A HIDDEN MARKOV MODEL FRAMEWORK FOR TRAFFIC EVENT DETECTION USING VIDEO FEATURES

Xiaokun Li[1] and Fatih M. Porikli[2]

[1]Department of ECECS, University of Cincinnati, Cincinnati, OH 45219, USA
[2]Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, USA

ABSTRACT

We present a novel approach for highway traffic event detection. Our algorithm extracts features directly from the compressed video and automatically detects traffic events using a Gaussian Mixture Hidden Markov Model (GMHMM) framework. First, a feature vector is computed for a group of picture from the Discrete Cosine Transform (DCT) coefficients and macro-block motion vectors after MPEG video bitstream is parsed. We show that the feature vector is robust towards different camera setups and illumination conditions such as sunny, overcast, dark, night, etc. Then, we use Viterbi algorithm to determine the most likely traffic condition. We define six traffic patterns, and each pattern is modeled by a separate GMHMM that is trained using the EM algorithm. The proposed system is efficient both in terms of computational complexity and memory requirement. The experimental results prove that the system has a high detection rate. The presented model-based system can be easily extended for detection of similar traffic events.

1. INTRODUCTION

Highway traffic inspection and event detection is an attractive research area where computer vision and image processing techniques are extensively used. Although the dominant technology for current detection system is loop detectors which are buried underneath the highways to count vehicles passing over them, video monitoring systems promise more advantages [1]. One important advantage is that video camera systems provide essential visual information that enables accurate understanding of the scene and the motion of the vehicles. Furthermore, installation of video cameras is much less disruptive and costly than loop detectors.

In current literature, most video-based systems for monitoring road traffic rely on stationary cameras [1, 6], and inspect traffic by tracking vehicles passing through the field of view of the camera. There are also a significant effort on the monitoring traffic and tracking vehicles from a moving camera mounted on a vehicle [4]. However, vehicle counting is computational expensive, and highway traffic engineers are more interested in inspection of the traffic condition in the entire highway. H. Taniguchi et al. [5] developed a rear-end-collision inspection system to detect collision event happening by using the information of whole inspected area. A non-parameter regression (NPR) method was used in [7] to forecast traffic incident on highway, which takes the information of inspected area as input. B. Maurin et al. [8] designed a system which addresses a multi-level approach to monitor traffic scenes using the technologies of optical flow, Kalman filtering, and blob merging to summary the information of the region of interest.

The paper aims to design a real-time and low-cost system to automatically detect highway traffic event, such as heavy congestion, high vehicle density with high speed, vacancy, etc. Instead of monitoring traffic by tracking vehicle individually, the algorithm considers the information of whole inspected area, extracts event features from DCT domain without decoding, and uses the trained GMHMM to detect traffic event in real-time. The merit of the low computational cost of the proposed algorithm enables our system simultaneously inspect multiple channels on a PC. The rest of the paper is organized as follows: Section 2 discusses feature extraction. Section 3 focuses on event modeling and detection. Experiment results are presented in Section 4. Conclusions are summarized in Section 5.

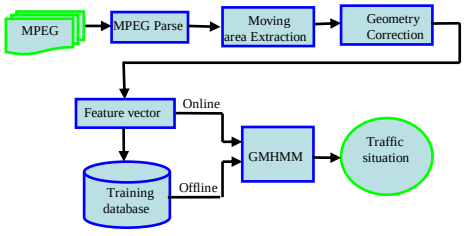


Figure 1 Algorithm flow chart

2. FEATURE EXTRACTION

2.1. MPEG Parser

MPEG is the most widely employed video compressed standards. Its compression scheme reduces the spatial redundancy in one frame by using DCT and temporal

redundancy between successive frames via motion compensation to achieve a low-bit rate compression [2]. The result of motion compensation is stored as motion vector (MV) in video. MPEG-1 video consists of a sequence of I frames with a number of B and P frames, where a P frame is predicted from the immediately preceding I or P frame, and a B frame is bi-directionally interpolated using the two I or P frames before and after it. The basic unit of a sequence is group of pictures (GOP) and its typical sequence order is: I B B P B B P B B P.

(a) A spatial frame (b) Image region matrix in DCT

Figure 2 Comparison of spatial image and DCT image

2.1.1. Image energy

DCT compressed video encodes an I frame image by dividing it into many blocks and using DCT coefficients of $\{C_{uv}\}$, 8×8 block, $\{I_{xy}, 0 \le x < N, 0 \le y < N\}$ to compress it.

$$C_{uv} = \frac{1}{N} K_u K_v \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} I_{xy} \cos\frac{\pi u(2x+1)}{2N} \cos\frac{\pi v(2y+1)}{2N} \qquad (1)$$

where u and v are the horizontal and vertical frequencies $(u, v = 0,1,..., N-1)$, $N = 8$, and $K_w = \frac{1}{\sqrt{2}}$, $(w \in \{u, v\})$. When $w = 0$, $K_w = 1$. When $u = v = 0$, the coefficients are called DC, the others are called AC. We extract the DCs from all blocks and save them into a small image region according to their block order. For the video with image size 352x240, the region size is 44x30. Repeat the same process to the other u, v to construct the other 63 image regions, thus create a 8x8 image region matrix shown in Fig. 2b. The top-let image region is DC image called image energy. It is actually a reduced image (1/64) of spatial image (Fig. 2a). Note that only Y channel information of the video is considered in our research.

2.1.2. Texture and edge

The other image regions ($C_{uv}, u \ne 0$ or $v \ne 0$), called AC images, store the information of edges and directionalities of spatial image. Since the main texture and edge information are mainly located in the first column and the first row of region matrix, only the first four AC images in the first column and the first four AC images in the first row are accumulated to get a mean of AC image called texture & edge map via Eq. (2). Furthermore, a directional Sobel filter is employed to enhance the map.

$$\overline{AC} = \frac{1}{2N}\left(\sum_{i=1}^{M} AC_{(i,0)} + \sum_{j=1}^{N} AC_{(0,j)}\right) \qquad (2)$$

where M and N are region number selected from the first row and the first column.(In our research, $M = N = 4$).

2.1.3. Motion vector

MV only exits in P and B frame and the MVs in P frame have a good property that the MVs describe the moving object in spatial domain. The magnitude of MV reflects the speed of the moving object and the direction of MV indicates the moving direction of the moving object. Moreover, the density of MV implies the density of moving objects in spatial domain. An illustration is given in Fig. 3. A trimmed mean filter is employed to remove the noise existed in P frame.

$$MV_{(i,j)} = \frac{1}{N-2} \sum_{m=2}^{N-1} SortedMV(m) \qquad (3)$$

where $SortedMV(m) \in$ { the eight neighbors of $MV_{(i,j)}$ plus $MV_{(i,j)}$ }, $N = 9$.

(a) High vehicle density with high speed (b) Low vehicle density with low speed

Figure 3 Example on motion vector

2.2. Moving area extraction and geometry correction

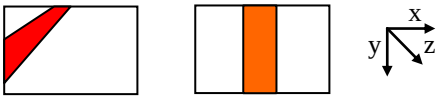
There are two ways to identify the moving area (inspected area) from video sequence. The first way is manually drawn by traffic engineer. The second is automatically decided by the background extraction and MV analysis (See [10] for details).

To make the extracted feature invariant to different data sources (Highways) as different highway has different angle and distance to camera, geometry correction is applied. Firstly, the data in inspected area are transformed into the earth coordinate system (a predefined coordinate in video camera system) via an affine transformation which includes a rotation matrix R and a translation vector T.

$$V_e = RV_s - T \qquad (4)$$

where $V_s = \begin{pmatrix} x \\ y \end{pmatrix}$ is the input data, $V_e = \begin{pmatrix} x \\ y \end{pmatrix}$ is the transformed data; $R = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$, θ is the angle between the traffic direction and the Y axis of the earth coordinate system; $T = \begin{pmatrix} D_x \\ D_y \end{pmatrix}$, T is the Euclidean distance between the center of inspected area and origin of the earth coordinate system. Since the size of far object to camera becomes smaller than the near object in video image, area compensation along the Z axis of camera coordinate system to the far

and near part of inspected area is also employed after the affine transformation. An illustration is given in Fig. 4.



(a) Inspected area before correction (b) Inspected area after correction

Figure 4 Example on geometry correction

2.3 Feature Vector

After the comparison of the traffic information in spatial domain to DCT domain in Section 2.1, it can be concluded that more vehicles on highway, more image energy in DC image. The same conclusion can be applied to the relationship of the number of vehicles and the value of texture & edge map. Therefore, image energy and texture & edge map are important features for traffic event detection. Since the state of MV in inspected area directly reflects the situation of traffic happened in inspected area, MV is a crucial feature that should be also considered. In our research we use the information of whole inspected area for traffic event detection by calculating the density of inspected area, such as density of image energy, density of texture & edge, etc. We construct a density feature vector to each inspected area of every GOP. And each vector contains four elements. The first element is the density of image energy, called ρ_{dc}. The second is the density of texture & edge map, called ρ_{ac}. The third is the density of the MVs whose magnitude is larger than a predefined threshold, called ρ_{Hmv}. The last is the density of the MVs whose magnitude is smaller than a predefined threshold, called ρ_{Lmv}. Note that the two predefined thresholds are not necessary to have the same value. The first two elements can be computed through Eq. (5). The feature vector is expressed in (6).

$$\rho = \frac{M}{N} \qquad (5)$$

where M is the total number of the pixels in inspected area whose residue is larger than zero, N is the total number of the pixels in inspected area.

$$V = [\rho_{dc} \quad \rho_{ac} \quad \rho_{Hmv} \quad \rho_{Lmv}]^T \qquad (6)$$

Since all components in vector are density values, the feature vector is invariant to the size of inspected area. Another advantage is the density values are insensitive to the change of illumination.

3. EVENT MODELING AND DETECTION

3.1. Event definition

Previously, we proposed a 5-events system [3]. In this paper, six traffic events are defined according to the number of vehicles and their speed in inspected area. They are heavy congestion (HC), high density with low speed (HDLS), high density with high speed (HDHS), low density with high speed (LDHS), low density with low speed (LDLS), and vacancy.

3.2. Modeling by HMM

Since traffic event is a temporal process and HMM is suitable to incorporate temporal continuity [9], we use HMM to model the traffic events. HMM consists of a finite set of states, each of them is associated with a probability distribution that models the distribution of the feature vector. Transitions among the states are governed by a set of probabilities called transition probabilities. To each state, the sum of outcome probabilities is equal to 1. In this work, we set up a HMM which has six states and the HMM topology is shown in Fig. 5. Each state represents an event and has four dimensional probabilities. We use either a single Gaussian or a Gaussian mixture distribution, called GM, to describe these state-conditional probability distributions. The feature vector of inspected area is used as observed data (input data) to GMHMM. On offline mode, the input data are training data for modeling learning. On online mode, the input data are testing data for event detection.

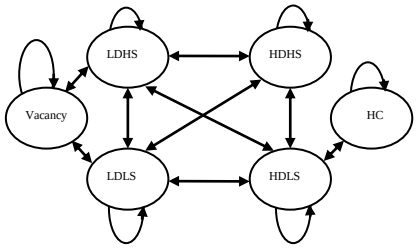


Figure 5 Hidden Markov Model for event detection

An EM method, Baum-Welsh algorithm, is used to learn the unknown GMHMM parameters (probabilities). EM algorithms perform an iterative computation of maximum likelihood estimation when the observed data are fed in. The aim of parameter learning is to find the model parameter λ which maximizes $\lambda = \arg\max(\log[p(V|\lambda)])$ for a given set V of observed data. The learning process produces a sequence of estimates for λ. Given a set of observed data V, the estimate λ^i has a greater value of $\log[p(V|\lambda)]$ than the previous estimate λ^{i-1}. The EM includes two parts:

- Preliminaries

$$\zeta_t(i,j) = P(q_t = i, q_{t+1} = j | V, \lambda) \qquad (7)$$

$$\gamma_t(i) = P(q_t = i | V, \lambda) \qquad (8)$$

where $V = \{v_1, ..., v_T\}$ is training sequence and T is the length of training sequence. $\zeta_t(i,j)$ and $\gamma_t(i)$ can be efficiently calculated by the forward-backward algorithm [9].

- Update Rules

$$\overline{\pi_i} = \gamma_1(i) \qquad (9)$$

$$\overline{\mu_i} = \frac{\Sigma_{t=1}^{T} v_t \gamma_t(i)}{\Sigma_{t=1}^{T} \gamma_t(i)} \quad (10)$$

$$\overline{\Sigma i} = \frac{\Sigma_{t=1}^{T} \gamma_t(i)(v_t - \overline{\mu_i})(v_t - \overline{\mu_i})^t}{\Sigma_{t=1}^{T} \gamma_t(i)} \quad (11)$$

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T} \zeta_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad (12)$$

After setting the initial value to λ, the parameter estimation process will repeat Eq. (7)-(12) until the $\log[p(V \mid \lambda)]$ reaches to a local maximum. One advantage of the EM algorithm is the convergence is guaranteed and the convergence time is short. Also, the local maximum is usually an adequate model for the data.

3.3. Detection

After all GMHMM's are trained, event detection is performed using the Viterbi algorithm [9], which is a standard technique for pattern recognition. Given a sequence of feature vectors, the Viterbi algorithm produces the sequence of states which are most likely to have generated these feature vectors. The state sequence is time-aligned with the feature sequence, so that the traffic video is detected according to the times corresponding to our defined events.

4. EXPERIMENTAL RESULTS

The algorithm was tested on a video database which has four video sources with different illumination situations, including US highway I5Ne, I5SR, SR520, and I405. The total time is 12 hours. The training set for our testing system is 20-minute video clip chosen from the video database, which covers all sources and illuminations. It is manually labeled by hand. The system was tested on 60-minute video clip of I5NE, 60 minutes of I5SR, 60 minutes of SR520, and 60 minutes of I405. The correct detection rate can reach 94% and the error rate is around 3%. All testing results are listed in Table 1&2. There also exist some detection results that are hard to judge they are right or not, as the traffic state at that time is in the middle state of two events. In Table 1 & 2, we also give out the results obtained from an optimum threshold method for comparison with the GMHMM method.

The testing system detects traffic event with real-time speed. Even with our C++&Matlab mixed program, the computational time for a 60-minute video clip which has three inspected areas is less than 15 minutes on a 1.6GHz-P4 PC. That enables the system to inspect four video channels simultaneously.

5. CONCLUSIONS

A GMHMM-based algorithm is presented. It directly extracts features from DCT coefficients and motion vectors of MPEG video sequence, and then use GMHMM to build event models. Geometry correction is employed for invariant feature extraction. The extracted feature vector is robust towards different illuminations. Experimental results show that the proposed real-time system can efficiently detect traffic event. In addition, it can be easily extended for detecting more traffic events.

	HMM		Threshold	
Data Set	Correct	Error	Correct	Error
I5NE (60 mins)	0.938	0.034	0.811	0.152
I5SR (60 mins)	0.943	0.033	0.793	0.174
SR520 (60 mins)	0.937	0.040	0.832	0.148
I405 (60 mins)	0.935	0.039	0.803	0.154

Table 1: Comparison of HMM detection to threshold method

	HMM		Threshold	
Illumination	Correct	Error	Correct	Error
Sunny (60 mins)	0.944	0.247	0.847	0.122
Cloud (60 mins)	0.938	0.341	0.797	0.162
Dark (60 mins)	0.916	0.408	0.729	0.228

Table 2: Comparison of HMM detection to threshold method under different illumination situations.

6. REFERENCES

[1] D. Beymer, P. McLauchlan, B. Coifman, J. Malik, "A Real-time computer vision system for Measuring Traffic Parameters," *IEEE Proc. CVPR*, pp. 495-501, 1997.

[2] Y. Q. Shi, H. Sun, *Image and Video Compression for Multimedia Engineering*, CRC Press, 2000.

[3] F. Porikli and X. Li, "Traffic congestion estimationusing HMM models without tracking", *IEEE Proc. Intelligent Vehicles*, 2004

[4] F. Dellaert, D. Pomerleau, C. Thorpe, "Model-based car tracking integrated with a road follower," *IEEE Proc. Robotics and Automation*, pp. 1189-1194, 1998.

[5] H. Taniguchi T. Nakamura, H. Furusawa, "Methods of Traffic Flow Measurement using Spatio-Timporal Image," *IEEE Proc. ICIP*, pp. 16-20, 1999.

[6] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic Monitoring and Accident Detection at Intersections," *IEEE Trans. Intelligent Transportation Systems*, Vol. 1, No. 2, pp.108-118, 2000.

[7] S. Tang, X. Gong, F. Wang "Traffic Incident Detection Algorithm Based on Non-parameter Regression," *IEEE Proc. Intelligent Transportation Systems (ITS)*, pp. 714-719, 2002.

[8] B. Maurin, O. Masoud, N. Papanikolopoulos, "Monitoring Crowded Traffic Scenes," *IEEE Proc. ITS*, pp. 19-24, 2002.

[9] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. of The IEEE*, pp. 257-286, 1989.

[10] G. M., I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia, "Event Detection and Analysis from Video Streams", *IEEE Trans. PAMI*, Vol. 23, No. 8, pp. 873-889, 2001.