Effective Gaussian Mixture Learning for Video Background Subtraction

Dar-Shyang Lee, *Member*, *IEEE*

Abstract—Adaptive Gaussian mixtures have been used for modeling nonstationary temporal distributions of pixels in video surveillance applications. However, a common problem for this approach is balancing between model convergence speed and stability. This paper proposes an effective scheme to improve the convergence rate without compromising model stability. This is achieved by replacing the global, static retention factor with an adaptive learning rate calculated for each Gaussian at every frame. Significant improvements are shown on both synthetic and real video data. Incorporating this algorithm into a statistical framework for background subtraction leads to an improved segmentation performance compared to a standard method.

Index Terms—Adaptive Gaussian mixture, online EM, background subtraction.

1 Introduction

THE ability to efficiently and accurately estimate nonstationary temporal distributions is a key element for any robust vision system. Adaptive Gaussian mixtures are commonly chosen for their analytical representation and theoretical foundations. For these reasons, they have been employed in real-time surveillance systems for background subtraction [3], [9] and object tracking [5]. However, in currently existing formulations, temporal adaptation is achieved at a price of slow convergence or large memory requirement, limiting their utility in real-time video applications. This paper proposes an effective online learning algorithm that significantly improves modeling convergence and accuracy for adaptive Gaussian mixture modeling of dynamic distributions. The technique is applied to video background subtraction under a statistical framework, and it led to improved performance over conventional methods.

Nearly all applications involving Gaussian mixtures are trained by some variant of the EM algorithm. The real-time and dynamic nature of surveillance applications prohibits direct usage of the batch EM [1] or incremental variant [6]. An online approximation that learns incrementally with each new observation and adapts locally within a temporal window without imposing a large memory requirement is needed. In the surveillance system of Stauffer and Grimson [9], which has become the standard formulation for the mixture approach in the field, an online EM approximation based on recursive filter was used to train the mixture background model. The rate of adaptation is controlled by a global parameter α that ranges between 0 and 1. In order to preserve a reasonably long learning history and maintain system stability, a very small constant is typically used for video applications. Unfortunately, this results in slow convergence whenever a Gaussian adapts to a new cluster. The effect of this learning inefficiency extends beyond system initialization; it applies to every new foreground object and background changes encountered later. Furthermore, it often adversely affects the learning dynamics such that the parameters converge to a poor solution. Choosing a larger value for α would improve the convergence rate. However, modeling becomes very sensitive to noise and retains too little history to be useful in practice. Few papers have proposed a clear solution that factored out the parameter convergence issue from that of system adaptability and stability.

In the work of Friedman and Russell for traffic monitoring [2], Gaussian mixtures were trained using a variant of EM where mixture parameters were computed from incrementally updated sufficient statistics. The resulting model reflected the long term cumulative distribution and could not adapt to distribution changes without a mechanism for "exponential forgetting." KaewTraKulPong and Bowden [4] proposed using expected sufficient statistics update equations at the initial learning stage to improve convergence and switching to recursive filter learning after sufficient samples were observed. However, this two-stage approach would not benefit learning new foreground objects at a later stage, where effective learning is most needed due to the small number of samples. Furthermore, update equations for the second stage were flawed and could lead to divergence. McKenna et al. offered a derivation for training mixture models to track video objects [5]. However, their solution required memory proportional to the temporal window size, making it unsuitable for applications where mixture modeling is pixel-based and over a long temporal window. The formulation of adaptive mixture learning with reassignments is also similar to the online EM algorithm proposed for Normalized Gaussian Networks by Sato and Ishii [8]. They proposed a mechanism for adding, deleting, and splitting Gaussians to handle dynamic distributions similar to Gaussian reassignments and suggested temporal adaptation could be achieved by manipulating a discount factor. However, it is unclear how to define the schedule of the discount factor to implement the behavior required in surveillance applications. Furthermore, this method requires storage for sufficient statistics in addition to model parameters. In the context of video data modeling, where every pixel is presented by a mixture, this can be a significant overhead.

The principle of our approach is to incorporate incremental EM type of learning into a recursive filter such that parameter learning of each Gaussian follows a 1/t schedule with the first few observations and gradually approaches the basic recursive filter learning after many observations. Details of the algorithm are described in Section 2. The proposed method is applied to video background subtraction in a statistical framework described in Section 3, and evaluated against the standard learning method using both synthetic data and in the context of background subtraction on real videos in Section 4. Conclusions are drawn in Section 5.

2 Adaptive Mixture Learning

The basic concept for different mixture learning approaches is best understood in terms of the recursive filter formulation with a learning rate schedule:

$$\theta(t) = (1 - \eta(t)) \cdot \theta(t-1) + \eta(t) \cdot \nabla(x(t); \theta(t-1)),$$

where the model at time t, $\theta(t)$, is updated by a local estimate $\nabla(x(t); \theta(t-1))$ at a rate controlled by $\eta(t)$. If $\eta(t) = 1/t$, as in the formulation of [2], [8] based on sufficient statistics, current parameter estimates are weighed proportionally to the number of observations, allowing it to quickly approach the expected value at the initial stage and converge to the local-optimal estimation over an infinite number of observations on a stationary distribution. If $\eta(t) = \alpha$, as in the formulation of [3], [9], the parameter estimates reflect the most recent observations within a roughly $L = 1/\alpha$ window with exponential decay. However, it takes proportionally longer for the parameters to converge.

• *The author is with Ricoh California Research Center, 2882 Sand Hill Road, Suite 115, Menlo Park, CA 94025. E-mail: dsl@rii.ricoh.com.*

Manuscript received 11 Sept. 2003; revised 30 Aug. 2004; accepted 9 Nov. 2004; published online 11 Mar. 2005.
Recommended for acceptance by K. Daniilidis.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0270-0903.

TABLE 1
The Proposed Mixture Learning Algorithm in Pseudocode

Control variables: $K\ V_0\ \alpha\ T_\sigma$
Initialization: $\forall_{j=1..K}\quad w_j = 0\quad \mu_j = Inf\quad \sigma_j = V_0\quad c_j = 0$

While new data x(t)

$$\forall_{j=1..K}\ \ p_j = \begin{cases} w_j \cdot g_j(x;\mu_j,\sigma_j) & \text{if } \dfrac{|x-\mu_j|}{\sigma_j} < T_\sigma \\ 0 & \text{otherwise} \end{cases}$$

If $\sum_{j=1}^{K} p_j > 0$ **Then** // at least one match is found

For (k=1; k<K; k++)

$q_k = p_k / \sum_{j=1}^{K} p_j$ // expected posterior of G_k

If Winner-Take-All **Then**

$$q_k = \begin{cases} 1 & \text{if } k = \arg\max_j \{p_j\} \\ 0 & \text{otherwise} \end{cases}$$

End If

$w_k(t) = (1-\alpha)\cdot w_k(t-1) + \alpha \cdot q_k$

If $q_k > 0$ **Then** // for matched Gaussians

$c_k = c_k + q_k \qquad \eta_k = q_k \cdot (\dfrac{1-\alpha}{c_k} + \alpha)$

$\mu_k(t) = (1-\eta_k)\cdot \mu_k(t-1) + \eta_k \cdot x$

$\sigma_k^2(t) = (1-\eta_k)\cdot \sigma_k^2(t-1) + \eta_k \cdot (x-\mu_k(t-1))^2$

End If

End For

Else // no match found

$\forall_{j=1..K}\quad w_j(t) = (1-\alpha)\cdot w_j(t-1)$

$k = \arg\min_j \{w_j\} \qquad w_k = \alpha \qquad \mu_k = x \qquad \sigma_k = V_0 \qquad c_k = 1$

End If

normalize w

End While

A solution that combines fast convergence and temporal adaptability is to use a modified schedule such that $\eta(t)$ converges to α instead of zero, and to compute $\eta(t)$ for each Gaussian based on current likelihood estimates in a manner consistent with the incremental EM algorithm. To account for the fact that not all Gaussians are updated with every sample and can be reassigned, $\eta(t)$ is computed for each Gaussian independently from the cumulative expected likelihood estimate. This ensures the same effective learning for each Gaussian component is applied through out all stages of the system, including reassignment and system reset. In the following subsections, we present details of the algorithm, then discuss its expected behavior and compare it to other methods.

2.1 Proposed Algorithm

The basic algorithm follows the formulation by Stauffer and Grimson [9]. Table 1 shows the proposed algorithm in pseudocode. For simplicity, the notation is based on a one-dimensional signal. Extension to a higher dimension is straight-forward.

Several differences between the solution in Table 1 and the basic recursive filter learning should be noted. The most important is the calculation of the learning rate η_k. A variable c_k is introduced to count the number of effective observations for Gaussian G_k and compute the appropriate learning rate. It is incremented when parameters of a Gaussian are updated. When the Gaussian is reassigned, it is reset to 1 since the old Gaussian has perished and a new one was started with a single data point. In all our simulations, this modification significantly improved the convergence speed and model accuracy with almost no adverse effects.

The solution above shows a general formulation where multiple Gaussians may be updated for a single sample. For efficiency reasons, the winner-take-all option where only a single best-matching component is selected for parameter update is typically used. However, if the distribution contains two significantly overlapping clusters, this strategy can lead to starvation where one Gaussian stretches with increasingly more weight to over-dominate the others. A solution to overcome this problem is to use soft-partition where all Gaussians that match a data point are updated by an amount proportional to their estimated posterior probability $P(G_k|x)$ [7]. At the cost of additional updates, this helps improve robustness in early learning stage for components whose variances are too large and weights too small to be the best match.

Another difference from the conventional formulation is the treatment of weight updates. Instead of initializing weights to an arbitrary W_0 and rely on renormalization, it can be shown that the weight updating equations are actually self-normalizing with $W_0 = \alpha$. There is also good justification in that it is the appropriate weighting for observing one data point. Furthermore, it is understood that context-dependent criteria, such as $\text{argmin}_j\{w_j/\sigma_j\}$ [9] or $\text{argmin}_j\{P(B|G_j)\}$ based on the likelihood of being background, can be used for selecting Gaussians for reassignment. In the experiments with video data, the generic $\text{argmin}_j\{w_j\}$ worked equally well.

2.2 Analysis and Related Work

We provide an informal analysis of the proposed algorithm and discuss its relationships to other methods. It should be recognized that c_k, which is the sum of the expected posterior of G_k, is one of the sufficient statistics computed in incremental EM

$$c_k(t) = \sum_{i=1}^{t} q_k(i) = \sum_{i=1}^{t} \hat{P}(G_k|x(i), \theta(i-1)).$$

It effectively serves as an individualized clock for the learning update of each Gaussian. From the learning rate update equation, it is clear that in the initial learning stage of a Gaussian when only a few samples have been observed, $\eta_k \approx 1/c_k$, and parameter updates closely approximate the expected value computed by sufficient statistics. As more samples are included in its estimation, η_k approaches α and behaves like the typical recursive learning.

More qualitatively, the proposed learning schedule differs from existing methods in the following way. The algorithm reported in [9] did in fact use a time varying learning rate $\eta_k(t) = \alpha \cdot g_k(x)$. Since $g_k(x) = P(x|G_k)$ is very small, this usually makes convergence slower than simply using $\eta_k(t) = \alpha$ [3], [4]. Another reasonable modification is to use $\eta_k(t) = \alpha \cdot P(G_k|x)$. Weighing by the expected posterior achieves the soft partitioning described earlier. Nonetheless, the convergence rate would be the same as using a static α.

In the method proposed in [4], learning at the initial stage is computed directly from sufficient statistics to obtain $\eta_k(t) = q_k/c_k(t)$. After the first $L = 1/\alpha$ samples, learning switches to the *L-window* rule which can be shown to be equivalent to

$$\mu_k(t) = (1-\alpha) \cdot \mu_k(t-1) + \eta_k \cdot x(t) \qquad \eta_k(t) = \alpha \cdot q_k/w_k(t).$$

Since $w_k(t) = c_k(t)/t$, it follows that $\eta_k(t) \approx \alpha \cdot t/c_k(t)$. However, this explicit division in learning stages applies only at system initialization, and does not benefit new or reassigned Gaussians adapting to new clusters. Furthermore, since η_k is unbounded and $(1-\alpha) + \eta_k$ may exceed 1 in the *L-window* rule, the algorithm sometimes display a diverging behavior. In contrast, the new method considers each Gaussian separately and applies the appropriate learning rate throughout all phases of learning.

3 Background Subtraction

To assess the impact of the learning algorithm in a real system, we applied mixture modeling to background subtraction using a statistical framework. It is understood that the goal of background segmentation is often application-driven and involves interaction with higher level semantics and domain-specific constraints as suggested in [10]. Nevertheless, the analytical form of mixture modeling allows an explicit expression of the Bayesian solution for the pixel-level segmentation problem in terms of mixture densities. The results in this paper can be augmented by problem-specific constraints and used for other applications.

At the pixel level, background segmentation involves a binary classification problem based on $P(B|x)$, where x is the pixel value at time t, and B represents the background class. With an explicit representation of the temporal distribution $P(x)$ as a mixture,

$$P(x) = \sum_{k=1}^{K} P(G_k)P(x|G_k) = \sum_{k=1}^{K} w_k \cdot g(x; \mu_k, \sigma_k),$$

the posterior probability can be expressed in terms of the mixture components $P(G_k)$ and $P(x|G_k)$, and a density estimate $P(B|G_k)$ as follows:

$$P(B|x) = \sum_{k=1}^{K} P(B|G_k)P(G_k|x) = \frac{\sum_{k=1}^{K} P(x|G_k)P(G_k)P(B|G_k)}{\sum_{k=1}^{K} P(x|G_k)P(G_k)}.$$

The mixture components are obtained through the learning algorithm described in Section 2. The estimation of $P(B|G_k)$ invariably involves heuristics encapsulating the domain knowledge of the *background* process. In [2], the Gaussians are manually labeled and remain fixed. In [9], $P(B|G_k)$ equals to 1 for Gaussians with the highest w/σ covering a certain percentage of observations, and 0 for all others. We train a sigmoid function on w/σ to approximate $P(B|G_k)$ using logistic regression

$$\hat{P}(B|G_k) = f(w_k/\sigma_k; a, b) = 1/(1 + e^{-a \cdot w_k/\sigma_k + b}).$$

Once $P(x)$ and $P(B|G_k)$ are estimated, the foreground region is composed of pixels where $P(B|x) < 0.5$. Obviously, there is a pragmatic issue that our estimate of $P(x)$ maybe inaccurate, even more so for $P(B|G_k)$. However, we found the default threshold of 0.5 worked quite well with a fitted sigmoid trained on some representative data.

To obtain a visual representation of the background model, an intuitive solution is to compute the expected value of the observations believed to be background. More precisely,

$$E[X|B] = \sum_{k=1}^{K} E[X|G_k] \cdot P(G_k|B) = \frac{\sum_{k=1}^{K} \mu_k \cdot P(B|G_k)P(G_k)}{\sum_{j=1}^{K} P(B|G_j)P(G_j)}.$$

The image is calculated as an average of the Gaussian means, weighted proportionally by their weights and posterior probabilities of being background.

4 Experimental Results

The proposed mixture learning is tested and compared to conventional methods using both simulation and real video data. In the first set of experiments, the effectiveness of the proposed mixture learning algorithm is verified using a set of one-dimensional synthetic data whose temporal distributions are generated to simulate real video data. In the second set of experiments, the proposed mixture learning is applied to background segmentation under the framework described in Section 3, and the results are compared to the commonly adopted formulation of [9].

4.1 Mixture Learning Experiment

First, the performance of the proposed mixture learning algorithm is evaluated through quantitative analysis on a set of synthetic data. Ten data files were created for each of the three types of distributions commonly observed in surveillance videos. Examples are shown in Fig. 1. The most typical distribution contains a dominant cluster representing the background, interspersed with occlusions from foreground objects, as shown in Fig. 1a. The background process can also drift over time due to lighting changes or shift suddenly due to occlusion by a new object, as shown in Fig. 1b. Fig. 1c illustrates a difficult case where the process is bimodal, as introduced by flickering monitors or fluorescence lights. These data sets are created by sampling from underlying distributions represented as Gaussian mixtures with known parameters.

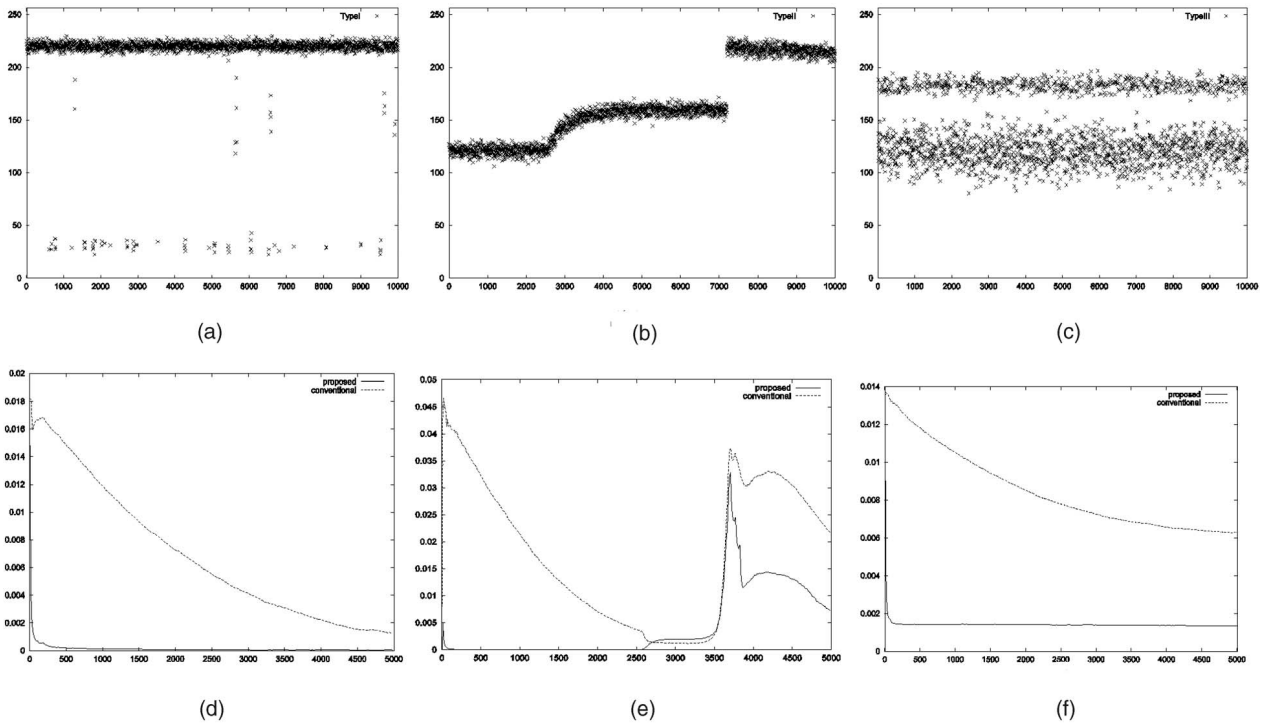

Fig. 1. (a), (b), and (c) Three types of synthetic distributions generated based on real video data. Pixel intensities on the Y-axis are plotted over time. (d), (e), and (f) show the distance (KL-divergence) between the estimated models and the ground truth for those distributions. The divergence achieved by the proposed (solid) and basic recursive filter (dashed) averaged over 10 data sets are plotted over time.

To evaluate the performance of learning methods, the estimated models are compared against the true distribution at every iteration and the difference is plotted over time. The winner-take-all version of the proposed method is compared to the basic recursive learning with a fixed α. Due to the approximation by finite support and winner-take-all update, it is easier for a wide Gaussians to shrink its variance than for a narrow Gaussian to widen its support. Consequently, without a priori knowledge of the distribution, the initial variances are customarily set to a large value. Altogether, over 100 trials were performed on 30 data files under different parameter settings with α between 0.001 and 0.1, K ranged 3 to 5, V_0 equaled 10 and 30. The contrast in performance is quite consistent. Figs. 1d, 1e, and 1f show the results for the three types of distributions using $\alpha = 0.001$, $K = 3$, $T_\sigma = 3$, and $V_0 = 30$. The Y-axis is the KL-divergence between the estimated model and the true distribution computed on a 256-point mesh, averaged over 10 data sets. The effectiveness of the proposed learning method is apparent in Fig. 1d, which achieved good estimation much faster than the conventional method. Another important difference not visible from the figure is that the proposed method obtained an accurate estimation of the small foreground clusters as well, whereas the other method could not find a good estimate due to their low frequencies. The same contrast can also be observed in Fig. 1e both at the beginning as well as the recovery from sudden distribution changes. The learning curve in the recovery portion is not monotonic because some datasets have randomly generated more distribution shifts in those regions. The proposed algorithm is also more robust when dealing with the difficult bimodal distributions. As shown in Fig. 1f, the proposed method not only converged faster but also achieved better accuracy. The standard method, represented by the dashed curve, would eventually stabilize at a much worse value. A closer examination reveals that it tends to merge data into a single cluster because of its slow convergence on the variance and often cannot recover from a local minimum.

4.2 Background Segmentation Experiment

The proposed solution for mixture learning was applied to background segmentation on traffic and meeting videos, and compared to the method of [9]. Although these two methods do not use the exact same framework, that difference mainly affects the labeling and visualization of the background model and has little direct impact on the segmentation results. The difference in segmentation performance is almost entirely due to the mixture learning algorithm.

Gaussian mixtures with K = 3 on *YUV* input and diagonal covariance matrices were learned using $\alpha = 0.005$. Ideally, the background classifier would be trained on ground-truthed samples. In these experiments, we bootstrapped this process using a training video by first learning $P(x)$ mixture, manually adjusting the threshold on w/σ to get a reasonable segmentation, then using the pixel labels to train the sigmoid. As a result, $a = 96$ and $b = 3$ with w_k/v_k as input where v_k is the determinant of the covariance matrix normalized by vector dimension. The same sigmoid function is then used for all pixel locations in the test videos. The same parameter settings are used for both methods on both video sequences. Furthermore, no domain-specific processing such as skin tone detection or shadow removal was applied. The system runs at 15fps on a 160 x 120 video on a 2GHz Pentium4 PC.

The segmentation results were much better using the proposed learning method. The proposed method was able to segment out the cars from an early stage; whereas the conventional learning method was able to pick out only very high contrast area. To get a sense of how well the underlying modeling process performed, we computed the likelihood of each frame predicted by the model estimated at the previous frame. Fig. 2 shows the log-likelihood of the proposed solution in solid line ("*proposed*"), conventional method in dotted line ("*conventional*"), plotted over one minute of the traffic sequence. The proposed method quickly converged to a good estimate in less than 100 frames, an order of magnitude faster than the standard formulation. Even after converging, the

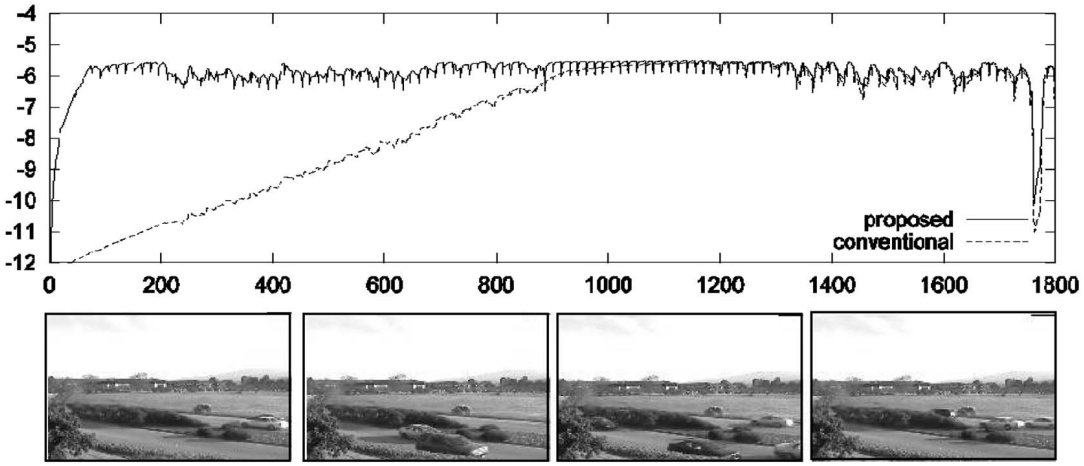

Fig. 2. Log-likelihood of the image computed by estimated models plotted over time for the traffic video. Using the proposed algorithm, shown by the solid curve (proposed), the model achieved good estimation very quickly in less than 100 frames. Learning with a basic recursive filter, shown by the dotted curve (conventional), is much slower.

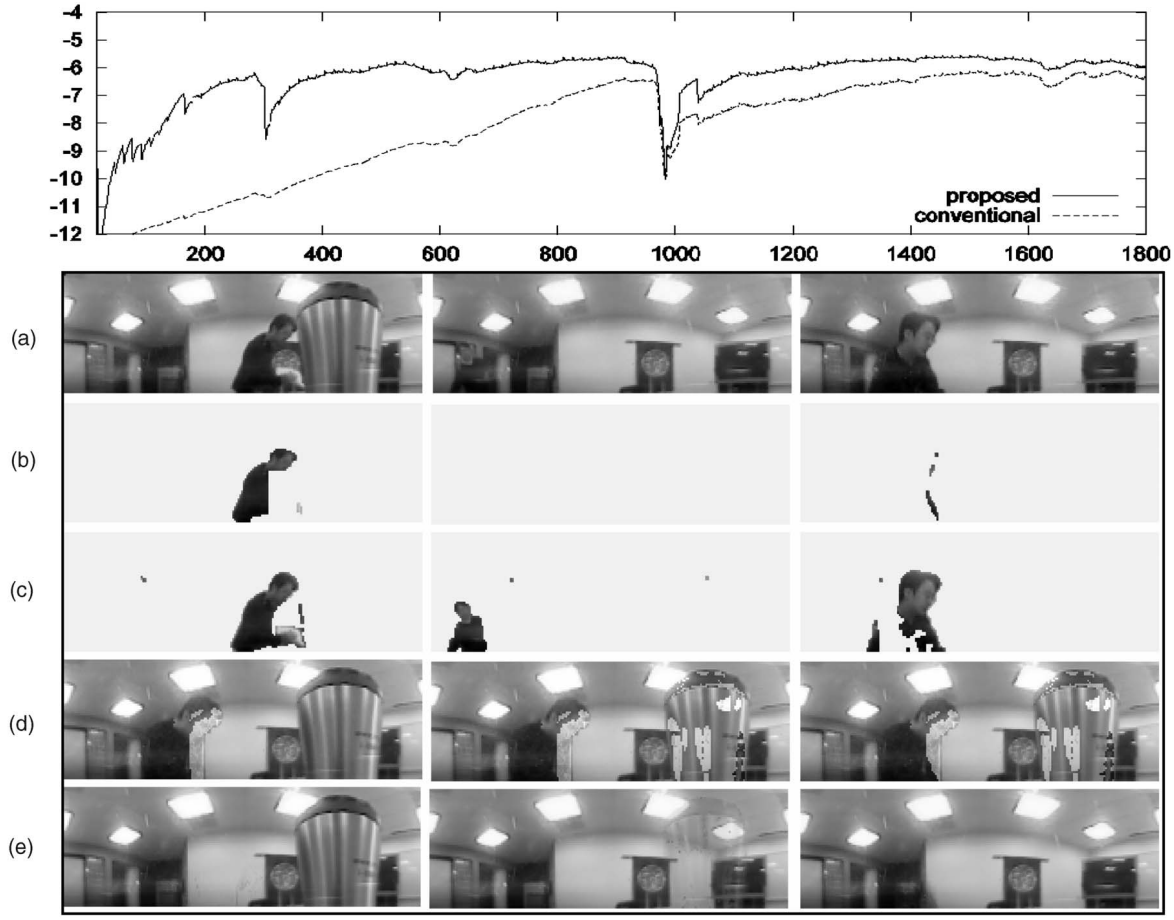

Fig. 3. A comparison of the log-likelihood achieved by the models (top) and segmentation results (bottom) on the meeting video. Row (a) shows three original frames, 20 seconds apart. Segmented foreground obtained by method of [9] shown in (b) detected only high contrast regions. The proposed method, shown in Row (c), achieved much better segmentation. Last two rows show background models produced by the labeling rule (d) and the proposed formulation (e), respectively.

proposed method adapted to changes caused by on-coming traffic better than the conventional method, as indicated by the higher likelihood at frame 1,300. The comb-like artifacts are periodic drops in likelihood attributed to mpeg compression.

The algorithms were also compared on a meeting video sequence recorded by an omni-directional camera. Unlike the traffic surveillance video where the background and lighting conditions change gradually, these indoor sequences can contain abrupt background changes due to displacements of objects on the table, the occlusion of overhead lighting, and the switching on or off a projector or monitor. Similar improvements are observed from the meeting video results shown in Fig. 3. In the three frames in Fig. 3a taken approximately 20 seconds apart, the conventional method Fig. 3b picked out the person only when he moved into a high contrast area; whereas the proposed method in Fig. 3c achieved almost perfect segmentation except in the last frame

where part of the person's body was missing because his shirt color matched the dark background. The effectiveness of the algorithms can be explained using the background models shown at the bottom Fig. 3d shows the mean of the background Gaussian produced by the labeling rule of [9], and Fig. 3e shows the background model obtained using the framework in Section 3. From the first column, which is 20 seconds into the video, it can be seen that the standard technique, shown in Fig. 3d, retained the initial value of the background from the first frame almost entirely. The person's original position at time zero was clearly visible. The proposed method, shown in Fig. 3e, however, obtained a good estimate of the room without a person although the room was never empty at any moment. Half way between the first column and the second column, the coffee mug was removed. As the background model began to shift, it can be seen from the middle column that the old method made the transition by leaving out fragments of the mug; whereas in the proposed framework, the transition was made by fading out the mug. Finally, in the last column, the previous method still had a poor estimate of the background after one minute and the proposed method had constructed a new background model without the mug. From the log-likelihood curves shown at the top of Fig. 3, similar advantage of the proposed method as in the traffic video can be seen, both at the beginning during self-initialization as well as during the recovery near frame 1,000 when the coffee mug was removed.

5 Conclusions

Online learning of adaptive Gaussian mixtures on nonstationary distributions is an important technique for many video applications. In this paper, we presented an effective learning algorithm that improved convergence rate and estimation accuracy over the standard method used today, offering an explicit formulation for incorporating incremental EM type of learning into a recursive filter without compromising system stability. The results were verified by a large number of simulations over a range of parameter settings and distributions. When applied to background subtraction in a statistical framework, the proposed solution significantly enhanced segmentation results over a commonly used method employed in mixture-based surveillance systems.

References

[1] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data Via the EM Algorithm," *J. Royal Statistical Soc. B*, vol. 39, pp. 1-38, 1977.

[2] N. Friedman and S. Russell, "Image Segmentation in Video Sequences: A Probabilistic Approach," *Proc. 13th Conf. Uncertainty in Artificial Intelligence*, Aug. 1997.

[3] M. Harville, G. Gordon, and J. Woodfill, "Foreground Segmentation Using Adaptive Mixture Models in Color and Depth," *Proc. ICCV Workshop Detection and Recognition of Events in Video*, July 2001.

[4] P. KaewTraKulPong and R. Bowden, "An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection," *Proc. European Workshop Advanced Video Based Surveillance Systems*, Sept. 2001.

[5] S.J. McKenna, Y. Raja, and S. Gong, "Object Tracking Using Adaptive Color Mixture Models," *Proc. Asian Conf. Computer Vision*, vol. 1, pp. 615-622, Jan. 1998.

[6] R.M. Neal and G.E. Hinton, "A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants," *Learning in Graphical Models*, 1998.

[7] S.J. Nowlan, "Soft Competitive Adaptation: Neural Network Learning Algorithms Based on Fitting Statistical Mixtures," PhD thesis, Carnegie Mellon Univ., 1991.

[8] M.A. Sato and S. Ishii, "Online EM Algorithm for the Normalized Gaussian Network," *Neural Computation*, vol. 12, pp. 407-432, 1999.

[9] C. Stauffer and W.E.L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 246-252, June 1999.

[10] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and Practice of Background Maintenance," *Proc. Int'l Conf. Computer Vision*, pp. 255-261, Sept. 1999.