Toward Dynamic Scene Understanding by Hierarchical Motion Pattern Mining

Lei Song, Fan Jiang, Zhongke Shi, Rafael Molina, *Member, IEEE*, and Aggelos K. Katsaggelos, *Fellow, IEEE*

***Abstract*—Our work addresses the problem of analyzing and understanding dynamic video scenes. A two-level motion pattern mining approach is proposed. At the first level, activities are modeled as distributions over patch-based features, including spatial location, moving direction, and speed. At the second level, traffic states are modeled as distributions over activities. Both patterns are shared among video clips. Compared to other works, one advantage of our method is that moving speed is considered to describe visual word. The other advantage is that traffic states are detected and assigned to every video frame. These enable finer semantic interpretation, more precise video segmentation, and anomaly detection. Specifically, every video frame is labeled by a certain traffic state, and the video is segmented frame by frame accordingly. Moving pixels in each frame, which do not belong to any activity or cannot exist in the corresponding traffic state, are detected as anomalies. We have successfully tested our approach on some challenging traffic surveillance sequences containing both pedestrian and vehicle motions.**

***Index Terms*—Anomaly detection, Latent Dirichlet Allocation (LDA), motion pattern analysis, video segmentation, visual surveillance.**

I. Introduction

In many surveillance scenarios, such as those involving a crowded traffic scene, a busy train station, or a shopping mall, various motions are involved. It is highly desirable to analyze the motion patterns and obtain some high-level interpretation of the semantic content. For example, in a video monitoring intersection, without any prior knowledge about the traffic rules in the specific scene, it is useful to discover typical vehicle behaviors and their dependencies involved in this scene and detect anomalous motion for security concerns.

Fig. 1. Activities and traffic states. Two traffic scenes illustrate activities and traffic states. (a) Arrows 1–7 and (b) 1–15 show single-agent motion patterns (activities), while arrows grouped by the same colors show interaction patterns (traffic states).

Motion patterns involved in a complex dynamic scene usually are of a hierarchical nature; that is, at low level, they consist of single-agent motion patterns, which are combined at a higher level to form interaction patterns. Typically, many objects (e.g., vehicles) are involved in the video scene. In terms of each single object, its motion might follow some regular streams, which are single-agent motion patterns. In addition, the cooccurrence of multiple objects at the same time might also be subject to constraints, which define interaction patterns. For example, in the traffic intersection scenario, the single-agent motion patterns are all the legal paths going through this intersection, which are named as "activities" [shown in Fig. 1(a) numbered from 1 to 7 and in Fig. 1(b) numbered from 1 to 15], whereas the interaction patterns are possible combinations of paths determined by the traffic lights, which are named as "traffic states." Fig. 1(a) has two traffic states, particularly paths 1–5 in red and paths 6–7 in yellow, while there are five traffic states in Fig. 1(b) represented separately in red, yellow, green, purple, and blue.

Considering this hierarchical nature of motion patterns, many works on scene understanding and motion pattern discovery are based on hierarchical modeling. One common approach is based on object trajectory analysis. Morris and Mohan [23] overviewed the work on trajectory learning and analysis for surveillance. Objects are tracked in videos, and an analysis and mining approach is applied to the object trajectories to discover motion patterns. For example, Jiang *et al.* [1] used a hidden Markov model (HMM) to characterize object trajectories and a Bayesian-information-criterion-based dissimilarity measure for highly recurrent events clustering. Duong *et al.* [2] introduced the switching hidden semi-Markov model for atomic activity modeling, and the high-level activities are modeled as a sequence of atomic activities. Jiang *et al.* [3] characterized the

Manuscript received December 1, 2012; revised June 12, 2013 and October 23, 2013; accepted December 22, 2013. Date of publication February 20, 2014; date of current version May 30, 2014. This work was supported in part by the China Scholarship Council, by the U.S. Department of Energy under Contract DE-NA0000431, by the "Ministerio de Ciencia e Innovación" under Contract TIN2010-15137, and by the CEI BioTic with the Universidad de Granada. The Associate Editor for this paper was M. M. Trivedi.

L. Song was with the School of Automation, Northwestern Polytechnical University, Xi'an 710072, China. She is now with The Third Research Institute of Ministry of Public Security, Shanghai 201204, China (e-mail: songlei@mail.nwpu.edu.cn).

F. Jiang was with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208 USA. He is now with Amazon.com, Seattle, WA 98101 USA (e-mail: fanjiang2008@u.northwestern.edu).

Z. Shi is with the School of Automation, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: zkeshi@nwpu.edu.cn).

R. Molina is with the Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, 18071 Granada, Spain (e-mail: rms@decsai.ugr.es).

A. K. Katsaggelos is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208 USA (e-mail: aggk@eecs.northwestern.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2299403

crowded motion by a patch-based local motion representation and clustered all patches into different motion patterns by spectral clustering. Basharat *et al.* [4] detected abnormal events based on local and global behavior of tracks. Instead of clustering tracks into major paths, they build local pixel-level probability density functions that capture a variety of tracks. Morris and Mohan [24] designed the path modeling block to learn the major scene routes by an HMM based on the vehicle tracking data. Wang *et al.* [25] treated the objects' trajectories as documents, clustered them into activities by the dual hierarchical Dirichlet processes (Dual-HDPs), and detected anomalies as trajectories with low likelihoods. Similarly, Jeong *et al.* [26] treated the observations on a trajectory as words in a document, and then, the latent Dirichlet allocation (LDA) model was adopted to model the topics, which are semantic regions. Tao and Gong [27] clustered behavior patterns into behavior classes through a spectral clustering algorithm. Hu *et al.* [28] first clustered the foreground pixels into trajectories, which were clustered hierarchically into motion patterns based on spatial and temporal information. Piciarelli *et al.* [29] extracted the trajectories of moving objects from video and clustered them into groups by support vector machine according to their similar features, while trajectories without these features were detected as anomalies.

Object tracking methods, however, are sensitive to object detection, recognition, and tracking errors, and they usually fail in complicated or crowded scenes due primarily to occlusions. To improve robustness, statistical methods have been devised that work directly on quantized pixel data or other low-level features in videos, such as object location and intensity gradient. These methods typically employ probabilistic topic models adapted from the text and image mining communities. Low-level features are considered as visual words in video sequences, which are treated as documents. Motion patterns can be discovered as topics (groups of visual words) shared by all documents. Yang *et al.* [5] used diffusion maps to embed the words into a lower dimensional space and to cluster them into motion patterns, while video clips are clustered to determine cooccurring motion patterns. Saleemi *et al.* [6] adopted a Gaussian mixture model for pixel-level representation of motion patterns in a hierarchical unsupervised fashion. In [7], a Markov Clustering Topic Model was proposed, which builds on LDA and Markov chains. Visual words are clustered into actions, and clips are clustered into behaviors over cooccurring actions. Both [8] and [9] used an HDP-HMM for state detection. In this model, the HDP can automatically decide the number of states for the HMM. Kuettel *et al.* [10] first learned activities using an HDP model and then found the activity dependencies by a dependent Dirichlet process HMM (DDP-HMM). Emonet *et al.* [11] proposed a model that relies on a Dirichlet process to discover the activities, their number, and their occurrences. Wang *et al.* [12] used hierarchical Bayesian models to classify surveillance video into two levels: atomic activity, which is represented by distribution over low-level visual features on a pixel basis, and interaction, which is modeled by distribution over atomic activities. This two-level motion analysis provides a good representation of the hierarchical nature of the video scene and enables video anomaly detection.

Following this hierarchical interpretation, we propose a novel two-level motion pattern analysis method based on the LDA model. Our approach is different from the work of Wang in the modeling of the interactions. The work of Wang is based on a DDP and HDP. Interactions are modeled as clusters of video clips. Since video clips are the basic processing unit for interaction learning, a problem appears when interaction transition happens within one clip. Video anomaly can only be detected and localized per video clip, which still includes a number of frames. In contrast, our approach utilizes LDA modeling for the interaction-level processing. Interactions are modeled as clusters of atomic activities and are shared among video clips. Then, they are assigned to every video frame, thus enabling frame-based, rather than video-clip-based, video segmentation and anomaly detection. In addition to location and direction, moving speed is also considered when we form the visual words. Our work achieves a finer semantic interpretation of a dynamic scene. Experiments with real traffic surveillance videos demonstrate that our approach is able to interpret every video frame by different interaction patterns and detect anomalies in each frame.

The rest of the paper is organized as follows: Visual word detection is presented in Section II. In Section III, a two-level motion pattern mining method is introduced. Experiment results are shown in Section IV. Some issues are discussed in Section V, and we conclude the paper in Section VI.

II. Visual Word Detection

A. Motion Detection

Based on the Lucas–Kanade (LK) optical flow estimation algorithm [13], we adopt the multiresolution LK (MLK) algorithm [30] in this paper to detect moving patches and their properties, such as location, moving direction, and speed. The MLK algorithm can reduce the resolution of images to make motions small enough when calculating the optical flow. It is very effective in traffic monitoring when the camera is installed on a high spot. Let L_{OF} be the number of layers in the pyramid structure. We use the LK algorithm on every block at the first layer (the highest level) and set the size of the window to $ofw_1 \times ofw_1$ when calculating optical flow. At layer $l_{\mathrm{OF}}(l_{\mathrm{OF}} = 1, 2, \ldots, L_{\mathrm{OF}})$, the size of the optical flow window is $ofw_{l_{\mathrm{OF}}} = ofw_1 \times 2^{l_{\mathrm{OF}}-1}$, and the analysis window is $d_{l_{\mathrm{OF}}} \times d_{l_{\mathrm{OF}}}$, where $d_{l_{\mathrm{OF}}} = 2^{l_{\mathrm{OF}}-1}$. The MLK algorithm is applied according to the following steps.

Step 1. Calculate the frame difference between two consecutive frames and keep those values larger than a threshold, which is defined as $diff_{L_{\mathrm{OF}}}$, since it is treated as layer L_{OF}.

Step 2. Perform downsampling of the frame difference image $L_{\mathrm{OF}} - 1$ times, to obtain the frame difference images $diff_{l_{\mathrm{OF}}}(l_{\mathrm{OF}} = L_{\mathrm{OF}} - 1, L_{\mathrm{OF}} - 2, \ldots, 1)$ for layer l_{OF}.

Step 3. Compute the LK algorithm on each block at the highest layer (the first layer) if its frame difference $(diff_1)$ is nonzero and then obtain the optical flow vector with components u_1 and v_1.

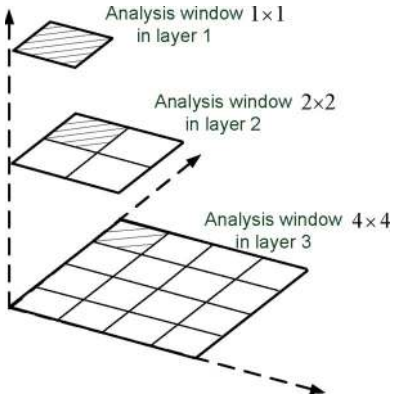


Fig. 2. Analysis window and the first block (shadowed) in each layer (when $L_{\mathrm{OF}} = 3$).

Step 4. For $l_{\mathrm{OF}} = 2, 3, \ldots, L_{\mathrm{OF}}$ do

- Take flow $u_{l_{\mathrm{OF}}-1}$ and $v_{l_{\mathrm{OF}}-1}$ from layer $l_{\mathrm{OF}} - 1$;
- Upsample the flow to create $u^*_{l_{\mathrm{OF}}}$ and $v^*_{l_{\mathrm{OF}}}$ of twice the resolution for layer l_{OF};
- Multiply $u^*_{l_{\mathrm{OF}}}$ and $v^*_{l_{\mathrm{OF}}}$ by 2;
- For the first block $p_{l_{\mathrm{OF}}}(x, y)$ in every $d_{l_{\mathrm{OF}}} \times d_{l_{\mathrm{OF}}}$ analysis window, if $diff_{l_{\mathrm{OF}}}(x, y)$ is nonzero, compute its I_t (the partial derivatives of the image I with respect to time t) from an optical flow window displaced by $u^*_{l_{\mathrm{OF}}}(x, y)$ and $v^*_{l_{\mathrm{OF}}}(x, y)$, and the size of the optical flow window is $ofw_{l_{\mathrm{OF}}} \times ofw_{l_{\mathrm{OF}}}$;
- Apply the LK algorithm to get $u'_{l_{\mathrm{OF}}}$, $v'_{l_{\mathrm{OF}}}$ (the correction in flow);
- Add corrections $u'_{l_{\mathrm{OF}}}$ and $v'_{l_{\mathrm{OF}}}$, to obtain the flow $u_{l_{\mathrm{OF}}}$ and $v_{l_{\mathrm{OF}}}$: $u_{l_{\mathrm{OF}}} = u'_{l_{\mathrm{OF}}} + u^*_{l_{\mathrm{OF}}}$, $v_{l_{\mathrm{OF}}} = v'_{l_{\mathrm{OF}}} + v^*_{l_{\mathrm{OF}}}$.

End for

Notice that, in step 4, instead of applying the LK algorithm to all blocks, we apply it only to the first block in each $d_{l_{\mathrm{OF}}} \times d_{l_{\mathrm{OF}}}$ analysis window to reduce the calculation. The block size will affect the optical flow performance, and the number of layers is decided based on the size of the images and the block size. Fig. 2 shows, as shadow squares, the first blocks of the analysis windows in three layers.

Finally, the speed of a moving pixel is calculated utilizing $spd(x, y) = \sqrt{u^2_{L_{\mathrm{OF}}}(x, y) + v^2_{L_{\mathrm{OF}}}(x, y)}$.

B. Visual Word

In our work, the whole video sequence is divided into short clips with fixed length, which are regarded as documents. In addition, each frame is divided into patches. We use the features of the first pixel in each patch to describe the patch's motion, which include spatial location (x, y), moving direction dir (quantized to north, south, east, and west), and moving speed (quantized to five grades). Consequently, a visual word is represented by the vector (x, y, dir, spd). Applying overhead-view video for speed calculation will provide more precise results by removing perspective effect.

III. Two-Level Motion Pattern Mining

The proposed approach includes two levels of motion pattern mining. At each level, the LDA model, with different definitions of words and topics, is used to discover the frequent motion patterns that exist in video data. The flowchart of our approach is shown in Fig. 3.

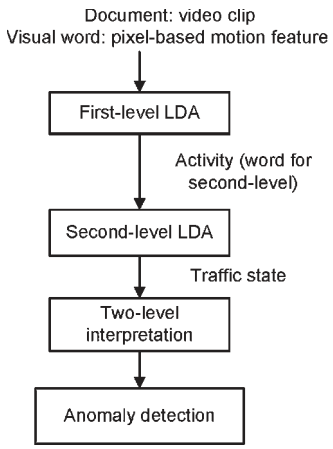


Fig. 3. Flowchart of two-level motion pattern mining.

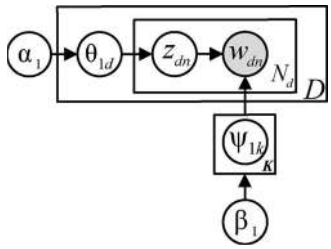


Fig. 4. First-level LDA model. Circle represents variable, whereas shaded circle represents observation. Rectangle represents replicate.

At the first level, video sequences are divided into short clips, which are regarded as documents. Patch-based motion features are regarded as visual words. In the first-level LDA modeling, activities (topics) are represented as distributions over visual words.

At the second level, we keep the same video clips as documents but consider the activities discovered by the first-level LDA as words. Due to this second-level LDA modeling, traffic states (clusters of topic) are discovered, which are represented as distributions over activities.

With the two-level motion pattern discovery, videos can be interpreted by the following hierarchical structure: patch-based motion features (visual words), activities (topics), and traffic states (clusters of topic). Specifically, every motion patch at each frame can be assigned to a certain activity. In addition, every activity appearing at one frame can be assigned to a certain traffic state. Therefore, the video can be segmented based on the assignment, and motion anomalies can be detected at two levels.

A. First-Level LDA

LDA is a generative probabilistic model for collections of discrete data (e.g., text corpora) [14]. Its graphical model is shown in Fig. 4. In the LDA model, the corpus is a collection of D documents over a word vocabulary of size W; each document $d(d = 1, \ldots, D)$ is a sequence of unordered words $\boldsymbol{w}_d = \{w_{dn}\}(n = 1, \ldots, N_d)$, where N_d is the number of words in document d, and w_{dn} represents the nth word in document d. Given the documents, LDA modeling can find out groups of cooccurring words, which are called "topics."

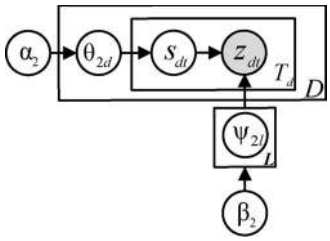


Fig. 5. Second-level LDA model. Circle represents variable, whereas shaded circle represents observation. Rectangle represents replicate.

The number of topics K is assumed known and fixed. Then, each topic is represented by a multinomial distribution over the word vocabulary, whereas each document d is represented as a multinomial distribution over topics $\boldsymbol{z}_d = \{z_{dn}\}$, where z_{dn} is the topic to which word w_{dn} is assigned. The relationships among these variables are given by

$$\begin{aligned} \psi_{1k}|\beta_1 &\sim \text{Dirichlet}(\beta_1) \\ \theta_{1d}|\alpha_1 &\sim \text{Dirichlet}(\alpha_1) \\ z_{dn}|\theta_{1d} &\sim \text{Multinomial}(\theta_{1d}) \\ w_{dn}|z_{dn}, \psi_{11:1K} &\sim \text{Multinomial}(\psi_{z_{dn}}). \end{aligned}$$

Here, ψ_{1k} is the distribution of words in topic k, which is drawn from a Dirichlet distribution with parameter β_1; θ_{1d} is the distribution of topics in document d, which is drawn from a Dirichlet distribution with parameter α_1. Both α_1 and β_1 are hyperparameters; θ_{1d} and ψ_{1k} are parameters to be estimated; z_{dn} is a latent variable; and w_{dn} is the observed data.

In our work, the goal is to find out the posterior distribution over the assignments of words to topics $p(\boldsymbol{z}|\boldsymbol{w})$, where $\boldsymbol{z} = \{\boldsymbol{z}_d\}$ and $\boldsymbol{w} = \{\boldsymbol{w}_d\}(d = 1, \ldots, D)$. Unfortunately, the distribution cannot be computed directly. Following [15]–[18], collapsed Gibbs sampling is used here to discover topics.

In the first-level LDA, topics are discovered as frequent cooccurring words shared by all documents. In fact, these topics are activities modeled as distributions over visual words and shared by all video clips.

B. Second-Level LDA

At the second-level LDA, our goal is to find out the interaction patterns defined by certain combinations of activities occurring at one time. The second-level LDA model is shown in Fig. 5.

We treat the topic discovered by the first-level LDA as the observed variable, where there are totally K topics. Assume that the number of topic clusters is L. Each cluster is represented by a multinomial distribution over topics. Each document $d(d = 1, \ldots, D)$ contains T_d topics $\{z_{dt}\}(t = 1, \ldots, T_d)$, and it can be represented by a multinomial distribution over latent clusters of topics $\{s_{dt}\}$. The relationships among these variables are given by

$$\begin{aligned} \psi_{2t}|\beta_2 &\sim \text{Dirichlet}(\beta_2) \\ \theta_{2d}|\alpha_2 &\sim \text{Dirichlet}(\alpha_2) \\ s_{dt}|\theta_{2d} &\sim \text{Multinomial}(\theta_{2d}) \\ z_{dt}|s_{dt}, \psi_{21:2L} &\sim \text{Multinomial}(\psi_{s_{dt}}). \end{aligned}$$

By performing the second-level LDA, the cooccurring activities are discovered and modeled as interactions.

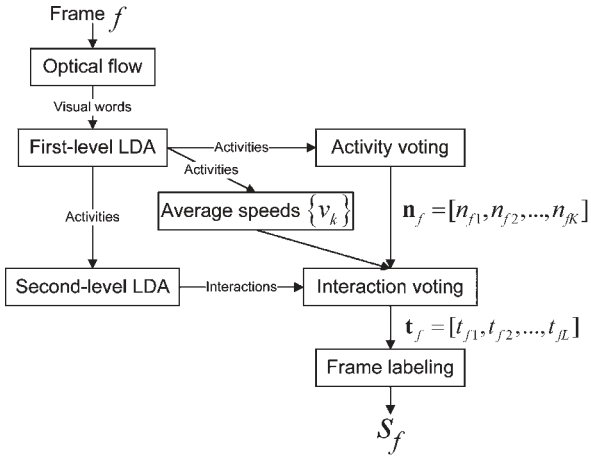


Fig. 6. Flowchart of interaction labeling.

C. Two-Level Interpretation

From the two-level LDA model, we obtain a hierarchical representation of the dynamics contained in the video: the activities modeled as distributions over visual words and the interaction patterns modeled as distributions over activities. For each video clip, we can figure out which activities a visual word belongs to and which interaction patterns an activity is assigned to. A visual word may belong to different activities in different frames. Similarly, an activity may be assigned to different interaction patterns. Furthermore, the average speed $v_k(k = 1, \ldots, K)$ of visual words in activity k can be calculated.

Based on the previous discussion, each video frame f, which belongs to video clip d, is labeled by one interaction pattern as follows, and the flowchart is shown in Fig. 6.

Step 1. Detect all visual words in frame f.

Step 2. For each word w_{dn}, find out the activities that it is assigned to, according to the first-level LDA modeling results. Count one vote for each such assignment. Obtain a vector $\mathbf{n}_f$ for frame f, which contains the sum of the votes $n_{fk}(k = 1, \ldots, K)$ for each activity k.

Step 3. Based on the vector $\mathbf{n}_f$, for each activity, find out the interactions that it is assigned to, according to the second-level LDA modeling results, and the vote $n_{fk} \times v_k$ for each possible interaction. Then, obtain the vector $\mathbf{t}_f$, which contains the sum of the votes $t_{fl}(l = 1, \ldots, L)$ for each interaction l in frame f.

Step 4. Label frame f by interaction l, which receives the highest vote among $\{t_{fl}\}$.

By repeating steps 1–4 for all frames in the video, the whole video is labeled by interaction patterns frame by frame, and it is represented as a vector $\mathbf{states}_0 = [s_1, \ldots, s_f]$, $s_f \in \{1, \ldots, L\}$, where s_f is the interaction label of frame f.

Notice that, in step 3, we add $n_{fk} \times v_k$ votes to t_{fl}, which means that the most frequent activities with faster speeds will get higher votes. It is based on the fact that, in our traffic videos, most vehicles follow the traffic rules very well, but some of the pedestrians who usually have slower speeds compared to vehicles do not. Hence, in our experiments, we define $n_{fk} \times v_k$ as the vote contributing to t_{fl} to pay more attention to moving vehicles, thus reducing the effect of pedestrian motions when we analyze interaction pattern (traffic state) labeling.

Furthermore, it also reduces the effect of vehicle motions with slow speeds. It is very helpful when considering, for example, the motion of vehicles approaching a red light; in this case, vehicles move slowly until they stop, which presents interference to traffic state labeling. If we use $n_{fk} \times (SP - v_k + 1)$ as a vote, where SP is the number of grades of quantized speed, then activities with lower speeds will get more attention. It may be used to detect speeding in some traffic scenes.

D. Temporal Constraint

In our traffic scenario, the traffic states we are detecting are naturally related to the traffic activity controlled by traffic lights. Therefore, we need to consider the temporal constraint for state detection. For example, every state in a traffic scene should have certain duration according to traffic lights transition, such as 30 or 60 s. Some labeled traffic states lasting a very short time can be errors due to noise or too few activities in the frame (not enough data to determine a traffic state). For instance, if $\mathbf{states}_0 = [1, 1, 1, 2, 1, 1, 1]$, then state 2 in vector $\mathbf{states}_0$ is probably a detection error. To address this problem, we model the traffic state transition using an HMM.

Specifically, the cooccurrence of activities in each frame can be treated as an observation $o_f, (f = 1, \ldots, F)$ (F is the total number of frames in the sequence). Then, the whole frame sequence can be treated as an observation sequence $\mathbf{O} = (o_1, o_2, \ldots, o_F)$ generated from an HMM. The hidden states correspond to the traffic states. Then, the Viterbi algorithm [19] can be used to find the most probable hidden states $\mathbf{q} = (q_1, q_2, \ldots, q_F)$.

In the Viterbi algorithm, the observation $\mathbf{O} = (o_1, o_2, \ldots, o_F)$ and model $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ are given. $\mathbf{A}$ is the state transition probability matrix. Its element $a_{ij} = P(q_f = j | q_{f-1} = i)$, $(1 \le i, j \le L)$ denotes the transition probability from state i to state j. $\mathbf{B}$ is the observation probability distribution. Here, $b_j(o_f) = P(o_f | q_f = j)$ is the probability of state j emitting the observation o_f. Finally, $\boldsymbol{\pi}$ is the initial state distribution, where $\pi_i = P(q_1 = i)$.

Then, the probability of the most probable state sequence for the first f observations $\delta_f(j)$ is given by

$$\delta_1(i) = \pi_i b_i(o_1) \tag{1}$$

$$\delta_2(j) = b_j(o_2) \max_i (a_{ij} \delta_1(i)) \tag{2}$$

$$\cdots$$

$$\delta_f(j) = b_j(o_f) \max_i (a_{ij} \delta_{f-1}(i)) . \tag{3}$$

Then, the Viterbi path $\mathbf{q} = (q_1, q_2, \ldots, q_F)$ can be retrieved, by saving all states used in (1)–(3).

We use an iterative approach similar to [20], to determine the transition probability $\{a_{ij}\}$ and to decode state sequence as follows.

Step 1. Estimate $\mathbf{B}$ from vector $\mathbf{t}_f$, i.e., $b_j(o_f) = t_{fj} / \sum_{j=1}^{L} t_{fj}$, which is to quantize the votes of states in frame f.

Step 2. Initialize π_i by the ratio between the number of frames labeled by state i and the total number of frames L in sequence $\mathbf{states}_0$.

Step 3. Initialize $\{a_{ij}\}$ by taking the ratio between the numbers of transitions from state i to state j and the total number of any transitions from state i in sequence $\mathbf{states}_0$.

Step 4. Decode states by formula (1)–(3) and gain a new sequence $\mathbf{states}^{\text{new}} = (q_1, q_2, \ldots, q_F)$;

Step 5. Recalculate $\{\pi_i\}$ and $\{a_{ij}\}$ based on $\mathbf{states}^{\text{new}}$ to get $\boldsymbol{\pi}^{\text{new}}$ and $\mathbf{A}^{\text{new}}$. For each element, if the difference of a_{ij} and a_{ij}^{new} is not small enough, go to step 4. Otherwise, convergence is reached; $\mathbf{states}^{\text{new}}$ is the interaction labels of our frame sequence.

E. Anomaly Detection

In our work, every visual word has been associated with one activity, and every frame has been associated with one of the interaction patterns. Thus, we can detect motion anomalies at two levels, as follows:

- *activity anomaly*: visual words do not belong to any of the activities;
- *interaction anomaly*: activities cannot coexist with others in that frame according to the corresponding interaction pattern.

Specifically, if a frame is classified to interaction l, all visual words in l are $\mathbf{W}_l$, the vocabulary is $\mathbf{W} = \{\mathbf{W}_1, \ldots, \mathbf{W}_l, \ldots, \mathbf{W}_L\}$, and a visual word in the frame is w, then

$$\text{if} \begin{cases} w \notin \mathbf{W}, & \text{activity anomaly} \\ w \in \mathbf{W} \,\&\, w \notin \mathbf{W}_l, & \text{interaction anomaly} \\ \text{otherwise}, & \text{normality.} \end{cases}$$

IV. Experimental Results

To analyze and understand a new monitoring scene, activities and traffic states are first discovered. Then, the video is segmented according to the labeled frames. Finally, anomalies are detected. The video data utilized come from the Next Generation Simulation (NGSIM) program [21], which is captured from the roof of a 36-story building with overhead view of the streets.

A. Words in Documents

The video is 2160 s long, and the frame rate is 10 frames/s. For a fair comparison with the method in [12], the whole video is divided into 216 clips, i.e., 10 s long each. The size of each frame is 640×480 pixel. For processing, we do not consider the boundaries of the image, so that the region of interest is 600×388 pixel.

In our experiment, $L_{\text{OF}} = 3$ denotes that we totally have three layers in the multiresolution image pyramid. At the first layer, the size of the processing unit is 1×1 block, and the optical flow window is set to 6×6 blocks. Accordingly, at the third layer, the processing unit is 4×4 blocks (pixels), and the optical flow window is 24×24 blocks (pixels). Therefore, we set the patch as 4×4 pixels, and there are 150×97 patches. Optical flows detected by the LK and MLK algorithms are shown in Fig. 7. In our case, the MLK algorithm works much better than the LK algorithm since the movements within two consecutive frames are not slow enough for the LK algorithm.

Moving directions are quantized into north, south, east, and west. Speed is quantized into five levels, using the thresholds 0.582, 2.65, 4.56, and 6.58, which represent the local minimum values of the speed histogram, as shown in Fig. 8. The statistical results are retrieved from an 180-s-long video, which contains two cycles of traffic signal change. The more levels we use, the

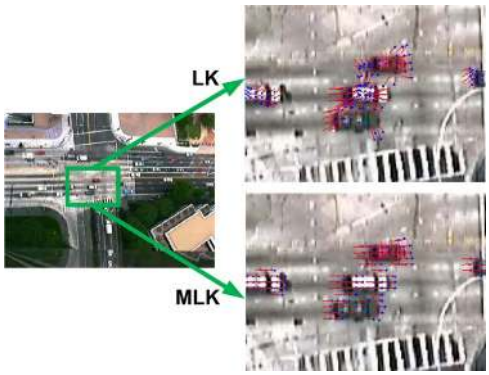


Fig. 7. LK and MLK optical flow results. Patch movements are represented by red lines, and their moving directions are represented by blue arrows.

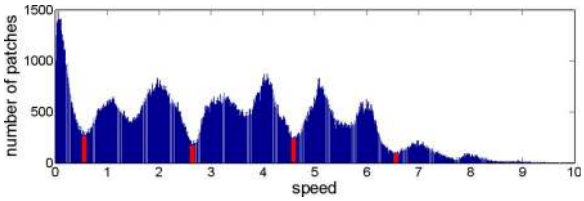


Fig. 8. Patch distribution over nonzero speeds. The X-coordinate represents the speed, whereas the Y-coordinate represents the number of patches at each speed. The red lines indicate local minimum values, which are the thresholds for speed quantization.

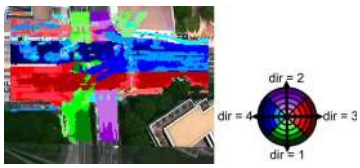


Fig. 9. Most possible motions in each patch. Four directions are represented by colors of red (east, $dir = 3$), purple (north, $dir = 2$), blue (west, $dir = 4$), and green (south, $dir = 1$). The five speed levels are indicated by five grades of color in each direction, in which lighter colors indicate lower speeds.

more details we are able to obtain and the more words we will have, which causes more calculations. In our experiment, the number of words is $150 \times 97 \times 4 \times 5$.

By analyzing all the words appearing in the whole video, we can find the most likely motion in each patch, which is shown in Fig. 9. Four directions are represented by colors of red (east, $dir = 3$), purple (north, $dir = 2$), blue (west, $dir = 4$), and green (south, $dir = 1$). The five speed levels are indicated by five grades of color in each direction, in which lighter colors indicate lower speeds.

Words in each speed level are shown in Fig. 10(a)–(e), and the distribution over speeds and directions is shown in Fig. 10(f). In Fig. 10(f), it is shown that the traffic volume descends from direction 4, direction 3, direction 1 to direction 2. In addition, the average speed in direction 2 is also the lowest because most motions in this direction are in lower speed levels as speed 1, 2, and 3.

B. Activity Learning by the First-Level LDA

Based on a number of experiments, we have found that setting the number of activities K equal to 32 is a reasonable choice for the data at hand. α_1 and β_1 are initialized as $\alpha_1 = 50/K$ and $\beta_1 = 200/W$, respectively [15]–[18]. According to our experiments and [31], larger α_1 will model each video clip

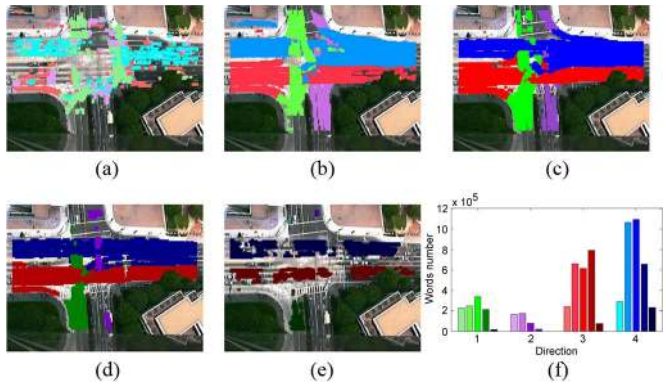


Fig. 10. Word distribution over five speed levels and four directions. (a)–(e) Words in each speed level. (f) Word distribution over speeds and directions. Colors represent speeds and directions, as shown in Fig. 9.

with more activities, and larger β_1 will force the model to assign more visual words to each activity. The discovered activities are shown in Fig. 11. We use a threshold equal to 30 to remove the noise for all these activities, i.e., to remove the visual words that appear less than 30 times in each activity. Some activities represent single trajectory (e.g., activities 15 and 28), whereas some represent multiple trajectories (e.g., activities 3 and 27). Activities 9, 18, 26, and 28 are mainly vehicles making right turns. Activities 13, 15, 21, and 23 are vehicles making left turns, while activities 1, 2, 6, and 12 are vehicles crossing the intersection. The average speed of all moving patches in each activity is calculated and shown in Fig. 11.

C. Interaction Learning by Second-Level LDA

According to the traffic signal at the intersection, it is clear that there are five traffic states in the video, which are described by the trajectories' diagrammatic sketches in Fig. 12(a). Red solid lines are trajectories of vehicles, whereas blue dotted lines are trajectories of pedestrians. The discovered traffic states are shown in Fig. 12(b). Since states are distributions over activities, these distributions are shown in Fig. 12(c) as well. The activities in each state are shown in Table I. We can easily notice that some activities (shown in color) appear in multiple states. With respect to the rules of the road, this appearance is reasonable, as explained next.

Activity 10 is included in all five states, which means that it can always happen during the video. As shown in Fig. 11, activity 10 presents vehicles moving from east to west at a lower speed. This activity can be part of different trajectories. Particularly in states 1, 2, and 4, it describes vehicles slowing down and waiting at the stop line for the green light. However, in states 3 and 5, it describes the vehicles speeding up to cross the intersection. Activity 28 describes a right turn, which is legal to take place in states 2, 3, and 4 at that intersection. In addition to these activities, there are some activities only assigned to a certain state. Those special activities are key points to distinguish the states; hence, we call them "key activities." For example, when activity 23 takes place, the system is in state 1, regardless of whether other activities in state 1 are taking place. But if none of those "key activities" are taking place in a frame, it would be hard to decide which state the system is in. This is a common source of labeling errors.

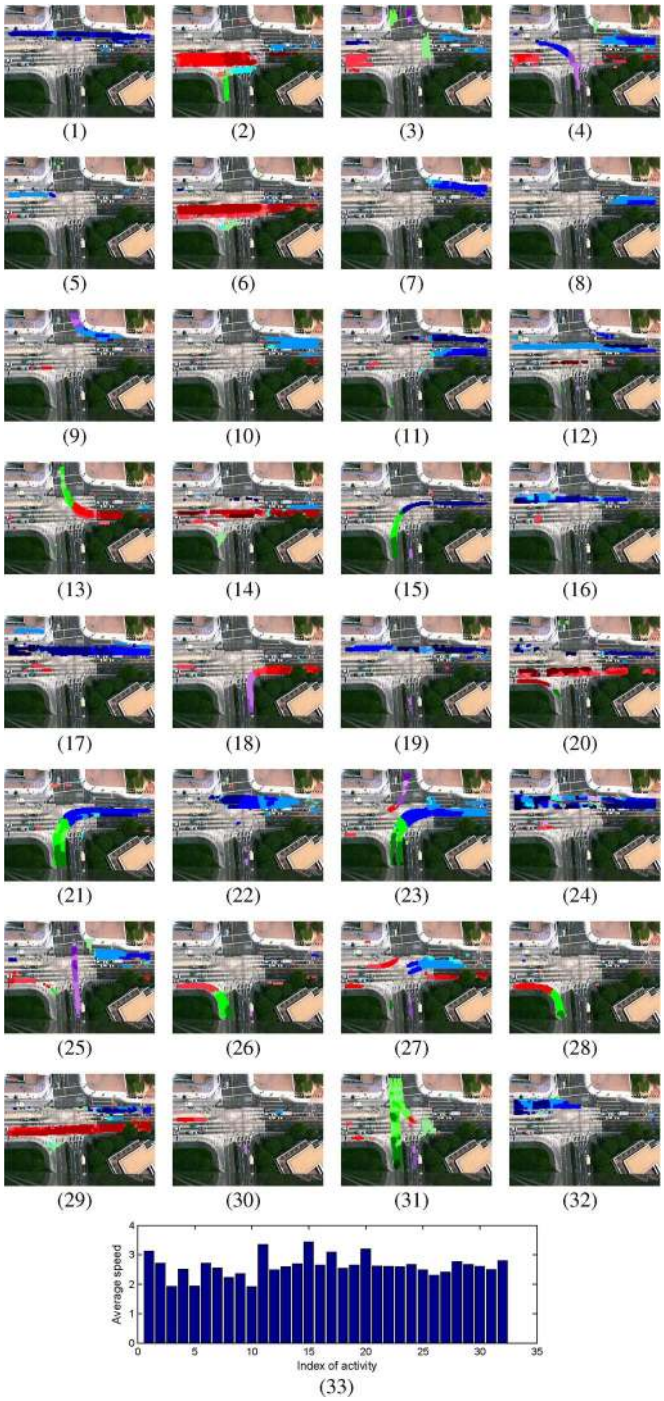


Fig. 11. Thirty-two activities and their average speeds. (1)–(32) Activities learned by the first-level LDA. (33) Average speed of all moving patches in each activity. The X-coordinate represents the activity index, whereas the Y-coordinate represents the average speed of each activity.

Turning right from west to south is split into activities 26 and 28 only because of the different speeds. The right turn appears in states 2, 3, and 4, but activity 26 only happens in state 4. This is because vehicles making a right turn from west to south should slow down and wait until no car is moving from north to south, which is not necessary in states 2 and 3. Thus, compared to other recent research results, the speed feature adopted in this paper provides a mean to accurately characterize activities in a scene.

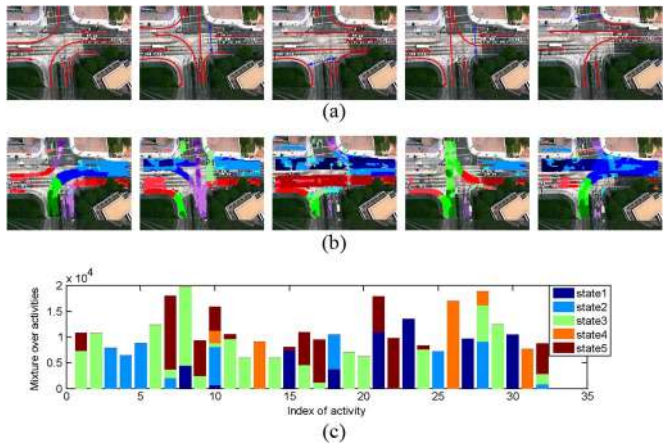


Fig. 12. Five states discovered by the second-level LDA. (a) Trajectory diagrammatic sketches in (b) traffic states. The states are numbered by 1 to 5 from left to right. (c) Mixture of activities in states, where color indicates state. The X-coordinate represents the activity, whereas the Y-coordinate represents the mixture over activities. (a) Trajectory diagrammatic sketches of traffic states (from 1 to 5). (b) Traffic states (from 1 to 5). (c) Mixtures of activities in states.

D. Video Segmentation

Each video frame is labeled by the state it belongs to, i.e., from state 1 to 5. This provides a video segmentation, as shown by the bar graph in Fig. 13. The X-coordinate represents the frame number, whereas the Y-coordinate represents the state index. The black red bars at the bottom indicate labeling errors at the corresponding frames. The accuracy of our frame-level segmentation is 81.25% for the whole video. It is calculated by the ratio between the number of correctly labeled frames and the total number of frames, which is 17551/21600 in the experiment. Most of the errors occur at the transitions between states. However, since the ground truth is labeled according to the traffic signal timing record, in real traffic scenes, there is reaction time for objects to start moving or stopping; this causes time delay.

Three segments in Fig. 13 are zoomed in and shown in Fig. 14. Red bars represent labeling errors. Since the frame is our basic segmenting unit, a video clip could be segmented into different sections. In other words, with our method, state switches can take place within a video clip. For example, the traffic state changes into state 2 within clip 43 (frame 4200 to 4300), as shown in Fig. 14(a). The labeling errors in Fig. 14(b) show a late transition from state 1 to state 5 during clips 178 and 179. The 103 state switches were detected in the whole video, but none of them happened just right at the boundary of two consecutive video clips.

The plot on the top in Fig. 15 shows the difference in errors with and without the use of the Viterbi algorithm. Green lines are the frames that were mistakenly labeled and corrected by the Viterbi algorithm, whereas the pink lines are new errors due to the use of the Viterbi algorithm. To show more details, three parts are zoomed in on the second row. The corresponding frame labeling results are shown in the third and fourth rows, in which red lines represent errors and blue lines correctly labeled frames. It is clear to see that, in the second and third columns, all the errors are corrected by considering the temporal constraint. However, in the first column, some new errors are

TABLE I
ACTIVITIES IN EACH STATE

	Activity index (probability from high to low)																	
State 1	23	21	30	27	15	8	18	10										
State 2	28	5	3	10	25	18	4	7	32									
State 3	8	29	6	2	11	24	1	28	19	20	14	12	16	9	32	7	17	10
State 4	26	13	31	28	10													
State 5	7	22	17	21	9	16	32	10	1	11	24	15						

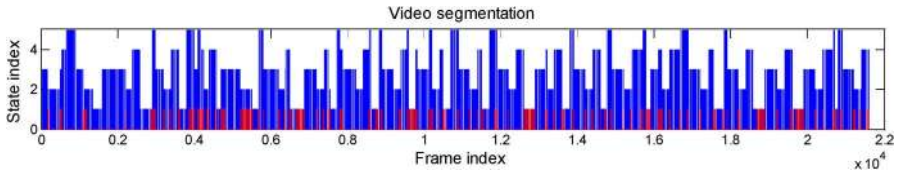


Fig. 13. Video segmentation results. The X-coordinate represents the frame index, whereas the Y-coordinate represents the state index. Red bars represent labeling errors.

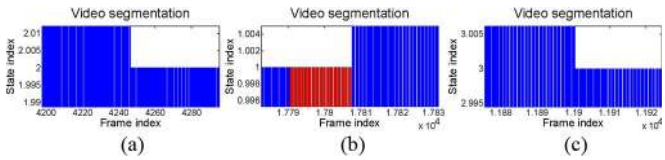


Fig. 14. State switch within a video clip. The X-coordinate represents the frame index, and the Y-coordinate represents the state index. Red bars represent labeling errors.

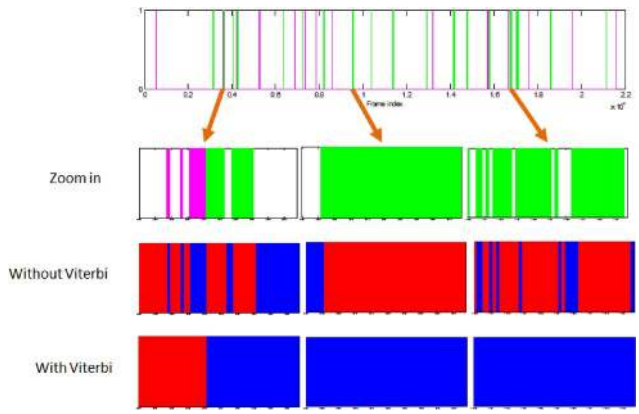


Fig. 15. Video segmentation with and without the use of the Viterbi algorithm. Corresponding to Fig. 13, the plot on the top shows the difference in errors with and without the use of the Viterbi algorithm. Green lines are the frames that were mistakenly labeled and corrected by the Viterbi algorithm, whereas the pink lines are new errors due to the use of the Viterbi algorithm. Three parts are zoomed in on the second row. Figures in the bottom rows represent frame labeling results of these three parts without and with the Viterbi algorithm. Red lines are errors, whereas blue lines are correctly labeled frames.

introduced. However, the count of correct frames is increased from 17269 to 17551 frames, and the accuracy of the frame labeling is accordingly improved from 79.95% to 81.25% by applying the Viterbi algorithm.

In Table II, the percentage of errors is shown, which is calculated as the number of errors in each situation versus the total number of errors. In particular, the number in the first column and the last row indicates that, from all the labeling errors, 30.2% of them are due to state 5 mistakenly labeled as state 1. The error happens at such a high percentage mainly because there is no key activity taking place in these frames.

TABLE II
PERCENTAGE OF ERRORS

Actual \ Labeled	State 1	State 2	State 3	State 4	State 5
State 1	0	0	0	0.0212	0.0242
State 2	0	0	0.0047	0.0141	0
State 3	0.0284	0.1119	0	0.0509	0.1190
State 4	0.1005	0.1348	0.0104	0	0.0089
State 5	0.3020	0	0.0311	0.0378	0

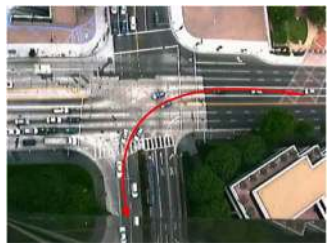


Fig. 16. Activity in frame 1550.

The only activity taking place in those frames is a left turn from east to south, which belongs to either state 1 or state 5, as shown in Fig. 16. It becomes a confusing situation during labeling.

Four frames with detected events are shown in the first column in Fig. 17. Moving pixels are colored according to their moving direction and speed. In the second and third columns, the bar graphs show the votes on per activity and the states for each frame, as described in Section III-C. Most of the frames are easily and correctly labeled according to their corresponding state votes, except for Figs. 17(a) and (c), which obtain close votes on two states. In Fig. 17(a), the key activity 25 in state 2 (a car is crossing the intersection from south to north) is taking place, while activity 26 voting for state 4 (right turn from west to south at lower speeds) also appears. The vote for activity 2 is very close to activity 4, which nearly causes a labeling error. Similarly, in Fig. 17(c), several cars are crossing the intersection from east to west, which is the key activity 22 in state 5, while other cars are making left turns, which belong to both activities 21 and 23. However, activity 23 is the key activity in state 1, which obtained higher votes in this frame. Thus, the frame is mistakenly labeled by state 1.

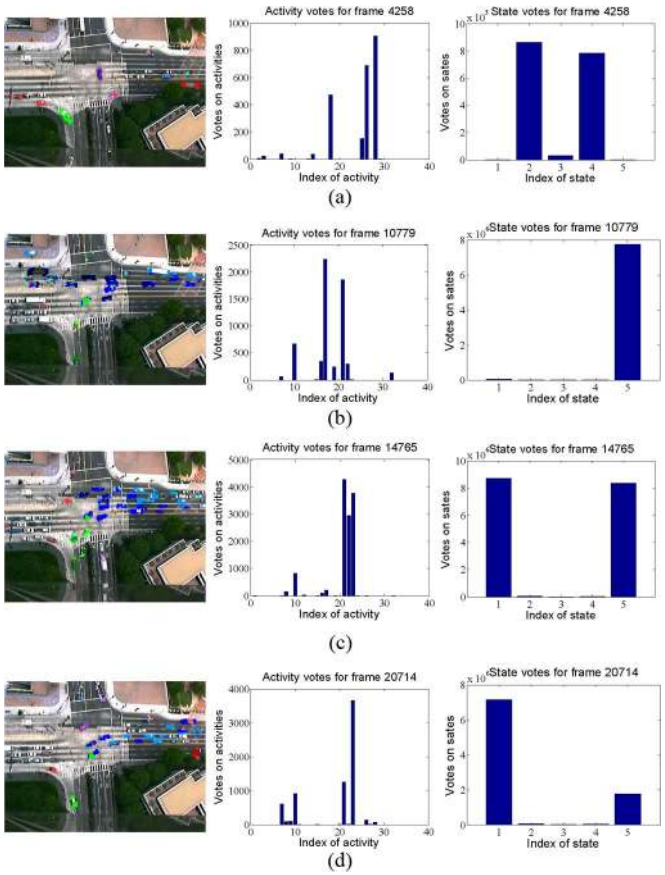


Fig. 17. Event detection per frame. For each frame, the detected visual words are shown in the first column, and the votes on activities are shown in the second column. Votes on states are shown in the third column. (a) Frame 4258, labeled as state 2. (b) Frame 10779, labeled as state 5. (c) Frame 14765, labeled as state 1. (d) Frame 20714, labeled as state 1.

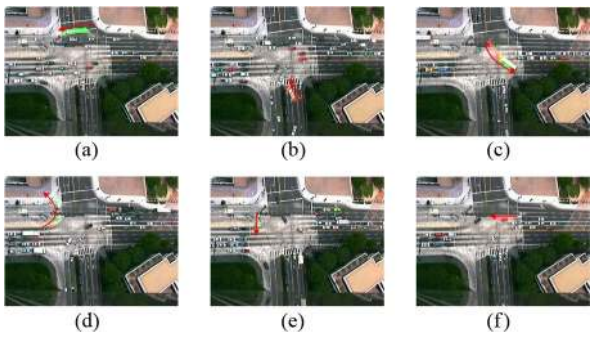


Fig. 18. Anomaly detection. Colored patches indicate detected anomalies, and arrows are used to describe the motions. Green patches are activity anomalies, whereas the red patches are interaction anomalies.

E. Anomaly Detection

Anomaly detection results for some consecutive frames are shown in Fig. 18. Colored patches indicate detected anomalies. Green patches are activity anomalies, whereas the red patches are interaction anomalies. The red arrows represent their trajectories. Particularly, in Fig. 18(a), an activity anomaly is shown that of a car moving along the crosswalk. It is not one of the legal 32 activities shown in Fig. 11. In Fig. 18(b), a car is changing lane when it is waiting at the stop line. Fig. 18(c) shows a bus blocking the way of vehicles moving from east to south. Some motion patches are activity anomalies, whereas others are interaction anomalies. The trajectory of the bus matches with

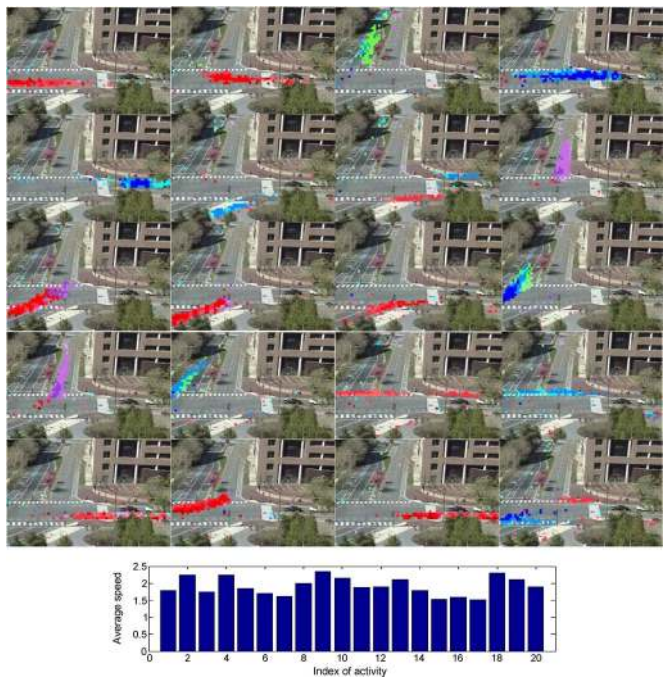


Fig. 19. Twenty activities and their average speeds. The activities are numbered from left to right and up to down.

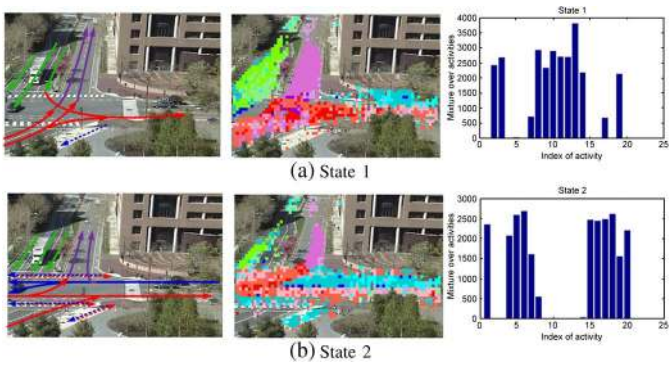


Fig. 20. Two states and the distributions over activities. (Left to right) Trajectory diagrammatic sketches, learned traffic states, and their distributions over activities. Dotted lines represent trajectories of pedestrians.

activity 13, but some patches correspond to speeding, which are detected as activity anomalies. Notice that motion cannot appear in state 1 when activities 23 or 27 are happening. In Fig. 18(d), a car is making a U-turn, while others are making a left turn. In Fig. 18(e), a pedestrian is crossing the street but not on a crosswalk. Finally, in Fig. 18(f), a car is crossing the intersection from east to west, which is an interaction anomaly that cannot exist in the state (state 1) as labeled.

F. Additional Results

1) Activity Learning by the First-Level LDA: To compare with the work of Wang, we applied our approach to the MIT video from [12], which is 5500 s long. By dividing the video into 10-s clips as Wang did, there are 550 clips in all. We consider that the size of pixel patch is 10×10, which is the same as the work of Wang. Since the image size is 720×480 and directions and speeds are quantized into four and five bins, the code book is of size $72 \times 48 \times 5 \times 4$. The number of activities K is set equal to 20. The discovered activities are shown in Fig. 19.

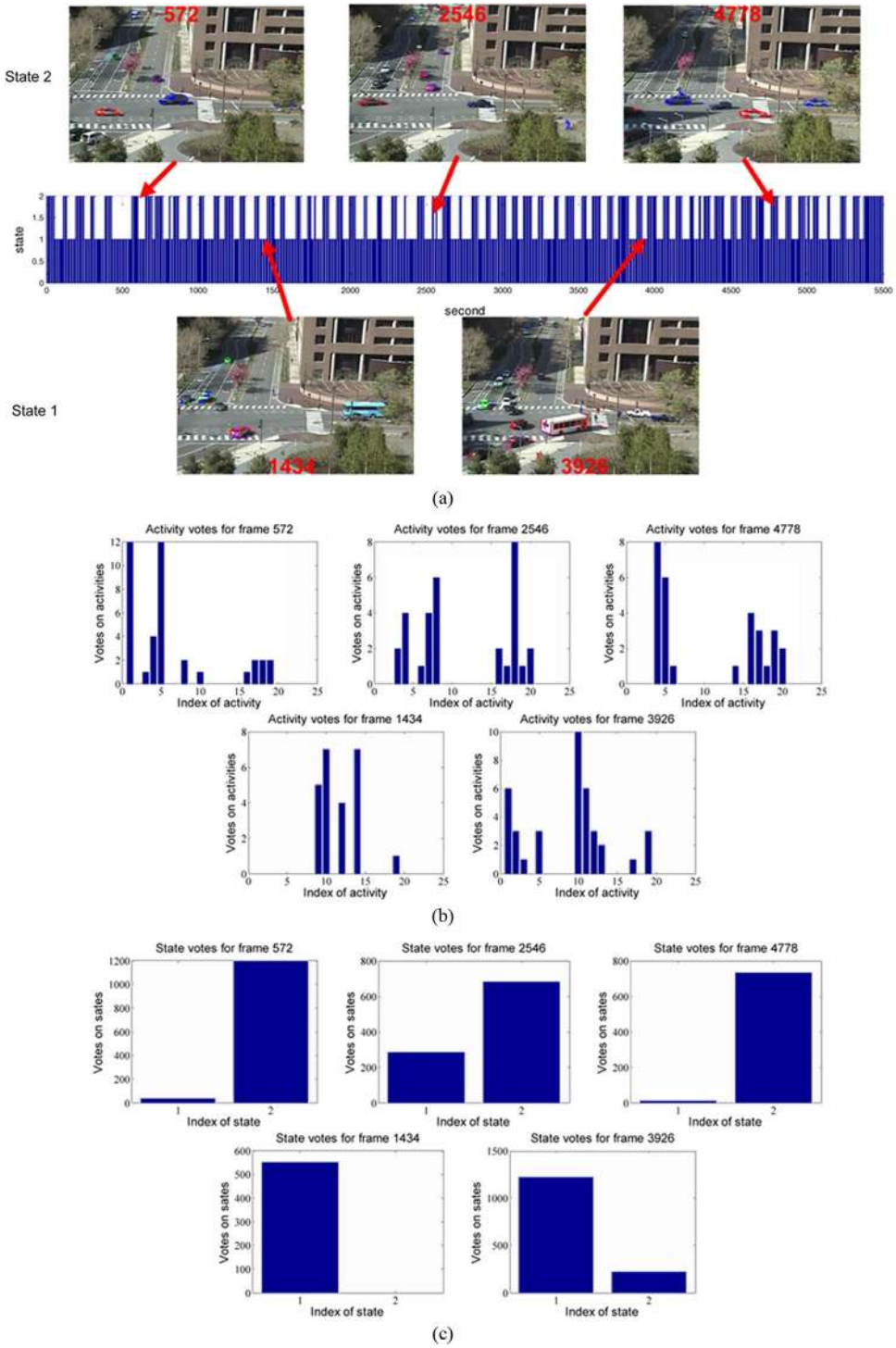


Fig. 21. Results of video segmentation and event detection. The video segmentation results are shown in the middle in (a). The X-coordinate represents the second index, and the Y-coordinate represents the state index. The three frames on the top in (a) are from state 1, while the two frames at the bottom are from state 2. Their activity and state votes are shown in (b) and (c), respectively. (a) Video segmentation and detected words in five frames. (b) Activity votes. (c) State votes.

Among these, activities 6, 7, 15, and 16 are activities of pedestrians walking on sidewalks or crosswalks, whereas the rest are vehicle activities. Different from the work of Wang, we considered moving speed as one of the parameters in visual word.

2) Interaction Learning by Second-Level LDA: There are two traffic states in this video, i.e., horizontal and vertical. Thus, the number of states is set equal to 2. The discovered two traffic states are shown in Fig. 20. It is easy to figure out that state 1 shown in Fig. 20(a) represents vertical traffic, whereas state 2 shown in Fig. 20(b) represents horizontal traffic. The bar graph to the right of each image is the corresponding distribution of activities.

3) Video Segmentation: The video segmentation results are shown by the bar graph in Fig. 21(a). All video frames are labeled by states 1 and 2. Five frames are shown with detected events. The three frames on the top are from state 1, while the two frames at the bottom are from state 2. Their activity and state votes are shown in Fig. 21(b) and (c). The accuracy of our segmentation is 84.14%.

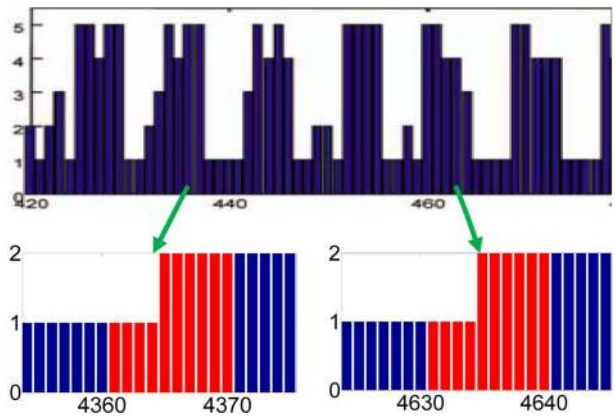


Fig. 22. Video segmentation in video clip. The figure on the top shows the clip-based (each clip contains 10 s) video segmentation obtained by [12]. The X-coordinate represents the video clip index, and the Y-coordinate represents the interaction index. Figures on the bottom show two parts of our frame-based video segmentation. The X-coordinate represents the second index (only the first frame's labeling result is sampled to show in each second), and the Y-coordinate represents the interaction index. The frames in a clip, in which a state transition occurred, are shown in red (clip 437 is shown on the bottom left, and clip 464 is shown on the bottom right).

Fig. 23. Anomaly detection. Red patches indicate detected anomalies, whereas blue dotted lines represent their trajectories.

In [12], the video was segmented based on clip clustering with five interactions. The segmentation provided by [12] is shown on the top in Fig. 22. Notice that clips 437 and 464 are labeled by interactions 5 and 4, respectively. However, since frame is our basic segmentation unit, a video clip could be segmented into different sections. In other words, our method can figure out state switches within a video clip. Our segmentation for these two clips is shown on the bottom in Fig. 22, where frames are labeled by the two states shown in Fig. 20. State transitions within clips 437 and 464 are clearly shown.

4) Anomaly Detection: Anomaly detection results are shown in Fig. 23. In Fig. 23(a), an activity anomaly is shown that of a car making a U-turn at the intersection. In Fig. 23(b), interaction anomalies are shown that of pedestrians walking across the road in the west–east direction, which is not allowed in the vertical traffic state.

V. Discussion

A reasonable question arising from the presented work is why is the two-level LDA needed. What kind of results would we obtain if we were to use one-level LDA and set the number of topics equal to 5 for our first data set? We address these questions by looking at the experimental results under the two situations. Fig. 24 shows the topics detected by LDA, when the number of topics is set equal to 5. Comparing with our five states detected by two-level LDA, which are shown in Fig. 12, there is no much difference, except for some details, e.g., left

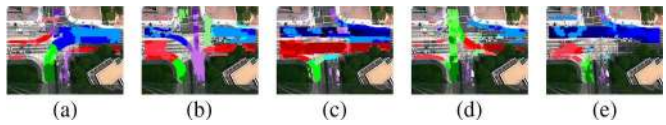


Fig. 24. Topics detected by one-level LDA.

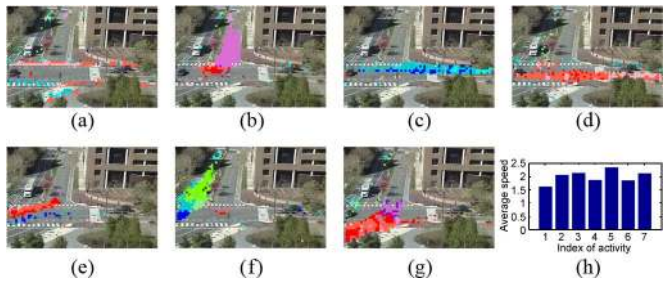


Fig. 25. Activities learned by the first-level HDP and their average speeds.

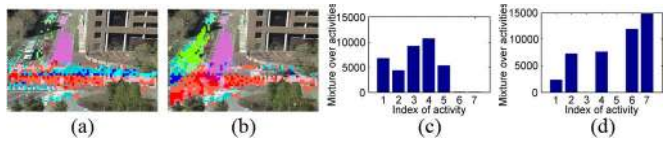


Fig. 26. Traffic states learned by the second-level HDP and their distributions over activities.

turn from east to south is not well shown in Fig. 24(e). It is because, in that topic, the left turn contains much fewer words than the rest part; these words are abandoned when we apply a threshold to remove the noise. In our two-level LDA, we have more topics in the first level and applied a threshold to all topics at the first level, which makes the result better at details. Since some parts in Fig. 24 are overlapped, when a word happens, its harder to decide which topic it belongs to. Therefore, more topics we have, less overlaps may occur, and easier decision can be made during labeling process. According to the states in Fig. 24, the accuracy of video segmentation is 72.70%. Furthermore, based on the two-level structure, we can classify the anomalies into either activity or interaction anomalies, which cannot be achieved by one-level structure.

The other question is how we decide the number of the topics in each level. As we discussed, more topics make less overlaps. However, more topics may also bring more calculation. In our experiments, we are trying to find a number of activities, which can balance these two issues well. We also set the number of activities as 16 and 48. The segmentation accuracies are 74.35% and 74.37%, respectively.

Moreover, some models, such as the HDP [22], can be adopted in the two-level structure, which can decide the number of topics during modeling process. Thus, the number of activities and traffic states can be learned automatically. Particularly for the MIT video data, Fig. 25 shows the seven activities discovered by the first-level HDP model with the initialization of 20 activities. In addition, the two traffic states learned by the second-level HDP are shown in Fig. 26.

Let us now discuss how to use our method in a realistic scenario of infinite video sequence. The method introduced previously is applied to analyze a finite video, including activity and traffic state mining, video segmentation, and anomaly

detection. It is an offline process. However, the infinite video processing contains two steps: training and testing phases. In the training phase, finite training video data are captured beforehand. Then, the two-level LDA model is adopted to learn the distribution of activities over pixel patches and traffic states over activities from the training video. Notice that the training video and the testing infinite video scenario are captured at the same place, with the same view of angle and camera parameters. It is better for the training video to contain more events. At least, it should cover a whole traffic cycle and with all kinds of legal motions. In the testing phase, the infinite scenario is treated as the testing data. When a frame is captured by the camera, we first detect the visual words by the MLK algorithm and apply the voting processing introduced in Section III-C, according to the distributions learned in the training phase. Then, label the frame by the traffic state that gains the highest vote. Thus, the scenario sequence is segmented frame by frame, and anomalies are detected as well according to our proposed method in Section III-E.

VI. Conclusion

We have proposed a hierarchical motion pattern mining approach to interpret a dynamic video scene. The LDA model is adopted to discover both activities and interactions in videos. The advantage of our method is that moving speed is considered in visual word and interactions are detected and assigned to every video frame. This enables a finer semantic interpretation and more precise anomaly detection. Experiments on real surveillance videos show that our approach is able to interpret every video frame by different traffic states and detect anomalies in each frame.

References

[1] F. Jiang, Y. Wu, and A. K. Katsaggelos, "A dynamic hierarchical clustering method for trajectory-based unusual video event detection," *IEEE Trans. Image Process.*, vol. 18, no. 4, pp. 907–913, Apr. 2009.

[2] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, "Activity recognition and abnormality detection with the switching hidden semi-Markov model," in *Proc. IEEE Conf. CVPR*, San Diego, CA, USA, Jun. 2005, vol. 1, pp. 838–845.

[3] F. Jiang, Y. Wu, and A. K. Katsaggelos, "Detecting contextual anomalies of crowd motion in surveillance video," in *Proc. IEEE Conf. Image Process.*, Nov. 2009, pp. 1117–1120.

[4] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Aug. 2008, pp. 1–8.

[5] Y. Yang, J. Liu, and M. Shah, "Video scene understanding using multi-scale analysis," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 1669–1676.

[6] I. Saleemi, L. Hartung, and M. Shah, "Scene understanding by statistical modeling of motion patterns," in *Proc. IEEE Conf. CVPR*, 2010, pp. 2069–2076.

[7] T. Hospedales, S. Gong, and T. Xiang, "Video behavior mining using a dynamic topic model," *Int. J. Comput. Vis.*, vol. 98, no. 3, pp. 303–323, Jul. 2012.

[8] D. H. Hu, X. Zhang, V. W. Zheng, and Q. Yang, "Abnormal activity recognition based on HDP-HMM models," in *Proc. 21st Int. Joint Conf. Artif. Intell.*, 2009, pp. 1715–1720.

[9] X. Zhang, H. Liu, Y. Gao, and D. H. Hu, "Detecting abnormal events via hierarchical Dirichlet processes," in *Proc. 13th Pac.-Asia Conf. Knowl. Discov. Data Mining*, 2009, pp. 278–289.

[10] D. Kuettel, M. Breitenstein, L. V. Gool, and V. Ferrari, "What's going on? Discovering spatio-temporal dependencies in dynamic scenes," in *Proc. IEEE Conf. CVPR*, Jun. 2010, pp. 1951–1958.

[11] R. Emonet, J. Varadarajan, and J. Odobez, "Extracting and locating temporal motifs in video scenes using a hierarchical non parametric Bayesian model," in *Proc. IEEE Conf. CVPR*, Jun. 2011, pp. 3233–3240.

[12] X. Wang, X. Ma, W. Eric, and L. Grimson, "Unsupervised activity perception in crowded and complicated scenes using hierarchical Bayesian models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 3, pp. 539–555, Mar. 2009.

[13] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Imaging Understand. Workshop*, 1981, pp. 121–130.

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[15] T. Griffiths, Gibbs sampling in the generative model of Latent Dirichlet Allocation, 2002. [Online]. Available: http://people.cs.umass.edu/~wallach/courses/s11/cmpsci791ss/readings/griffiths02gibbs.pdf

[16] M. Steyvers and T. Griffiths, "Probabilistic topic models," in *Latent Semantic Analysis: A Road to Meaning*, T. Landauer, D. McNamara, S. Dennis, and W. Kintsch, Eds. Mahwah, NJ, USA: Laurence Erlbaum, 2007.

[17] T. Griffiths, M. Steyvers, and J. B. T. Tenenbaum, "Topics in semantic representation," *Psychol. Rev.*, vol. 114, no. 2, pp. 211–244, Apr. 2007.

[18] T. Griffiths and M. Steyvers, "Finding scientific topics," *Proc. Nat. Acad. Sci. USA*, vol. 101, no. S1, pp. 5228–5235, Apr. 2004.

[19] R. M. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[20] F. Jiang, J. Yuan, S. A. Tsaftaris, and A. K. Katsaggelos, "Anomalous video event detection using spatiotemporal context," *Comput. Vis. Image Understand.*, vol. 115, no. 3, pp. 323–333, Mar. 2011.

[21] [Online]. Available: http://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm

[22] T. Y. Whye, M. I. Jordan, M. J. Beal, and D. M. Blei, "Hierarchical Dirichlet processes," *J. Amer. Stat. Assoc.*, vol. 101, no. 476, pp. 1566–1581, Dec. 2006.

[23] B. T. Morris and M. T. Mohan, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 8, pp. 1114–1127, Aug. 2008.

[24] B. T. Morris and M. T. Mohan, "Learning, modeling, and classification of vehicle track patterns from live video," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 425–437, Sep. 2008.

[25] X. Wang, K. T. Ma, G. W. Ng, and W. E. L. Grimson, "Trajectory analysis and semantic region modeling using nonparametric hierarchical Bayesian models," *Int. J. Comput. Vis.*, vol. 95, no. 3, pp. 287–312, Dec. 2011.

[26] H. Jeong, H. J. Chang, and J. Y. Choi, "Modeling of moving object trajectory by spatio-temporal learning for abnormal behavior detection," in *Proc. IEEE 8th Int. Conf. AVSS*, 2011, pp. 119–123.

[27] X. Tao and S. Gong, "Video behavior profiling for anomaly detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 893–908, May 2008.

[28] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1450–1464, Sep. 2006.

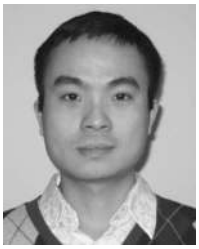
[29] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-based anomalous event detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1544–1554, Nov. 2008.

[30] N. Chiba and T. Kanade, "A tracker for broken and closely-spaced lines," in *Proc. Int. Arch. Photogramm. Remote Sens.*, 1998, vol. 32, pp. 676–683.

[31] G. Heinrich, Parameter estimation for text analysis, 2005. [Online]. Available: http://www.arbylon.net/publications/text-est.pdf

Lei Song was born in Xuzhou, China. She received the B.S. degree in information engineering, the M.S. degree in transportation planning and management, and the Ph.D. degree in communication and transportation engineering from Northwestern Polytechnical University, Xi'an, China, in 2005, 2008, and 2013, respectively.

From 2009 to 2012, she was a Joint Education Ph.D. Student with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA. She is now with The Third Research Institute of Ministry of Public Security, Shanghai, China. Her current research interests include traffic surveillance, Bayesian methods, machine learning, and crowd analysis.



Fan Jiang was born in Chengdu, China. He received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2002 and 2005, respectively, and the Ph.D. degree in electrical engineering and computer science from Northwestern University, Evanston, IL, USA, in 2011.

He is currently with Amazon.com on web services. His research interests include content-based video analysis, image/video data mining, machine learning, and computer vision.

Zhongke Shi was born in 1956. He received the B.S., M.S., and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 1981, 1985 and 1994, respectively, all in control theory and control engineering.

In 1996, he became a Professor of traffic and transportation with Northwestern Polytechnical University, Xi'an, China. His research interests include control theory, control engineering, and traffic information engineering.

Rafael Molina (M'88) was born in 1957. He received the degree in mathematics (statistics) in 1979 and the Ph.D. degree in optimal design in linear models in 1983.

In 2000, he became a Professor of computer science and artificial intelligence with the University of Granada, Granada, Spain. He was a Former Dean with the Computer Engineering School, University of Granada, from 1992 to 2002, and the Head of the Computer Science and Artificial Intelligence Department, University of Granada, from 2005 to 2007. His current research interest focuses mainly in using Bayesian modeling and inference in problems, such as image restoration (applications to astronomy and medicine), super resolution of images and video, blind deconvolution, computational photography, source recovery in medicine, compressive sensing, low-rank matrix decomposition, active learning, and classification.

Dr. Molina served the IEEE and other professional societies: Associate Editor of *Applied Signal Processing* from 2005 to 2007; Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING since 2010; Associate Editor of *Progress in Artificial Intelligence* since 2011; Area Editor of *Digital Signal Processing* since 2011. He was a recipient of an IEEE International Conference on Image Processing Paper Award in 2007, an IEEE International Symposium on Image and Signal Processing and Analysis (ISPA) Best Paper Award in 2009, and a European Signal Processing Conference Paper Award in 2013. He is a coauthor of a paper awarded the runner-up prize from the Reception for Early-Stage Researchers at the House of Commons.

Aggelos K. Katsaggelos (F'98) received the Diploma degree in electrical and mechanical engineering from the Aristotelian University of Thessaloniki, Thessaloniki, Greece, in 1979, and the M.S. and Ph.D. degrees in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 1981 and 1985, respectively.

In 1985, he joined the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA, where he is currently a Professor and an AT&T Chair. He was the Ameritech Chair of information technology from 1997 to 2003. He is also the Director of the Motorola Center for Seamless Communications, Northwestern University; a member of Academic Staff with the NorthShore University Health System, Evanston, IL, USA; and an Affiliated Faculty with the Department of Linguistics, Northwestern University. He has an appointment with the Argonne National Laboratory, Lemont, IL, USA. He has published extensively in the areas of multimedia signal processing and communications over 180 journal papers, 450 conference papers, and 40 book chapters. He is the holder of 20 international patents. He is the coauthor of *Rate-Distortion Based Video Compression* (Kluwer, 1997), *Super-Resolution for Images and Video* (Claypool, 2007), and *Joint Source-Channel Video Transmission* (Claypool, 2007).

Dr. Katsaggelos was the Editor-in-Chief of the IEEE Signal Processing Magazine from 1997 to 2002, a Board of Governors Member of the IEEE Signal Processing Society from 1999 to 2001, and a member of the Publication Board of the IEEE Proceedings from 2003 to 2007. He is a Fellow of the SPIE (2009). He is a recipient of the IEEE Third Millennium Medal in 2000, the IEEE Signal Processing Society Meritorious Service Award in 2001, the IEEE Signal Processing Society Technical Achievement Award in 2010, an IEEE Signal Processing Society Best Paper Award in 2001, an IEEE International Conference on Multimedia and Expo (ICME) Paper Award in 2009, an IEEE International Conference on Image Processing (ICIP) Paper Award in 2007, an IEEE International Symposium on Image and Signal Processing and Analysis (ISPA) Paper Award in 2009, and a European Signal Processing Conference (EUSIPCO) Paper Award in 2013. He was a Distinguished Lecturer of the IEEE Signal Processing Society from 2007 to 2008.