$ for $i = 1,...n$. Without further derivation, we give the maximum value here directly, that is $\frac{(L-1)^2}{4}$ .Therefore, normalized version of equation (10) is $M_2 = \frac{4}{(L-1)^2} \left(\sum_{i=1}^{L} \left(i^2 p(i) \right) - M_1^2 \right)$

D. Selection criterion

We test each of these measurement using two sets of samples which are manually labelled beforehand. The quality of each feature is judged by Fishers criterion, $J = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$ where μ_i and σ_i^2 are mean and variance of feature value of class respectively. Large means large inter-class distance and small intra-class variation. All the feature measurements are tested on a training set containing 4000 vehicle images and 2000 lane images according to Fishers criterion. The testing result is given in table 1, and Fig 4 shows the separating distance in the form of probability density function (PDF).

Note that all features are normalized before being assembled as a feature vector. As we use Euclidean distance, non-normalized feature vector will lead to focus on larger scale feature component.

TABLE I
AN EXAMPLE OF A TABLE

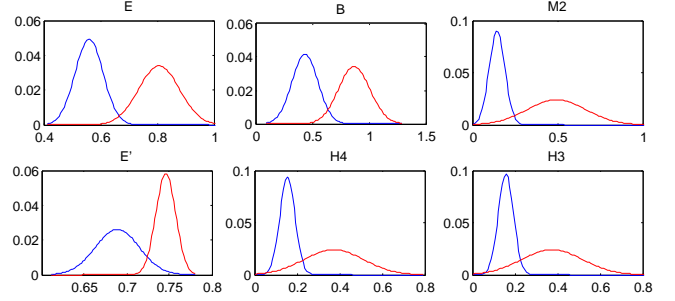| Symbol	Meaning	Separating distance
E	Entropy	2.0267
B	Total non-zero histogram bins	1.7142
M_2	Second order central moment	1.5166
$\Gamma(h_4)$	Right diagonal edge	1.2329
$\Gamma(h_3)$	Left diagonal edge	1.2047
M_1	First order moment(mean)	0.8501
$\Gamma(h_2)$	Vertical edge	0.7979
$\Gamma(h_1)$	Horizontal edge	0.8353

Fig. 4. The red curve and blue curve represent the distribution of each feature extracted from training samples which contain vehicles and those contain only background, respectively. These visualized distributions provide an intuition on which features are superior for classification. The title of each chart follows symbols given in table I.

V. CLUSTERING AND ONLINE EM ALGORITHM

A. Clustering

We use the standard K-means clustering algorithm [1] to coarsely estimate the cluster centroids which serves as an initial guess for the subsequent EM Clustering algorithm. Without discussing algorithm details, we give the K-means parameter values in our case here directly. 1. K=2, i.e. there are two clusters: vehicle and lane 2. Cluster distance is measured by Euclidean distance 3. Repeat the algorithm N times to avoid trapping into local maximums.(N=3 is sufficient)

B. EM for GMM

The output of K-means algorithm are two cluster centroids, as well as labels of data indicating which cluster they belong too. The two clusters are further modelled with Gaussian distribution, i.e. $p(x|x \in c; \mu_c, \Sigma_c) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_c|} \exp\left(-\frac{1}{2}(x-\mu_c)^T \Sigma_c^{-1}(x-\mu_c)\right)$ where $c \in \{l, v\}$ represents lane and vehicle respectively. $x \in \mathbf{R}^n$ is the feature vector. $\mu_c \in \mathbf{R}^n$ is the mean vector, and $\Sigma_c \in S_{++}^n$ denotes the covariance matrix, which is positive definite. We further simplify the covariance matrix as a diagonal matrix to lower computation complexity without much loss of accuracy.

Suppose that we have m feature vectors which we want to divide into two classes and model them with two separate Gaussian distributions. To put it another way, the problem is to find μ_c, Σ_c and more accurately determine memberships of all data to the two models . By using the well-known EM
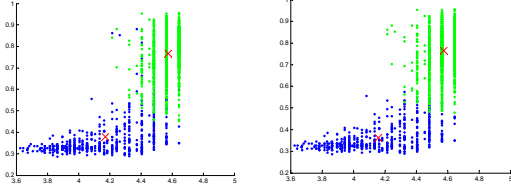
Fig. 5. Clustering result for the first two dimensions of feature vectors are demonstrated. The left one is k-means clustering result; The right one is EM clustering result. It can be seen that there are considerable difference between them.

algorithm, we can easily solve this mixture models problem [5]. First, we define the class prior density as $p(x \in c) = \Phi_c$ with $\sum_{c=\{l,v\}} \Phi_c = 1$ and $\Phi_c > 0$.

1. E-step: For every i and c, estimate memberships according to

$$
\begin{aligned}
\omega_c^{(i)} &= p(x^{(i)} \in c | x^{(i)}; \Phi_c, \mu_c, \Sigma_c) \\
&= \frac{1}{D} p(x^{(i)} | x^{(i)} \in c; \mu_c, \Sigma_c) p(x^{(i)} \in c; \Phi_c) \quad (5)
\end{aligned}
$$

2. M-step:

$$
\Phi_c = \frac{1}{m} \sum_{i=1}^{m} \omega_c^{(i)} \tag{6}
$$

$$
M_c = \sum_{i=1}^{m} \omega_c^{(i)} \tag{7}
$$

$$
\mu_c = \frac{1}{M_c} \sum_{i=1}^{m} \omega_c^{(i)} x^{(i)} \tag{8}
$$

$$
\Sigma_c = \frac{1}{M_c} \sum_{i=1}^{m} \omega_c^{(i)} (x^{(i)} - \mu_c)(x^{(i)} - \mu_c)^T \tag{9}
$$

Then, a common used EM-based GMM algorithm is listed here without proving it.

In the above algorithm, $x^{(i)}$ denotes the ith feature vector. D is denominator which normalizes $\omega_c^{(i)}$ so that $\sum_c \omega_c^{(i)} = 1$ for every $i = 1, ..., m$. The initial value for Φ_c is 0.5, for μ_c is the output of cluster centroids from k-means, and for Σ_c is just the identity matrix. A snapshot of k-means and EM clustering algorithm is taken from our system and showed in figure 5, which shows the cluster distribution of the first two components in the feature space.

VI. ON-LINE EM

Suppose that we already have samples in hand, and the GMM is well modeled accordingly. As the monitoring system runs, the environment (mostly the lighting condition) will gradually change. If no updating is made, the system probably will fail when the change becomes obvious. Therefore, an on-line learning phase is adopted to update the parameters of GMM.

Derivation of the on-line EM algorithm is intuitive. Let $x^{(m+1)}$ denotes the incoming sample, following the notations

of previous EM algorithm, we have the following on-line learning algorithm,

1. Estimate $\omega_c^{(m+1)}$ according to (5).
2. Update model parameters

$$
\Phi_{c,m+1} = \frac{1}{m+1}(\Phi_{c,m}m + \omega_c^{(m+1)}) \tag{10}
$$

$$
M_{c,m+1} = M_{c,m} + \omega_c^{(m+1)} \tag{11}
$$

$$
\mu_{c,m+1} = \frac{1}{M_{c,m+1}}(\mu_{c,m}M_{c,m} + \omega_c^{(m+1)}x^{(m+1)}) \tag{12}
$$

$$
\Sigma_{c,m+1} = \frac{1}{M_{c,m+1}}(\Sigma_{c,m}M_{c,m} +
$$
$$
\omega_c^{(m+1)}(x^{(m+1)} - \mu_{c,m+1})(x^{(m+1)} - \mu_{c,m+1})^T) \tag{13}
$$

One problem of On-line EM is that, when m is too large, $M_{c,m}$ will be also too large that the incoming new sample barely influences the model parameters. To avoid this, we fix $\omega_c^{(m+1)}/M_{c,m} = \lambda = m/(m+1)$ and then adjust the model updating equations accordingly.

VII. BAYES CLASSIFIER

Given a sample feature vector and two class models that have been trained, according to the Bayes rule,

$$
p(x \in c | x; \theta_c) = \frac{p(x | x \in c; \theta_c)p(x \in c; \theta_c)}{\sum_c p(x | x \in c; \theta_c)p(x \in c; \theta_c)} \tag{14}
$$

for $c \in \{l, v\}$. $p(x | x \in c; \theta_c) = N(x; \mu_c, \Sigma_c)$ is the likelihood term. $\theta_c = [\mu_c, \Sigma_c, \Phi_c]$. $p(x \in c; \theta_c) = \Phi_c$ is the prior density. A feature vector is said to be from class v if $p(x \in v | x; \theta_v) > p(x \in l | x; \theta_l)$.

Another form of (14) is the discriminate function $f(x) = \log p(x \in v | x; \theta_v) - \log p(x \in l | x; \theta_l) > 0$ which is further simplified as

$$
\begin{aligned}
f(x) &= \log\frac{|\Sigma_l|}{|\Sigma_v|} + \log\frac{\Phi_v}{\Phi_l} + \frac{1}{2}(x - \mu_l)^T \Sigma_l^{-1}(x - \mu_l) \\
&\quad - \frac{1}{2}(x - \mu_v)^T \Sigma_v^{-1}(x - \mu_v) \tag{15}
\end{aligned}
$$

Every incoming feature is classified to be vehicle or lane according to (15).

VIII. EXPERIMENTAL RESULT

Although our video sources contain hue information, we utilized only the intensity channel. The image sequence are in size and 25FPS. Because high chronological preciseness is not necessary, the video rate is down-sampled to 5FPS, which lowers the consumption of computational resource tremendously. The values of aforementioned parameters are sum up in table II.

IX. CONCLUSION

In this paper, we have proposed a congestion estimation system which has wide application in urban traffic scenes. By using a traffic monitoring camera, it can effectively estimate the current traffic state, measure the length of vehicle queue and provide useful information for the traffic control departments. We simplify the flow chart of our system by dividing

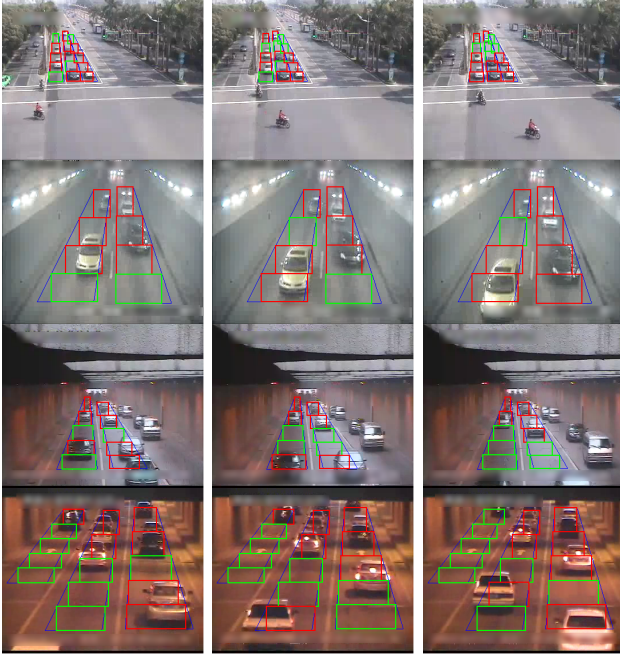| parameter	value	parameter	value
L	256	w'	2
φ	30	h'	2
w	352	L'	32
h	288	λ	0.05

Fig. 6. Four different scenes are presented here. The red square means vehicle exists whereas the green square means otherwise.

lanes into blocks, which eliminates the need for vehicle segmentation. However, it brings up another new challenge, that is, to detect partly visible objects. By incorporating GMM and Bayesian into our system, we have successfully tackled the detection of vehicle existence, assuming that only lanes and vehicles are presented. In the future research, vehicle type recognition and pedestrian detection within lane blocks will be of great interest to us.

ACKNOWLEDGMENT

REFERENCES

[1] P. Michalopoulos, "Vehicle detection video through image processing: the autoscope system," *Vehicular Technology, IEEE Transactions on*, vol. 40, no. 1, pp. 21–29, 1991.

[2] T. Bragatto, G. Ruas, V. Benso, M. Lamar, D. Aldigueri, G. Teixeira, and Y. Yamashita, "A new approach to multiple vehicle tracking in intersections using harris corners and adaptive background subtraction," in *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE, 2008, pp. 548–553.

[3] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE, 1997, pp. 495–501.

[4] W. Hsu, H. Liao, B. Jeng, and K. Fan, "Real-time vehicle tracking on highway," in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, vol. 2. IEEE, 2005, pp. 909–914.

[5] B. Morris and M. Trivedi, "Real-time video based highway traffic measurement and performance monitoring," in *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*. IEEE, 2007, pp. 59–64.

[6] ——, "Learning, modeling, and classification of vehicle track patterns from live video," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 9, no. 3, pp. 425–437, 2008.

[7] Y. Jung and Y. Ho, "Traffic parameter extraction using video-based vehicle tracking," in *Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on*. IEEE, 1999, pp. 764–769.

[8] X. Song and R. Nevatia, "A model-based vehicle segmentation method for tracking," 2005.

[9] Z. Yang, H. Meng, Y. Wei, H. Zhang, and X. Wang, "Tracking ground vehicles in heavy-traffic video by grouping tracks of vehicle corners," in *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*. IEEE, 2007, pp. 396–399.

[10] M. Zanin, S. Messelodi, and C. Modena, "An efficient vehicle queue detection system based on image processing," in *Image Analysis and Processing, 2003. Proceedings. 12th International Conference on*. IEEE, 2003, pp. 232–237.

[11] F. Porikli and X. Li, "Traffic congestion estimation using hmm models without vehicle tracking," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 188–193.

[12] J. Kato, T. Watanabe, S. Joga, J. Rittscher, and A. Blake, "An hmm-based segmentation method for traffic monitoring movies," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 9, pp. 1291–1296, 2002.

[13] H. Shi, Y. Zou, Y. Wang, and G. Shi, "A robust traffic parameter extraction method using texture and entropy," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 237–241.