Online Adaptive Gaussian Mixture Learning for Video Applications

Dar-Shyang Lee

Ricoh California Research Center
2882 Sand Hill Road, Menlo Park, CA 94025, USA
dsl@rii.ricoh.com

Abstract. This paper presents an online EM learning algorithm for training adaptive Gaussian mixtures for non-stationary video data. Existing solutions are either slow in learning or computationally and storage inefficient. Our solution is derived based on sufficient statistics of the short-term distribution. To avoid unnecessary computation or storage, we show that the equivalent estimates can be accomplished by a set of recursive parameter update equations with one additional variable. The solution is evaluated against several existing algorithms on both synthetic data and surveillance videos. The results showed remarkable learning efficiency and robustness over current solutions.

1 Introduction

Adaptive Gaussian mixtures are becoming popular for modeling the temporal distribution of video data because of their compact and analytical representation. The nature of many vision applications requires the models to be adaptive over time and learning to be performed online. This prevents direct application of the traditional batch learning by Expectation Maximization [1] or incremental EM [7] developed for stationary distributions. The most common solution today for online adaptive mixture learning for non-stationary distributions employs exponential decay through recursive filtering to achieve temporal adaptability [8]. However, this is notoriously slow in convergence. Other solutions either require additional computation and storage overhead for buffering statistics in the adaptive window [6] or provide approximations without good theoretical foundation [4],[5].

In this paper, we present an online EM learning algorithm for non-stationary distributions formulated on short-term sufficient statistics inside the temporal adaptive window. Under explicit assumptions, we show that a set of equivalent recursive parameter update equations can be derived that has practically no storage or computational overhead. We compared the derived optimal learning algorithm with previously proposed methods on both synthetic and real video data. Experimental results showed excellent robustness and efficiency compared to other methods.

The rest of the paper is organized as follows. In Section 2, we formulate the online adaptive mixture learning problem and present the derivation of the learning

equations. A comparative analysis of our derived solution against other related work is provided in Section 3. We evaluate several algorithms using large simulation as well as real video data. Experimental results are presented in Section 4, followed by conclusions.

2 Online EM Learning for Adaptive Gaussian Mixtures

In vision applications, adaptive Gaussian mixtures offer a compact and analytical representation of input data distribution. In order to work over a long duration, it must be able to adapt to condition changes.

The problem can be stated as follows. We would like to model the short-term distribution of the intensity values observed at a single pixel location in the video over time using a Gaussian mixture. The model must be adaptive to reflect the non-stationary nature of the process, and learning must be done online without a presumptuous storage requirement.

2.1 Case 1 : O(*LK*) Solution

Let $\mathbf{x}_1..\mathbf{x}_n$ be a sequence of data points whose temporal distribution we are trying to approximate by a mixture density function of K Gaussian components

$$P(\mathbf{x}) = \sum_{j=1}^{K} G(\mathbf{x};\theta_j) = \sum_{j=1}^{K} w(j)\cdot g(\mathbf{x};\mu(j),\Sigma(j)) \text{ and } \sum_{j=1}^{K} w(j) = 1 \tag{1}$$

where $g(x;\mu,\Sigma)$ is a single normal distribution. Since the assignment of the Gaussian components is not directly observed, maximum likelihood parameter estimates are obtained through an iterative Expectation-Maximization procedure [1]. For stationary distributions, stochastic versions such as the incremental EM algorithm [4] have been shown to achieve the same result by computing a set of sufficient statistics.

Let $q_n(j)$ be the expected posterior probability of component G_j for data point $\mathbf{x}_n$

$$q_n(j) \equiv P(G_j \mid \mathbf{x}_n) = \frac{w_n(j)\cdot g(\mathbf{x}_n;\mu_{n-1}(j),\Sigma_{n-1}(j))}{P(\mathbf{x}_n)}, \tag{2}$$

and let $Q_n(j)$, $M_n(j)$, $V_n(j)$ be sufficient statistics defined as

$$\begin{aligned} Q_n(j) &\equiv \textstyle\sum_{i=1}^{n} q_i(j) \\ M_n(j) &\equiv \textstyle\sum_{i=1}^{n} q_i(j)\cdot \mathbf{x}_i \\ V_n(j) &\equiv \textstyle\sum_{i=1}^{n} q_i(j)\cdot(\mathbf{x}_i-\mu_i(j))(\mathbf{x}_i-\mu_i(j))^T. \end{aligned} \tag{3}$$

Then mixture parameters are computed by

$$
\begin{aligned}
w_n(j) &= Q_n(j) / \textstyle\sum_{k=1}^{K} Q_n(k) \\
\mu_n(j) &= M_n(j) / Q_n(j) \\
\Sigma_n(j) &= V_n(j) / Q_n(j).
\end{aligned} \tag{4}
$$

In order for the model to adapt to distribution changes, we must discount older statistics as more samples are observed. To model the short-term distribution within a window of L recent samples $\mathbf{x}_{n-L+1}..\mathbf{x}_n$, the sufficient statistics for data in that window can be computed by

$$
\begin{aligned}
Q_n^L(j) &\equiv Q_n(j) - Q_{n-L}(j) \\
M_n^L(j) &\equiv M_n(j) - M_{n-L}(j) \\
V_n^L(j) &\equiv V_n(j) - V_{n-L}(j).
\end{aligned} \tag{5}
$$

Based on these quantities, similar equations as Eq.(4) can be formulated to compute the parameters for an adaptive mixture model

$$
\begin{aligned}
w_n^L(j) &= Q_n^L(j) / \textstyle\sum_{k=1}^{K} Q_n^L(k) \\
\mu_n^L(j) &= M_n^L(j) / Q_n^L(j) \\
\Sigma_n^L(j) &= V_n^L(j) / Q_n^L(j).
\end{aligned} \tag{6}
$$

However, computing Eq.(5) would require storing sufficient statistics for all points within the most recent L-window, or O(LK) amount of storage, which is prohibitive in video applications. We would like to develop an approximation that is more efficient.

2.2 Case 2 : O(K) Approximation

Since we are computing the statistics within a sliding window, we can rewrite Eq.(5) recursively to get

$$Q_n^L(j) \equiv Q_{n-1}^L(j) - q_{n-L}(j) + q_n(j) \tag{7}$$

$$M_n^L(j) \equiv M_{n-1}^L(j) - q_{n-L}(j) \cdot \mathbf{x}_{n-L} + q_n(j) \cdot \mathbf{x}_n \tag{8}$$

$$
\begin{aligned}
V_n^L(j) \equiv V_{n-1}^L(j) &- q_{n-L}(j) \cdot (\mathbf{x}_{n-L} - \mu_{n-L}^L(j))(\mathbf{x}_{n-L} - \mu_{n-L}^L(j))^T \\
&+ q_n(j) \cdot (\mathbf{x}_n - \mu_n^L(j))(\mathbf{x}_n - \mu_n^L(j))^T.
\end{aligned} \tag{9}
$$

Considering that the distribution is quasi-stationary, it is reasonable to assume $q_{n-L}(j) \approx Q_{n-1}^L(j)/L$, then Eq.(7) simplifies to

$$Q_n^L(j) \equiv Q_{n-1}^L(j) - q_{n-L}(j) + q_n(j) \approx \tfrac{L-1}{L} Q_{n-1}^L(j) + q_n(j). \tag{10}$$

We can make similar assumptions to simplify M_n^L and V_n^L. Assuming $q_{n-L}(j)\mathbf{x}_{n-L} \approx M_{n-1}^L(j)/L$, then Eq.(8) simplifies to

$$M_n^L(j) \approx \tfrac{L-1}{L} M_{n-1}^L(j) + q_n(j)\cdot \mathbf{x}_n \,. \tag{11}$$

We further assume $q_{n-L}(j)\cdot(\mathbf{x}_{n-L}-\mu_{n-L}^L)^T(\mathbf{x}_{n-L}-\mu_{n-L}^L) \approx V_{n-1}^L(j)/L$, so

$$V_n^L(j) \approx \tfrac{L-1}{L} V_{n-1}^L(j) + q_n(j)\cdot(\mathbf{x}_n-\mu_n^L(j))(\mathbf{x}_n-\mu_n^L(j))^T \,. \tag{12}$$

From Eq.(10),(11),(12) the adaptive mixture parameters can be computed using Eq.(6). The amount of extra storage required in addition to mixture parameters is reduced to O(*K*). However, for video applications where every pixel is represented by a mixture model, maintaining and updating both sufficient statistics and model parameters, especially the vector manipulations involving M_n^L and V_n^L, still represent a significant overhead in storage and computation. We can eliminate the need for storing M_n^L and V_n^L from the following derivation.

Using Eq.(10), the weight calculation in Eq.(6) can be written as

$$w_n^L(j) = [\tfrac{L-1}{L} Q_{n-1}^L(j) + q_n(j)]/L = \alpha\cdot w_{n-1}^L(j) + (1-\alpha)\cdot q_n(j) \tag{13}$$

where α=(*L*-1)/*L* controls the rate of adaptation. Substituting Eq.(10) and (11) into Eq.(6), the recursive equation for computing the mean can be derived

$$\begin{aligned}\mu_n^L(j) &= \frac{\frac{L-1}{L} M_{n-1}^L(j) + q_n(j)\cdot\mathbf{x}_n}{\frac{L-1}{L} Q_{n-1}^L(j) + q_n(j)} = \frac{\alpha\cdot\mu_{n-1}^L(j) + \frac{q_n(j)}{Q_{n-1}^L(j)}\cdot\mathbf{x}_n}{\alpha + \frac{q_n(j)}{Q_{n-1}^L(j)}} \\ &= (1-\eta_n^L(j))\cdot\mu_{n-1}^L(j) + \eta_n^L(j)\cdot\mathbf{x}_n\end{aligned} \tag{14}$$

where the learning rate η can be simplified using Eq.(10)

$$\eta_n^L(j) = \frac{\frac{q_n(j)}{Q_{n-1}^L(j)}}{\alpha + \frac{q_n(j)}{Q_{n-1}^L(j)}} = \frac{q_n(j)}{\alpha\cdot Q_{n-1}^L(j) + q_n(j)} = \frac{q_n(j)}{Q_n^L(j)} \,. \tag{15}$$

Similarly, substituting Eq.(10) and (12) into the variance update equations results in

$$\begin{aligned}\Sigma_n^L(j) &= \frac{\frac{L-1}{L} V_{n-1}^L(j) + q_n(j)(\mathbf{x}_n-\mu_n^L(j))(\mathbf{x}_n-\mu_n^L(j))^T}{\frac{L-1}{L} Q_{n-1}^L(j) + q_n(j)} \\ &= (1-\eta_n^L(j))\cdot\Sigma_{n-1}^L(j) + \eta_n^L(j)\cdot(\mathbf{x}_n-\mu_n^L(j))(\mathbf{x}_n-\mu_n^L(j))^T .\end{aligned} \tag{16}$$

The solution in Eq.(13)-(16) shows that we do not need to compute M_n^L and V_n^L, but only keep *K* extra scalar variables $Q_n^L(j)$ updated by Eq.(10), to compute the parameter estimates.

2.3 Case 3 : O(1) Approximation

Since the weight is computed from $Q_n^L(j)$, we can in fact further eliminate the need to store $Q_n^L(j)$. However, we must consider the initialization phase when less than L samples have been processed. If the relation in Eq.(6) holds for all cases, we can replace $Q_n^L(j)$ in the learning rate Eq.(15) by $w_n^L(j)$.

Let $c(n)$ be the size of applicable window at time n, and let $\overline{Q}_n^{c(n)} = \sum_{k=1}^{K} Q_n^{c(n)}(k)$ be the cumulative posterior probability for up to L samples summed over all components. Since $\sum_{k=1}^{K} q_i(k) = 1$ for all i, then

$$\overline{Q}_n^{c(n)} = \sum_{k=1}^{K} Q_n^{c(n)}(k) = \sum_{k=1}^{K}\left[\sum_{i=n-c(n)+1}^{n} q_i(k)\right] = \sum_{i=n-c(n)+1}^{n} 1 = c(n). \tag{17}$$

Therefore, the weight computed by Eq.(6) can be expressed as for all n

$$w_n^{c(n)}(j) = \frac{Q_n^{c(n)}(j)}{\sum_{k=1}^{K} Q_n^{c(n)}(k)} = \frac{Q_n^{c(n)}(j)}{c(n)}. \tag{18}$$

We also need to verify the recursive equation in Eq.(13) for all n. Since $c(n)=L$ for $n > L$, it is trivial to replace L by $c(n)$ to verify Eq.(10) and (13). For $n \le L$, $c(n)=n$, so $c(n)=c(n\text{-}1)+1$. Since there is no old data to be discounted, $q_{n-L}(j) = 0$, and $Q_n^{c(n)}(j)$ is computed exactly by

$$Q_n^{c(n)}(j) = Q_{n-1}^{c(n-1)}(j) + q_n(j). \tag{19}$$

Consequently, we can derive the recursive weight update equation as follows.

$$\begin{aligned} w_n^{c(n)}(j) &= \frac{Q_n^{c(n)}(j)}{c(n)} = \frac{Q_{n-1}^{c(n-1)}(j) + q_n(j)}{c(n)} = \frac{c(n-1)}{c(n)} \frac{Q_{n-1}^{c(n-1)}(j)}{c(n-1)} + \frac{q_n(j)}{c(n)} \\ &= \frac{c(n)-1}{c(n)} w_{n-1}^{c(n-1)}(j) + \frac{q_n(j)}{c(n)} = \alpha_n \cdot w_{n-1}^{c(n-1)}(j) + (1-\alpha_n)\cdot q_n(j) \end{aligned} \tag{20}$$

where $\alpha_n = (c(n)-1)/c(n)$. Based on this proof that Eq.(18) and (20) holds for all n, we can replace $Q_n^L(j)$ in the learning rate equation Eq.(15) by $c(n)\cdot w_n^{c(n)}(j)$

$$\eta_n^{c(n)}(j) = \frac{q_n(j)}{c(n)\cdot w_n^{c(n)}(j)}. \tag{21}$$

By definition, $1 \le c(n) \le L$. Since $w_n^{c(n)}(j)$ is updated based on $q_n(j)$, $w_n^{c(n)}(j) > 0$ if $q_n(j)>0$. If $q_n(j)=0$, then $\eta_n^{c(n)}(j)$ is set to 0. It is straight-forward to replace L with $c(n)$ to derive the same updating equations for mean and variance as Eq.(14) and (16).

$$\mu_n^{c(n)}(j) = (1-\eta_n^{c(n)}(j)) \cdot \mu_{n-1}^{c(n-1)}(j) + \eta_n^{c(t-1)}(j) \cdot \mathbf{x}_n \quad (22)$$

$$\Sigma_n^{c(n)}(j) = (1-\eta_n^{c(n)}(j)) \cdot \Sigma_{n-1}^{c(n-1)}(j) + \eta_n^{c(n)}(j) \cdot (\mathbf{x}_n - \mu_n^{c(n)}(j))(\mathbf{x}_n - \mu_n^{c(n)}(j))^T \quad (23)$$

This completes the derivation for the recursive mixture update equations based on short-term sufficient statistics with the addition of a single scalar variable $c(n)$. At each iteration, the weights are first updated based on $c(n)$ using Eq.(20), then the means and variances are updated using Eq.(22) and (23) based on learning rate calculated by Eq.(21).

2.4 Remaining Issues

In the discussion above, we have not addressed the situation when $P(\mathbf{x})=0$. Since Gaussians have infinite support, this would not occur in theory and allows us to perform the above analysis. However, in practical implementations, finite supports are imposed to determine if a match is good enough and whether an assignment should be carried out. Consequently, a data point may fall outside the boundary of all Gaussians and results in $P(\mathbf{x})=0$. In this case, one of the Gaussians G_j is set to center at the data point with a large initial variance. It can be shown that the appropriate initial weight should be $1-\alpha_n$. All other weights are updated the same way followed by normalization.

The assignment of a Gaussian does not introduce a problem when the weight of the assigned Gaussian is zero, as in the case of initialization. However, since we use a finite number of Gaussians, an existing component is reassigned when no more unused Gaussians are available. The original weight associated with that Gaussian is then implicitly distributed to all Gaussians through the normalization process. This problem is inherent to any methods using a finite number of Gaussians, and is not particular to our solution. Fortunately, since reassignment occurs only when no existing Gaussians match the data point, and does not involve mean or variance updates other than the assigned Gaussian, Eq.(21) is not used.

Another issue to be considered is whether to use the newly updated mean or the old mean when updating the variance. Since the variance is computed based on the estimated mean, the degrees of freedom are reduced by one. An unbiased estimate would subtract the old mean estimate rather than the updated one. Both formulations have been used in the literature. The choice of this estimate does not affect our derivation. In practice, there is little difference between the two as long as the variance is not initialized to 0 when the biased estimation is used.

Finally, we should mention another implicit assumption made through out our analysis. That is, the calculation of the statistics for x_t, in the expectation step, such as $q_t(k)$, should depend on the most recent parameter estimates $q_t^{t+n}(k)$ where $t+n$ is the current time. Therefore, it should be re-evaluated before including in the window. However, this is inherent in the incremental approach and it is commonly assumed that $q_t^{t+n}(k) = q_t^t(k)$.

3 Comparative Analysis

In this section, we first summarize existing solutions for online adaptive mixture learning, followed by a more detailed analysis of other algorithms based on the results derived in the previous section.

3.1 Related Work

Current applications of Gaussian mixtures for temporal distribution modeling in vision systems are exemplified by two approaches. In early application of Gaussian mixtures for video processing, mixture parameters were incrementally computed from sufficient statistics [2]. The validity of such incremental algorithms has been shown to be a stochastic approximation to the batch EM algorithm and will converge to a maximum likelihood solution [7]. However, such a formulation based on sufficient statistics is cumulative over all observed samples and cannot adapt to distribution changes. An online learning algorithm for adaptive mixtures was proposed by [8] where temporal adaptation is achieved by recursive learning. The method was shown effective and is commonly used in many vision systems. Unfortunately, learning by recursive filter converges very slowly, requiring data distribution to remain stable over a longer duration for accurate modeling. The problem was observed by [4] who proposed computing parameters based on sufficient statistics in the early stage to achieve fast convergence, then switching to recursive filter type of learning afterwards. Although their method does improve initial convergence quite significantly, we have found the formulation of the recursive learning stage to be flawed and can lead to divergence. Another approach was our own earlier work [5] where we used a modified learning rate schedule to achieve fast convergence during initial learning and gradually became recursive filter learning after many samples were observed. Although the method showed good performance experimentally, the formulation is ad hoc and no theoretical support was provided. Another formulation of adaptive mixture based on a short-term distribution was proposed by [6]. Although a simplification assumption similar to ours was made on the expected posterior, parameters calculations were formulated based on exact solutions that required O(LK) amount of storage, similar to the solution in Case 1, which would be prohibitive for many video applications. The recursive updates suggest another similar derivation by applying an exponentially decaying envelop on recent samples, as suggested by [3]. In that formulation, parameters are calculated from recursively updated sufficient statistics (moments), requiring O(K) overhead as in Case 2 of our solution. Furthermore, there was no description of the initial stage (or after reset) where convergence has the biggest effect. It can be shown that a 1/t-type learning in those stages would require a uniformly weighted envelop instead of exponential.

3.2 Qualitative Analysis

All online mixture learning algorithms use the same basic adaptive filtering equation where Gaussian parameters $\theta_n(j)$ are recursively updated based on new samples

$$\theta_n(j) = (1-\eta_n(j)) \cdot \theta_{n-1}(j) + \eta_n(j) \cdot \nabla(\mathbf{x}_n; \theta_{n-1}(j)) . \quad (24)$$

Algorithms differ in the choice of $\eta_n(j)$. In the approach of [2] where parameters are calculated from incrementally updated sufficient statistics, $\eta_n(j)=1/n$. This stochastic approximation of online EM with $1/t$-type of learning rate will converge to a local maximum likelihood solution [7]. However, the estimates are cumulative and cannot adapt to distribution changes. In the work of [8], temporal adaptation is achieved using the proposed $\eta_n(j)=\alpha \cdot g(\mathbf{x};\theta_j)$. Since the conditional probability $g(\mathbf{x};\theta_j)$ is usually very small, several researchers have found it is more efficient to simply use $\eta_n(j)=\alpha$. This type of exponential decay with fixed rate is commonly employed in many mixture-based surveillance systems for temporal adaptation. The disadvantage of this approach is slow convergence. Since α is typically set to a very small value to provide stability, any change in distribution requires a long period of learning before it is reflected in the model.

These two basic approaches can be combined to provide fast, robust adaptive mixture learning by incorporating exponential decay into the incremental EM estimates or employing a $1/t$-type of learning rate the recursive filter learning. In [4], a two-stage learning process is used. In the first stage when less than L samples are observed, the parameters are computed by incremental EM. After more than L-samples are observed, a different set of learning equations based expected sufficient statistics for the L-recent window is applied. The approach is similar to ours. However, we have shown that a distinct separation of learning stages is unnecessary and derived the optimal learning rate. More importantly, we found their formulation for the second stage of learning is flawed. Their update equations for the mean and variance can be written as

$$\theta_n(j) = \alpha \cdot \theta_{n-1}(j) + (1-\alpha) \cdot \frac{q_n(j)}{w_{n+1}(j)} \cdot \nabla(\mathbf{x}_n; \theta_{n-1}(j)) . \quad (25)$$

Since the mixing coefficients do not sum to one, we find it sometime displays divergent behavior after L samples. Another similar method was proposed by [5], who used a modified $1/t$ learning rate schedule that converges towards $(1-\alpha)$ instead of 0

$$\eta_n(j) = q_n(j)\left(\frac{\alpha}{Q_n^L(j)} + (1-\alpha)\right) . \quad (26)$$

Although it showed good performance in experiments, no theoretical justification was provided. Moreover, it requires storing an extra scalar for each Gaussian $O(K)$, similar to the solution we found in Case 2.

To illustrate the differences among these approaches, consider the following example. If a Gaussian is initialized to $\mathbf{x}$ with a variance of 1, and the same data point is observed over and over again, how fast would the variance converge to 0? Fig.1 shows the value of η_n plotted over time. The $1/t$ curve for [2] and the line $(1-\alpha)$

exemplify the two basic approaches; whereas the other three curves converge towards (1-α) at roughly 1/t rate. The middle line is the solution derived in this paper. The method of [4] follows 1/t for the first L samples, then follows α on the first term, and the derived line for the second term. The curve for [5] is similar to our derived solution but converges at a slower rate. For α=0.999, it takes 100 iterations for [2][4], 91 iterations for [5] and 95 iterations for Eq.(21) for σ to reach 0.01, compared to 4603 iterations for the commonly used method in [8].

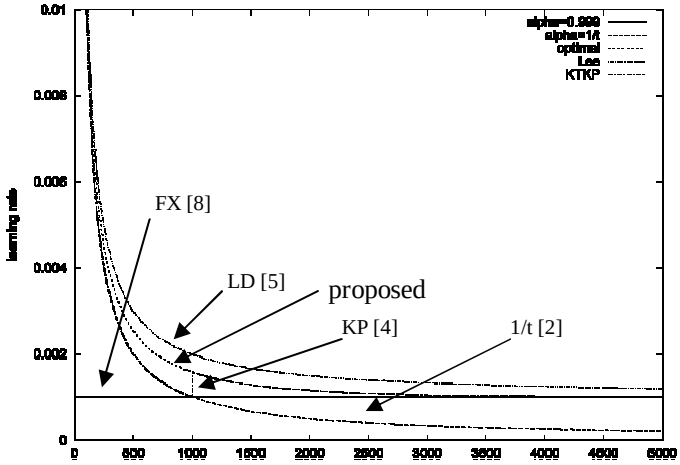

Fig. 1. A comparison of effective learning rates of several algorithms for α=0.999.

4 Experimental Results

For a quantitative comparison, we evaluated various algorithms on a set of synthetically generated data with known ground truth. Each data set simulates pixel intensity values in video over time and is generated by randomly drawing a value between 0 and 255 from a time-varying mixture distribution with known parameters. These synthetic data models are created based on observations of real video data. Fig.2 shows some examples where the intensity value (y-axis) is plotted over time (x-axis). There are three types of data models. Type 1 consists of a tri-modal mixture density with one dominant cluster and two smaller clusters. This distribution is representative of regions in a surveillance video where foreground activities are observed. Type 2 data is composed of a single migrating distribution, aimed at testing the temporal adaptability of the models. Type 3 data consists of a bimodal distribution observed in real videos due to monitor flickering or swaying trees.

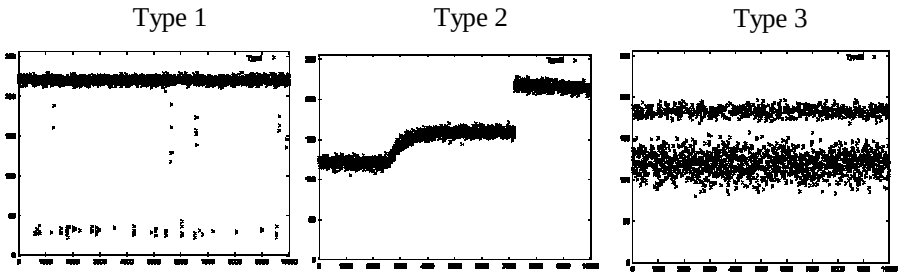

Fig. 2. Examples of synthetic data sets modeling pixel intensity values in video.

Four adaptive algorithms were tested: the fixed rate method in [8] as a baseline (FX), two previously proposed methods for improving convergence [4] (KP) and [5] (LD), and the solution derived in this paper (Proposed). At every iteration, we compute the KL-divergence between the estimated model and the ground truth. The algorithms are trained for $5L$ iterations where L is the adaptive window size. To get a sense of how fast the algorithm converges and adapts, we compute the average divergence over the entire learning window. The results are summarized in Table 1. Each entry is averaged over 10 data sets.

Table 1. Performance comparison for different adaptive learning algorithms. Each entry in the table is the KL-divergence between the estimated model to the distribution ground truth, averaged over 10 data sets on $5L$ iterations, where $L=1/(1-\alpha)$ is the adaptive window size.

KL-Divergence	FX	KP	LD	Proposed
(a) Type1,α=0.999,K=3	0.006866	0.007563	0.000148	0.000148
(b) Type1,α=0.999,K=5	0.006866	0.008976	0.000123	0.000124
(c) Type1,α=0.99,K=3	0.006453	0.003868	0.000644	0.000725
(d) Type1,α=0.99,K=5	0.006453	0.004885	0.000644	0.000726
(e) Type2,α=0.99,K=3	0.010331	0.000520	0.000471	0.000520
(f) Type2,α=0.999,K=3	0.018361	0.015017	0.003878	0.004363
(g) Type3,α=0.99,K=3	0.008844	0.010889	0.001666	0.001668
(h) Type3,α=0.999,K=3	0.008494	0.010102	0.001413	0.001426

The experimental results are consistent with our analysis in the previous section. The commonly used fixed rate method (FX) is orders of magnitude slower in convergence than any other adaptive methods, resulting in proportionally larger average divergence. The KP method performed well in the initial learning stage. However, learning performance in the second stage is sensitive to data characteristics and parameter settings. For example, its performance in trial (e) is comparable to the LD and proposed method. However, a similar run with different parameters in (f) is much worse relative to others because of several diverging cases. The unstable nature of the algorithm can be seen by the inconsistent results shown across different tests, as predicted in our analysis. The LD, which was our own method developed

earlier, and the proposed methods performed equally well across all test sets. The slight advantage for the LD method can be attributed to a slower decreasing learning rate (as shown in Fig.1). However, it requires additional storage of O(K) variables. In addition, without any theoretical backing, it is difficult to quantify its expected behavior.

We also tested the algorithms on real surveillance videos, as shown in Fig.3. An adaptive mixture is used to model the color distribution at a pixel in the video over time. We used a separate mixture for each pixel with 3 Gaussian components in the YUV space and a diagonal covariance matrix, and α=0.995. We compute the likelihood of each new frame under the previous estimates and plotted over time, shown on the right hand side. The observation again is consistent with our prediction. Model trained under the FX method improved very slowly compared to others. The KD method performed quite well except for frames between 300 and 800 when moving traffic come into view. Similar to the previous tests on Type 1 data, the algorithm diverged at several locations in the video and lowered the overall frame likelihood. The LD and proposed method were again on par with each other, with the LD method converging very slightly ahead at the beginning because of a slower decreasing learning rate. The exact same observations can be made on the second sequence as well.

5 Conclusion

We presented an online EM learning algorithm for training adaptive Gaussian mixtures. We derived a set of recursive parameter update equations based on short term sufficient statistics that can be computed without additional storage of auxiliary variables. To our knowledge, this is the first such derived formulation of its kind. The proposed solution was evaluated against existing algorithms and showed superior efficiency and robustness on large simulations as well as real video data.

References

1. Dempster, A.P., Laird, N.M. and Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. of the Royal Statistical Society B, vol. 39 (1977) 1-38.
2. Friedman, N. and Russell, S.: Image segmentation in video sequences: a probabilistic approach. In: Proc. 13th Conf. Uncertainty in Artificial Intelligence, (1997).
3. Jepson, A.D., Fleet, D.J., El-Maraghi, T.: Robust online appearance models for visual tracking**.** IEEE Trans. on PAMI Vol. 25, No.10, (2003) 1296-1311.
4. KaewTraKulPong, P. and Bowden, R.: An improved adaptive background mixture model for real-time tracking with shadow detection. In: Proc. of 2nd Euro. workshop on Advanced Video Based Surveillance Systems (2001).
5. Lee, D.S.: Improved adaptive mixture learning for robust video background modeling. In: Proc. of IAPR Workshop on Machine Vision for Applications (2002) 443-446.

6. McKenna, S.J., Raja Y. and Gong, S.: Object tracking using adaptive colour mixture models. In: Proc. of ACCV, vol.1, (1998) 615-622.
7. Neal, R.M. and Hinton, G.E.: A view of the EM algorithm that justifies incremental, sparse, and other variants. In: Learning in Graphical Models (1998).
8. Stauffer, C. and Grimson, W.E.L.: Adaptive background mixture models for real-time tracking, In: Proc. of CVPR, vol.2, (1999) 246-252.

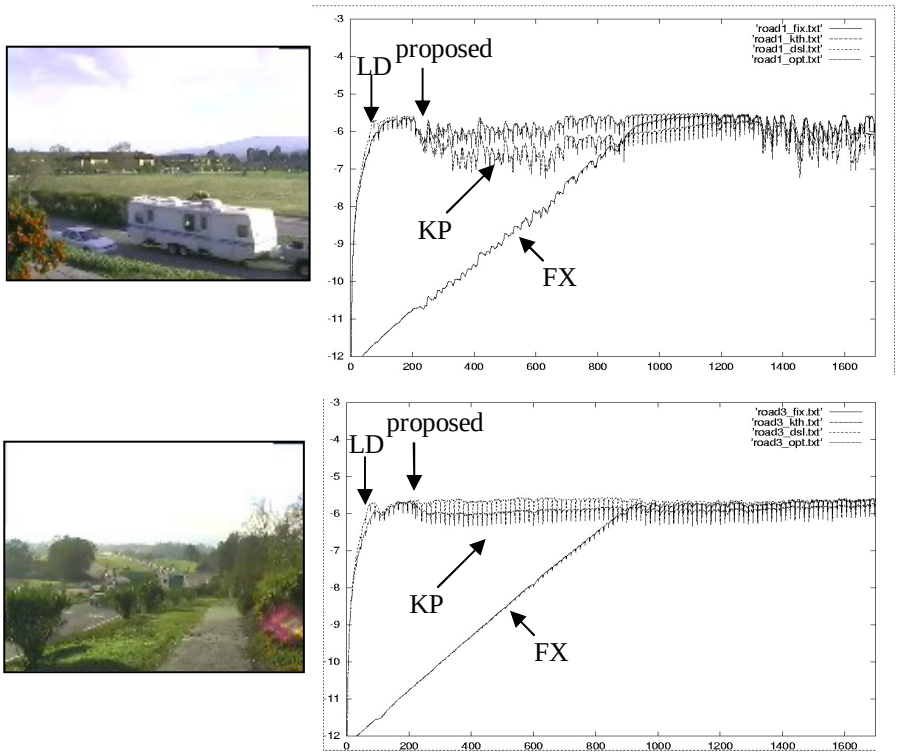

Fig. 3. Testing on traffic monitor video. The log likelihood curve of video predicted by various algorithms are compared.