

# An ICA Mixture Hidden Markov Model for Video Content Analysis

Jian Zhou, *Member, IEEE*, and Xiao-Ping Zhang, *Senior Member, IEEE*

## Abstract

In this paper, a new theoretical framework based on hidden Markov model (HMM) and independent component analysis (ICA) mixture model is presented for content analysis of video, namely ICAMHMM. Unlike the Gaussian mixture observation model commonly used in conventional HMM applications, the observations in the new ICAMHMM are modeled as a mixture of non-Gaussian components, which are better representation of video components. Each non-Gaussian component is formulated by an ICA mixture, reflecting the independence of different components across video frames. In addition, to construct a compact feature space to represent a video frame, ICA is applied on video frames and the ICA coefficients are used to form a compact feature subspace. Each frame is then represented by a two dimensional vector that makes the subsequent modeling algorithms computationally efficient. The model parameters can be identified using supervised learning by the training sequences. The new re-estimation learning formulae of iterative ICAMHMM parameter estimation are derived based on a maximum likelihood function. Employing the identified model, maximum likelihood algorithms are developed to detect and recognize video events. As a case study, golf video sequences are used to test the effectiveness of the proposed algorithm. Experimental results show that the presented method can effectively detect and recognize the recurrent event patterns in video data. The presented new ICAMHMM is generic and can also be applied to sequential data analysis in other video content analysis applications.

## Keywords

hidden Markov model, independent component analysis (ICA) mixture model, video content analysis, expectation-maximization (EM) algorithm, sequential data analysis

A shorter version appears in *IEEE Trans. on Circuit and Systems for Video Technology, Special Issue on Event Analysis in Videos*, vol. 18, no. 11, pp. 1576-1586, November 2008.

The authors are with the Department of Electrical & Computer Engineering, Ryerson University, 350 Victoria Street, Toronto, Ontario, CANADA, M5B 2K3. E-mail : xzhang@ee.ryerson.ca

This work is supported in part by Natural Sciences and Engineering Research Council of Canada, Grant No. RGPIN239031.

## I. INTRODUCTION

Content analysis of video is to find meaningful structures and patterns from visual data and represent them in a compact form such that efficient indexing, classification, and retrieval of video content are possible. Various techniques have been developed to analyze the content of video data. Early works mainly focus on video transition detections [1], [2], [3]. Transitions or shot boundaries are detected such that post production of video can be recovered. After video shots are identified, one or several video frames are selected as key-frames to represent the video shots for indexing and retrieval purpose. Besides video temporal segmentation, localized features such as object-based representation of video have also been investigated by researchers. In [4] and [5], the trajectories of objects based on object segmentation and tracking are used for video indexing. Such object-based representations fully utilize the spatial-temporal information. However, the intra-shot analysis only provides a very small time scale access for video, and these approaches that use localized features are still low level analysis.

In the past few years there has been increased interest in semantic event detection and recognition from video data. A semantic video event can be described by low-level features. To bridge the semantic gap, new tools and models need to be developed. Several directions have been studied recently. One direction is to represent the high-dimensional video data in a compact representation, and thus make it possible to index, analyze and retrieve the elements efficiently. In [6], principal component analysis (PCA) is used to reduce the dimension of features of video frames, and two applications were demonstrated. One is high-level scene analysis, and the other is sports video classification. On the other hand, other researchers are utilizing new models to analyze the semantics from video. In [7], hidden Markov model (HMM) is applied to detect play and break event for soccer video. Dominant color ratio and the magnitude of the motion vectors are used as features. The observations are modeled as a mixture of Gaussians with two mixtures per state. The sequences are segmented by computing the maximum likelihood to classify the video segments. In [8], HMM and audio features are used to classify TV programs into commercial, basketball, football, news and weather video. In [9], MPEG-7 audio features and entropic prior HMM models are used to recognize common audio events such

as applause and cheering.

Most existing HMM-based video analysis systems mainly focus on video classification tasks. In [7], semantic events are identified by using dominant color ration and motion intensity based on HMM in soccer domain. In [7] and [10], hierarchical HMM structures are used to model the events. However, the tree-like hierarchical HMM is very complex by nature, and such structures may not be practical due to the computation burden. Also, the feature space and pre-defined events are domain dependent and may not be generalized to other domains.

In this paper, we explore a general framework to detect and recognize semantic events. The task is to recognize and identify the known semantic events from video data. Note that finding a good feature space and representing the video data in an efficient way is crucial for recognition systems.

We present a compact feature space to represent video data based on independent component analysis (ICA). The new representation makes it easier to analyze the dynamics and characteristics contained in video data. In the feature space, we develop a new statistical model by combining ICA mixture model [11] and HMM modeling, namely ICA mixture HMM or ICAMHMM. Note that ICA techniques are used twice in our model. At the feature extraction step, ICA is used as a preprocessing filter to decompose the video signals. During the modeling step, ICA mixture model are integrated into HMM framework to capture the spatial and temporal characteristics. We use HMM framework to grasp the temporal structures of video data since HMM is well-known for its capability to capture the temporal statistics of stochastic process and it is widely used in pattern recognition field. Since the features are already represented in an efficient way, we only consider simple HMM models with ergodic and left-right structures. Simulations show that these structures are good enough to model the semantic events. In the ICAMHMM modeling, each semantic event is described by one HMM model, and its parameters are learnt through training sequences. The maximum likelihood criteria is used to evaluate how well an unknown video segment matches the model. Sequences with larger likelihood are considered to be more likely to contain the pre-defined semantic events. As a case study, golf video sequences are used to test the effectiveness of the proposed algorithm.

Different from football, soccer, and tennis video, golf video has not been well analyzed in the literature. The content analysis and event detection in golf domain could provide potential applications for home video and entertainment.

The paper is organized as follows. In section II, the proposed framework of ICAMHMM for content analysis of video is presented. The detailed algorithm is presented in section III. Simulations are performed to demonstrate and analyze the effectiveness of the new algorithms in section IV. In section V, golf video is analyzed as a case study and experimental results are presented and discussed. Conclusion remarks are presented in section VI.

## II. A NOVEL FRAMEWORK OF ICA MIXTURE HIDDEN MARKOV MODEL

### A. Problem Formulation

For a video sequence with  $T$  video frames, we assume a feature vector of length  $L$  can be extracted from each frame. Let  $\mathbf{o}_t$  ( $1 \leq t \leq T$ ) be the feature vector for the  $t$ -th video frame. Each feature vector can be considered as one observation in the  $L$ -dimensional feature space. An event is defined as a video segment which has certain semantic meanings. In this paper, we are only interested in supervised learning techniques. Thus, the training sequences that define the events are known in advance. Let  $\mathbf{E}_d$  ( $1 \leq d \leq D$ ) denote a possible event where  $D$  is the total number of all possible events for a given video set. We assume a semantic event  $\mathbf{E}_d$  can be described by an observation sequence, i.e.,

$$\mathbf{E}_d : \{\mathbf{o}_{z_d}, \mathbf{o}_{z_d+1}, \dots, \mathbf{o}_{z_d+Z_d-1}\} \quad (1)$$

where  $\mathbf{o}_{z_d}$  is the first frame for the event  $\mathbf{E}_d$ , and  $Z_d$  is the number of frames of the observation sequence. For a given video sequence, the objective of event detection and recognition is to first identify the event boundaries and then classify each video segments into one of the  $D$  possible known events. In the following sections, mathematical models are developed to represent the observation sequence. In our method, the event is being modeled as by model parameters. Equation (1) shows that our semantic events are defined on frame-level. To simplify the detection for event boundaries, we make the assumption that the beginning and ending frames of an event are located at shot boundaries. The assumption allows the shot-level information to be utilized for event boundary detections.

From the above problem formulation, it can be seen that the event detection and recognition consists of three major parts. The first part is to identify the event boundaries by using shot-level information for the given video sequence. The second part is model identification, i.e., semantic events are represented as model parameters using the training sequences. The third part is to compute the likelihood given each model, and the sequence is classified to the event if its model parameters produce the largest likelihood.

### B. The ICA Mixture Hidden Markov Model (ICAMHMM)

Our motivations come from HMM’s great success in modelling the temporal structure in sequential data. For classical applications such as speech recognition [12], HMMs have been successfully applied in that domain for recognitions. For video content analysis, a similar concept can also be adopted since the video sequence can be treated as sequential data. Many videos such as news videos and sports videos often contain recurrent patterns and temporal structures. For example a news video often has interleaved segments such as scenes with news anchors, commercial breaks, live interviews in split screen, weather forecast and reports from the field. A sports video often contains recurring events depending on how the game being played. In the context of content analysis or event detection, these temporal structures can be learnt by the HMM and the event detection becomes a classification problem when each event is described by one set of HMM parameters.

#### B.1 Modeling the Event using HMM

We denote a given observation sequence as  $\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T$ , where  $T$  is the length of the sequence and  $\mathbf{o}_t, 1 \leq t \leq T$ , is the  $L$ -dimensional feature vector for the  $t$ -th video frame. Let  $q = q_1, q_2, \dots, q_T$  be the hidden state sequence. An HMM model with  $N$  states is determined by the parameters  $\lambda = (A, B, \pi)$ , where  $A = \{a_{ij}\}, 1 \leq i, j \leq N$  is the state transition probability matrix,  $a_{ij} = P(q_{t+1} = j | q_t = i)$  is the probability of state  $j$  at time  $t + 1$  given the state is  $i$  at time  $t$ .  $B = \{b_j(\mathbf{o})\}, 1 \leq j \leq N$ , is the density parameter, where  $b_j(\mathbf{o})$  is the probability density function of the observation at state  $j$ . And  $\pi = \{\pi_i\}, 1 \leq i \leq N$  is the initial state distribution where  $\pi_i = P(q_1 = i)$ .

The HMM is suitable for modeling temporal structures of sequential data. There are different techniques that can be used to model a semantic event under the HMM framework.

For example, an event can be presented by the model structure. For unsupervised learning, the structure can be learnt from the video data. The unsupervised learning methods generally require complex graphic models and are computationally expensive. Also, the prior knowledge generally available for event detection and recognition is not well utilized. Besides the unsupervised structure learning, a hidden state can also be used to describe an event. However, the states are “hidden” in nature, and there is no way to find the “correct” state sequence. The choice of an optimality criterion is a strong function of the intended use for the uncovered state sequence [12]. Therefore, in our method, we choose to use one HMM model to describe one event, and different events are represented by different model parameters. This method can be considered as supervised learning since the model structure is pre-defined. The advantages of this method are that it is relatively robust to the small variations in training data, and it is generally not computationally demanding. We model the video signal using continuous observation models because we believe video signals in feature space generally are continuous in nature. In this paper, we choose the continuous observation densities to avoid the errors introduced by quantization in discrete HMM. The probability density functions are generally represented in parametric form. Thus, the whole parameter set for continuous HMM modeling to describe a set of semantic events  $\mathbf{E}_d$  can be written as

$$\mathbf{E}_d : \lambda_d = (A_d, \Theta\{B_d\}, \pi_d), \quad 1 \leq d \leq D, \quad (2)$$

where  $D$  is the number of events,  $\lambda_d = (A_d, \Theta\{B_d\}, \pi_d)$  are the HMM model parameters for the event  $\mathbf{E}_d$ .  $A_d$  is the transition matrix for the  $d$ -th event.  $\pi_d$  is the initial state distributions for the  $d$ -th event.  $B_d$  is the probability density function, and  $\Theta\{B_d\}$  is the parameters set which uniquely determines continuous observation densities in all states.

The identification of the model is to find the model parameters  $\lambda_d^*$  that gives the maximum likelihood,

$$LH_{\lambda_d}(\mathbf{O}) \equiv P(\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_N | \lambda_d) = P(\mathbf{O} | \lambda_d), \quad 1 \leq d \leq D, \quad (3)$$

$$\lambda_d^* = \arg \max_{\lambda_d} LH_{\lambda_d}(\mathbf{O}), \quad 1 \leq d \leq D. \quad (4)$$

To apply HMM modeling techniques for video event detection, our interests include: 1) identification of model given known observations, and 2) detection of an event given the

model. The former is to adjust the model parameters to maximize the likelihood of the observations. The latter is to evaluate how likely the sequence is produced by the model.

Given the observation sequence  $\mathbf{O}$ , the description of the semantic event is essentially to build the model, and thus to estimate the model parameters. The Baum-Welch method can be used to estimate the model parameters  $\lambda$  such that the joint probability  $P(\mathbf{O} | \lambda)$  is maximized. After convergence, an event described by  $\mathbf{O}$  is represented by HMM model parameters. For the second problem, given the unknown observation sequence  $\mathbf{O}'$ , and a model  $\lambda = (A, B, \pi)$ , the likelihood  $P(\mathbf{O}' | \lambda)$  determines how well the unknown observation sequence matches the given model. A larger likelihood implies the sequence is more likely to have the event described by the model parameters. The likelihood can be computed using *the Forward-Backward Procedure* [12]. For supervised learning, besides the HMM model parameters  $\lambda$ , the number of classes (i.e. the number of events) and the training sequences for each class still need to be determined.

## B.2 Continuous Observation Densities using Gaussian Mixture Model

Continuous observation densities can be used in HMM to avoid the quantization errors introduced by vector quantization in discrete HMM [12]. The continuous observation model has been formulated in [13], and the continuous densities based on Gaussian mixture model have been formulated in [14]. Thus, for Gaussian mixture model, the observation densities are of the form

$$b_j(\mathbf{o}) = \sum_{k=1}^K P(C_{jk}) \cdot p(\mathbf{o} | C_{jk}), \quad 1 \leq j \leq N, \quad (5)$$

where

$$p(\mathbf{o} | C_{jk}) = \frac{1}{(2\pi)^{d/2} [\det\{\boldsymbol{\Sigma}_{jk}\}]^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{o} - \boldsymbol{\mu}_{jk})^T \boldsymbol{\Sigma}_{jk}^{-1}(\mathbf{o} - \boldsymbol{\mu}_{jk})\right). \quad (6)$$

The multivariate variable  $\mathbf{o}$  can be considered as the observation vector being modeled,  $K$  is the number of mixtures,  $C_{jk}$  is the  $k$ -th mixture component in state  $j$ ,  $P(C_{jk})$  is the probability of choosing the  $k$ -th mixture component in state  $j$ ,  $p(\mathbf{o} | C_{jk})$  is a Gaussian density with mean vector  $\boldsymbol{\mu}_{jk}$  and covariance matrix  $\boldsymbol{\Sigma}_{jk}$  for the  $k$ -th mixture component in state  $j$ .

### B.3 Continuous Observation Densities using non-Gaussian Mixture Model

Even though the Gaussian mixture model can approximate arbitrarily closely any continuous density function for a sufficient number of mixtures, the results may highly depend on how the number of mixtures is chosen and the how the parameters are estimated. In this paper, we introduce a new HMM observation model using non-Gaussian mixtures. Each non-Gaussian mixture is associated with a standard ICA. Thus we call this new continuous observation model as non-Gaussian mixture observation model or ICA mixture observation model for HMM. The observations are modeled as a mixture of non-Gaussians since the distribution of video frames in the feature space generally shows non-Gaussian characteristics, and such higher- order statistics can be captured by ICA blindly. The reason of using *ICA mixture model* instead of *ICA* is that the observed video data can be categorized into mutually exclusive classes, similarly like Gaussian mixture models. Such characteristics are often true in video data since the separate stories are often interlaced in video sequences. The ICA mixture model first divides observed data into mutually exclusive classes, and then models each class as a linear combination of independent, non-Gaussian sources. This allows modeling classes with non-Gaussian structures. The observation densities described by ICA mixture can model a broader range of probability density functions, and can be considered as a complement of Gaussian mixture modeling. When using ICA mixture to capture non-Gaussian structures and classify the data, better results were reported in [11], compared with Gaussian mixture. As shown in Fig. 1, the structure of continuous HMM based on non-Gaussian mixture model is similar as the configuration of Gaussian mixture based HMM.

In the new ICA mixture observation model, we are not really interested in the recovered sources or their physical meanings. Our major concern is to estimate the observation densities based on ICA mixture learning algorithms and the source models we selected. The goal is to develop a parametric form of representation of observation densities and then derive HMM learning algorithms for HMM models.

Let  $q_t$  be the hidden state at time  $t$ , the proposed non-Gaussian mixture observation model that brings ICA mixture model into HMM framework to capture the non-Gaussian

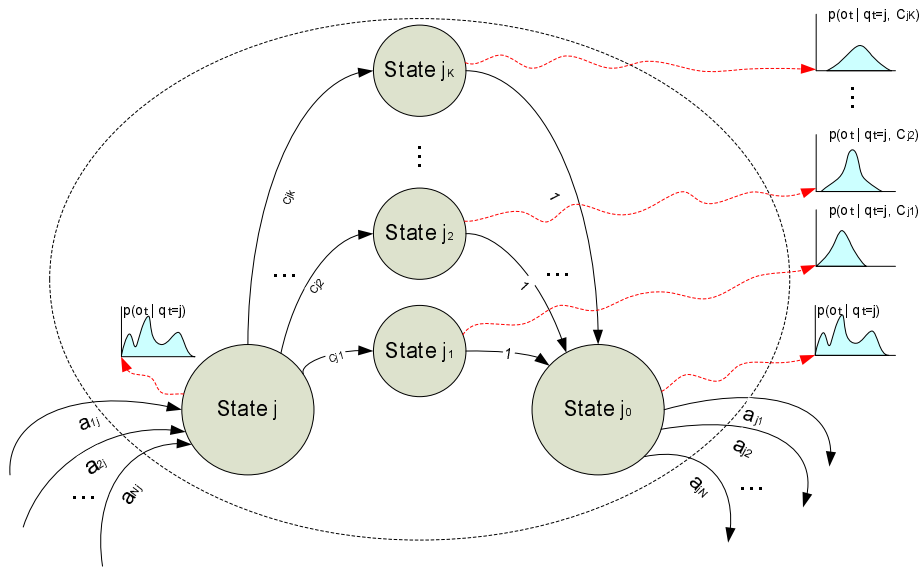


Fig. 1. Equivalent  $K$  state re-configuration for state  $j$  with  $K$  Gaussian or non-Gaussian mixtures.

structures can be represented as follows

$$b_{q_t}(\mathbf{o}) = \sum_{k=1}^K p(\mathbf{o} | C_{q_t k}, \theta_{q_t k}) \cdot P(C_{q_t k}), \quad (7)$$

where  $\mathbf{o}$  is the vector being modeled.  $P(C_{q_t k})$  is the class probability to the  $k$ -th class;  $p(\mathbf{o} | C_{q_t k}, \theta_{q_t k})$  is a non-Gaussian probability density function that describes the statistics of the observation for the  $k$ -th class at time  $t$ , given the state at time  $t$  is  $q_t$ , where  $\theta_{q_t k}$  represents the parameters of the densities in state  $q_t$ . Note that (7) is similar to (5) since both are essentially mixture models. The key difference between (7) and (5) is that the mixture component  $p(\mathbf{o} | C_{q_t k}, \theta_{q_t k})$  in (7) is a non-Gaussian density function. The challenges and difficulties reside in the inferring the non-Gaussian density analytically. However, the non-Gaussianities can be captured by ICA by seeking statistically independent sources. Thus, each non-Gaussian mixture component density in (7) is further modeled as a standard ICA. Therefore, the continuous observation densities using non-Gaussian mixture model is essentially an ICA mixture model.

The non-Gaussian distributions can be further decomposed into functions through standard ICA as follows. In classical ICA without considering the mixture modeling, the observation sequence  $\mathbf{O} = \mathbf{o}_1, \dots, \mathbf{o}_T$  is modeled as an  $L$ -dimensional random variable  $\mathbf{o}_t$

which is further modeled as a linear combination of  $L$  statistically independent sources  $\mathbf{s}_t$  plus the bias  $\boldsymbol{\mu}$ , i.e.:

$$\mathbf{o}_t = \mathbf{M}\mathbf{s}_t + \boldsymbol{\mu}, \quad t = 1, \dots, T. \quad (8)$$

where  $\mathbf{M}$  ( $L \times L$ ) is known as the *mixing matrix*. To avoid the ambiguity of the terms, we refer to the mixing matrix  $\mathbf{M}$  as the basis matrix to distinguish the word ‘‘mixture’’ in the mixture model. The ICA task is to find the filter matrix  $\mathbf{W} \approx \mathbf{M}^{-1}$  using only the observed signals  $\mathbf{O}$ . Since the observed signals are a linear transformation of the sources, their multivariate probability density functions satisfy the following relationship:

$$p(\mathbf{o}) = \frac{p(\mathbf{s})}{|\det\{\mathbf{J}\}|} \quad (9)$$

where  $|\cdot|$  denotes the absolute value and  $\mathbf{J}$  is the Jacobian of the transformation. Therefore, the log likelihood can be written as:

$$\log p(\mathbf{o} | \theta) = \log p(\mathbf{s}) - \log(\det\{\mathbf{M}\}) \quad (10)$$

where  $\theta$  denotes the model parameters  $\{\mathbf{M}, \boldsymbol{\mu}, \mathbf{s}\}$ . Equations (8)-(10) describe the case when all the observations are assumed to be generated from one class (i.e.  $K = 1$ ). The observation model introduced in (7) requires a mixture modeling since the observations are assumed to be generated from multiple classes. The standard ICA can be generalized into ICA mixture model that allows modeling of classes with non-Gaussian structures. The ICA mixture model, originally proposed by Lee and Lewicki [11], assumes that the observed data  $\mathbf{O} = \mathbf{o}_1, \dots, \mathbf{o}_T$  are drawn independently and generated by a mixture density model. The likelihood of the data is given by the joint density:

$$p(\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T | \Theta) = \prod_{t=1}^T p(\mathbf{o}_t | \Theta), \quad (11)$$

and if a mixture density has the following form:

$$p(\mathbf{o}_t | \Theta) = \sum_{k=1}^K p(\mathbf{o}_t | C_k, \theta_k) p(C_k) \quad (12)$$

where  $\mathbf{o}_t$  is a multivariate variable and  $\Theta = (\theta_1, \dots, \theta_k)$  is the unknown parameter set for each  $p(\mathbf{o}_t | C_k, \theta_k)$ .  $C_k$  denotes the class  $k$  and  $K$  is the number of classes. Then the data

in each class can be described as:

$$\mathbf{o}_t = \mathbf{M}_k \mathbf{s}_k + \boldsymbol{\mu}_k, \quad \mathbf{o}_t \in C_k, \quad (13)$$

where  $\mathbf{M}_k$  is a square basis matrix for the  $k$ -th class, and  $\boldsymbol{\mu}_k$  is the bias vector for class  $k$ . The task is first to classify the unlabeled data points into one of the  $K$  classes, and then to determine the parameters for each classes. The parameters include  $(\mathbf{M}_k, \boldsymbol{\mu}_k)$  for the  $k$ -th class, and the class probability  $p(C_k | \mathbf{o}_t, \theta_k)$  for each data point. Within each class, the data points are modeled by standard ICA. Therefore, considering (10), we rewrite the total likelihood of the data based on ICA mixture model for state  $j$  as:

$$\log p_j(\mathbf{o} | \Theta) = \sum_{t=1}^T \log \left( \sum_{k=1}^K p(C_{jk}) \exp(\log p(\mathbf{s}_{jk}) - \log(\det\{\mathbf{M}_{jk}\})) \right). \quad (14)$$

Note that the mixture density is represented in a form of exponential functions. For ICA problems, as analyzed in [15] and [16], a Laplacian density model can be used to approximate super-Gaussian densities, and a bimodal density can be used to approximate sub-Gaussian densities. The combination is the extended informax ICA learning rule for a standard ICA problem. For the ICA mixture model, the estimation of each mixture density is one standard ICA problem.

The ICA mixture model described above can be further integrated into HMM framework to model the observation densities. Classic Gaussian mixture models (GMM) based on maximum likelihood estimation are formulated in [14] to describe continuous observations for Markov chains. We consider our ICA mixture based observation models as a further extension to better describe non-Gaussian observations for Markov chains. When observed data show strong non-Gaussian properties, how well Gaussian mixture models fit the data may depend on the number of classes we choose and how the parameters are estimated. However, using the ICA mixture, i.e. the non-Gaussian mixture observation model, we may be able to get a better fit of the data with a smaller number of classes. A major application motivation for ICAMHMM is that we believe it may have potentials for video analysis. In video data, each frame generally has different properties in different regions and they may all change with time. In the high dimensional feature space, if observed data has strong non-Gaussian characteristics, or if we are interested in understanding its non-Gaussian properties, we can choose ICAMHMM as the model, and may get a better fit of

the data especially when we want to limit the number of classes to reduce the complexity of the system.

### III. ALGORITHMS FOR ICAMHMM

#### A. Model Parameters

The proposed *ICAMHMM* framework has the following model parameters:

$$\lambda = (A, C, \mu, M, \pi), \quad (15)$$

where  $A = \{a_{ij}\}, 1 \leq i, j \leq N$  is the transition matrix.  $C = \{P(C_{jk})\}, 1 \leq j \leq N, 1 \leq k \leq K$  is the mixture matrix;  $\mu = \{\boldsymbol{\mu}_{jk}\}, 1 \leq j \leq N, 1 \leq k \leq K$  is the mean coefficients for mixture densities, where  $\boldsymbol{\mu}_{jk}$  is the  $L$ -dimensional mean vector for the  $k$ -th mixture component at state  $j$ ;  $M = \{\mathbf{M}_{jk}\}, 1 \leq j \leq N, 1 \leq k \leq K$  is the ICA basis coefficients, where  $\mathbf{M}_{jk}$  is the  $L \times L$  basis matrix for the  $k$ -th ICA source at state  $j$ .  $\pi = \{\pi_i\}, 1 \leq i \leq N$ , is the initial state distribution. Note that  $C$ ,  $\mu$ , and  $M$  are essentially the parameters for the modeling of the observation distributions in parametric form.

The identification of the ICAMHMM model is to infer the parameters based on training data. In [14], an interactive procedure to update parameters based on Gaussian mixtures has been formulated and derived to model the continuous observation densities in HMM framework. In ICAMHMM, the new observation model is based on non-Gaussian mixtures, and each non-Gaussian mixture component is associated with a standard ICA. In this section, the re-estimation formulae for learning all the model parameters of the ICAMHMM framework are derived as follows.

In order to develop the updating rules for model parameters  $\lambda$  in ICAMHMM framework, we first review some definitions that are required for our derivations. The forward variable  $\alpha_t(i)$  and backward variable  $\beta_t(i)$  are defined as [12]

$$\alpha_t(i) = P(\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t, q_t = i \mid \lambda), \quad (16)$$

$$\beta_t(i) = P(\mathbf{o}_{t+1}, \mathbf{o}_{t+2}, \dots, \mathbf{o}_T, \mid q_t = i, \lambda). \quad (17)$$

Using the forward variable and the backward variable defined above, two probabilities of the joint event can be defined [12]:

$$\xi_t(i, j) \equiv P(q_t = i, q_{t+1} = j \mid \mathbf{O}, \lambda) = \frac{\alpha_t(i) \cdot a_{ij} \cdot b_j(\mathbf{o}_{t+1}) \cdot \beta_{t+1}(j)}{P(\mathbf{O} \mid \lambda)}, \quad (18)$$

$$\gamma_t(i) \equiv P(q_t = i \mid \mathbf{O}, \lambda) = \frac{\alpha_t(i) \cdot \beta_t(i)}{P(\mathbf{O} \mid \lambda)}, \quad (19)$$

where (18) defines the probability of the joint event: a path passes through state  $i$  at time  $t$  and through state  $j$  at time  $t + 1$ , given the available sequence of observations  $\mathbf{O}$  and the parameters of the model  $\lambda$ . The (19) defines the probability of being in state  $i$  at time  $t$ , given the observation sequence  $\mathbf{O}$ , and the model  $\lambda$ .

In our non-Gaussian mixture observation model, we generalize the intermediate variable  $\gamma_t(j)$  to  $\gamma_t(j, k)$ . The variable  $\gamma_t(j, k)$  is defined as

$$\gamma_t(j, k) = \gamma_t(j) \cdot \frac{P(C_{jk}) \cdot p(\mathbf{o}_t \mid C_{jk}, \theta_{jk})}{\sum_{m=1}^K P(C_{jm}) \cdot p(\mathbf{o}_t \mid C_{jm}, \theta_{jm})}, \quad (20)$$

where  $p(\mathbf{o}_t \mid C_{jk}, \theta_{jk})$  is the non-Gaussian observation probability density function  $p(\mathbf{o} \mid C_{jk}, \theta_{jk})$  evaluated at  $\mathbf{o}_t$ . The term  $\gamma_t(j, k)$  can be interpreted as the probability of being in state  $j$  at time  $t$  with the  $k$ -th mixture component accounting for  $\mathbf{o}_t$ .

## B. Re-estimation Algorithms

### B.1 Maximum likelihood formulation

To derive ICAMHMM updating rules, we introduce the likelihood function (as defined in (3)) and partition the likelihood function over the state space. First, we denote the joint likelihood of observation and the state sequence as

$$LH_\lambda(\mathbf{O}, q) \equiv P(\mathbf{O}, q \mid \lambda). \quad (21)$$

Thus, the partition of the likelihood function over the state space is represented as a sum over the set,  $\ell$ , of all possible state sequences  $q$

$$LH_\lambda(\mathbf{O}) = \sum_{q \in \ell} P(\mathbf{O}, q \mid \lambda) = \sum_{q \in \ell} LH_\lambda(\mathbf{O}, q). \quad (22)$$

Note that the posterior likelihood defined in (22) is over all possible sequences of states. The objective is to maximize  $LH_\lambda(\mathbf{O})$  over all parameters  $\lambda$ . For a particular state sequence  $q = q_1, q_2, \dots, q_T$ , the probability  $P(q \mid \lambda)$  is:

$$P(q \mid \lambda) = \pi_{q_1} \prod_{t=2}^T a_{q_{t-1}q_t}. \quad (23)$$

By substituting (23) into (22), the likelihood function can be rewritten as:

$$LH_\lambda(\mathbf{O}) = \sum_{q \in \ell} P(\mathbf{O}, q | \lambda) = \sum_{q \in \ell} P(\mathbf{O} | q, \lambda) \cdot P(q | \lambda) = \sum_{q \in \ell} \pi_{q_1} \cdot \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \cdot \prod_{t=1}^T b_{q_t}(\mathbf{o}_t). \quad (24)$$

Note that  $b_{q_t}(\mathbf{o}_t)$  is the observation probability density function  $b_{q_t}(\mathbf{o})$  (see 7) evaluated at  $\mathbf{o}_t$ , given the state at time  $t$  is  $q_t$ . Recall (7), we model  $b_{q_t}(\mathbf{o})$  as ICA mixture models

$$b_{q_t}(\mathbf{o}_t) = \sum_{k=1}^K p(\mathbf{o}_t | C_{q_t k}, \theta_{q_t k}) \cdot P(C_{q_t k}), \quad (25)$$

where  $p(\mathbf{o}_t | C_{q_t k_t}, \theta_{q_t k_t})$  is the non-Gaussian probability density function  $p(\mathbf{o} | C_{q_t k_t}, \theta_{q_t k_t})$  evaluated at  $\mathbf{o}_t$ . Based on the above representation, the likelihood function can be further partitioned by choosing a particular classification sequence,  $\mathbf{K} = k_1, k_2, \dots, k_T$ , of mixture densities, where the values of  $k_t$  ( $1 \leq t \leq T$ ) can be chosen from  $\{1, 2, \dots, K\}$ . The mixture sequence  $\mathbf{K}$  determines which class each observation belongs to. We denote the set of all possible mixture sequences as  $\mathfrak{h}$ . By definition we have the following partitions of the likelihood function:

$$LH_\lambda(\mathbf{O}) = \sum_{q \in \ell} LH_\lambda(\mathbf{O}, q) = \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} LH_\lambda(\mathbf{O}, q, \mathbf{K}), \quad (26)$$

where  $LH_\lambda(\mathbf{O}, q, \mathbf{K})$  is the joint likelihood of  $\mathbf{O}, q, \mathbf{K}$  for some particular mixture sequence  $\mathbf{K} \in \mathfrak{h}$

$$LH_\lambda(\mathbf{O}, q, \mathbf{K}) = \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \prod_{t=1}^T p(\mathbf{o}_t | C_{q_t k_t}, \theta_{q_t k_t}) \cdot P(C_{q_t k_t}). \quad (27)$$

Thus, a complete representation of the likelihood function is represented as

$$LH_\lambda(\mathbf{O}) = \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \prod_{t=1}^T p(\mathbf{o}_t | C_{q_t k_t}, \theta_{q_t k_t}) \cdot P(C_{q_t k_t}). \quad (28)$$

To apply maximum likelihood estimation, we define the auxiliary function (the  $Q$ -function) with similar form as [13] [14]:

$$Q(\lambda, \lambda^*) = \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} LH_\lambda(\mathbf{O}, q, \mathbf{K}) \cdot \log LH_{\lambda^*}(\mathbf{O}, q, \mathbf{K}), \quad (29)$$

where  $\lambda$  is the current parameter set, and  $\lambda^*$  is the new parameter set. It has been proved that maximization of  $Q$ -function of the above form leads to increased likelihood [17] [14],

i.e.  $Q(\lambda, \lambda^*) > Q(\lambda, \lambda)$  implies that  $LH_{\lambda^*}(\mathbf{O}) > LH_{\lambda}(\mathbf{O})$ . Recall that the likelihood  $LH_{\lambda}(\mathbf{O})$  is first partitioned in the state sequence space, and then further partitioned in the mixture sequence space. Similarly, by definition the  $Q$ -function can also be partitioned as

$$Q(\lambda, \lambda^*) = \sum_{j=1}^N Q_j(\lambda, \lambda^*) = \sum_{j=1}^N \sum_{k=1}^K Q_{jk}(\lambda, \lambda^*), \quad (30)$$

where

$$Q_j(\lambda, \lambda^*) = \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} LH_{\lambda}(\mathbf{O}, q, \mathbf{K}) \cdot \log LH_{\lambda^*}(\mathbf{O}, q_t = j, \mathbf{K}), \quad (31)$$

and

$$Q_{jk}(\lambda, \lambda^*) = \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} LH_{\lambda}(\mathbf{O}, q, \mathbf{K}) \cdot \log LH_{\lambda^*}(\mathbf{O}, q_t = j, k_t = k). \quad (32)$$

Based on (30)-(32), we successfully partition the  $Q$ -function, and construct the function (32). The inner summation of (32) has the identical form to the one used by Liporace in [13]. In [13], Liporace has already proved that the  $Q$ -function of that form has a unique global maximum as a function of  $\lambda^*$ , and this maximum is at a critical point. Assuming the mixture densities in (25) are non-Gaussian but log-concave or elliptically symmetric, and based on Liporace's theorems, we can conclude our  $Q$ -function that takes summations over  $N$  and  $K$  also has a unique maximum at a critical point. Thus, the maximization of the  $Q$  function leads to increased likelihood, and the likelihood function converges to a relative maximum.

## B.2 Derivation of Re-estimation Formulae

Since we have proved the convergence of a relative maximum of the  $Q$ -function, we now can apply the standard Lagrange optimization technique to derive the re-estimation formulae for learning model parameters of the ICAMHMM.

To derive the re-estimation formulae, we can calculate the  $Q$ -function and split the

results into three terms as

$$\begin{aligned}
Q(\lambda, \lambda^*) &= \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \log \pi_{q_1}^* + \\
&\sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \left( \sum_{t=1}^{T-1} \log a_{q_t q_{t+1}}^* \right) + \\
&\sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \left\{ \sum_{t=1}^T \log [p(\mathbf{o}_t \mid C_{q_t k_t}^*, \theta_{q_t k_t}^*) \cdot P(C_{q_t k_t}^*)] \right\}.
\end{aligned} \tag{33}$$

Since the parameters  $\pi_i^*$ ,  $a_{ij}^*$ , and  $P(C_{jk}^*)$  are independently separated in the sum, we can optimize each term individually. The first term in (33) becomes

$$\sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \log \pi_{q_1}^* = \sum_{i=1}^N P(\mathbf{O}, q_1 = i \mid \lambda) \cdot \log \pi_i^*. \tag{34}$$

To optimize the right hand side of (34), we can add the Lagrange multiplier  $\psi$  using the constraint that  $\sum_i \pi_i^* = 1$ , and setting the partial derivative to zero. We get

$$\frac{\partial}{\partial \pi_i^*} \left[ \sum_{i=1}^N P(\mathbf{O}, q_1 = i \mid \lambda) \cdot \log \pi_i^* + \psi \left( \sum_{i=1}^N \pi_i^* - 1 \right) \right] = 0. \tag{35}$$

Calculate the derivative and sum to get  $\psi$  first, and then solve for each  $\pi_i^*$ , we get

$$\pi_i^* = \frac{P(\mathbf{O}, q_1 = i \mid \lambda)}{P(\mathbf{O} \mid \lambda)}. \tag{36}$$

The second term in (33) becomes

$$\sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \left( \sum_{t=1}^{T-1} \log a_{q_t q_{t+1}}^* \right) = \sum_{i=1}^N \sum_{j=1}^N \sum_{t=1}^{T-1} P(\mathbf{O}, q_t = i, q_{t+1} = j \mid \lambda) \log a_{ij}^*. \tag{37}$$

In a similar way, we use the Lagrange multiplier with the constraint  $\sum_{j=1}^N a_{ij}^* = 1$ , and get

$$a_{ij}^* = \frac{\sum_{t=1}^{T-1} P(\mathbf{O}, q_t = i, q_{t+1} = j \mid \lambda)}{\sum_{t=1}^{T-1} P(\mathbf{O}, q_t = i \mid \lambda)}. \tag{38}$$

The third term in (33) can be further split into two terms. Thus we get

$$\begin{aligned}
& \sum_{q \in \ell} \sum_{\mathbf{K} \in \mathfrak{h}} P(\mathbf{O}, q, \mathbf{K} \mid \lambda) \cdot \left\{ \sum_{t=1}^T \log [p(\mathbf{o}_t \mid C_{q_t k_t}^*, \theta_{q_t k_t}^*) \cdot P(C_{q_t k_t}^*)] \right\} \\
&= \sum_{j=1}^N \sum_{k=1}^K \sum_{t=1}^T P(\mathbf{O}, q_t = j, k_t = k \mid \lambda) \cdot \log P(C_{jk}^*) + \\
& \sum_{j=1}^N \sum_{k=1}^K \sum_{t=1}^T P(\mathbf{O}, q_t = j, k_t = k \mid \lambda) \cdot \log p(\mathbf{o}_t \mid C_{jk}^*, \theta_{jk}^*).
\end{aligned} \tag{39}$$

The first term on the right hand side of (39) can be optimized using Lagrange multiplier to get the  $P(C_{jk}^*)$

$$P(C_{jk}^*) = \frac{\sum_{t=1}^T P(q_t = j, k_t = k \mid \mathbf{O}, \lambda)}{\sum_{t=1}^T \sum_{k=1}^K P(q_t = j, k_t = k \mid \mathbf{O}, \lambda)}. \tag{40}$$

The second term on the right hand side of (39) is the representation based on non-Gaussian densities. As formulated earlier, we associate each non-Gaussian mixture component with a standard ICA. Thus, the non-Gaussian component can be decomposed as a linear combination of statistically independent sources. The inference of non-Gaussian component densities in parametric form is described as follows.

In the presented ICAMHMM, the spatial statistics of observations are exploited by ICA mixture model. The observation model is to represent the observation density functions as the weighted sum of non-Gaussian distributions. The mixture components can be considered as a combination of different classes. For each class, the non-Gaussian density is further represented as a linear transformation of statistically independent sources. Thus, without considering the state space, the data within the  $k$ -th ( $1 \leq k \leq K$ ) class are described by

$$\mathbf{o}_t = \mathbf{M}_k \cdot \mathbf{s}_k + \boldsymbol{\mu}_k, \tag{41}$$

where  $\mathbf{M}_k$  is the basis matrix for the  $k$ -th component,  $\mathbf{s}_k$  contains the statistically independent sources, and  $\boldsymbol{\mu}_k$  is the bias vector for class  $k$ .

Taking the state space into account and applying ICA mixture method, the observation density at time  $t$ , given state  $j$ , is computed as

$$\log p(\mathbf{o}_t \mid C_{jk}, \theta_{jk}) = \log p(\mathbf{s}_{jk}) - \log(\det\{\mathbf{M}_{jk}\}), \tag{42}$$

where  $\theta_{jk} = \{\mathbf{M}_{jk}, \boldsymbol{\mu}_{jk}\}$ . Note that  $\mathbf{M}_{jk}$  implicitly models the sources  $\mathbf{s}_{jk}$  (see (41)). The density of  $\mathbf{s}_{jk}$  can be approximated by super-Gaussian or sub-Gaussian densities depending on the source model.

The basis matrix  $\mathbf{M}_{jk}$  for the  $k$ -th class at state  $j$  can be learnt by using the standard ICA algorithm. We choose the extended infomax estimation algorithm [16] to learn the parameters in the ICA models. By selecting different sigmoidal nonlinearities in infomax, it is suitable to adapt the learning for both super-Gaussian and sub-Gaussian sources [11] [16]. The adaptation of the basis matrix is

$$\Delta \mathbf{M}_{jk} \propto p(C_{jk} | \mathbf{o}_t, \Theta_j) \cdot \frac{\partial}{\partial \mathbf{M}_{jk}} \log p(\mathbf{o}_t | C_{jk}, \theta_{jk}), \quad (43)$$

where  $\Theta_j = \{\theta_{j1}, \dots, \theta_{jK}\}$  are the parameters for each component density.  $p(C_{jk} | \mathbf{o}_t, \Theta_j)$  can be computed as

$$p(C_{jk} | \mathbf{o}_t, \Theta_j) = \frac{p(\mathbf{o}_t | \theta_{jk}, C_{jk}) \cdot p(C_{jk})}{\sum_{k=1}^K p(\mathbf{o}_t | \theta_{jk}, C_{jk}) \cdot p(C_{jk})}. \quad (44)$$

The mean vector for each mixture at state  $j$  is approximated by

$$\boldsymbol{\mu}_{jk} = \frac{\sum_{t=1}^T \mathbf{o}_t \cdot p(C_{jk} | \mathbf{o}_t, \Theta_j)}{\sum_{t=1}^T p(C_{jk} | \mathbf{o}_t, \Theta_j)}. \quad (45)$$

In the actual implementation, the calculation of the basis matrix  $M_{jk}$  is approximated using the extended infomax algorithm reported in [16]. For a given state  $j$ , each mixture component in the density model is an separate ICA model and finding the basis matrix is equivalent to finding its pseudoinverse, i.e. the filtering matrix  $\mathbf{W}_{jk}$  with  $\mathbf{W}_{jk} = \mathbf{M}_{jk}^{-1}$ . The updating rule for  $\mathbf{W}_{jk}$  at  $n + 1$  step is given by [11] [16]:

$$\mathbf{W}_{jk}^{[n+1]} = \mathbf{W}_{jk}^{[n]} + \rho \cdot (\mathbf{I} - \mathit{Signs} \cdot \tanh(\mathbf{u}) \cdot \mathbf{u}^T - \mathbf{u}\mathbf{u}^T) \mathbf{W}_{jk}^{[n]}, \quad (46)$$

where  $\rho$  is the learning rate and  $\mathbf{u}$  is the source estimate  $\mathbf{u} = \mathbf{W} \cdot \mathbf{o}_t$ . The *Signs* is a diagonal matrix with ones representing super-Gaussian distribution and negative ones representing sub-Gaussian distribution according to Kurtosis estimate.

### B.3 A Summary of the Algorithm

The ICA mixture model itself in observation space only gives us the spatial statistics. The temporal dynamics are not well exploited in ICA mixture model itself since generally

ICA algorithm ignores the order of the signals. However, by integrating ICA mixture model into HMM framework, the temporal information is modeled by the transition matrix and the state sequence. During implementation, we randomly initialize the following parameters: (i) the basis matrix coefficients  $M = \{\mathbf{M}_k\}$ ,  $1 \leq k \leq K$ ; (ii) the mean vector coefficients  $\mu = \{\boldsymbol{\mu}_k\}$ ,  $1 \leq k \leq K$ ; (iii) the observation densities  $b(\mathbf{o}_t)$ . To integrate them into ICAMHMM framework, these coefficients are duplicated for each state and used as initial values for each state. In the temporal modeling step, (36), (38), and (40) are implemented and calculated using the intermediate variables iteratively. To summarize the algorithms derived above, the re-estimation formulae for ICAMHMM are listed as follows:

$$\pi_i^* = \gamma_1(i) \quad (47)$$

$$P(C_{jk}^*) = \frac{\sum_{t=1}^T \gamma_t(j, k)}{\sum_{t=1}^T \sum_{k=1}^K \gamma_t(j, k)} \quad (48)$$

$$\boldsymbol{\mu}_{jk}^* = \frac{\sum_{t=1}^T \gamma_t(j, k) \cdot \mathbf{o}_t}{\sum_{t=1}^T \gamma_t(j, k)} \quad (49)$$

$$a_{ij}^* = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}. \quad (50)$$

Note that when calculating the intermediate variables like  $\alpha_t(i)$ ,  $\beta_t(i)$ ,  $\xi_t(i, j)$ ,  $\gamma_t(i, j)$ , the observation densities  $b_j(\mathbf{o}_t)$  is computed as

$$b_j(\mathbf{o}_t) = \sum_{k=1}^K P(C_{jk}^*) \cdot b_{jk}(\mathbf{o}_t), \quad (51)$$

where

$$b_{jk}(\mathbf{o}_t) = \exp(\log p(\mathbf{s}_{jk}) - \log(\det\{\mathbf{M}_{jk}\})). \quad (52)$$

Equation (52) can be interpreted as the  $k$ -th component density, given the state  $j$ . Thus, the weighted summation over  $k$  gives the overall observation density at state  $j$ . During the practical implementation, the  $\log$  is used to avoid the precision issues.

The procedures for ICAMHMM framework are summarized as follows:

1. Initialize the parameters, such as the number of mixtures  $K$ , the number of hidden states  $N$ , and the ICA source model  $p(\mathbf{s})$ .

2. Apply ICA mixture to model the observations, and calculate the parameters for mixture component densities.
3. Apply the derived re-estimation formulae (47)-(52) to compute all the parameters for ICAMHMM.

The detailed description of ICAMHMM learning is listed in Algorithm 1.

---

**Algorithm 1** ICAMHMM Learning

---

Select the number of mixtures  $K$  and the number of hidden states  $N$

Select ICA source model  $p(\mathbf{s})$

Initialize ICA mixture model parameters  $\mathbf{M}_{jk}, \boldsymbol{\mu}_{jk}, k = 1, \dots, K$

Randomly initialize the initial state distributions

Take input observation sequence  $\mathbf{o}_t, t = 1, \dots, T$

Repeat

Repeat

For  $j=1$  to  $N$  do

For  $k = 1$  to  $K$  do

Calculate  $p(\mathbf{s}_{jk})$  based on ICA source model

Calculate the mixture densities  $b_{jk}(\mathbf{o}_t)$  using (52)

Adapt the basis matrix  $\mathbf{M}_{jk}$  using (43) and (46)

Adapt the bias vector  $\boldsymbol{\mu}_{jk}$  using (45)

Update the class probability for each class using (44)

End

Calculate observation density using (51)

End

Until all the observation samples have been used

Calculate  $\alpha_t(i), \beta_t(i), \xi(i, j), \gamma_t(i), \gamma_t(j, k), 1 \leq i, j \leq N, 1 \leq k \leq K$

Calculate the likelihood of the observations sequence (the summation over  $\alpha_t(i)$ )

Adapt the transition matrix

Until the adaptation has converged or the maximum number of iterations is reached

---

### C. Likelihood Evaluation

The re-estimation formulae derived in the previous subsection can be used to compute all the parameters needed to represent a HMM model with non-Gaussian mixture observation densities. Each model (model parameters) represents one event. Given any observation sequence  $\mathbf{O}$ , the likelihood of producing the observation sequence is given by  $P(\mathbf{O} | \lambda)$ . The calculation of the likelihood is very similar to the classical *Forward-Backward Procedure*. The major difference is that the observation densities are computed from the sources and basis matrices. The likelihood  $P(\mathbf{O} | \lambda)$  is inductively solved as follows:

1) Initialization:

$$\alpha_1(i) = \pi_i \cdot \sum_{k=1}^K P(C_{ik}) \cdot \exp(\log p(\mathbf{s}_{ik}) - \log(\det\{\mathbf{M}_{ik}\})), \quad 1 \leq i \leq N. \quad (53)$$

2) Induction:

$$\alpha_{t+1}(i) = \left[ \sum_{i=1}^N \alpha_t(i) \cdot a_{ij} \right] \sum_{k=1}^K P(C_{jk}) \cdot \exp(\log p(\mathbf{s}_{jk}) - \log(\det\{\mathbf{M}_{jk}\})), \quad (54)$$

$$1 \leq t \leq T - 1, \quad 1 \leq i \leq N. \quad (55)$$

3) Termination:

$$P(\mathbf{O} | \lambda) = \sum_{i=1}^N \alpha_T(i). \quad (56)$$

The above iteration procedure gives one likelihood for  $\mathbf{O}$ , given the model parameters. For  $D$  events, we need to calculate  $D$  likelihood given all the model parameters. The one that give the maximum is considered as the detected event (see (3) and (4)).

### D. Discussions

The algorithms derived in this paper provides an alternative way to estimate mixture models under HMM framework other than the classic mixture of Gaussian (MoG) models. The ICA mixture model is able to decompose the observations into a representation of non-Gaussian sources, and this representation, when combined with HMM learning, is able to adapt the data in the spatial domain and the time domain simultaneously.

However to determine the number of mixture components is non-trivial in mixture models. In our experiments, the number of mixture is arbitrarily chosen and we generally use

two mixtures per each observation model to make the implementation easier and it seems sufficient for us to cover most cases both in simulations and in our collected video data. MoG HMM is sensitive to the initial parameter estimation and the number of mixtures since expectation-maximization (EM) algorithm can only find a local maximum. Our ICAMHMM model essentially inherits the same limitation since we plug our ICA mixture mode into the existing EM based HMM learning. There are some efficient algorithms [18] and [19] that can be used to estimate these initial parameters and thus can effectively help convergence and estimate the number of mixtures. In the present paper, we would focus on general model development.

#### IV. SIMULATION RESULTS USING ICA MIXTURE HIDDEN MARKOV MODEL

To verify the proposed model, we generate the following simulation data to validate the model and algorithms. We define two “hidden” states with each state being associated with a mixture model. The observations are defined in a two dimensional space. To show the effectiveness of the proposed model, we choose some common non-Gaussian distributions in images and videos and combine them together to form the observation distributions.

The observation density for state 1 has three mixtures with each mixture being generated from an independent ICA model. The first ICA mixture in state 1 is created by mixing two independent sources randomly drawn from a Laplacian distribution and a Tukey-Lambda distribution. The second ICA mixture is a mixing of two independent uniform sources. The third ICA mixture is a mixing of two independent Laplacian sources. Each of the above three mixtures has their own bias vector. The weights, or the class probabilities for each of the three mixtures are 40%, 10% and 50%, respectively. The observations for state 1 is shown in Fig. 2, and we also plot the directions and magnitudes of the basis matrix showing how the two statistically independent sources are being mixed or transformed through basis matrix to form two dimensional observations.

The observation density for state 2 has two mixtures with each mixture being generated from an independent ICA model. The first mixture component is created by mixing two independent sources randomly drawn from a Gamma distribution and a uniform distribution. The second mixture is created by mixing a uniform source and a binorm source. The weights are 50% for both mixture components. The observations for state 2 are shown in

Fig. 3.

In Fig. 4 we plot the complete observation space which includes all observations generated from state 1 and observations generated from state 2 based on a pre-defined state sequence. The pre-defined state sequence is shown in Fig. 5. We apply our ICAMHMM algorithm to the observations and then get the estimated state sequence as shown in Fig. 6. The results show that the hidden states of 99.85% observation samples have been correctly identified. The results show that ICA mixture model combined with existing EM based HMM framework is effective in learning the data and capture the temporal characteristics.

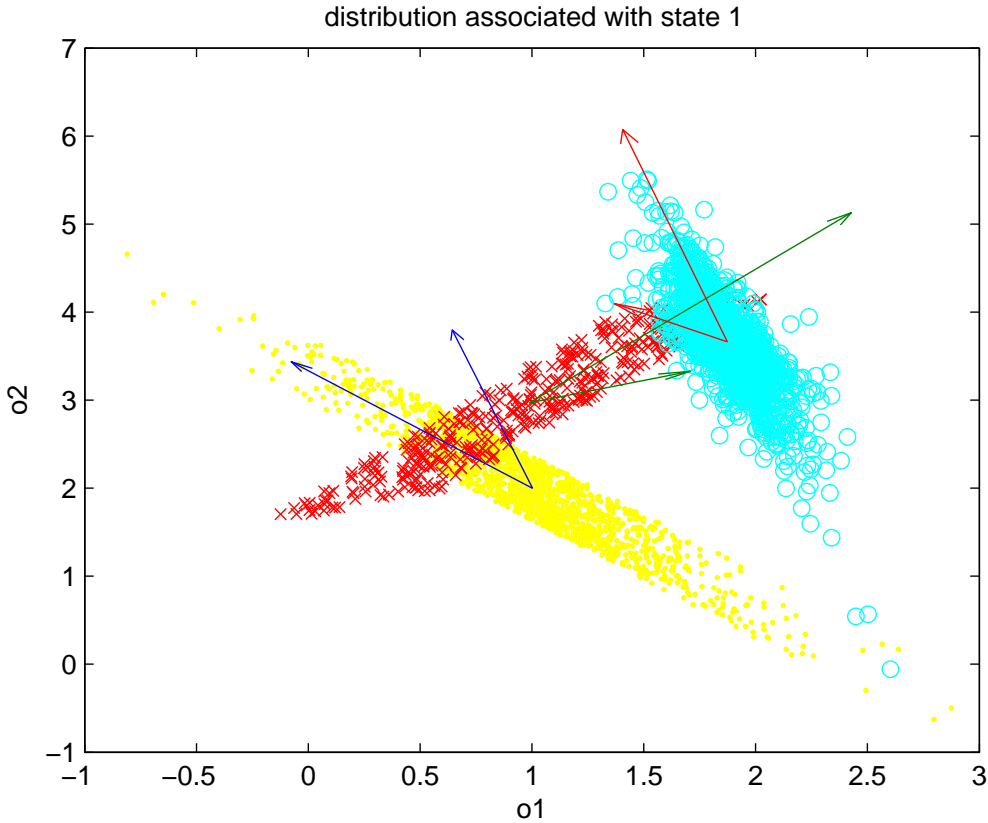


Fig. 2. The observations associated with state 1 with 3 mixtures.

The simulation results show that the ICAMHMM can effectively capture both the spatial (it is actually in the feature space rather than the real spatial domain as in images) characteristics and temporal characteristics, and converge to a local maximum. The recovered or learned data can be represented by a mixture model with each mixture component

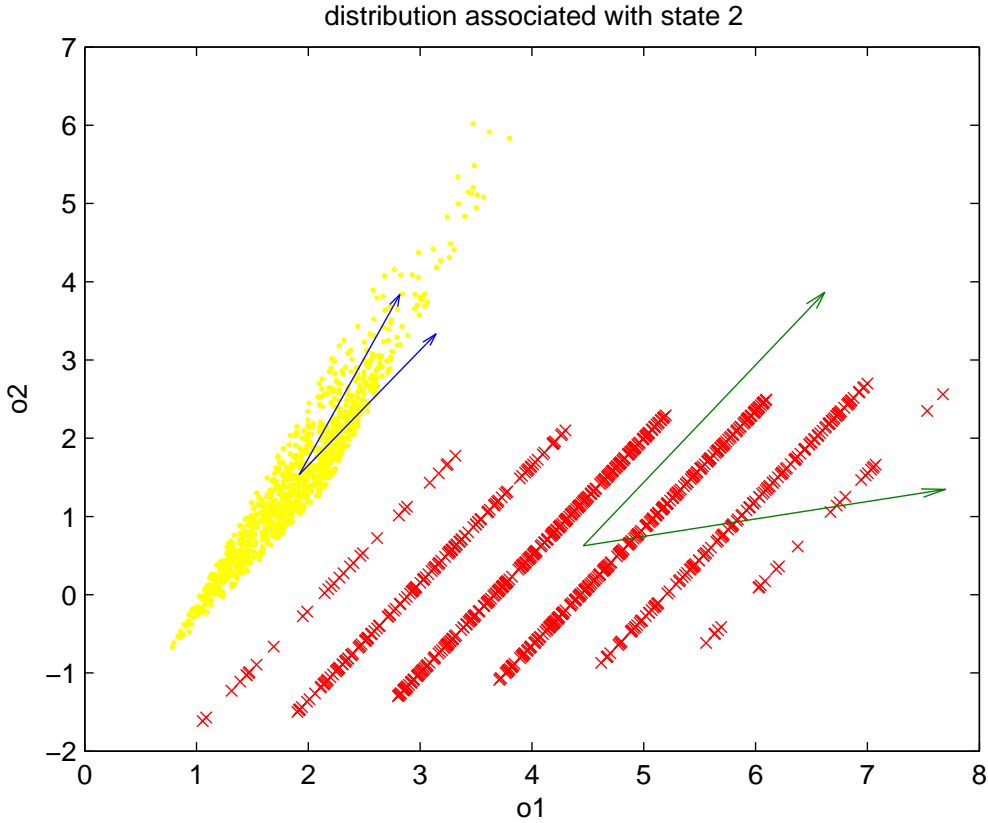


Fig. 3. The observations associated with state 2 with 2 mixtures.

being further described by a linear combination of two or more independent sources.

One key question is that how this new model performs against the classical MoG HMM method. We would like to clarify that the introduction of ICAMHMM is not intended to replace the MoG HMM, but rather is an extension of the MoG to bring additional mixture models into HMM framework. In practice when using the real world data, users might need to roughly check the distributions in the observation space. If the distribution show a good Gaussian shape or a mixture of Gaussian shape, then MoG HMM is more suitable. If the distribution shows a strong non-Gaussianity, for example some non-random structure, we believe our method is more suitable, or have a more reasonable representation in a form of non-Gaussian independent sources. To further quantify the amount of non-Gaussianity in the observed samples and then make decisions whether MoG HMM or ICAMHMM is more suitable can be a part of our future work.

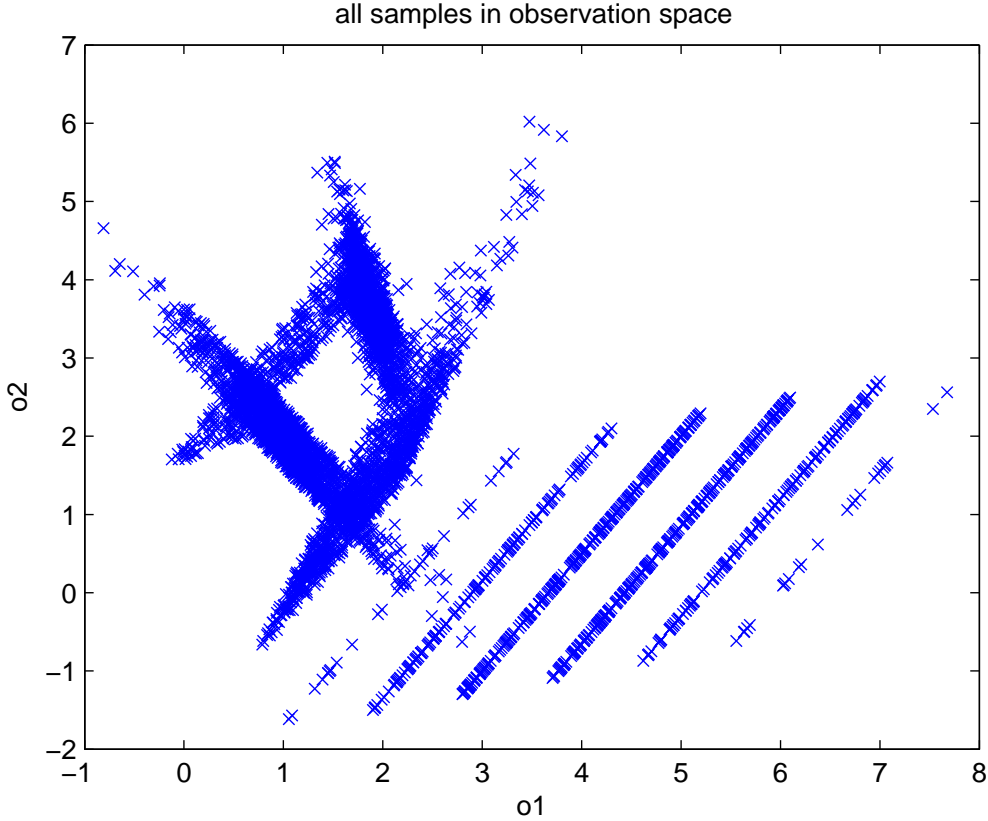


Fig. 4. All samples in the observation space.s

We compared ICAMHMM with MoG HMM and found that when both of them converge to the global maximum, the performances are similar. Note that we set the number of states as 2 and we use 2 Gaussian mixtures for each observation distribution. Using MoG HMM, the model can also adapt the simulation data and by average 99.98% of hidden states can be correctly estimated when the algorithm converges. However, we found that for the simulation data, MoG HMM has difficulties in convergence. We ran Monte-Carlo simulations on randomly chosen initial parameters on the same data set. The results show that the MoG HMM converges to the correct global maximum at the ratio of 33% while the ICAMHMM converges at the ratio of 52%.

The computation cost of the training process for ICAMHMM is higher than MoG HMM learning due to an additional ICA step. In our simulations, the ICAMHMM algorithm was implemented in Matlab without optimization and the average running time for each

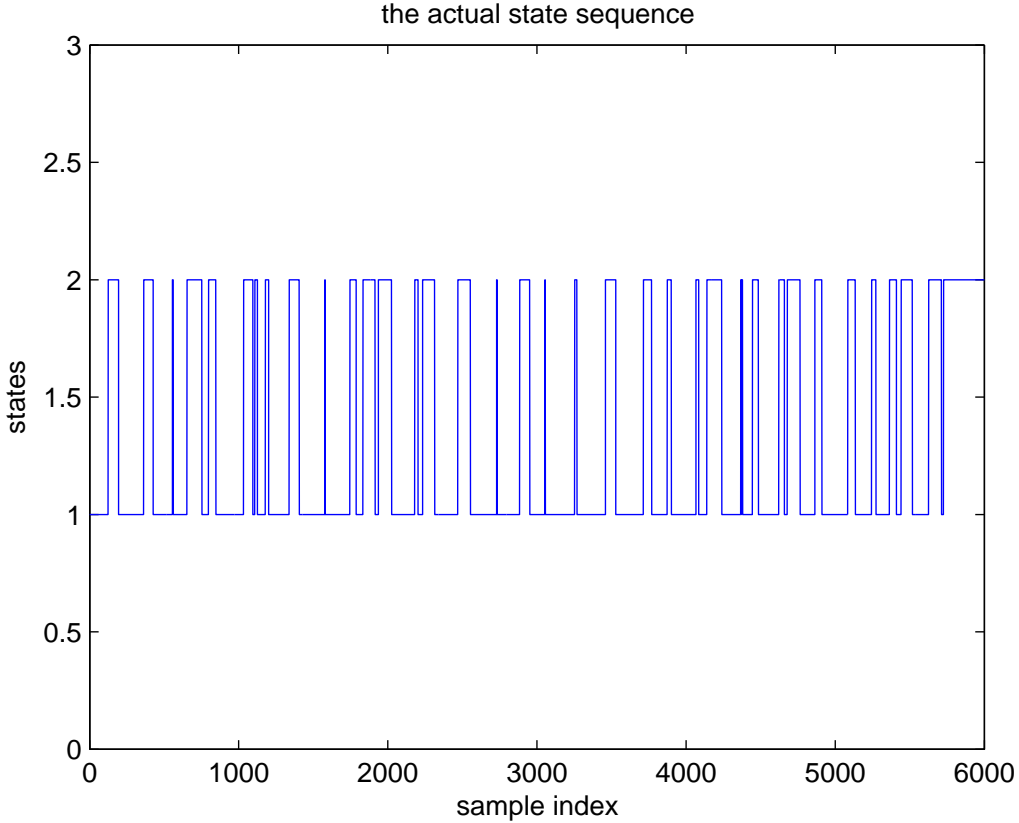


Fig. 5. The actual state sequence constructed for the simulation.

iteration is 16.23 seconds. For comparison, the average running time for each iteration in MoG HMM learning on the same hardware is 1.3 seconds. The performance of ICAMHMM highly depends on selected ICA algorithms. However, note that these calculations are in the training step and may only need to be done once in off-line mode.

In the feature extraction step, our selected features can be run in real-time since we only choose color histograms as the main features. In the event detection step, ICAMHMM does not have additional computation cost. The computation cost of the likelihood function is the same as MoG HMM.

## V. CONTENT ANALYSIS BASED ON ICA MIXTURE HIDDEN MARKOV MODEL

The proposed ICAMHMM framework is applied to video data for content analysis. We develop an event detection and recognition system which includes the following modules:

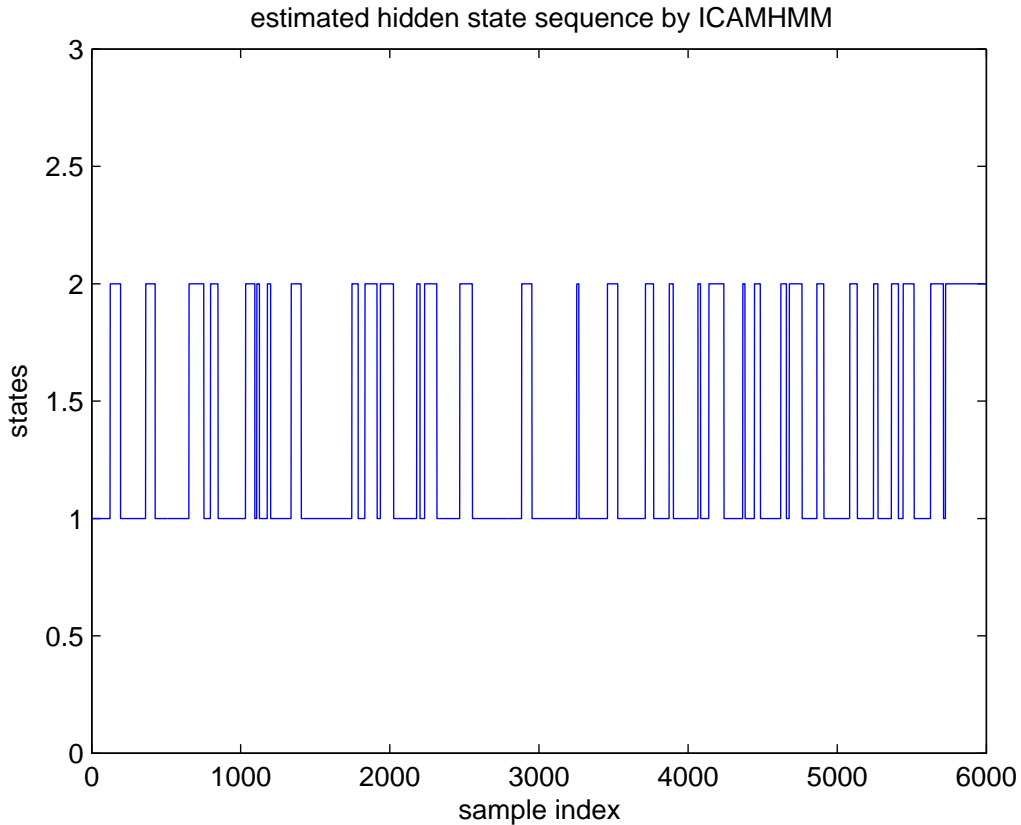


Fig. 6. The estimated state sequence learnt through ICAMHMM.

1) Feature extraction. 2) Shot boundary detection. 3) Observation density estimation and model training. 4) Event detection. First, we choose illumination invariant histograms to generate the raw features, and then ICA is applied to process these features and project the data into two dimensional ICA subspace. In the ICA subspace, shot boundaries are identified by using a clustering algorithm. Next, we assume that each event consists of one or multiple shots. Several training shots are selected to train the ICAMHMM model, and the rest of the video data are evaluated by the trained models. The model that gives the maximum likelihood is considered as the identified event. Note that ICA techniques are used twice in the event detection system. During feature extraction, the use of ICA can be considered as a pre-processing filter which creates a compact and efficient representation of the original data. During model learning process, ICA mixture is used to model the observation densities that are non-Gaussian distributions.

### A. Feature Extraction

In this paper, we use frame-based global features to analyze the content. However, the proposed framework can be easily extended to incorporate local features, such as object based features, to analyze the events which are associated to the specific video objects. Illumination change is an important factor that affects the performance of content based analysis of video data. To reduce the lighting effects, we choose the normalized chromaticity histograms [20] as our color features. Based on 3D RGB color space, the 2D illumination-invariant normalized chromaticity  $(r, g)$  is defined as,

$$r = R/(R + G + B), \quad g = G/(R + G + B). \quad (57)$$

Histograms with 256 bins are generated as features in the normalized chromaticity color space for each video frame. The dimension of the feature vector for each video frame is 256. In our algorithms, we apply ICA to extract the two independent components (ICs) from high-dimensional feature vector. The ICA task is to find filter matrix using only the observations. Each video frame is processed as one observation that can be considered as a linear combination of hidden basis functions. The ICA model assumes that the observations are a linear combination of statistically independent sources. The illumination invariant histograms are used as the input signal. We denote  $\mathbf{x}_t, t = 1, \dots, T$  as the input signal, and  $\mathbf{o}_t, t = 1, \dots, T$  as the output signals of ICA learning model, where  $T$  is the number of video frames. The ICA learning model is defined as

$$\mathbf{o}_t = W \cdot \mathbf{x}_t = W \cdot M \cdot \mathbf{s}_t, \quad (58)$$

where  $W$  is the filter matrix,  $\mathbf{s}_t$  is the statistically independent sources.  $\mathbf{o}_t$  is considered as recovered independent sources. The rows of the output signals are independent components (ICs). Since the time course is only associated with the ICs, we select the two most significant ICs as the new features instead of the basis functions. Thus, the observation feature space is reduced from high-dimensional to 2-dimensional.

### B. Shot Boundary Detection

Based on video frame distribution in the ICA subspace, a dynamic clustering algorithm [21] is applied to classify video frames into shots and detect the shot boundaries. Each

video frame is represented by a point in ICA subspace, and Euclidean distance is used as dissimilarity measure between two points. A dynamic clustering algorithm based on adaptive thresholding is employed to detect shot boundaries [21].

### *C. Model Training*

The output signals  $\mathbf{o}_t, t = 1, \dots, T$  from ICA learning in previous feature extraction step are used as observations in ICAMHMM framework. During ICAMHMM model training, the spatial characteristics of the signals are captured by ICA mixture model and the temporal characteristics of the observations are explored by a HMM modeling to find the most probable state sequence. The parameter re-estimation formulae are derived in earlier sections. The model training described in this section can be considered as supervised learning since the training sequences are manually annotated. For each event, a different model is trained, and the model parameters are used to represent the event.

### *D. Event Detection*

In the proposed event detection system, we assume each shot can be categorized into one of the candidate events. The detection of event is essentially a sequence classification since each event is represented as model parameters. For a new sequence, we evaluate the log-likelihood given each model. The event whose model gives the maximum log-likelihood will be declared as the detected event for the test sequence.

### *E. Experimental Results*

In order to test the effectiveness of the proposed algorithms, we choose golf video data as a case study. For the golf video, the recurrent patterns are generally very recognizable especially when the players hit long and straight shots. The first scene is relatively static when the payer prepares for his hit. After he swings and hits the ball, the next scene often contains high motion activities when the camera follows the ball. Finally, the scenes always focus on the golf court to track the ball or the player, and those generally contain low activities.

In the experiment, one hour golf video is captured from TV. The video data is encoded in MPEG-1 format with frame rate 29.97 frames per second. The video contains different

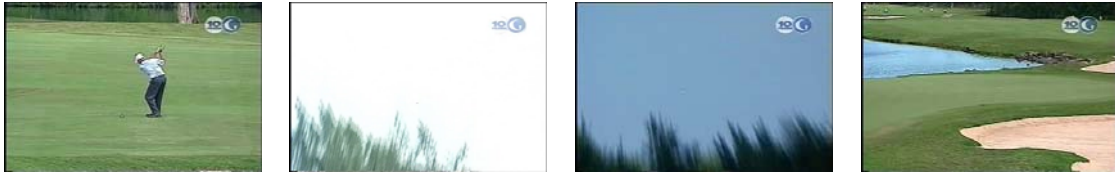


Fig. 7. Video pattern for a tee shot with full-swing.



Fig. 8. Video pattern for a fairway shot.



Fig. 9. Irrelevant events.

lighting conditions, multiple views in one window, and quick camera motions. Compared with surveillance or traffic video with relatively static background, this golf video data is a challenge to analyze. Even though the video contains many dynamics, some recurrent patterns are still recognizable by human perception. For example, “full-swing” scenes generally begin with a zoom-in to capture the player’s preparation for his hit, and then followed by a quick camera motion to track the ball. Finally, the last scenes are usually some zoom-ins to locate the slowly moving ball. Other events include random camera moves, audiences, invited talks, and natural views. We define three events: event 1 – “full-swing”, event 2 – “non-full-swing”, and event 3 – “irrelevant event”. “Full-swing” event is defined as the golf swing that produces long and straight shots with full-swings. “Non-full-swing” event is defined as the soft hit such as fairway shots and bunker shots, and generally the ball does not fly very high for such cases. The irrelevant event includes

all other scenes such as the invited talk, the scene of the audience, etc.

The one hour golf video is automatically segmented to video shots and then video shots are manually grouped and annotated as video events. Therefore, the event boundaries are assumed to be aligned with shot boundaries but each event may contain one or multiple video shots according to our definition. In this paper, we do not address the event segmentation issue. The experimental results are only used to verify the effectiveness of the new statistical model in learning the training data, and detecting or classifying the new data by exploring both the spatial and temporal features

We choose three video sequences (Training sequence 1, 2, and 3) which contain different events as our training data. The goal is to detect which event an unlabeled video sequence might belong to. The training results are shown in Table I. The log-likelihood of the training sequences for each model/event are listed in the second column in Table I. After the training, we use the rest of golf video data as the test sequence to verify the proposed event detection algorithms. The test sequence is also manually annotated and classified into one of the three events. The ground truth is shown in Table II. The event detection results are shown in Table III. To evaluate the performance of the proposed event detection algorithm, we introduce a detection rate

$$D = \frac{N_{correct}}{N_{total}}, \quad (59)$$

where  $N_{correct}$  is the number of correctly detected events, and  $N_{total}$  is the number of total events. The overall detection rate for the proposed framework is 70.79%.

After manually annotating the video, we classify each video shot into one of the three pre-defined events and then obtain a list of video shots with event labels as ground truth. Each event is supposed to have one or multiple video shots. Then based on selected training data, we learn three different models representing the three pre-defined events. For the test data, we use video shots as boundaries and we select a video segment with a length of three video shots each time. The learnt models will be applied to the video segments to calculate the likelihood values. The results are only applied to the middle video shot and based on maximum likelihood the video shot is classified to its corresponding event. After this calculation, we move forward one video shot to get the next video segment for classification of next middle video shot.

For golf videos, scenes with uniform distributions such as sky scenes and golf court scenes are very common. Those cases generally tend to generate errors in motion estimation algorithms since low frequency contents are dominant in video frames. Golf videos are also different from other games such as basketball videos in that motions either from players or from the ball tracking are often less intensive. Camera panning or slow zooming on relatively static scenes is also very common. However, we observe that in golf videos the patterns of camera movements from one scene to another seem to provide a good cue to identify the events. So, in our experimental results, we only apply the color histogram as the features. It is also possible to include more features, such as motion or even segmentation results to extend this analysis. The use of ICA in the feature extraction step will make the fusion of multiple features from different domains much easier since ICA filtering can automatically reduce the dimension and transform the feature space to ICA subspace and re-represent the features as the statistically independent components. Also note that in the feature extraction step, our selected features can be run in real-time since we only choose color histograms as the main features. Using more complex cues such as motion may not be as computationally efficient.

The temporal information, i.e., the state sequences learnt for the pre-defined three events are plotted in Fig. 10, Fig. 11 and Fig. 12. The distributions of video frames in ICA subspace are plotted in Figs. 13-18. Note that we also manually add arrows to show the temporal information and how video samples (feature vectors) move in the ICA subspace. In event detection step, both the features in the ICA subspace and the temporal information will be used to identify the event based on the trained model. It can be seen that the shapes and patterns are very similar. The temporal characteristics are captured by HMM models.

The detection results measured in confusion matrix is shown in Table IV. In the event detection step, the problem is essentially equivalently to classifying each video shot in one of the three pre-defined events. From the confusion matrix, we can see that many event 1 candidates are mis-classified to event 2, and some event 2 candidates are also mis-classified to event 1. We did some failure analysis and found that the errors are caused by the following factors: features, the definition of the events, and the “noise”. The reason

we choose normalized color histogram is because we want to eliminate the effect of the illumination changes and make maximum use of the color cues for golf video. In some cases, it is difficult to distinguish a full-swing from a regular shot because for some full-swing shots the camera did not track the ball in close-up, so the model learnt from golf court – camera panning to track the ball in the sky – ball falling down cannot capture those full-swings in long camera shot. Also, in some regular shots, there are some quick camera movements that confused the models and thus some regular shots are also mis-detected as full-swing shots. Possible improvements includes adding more features such as motion, but as analyzed earlier, motion is also likely to introduce noise in the feature space. For those failure cases the difference between a full-swing and a regular shot is less recognizable.

TABLE I  
LOG-LIKELIHOOD FOR THE TRAINING SEQUENCES.

	Log-likelihood for the trained model
Training sequence 1 for event 1	2369.1
Training sequence 2 for event 2	-396.06
Training sequence 3 for event 3	7914.3

TABLE II  
GROUND TRUTH FOR THE TEST SEQUENCES.

	The number of events
Event 1	54
Event 2	132
Event 3	16
Total events	202

TABLE III  
EVENT DETECTION RESULTS.

	The number of events
Event 1	51
Event 2	132
Event 3	19
Total events	202

TABLE IV  
EVENT DETECTION RESULTS MEASURED IN CONFUSION MATRIX.

	Event 1	Event 2	Event 3
Event 1	26	27	1
Event 2	23	104	5
Event 3	2	1	13

## VI. CONCLUSION

In this paper, a new video content analysis framework is presented based on HMM and ICA mixture model. The new framework combines ICA mixture model and HMM. The observation densities of HMM are presented by non-Gaussian mixtures and each non-Gaussian mixture density is learned by ICA. The new framework extended the classical continuous HMM using Gaussian mixtures, thus allows the approximations of a larger range of distributions. The spatial statistics are explored by ICA mixture model and the temporal characteristics are modeled by HMM transitions. Note that ICA is used twice in our video detection algorithm. First, ICA is applied on video frames and the ICA coefficients are used to form a compact feature subspace. Secondly, ICA mixture model is used as a density estimation method for parametric form representations of the distributions since ICA is an appropriate model to estimate the non-Gaussian source densities. We verify the proposed framework in a supervised learning to detect semantic video events.

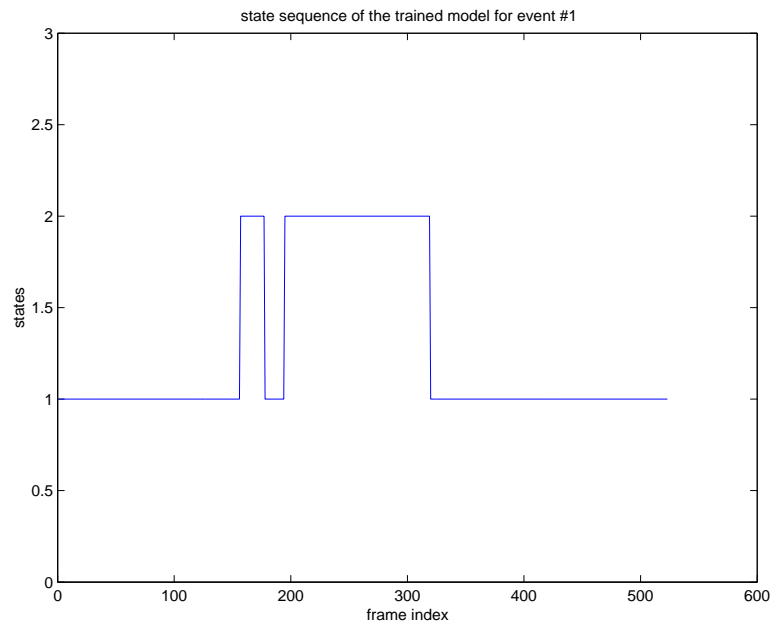


Fig. 10. State sequence for training sequence 1.

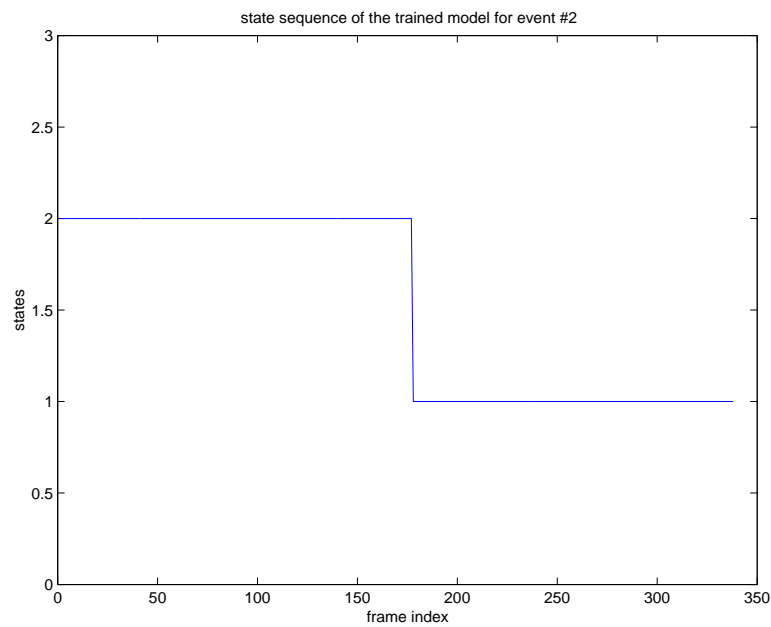


Fig. 11. State sequence for training sequence 2.

One HMM model is trained for each event. The model that gives the maximum likelihood for the test sequence is considered as the detected event. The experimental results show

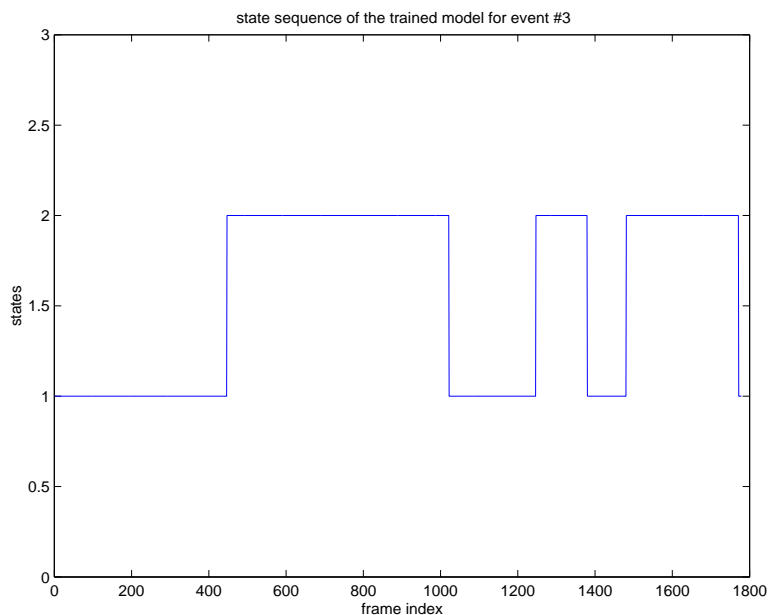


Fig. 12. State sequence for training sequence 3.

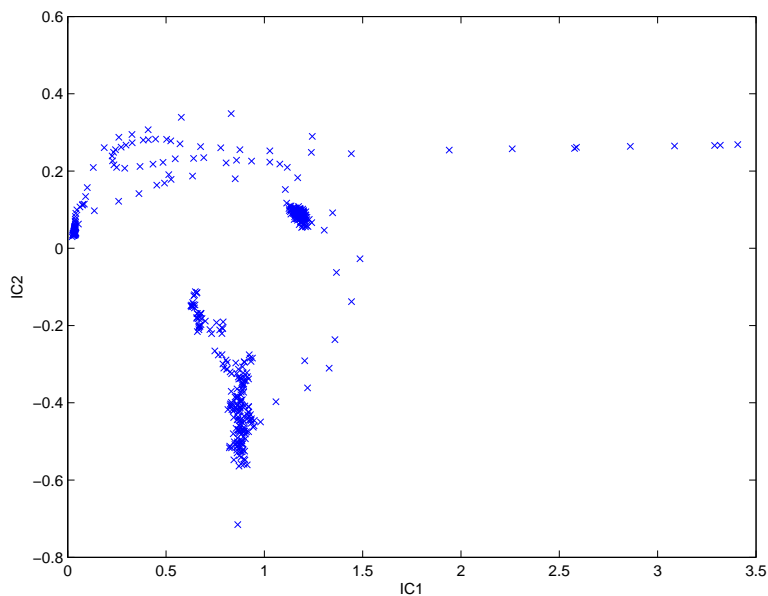


Fig. 13. ICA feature space trajectory of the training video sequence 1 (for event 1).

that the presented method can effectively detect and recognize the recurrent patterns in our golf video data. Also, the presented new ICAMHMM framework is general and can be applied to sequential data analysis in other video content analysis applications.

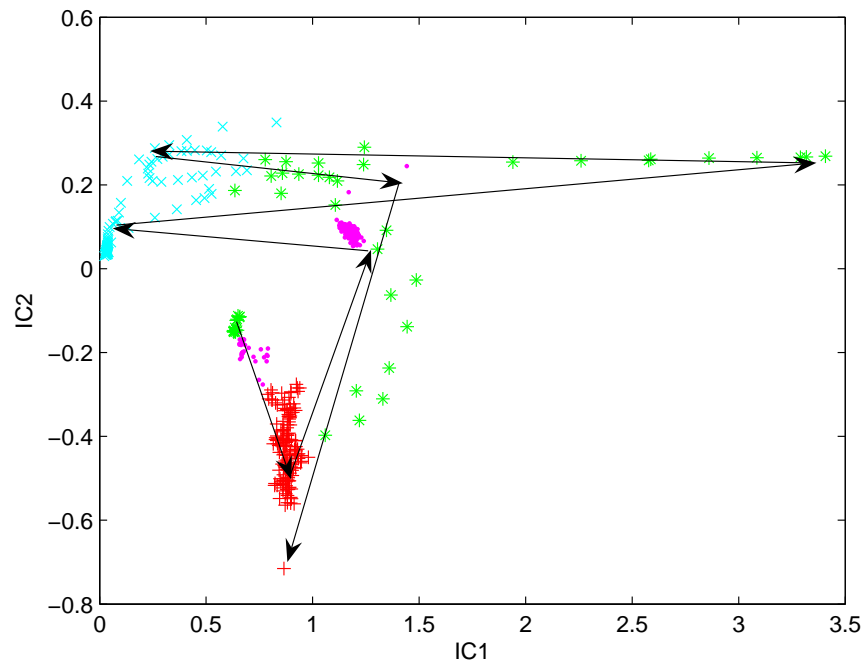


Fig. 14. Four classes are learned for training video sequence 1 in the ICA feature space.

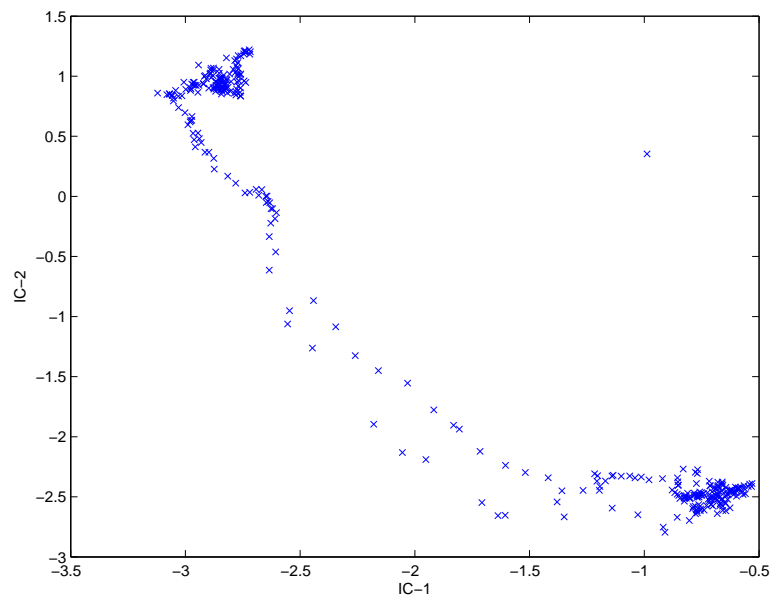


Fig. 15. ICA feature space trajectory of the training video sequence 2 (for event 2).

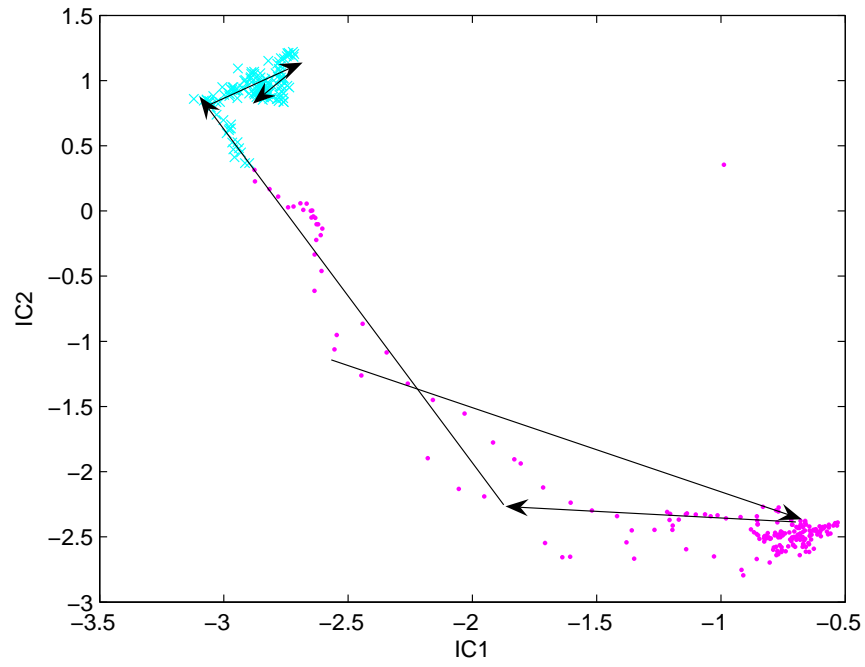


Fig. 16. Two classes are learned for training video sequence 2 in the ICA feature space.

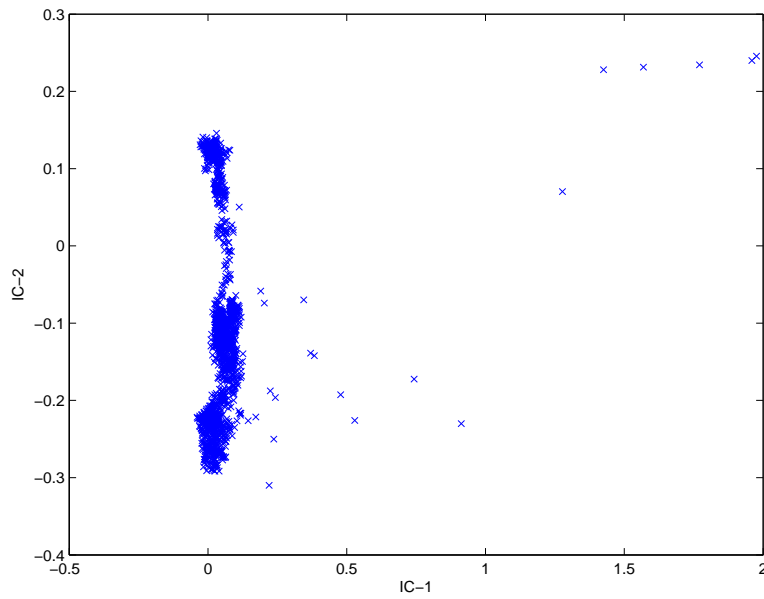


Fig. 17. ICA feature space trajectory of the training sequence 3 (for event 3).

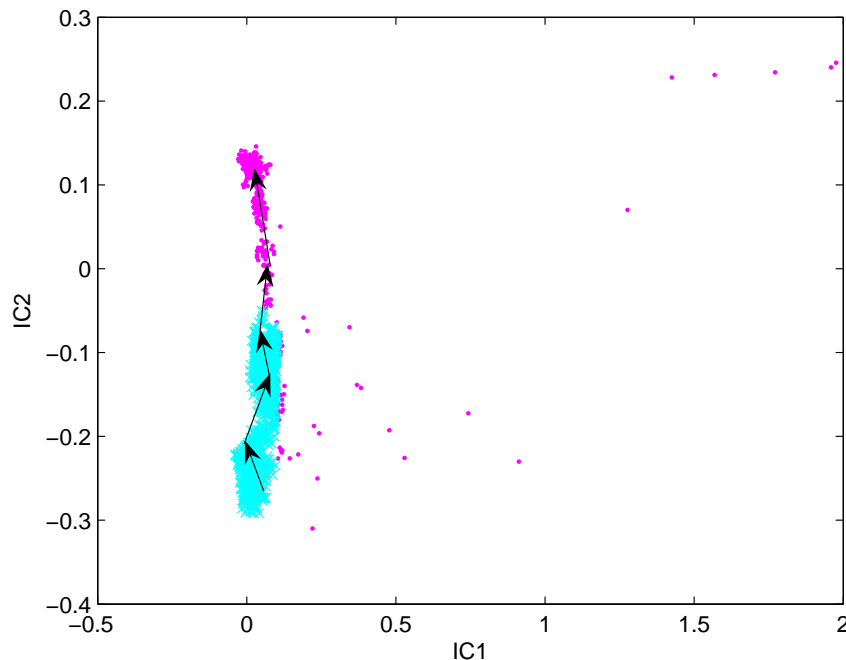


Fig. 18. Two classes are learned for training sequence 3 in the ICA feature space.

## REFERENCES

- [1] H.J. Zhang, A. Kankanhalli, and S. W. Smoliar, “Automatic partitioning of full-motion video,” *Multimedia Systems*, vol. 1, no. 1, pp. 10–28, June 1993.
- [2] B. Yeo and B. Liu, “Rapid scene analysis on compressed video,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 533–544, Dec. 1995.
- [3] A. Hanjalic, “Shot-boundary detection: Unraveled and resolved?,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 90–104, Feb. 2002.
- [4] N. Dimitrova and F. Golshani, “Motion recovery for video content classification,” *ACM Transactions on Information Systems*, vol. 13, no. 4, pp. 408–439, Oct. 1995.
- [5] S.-F. Chang, W. Chen, H.J. Meng, H. Sundaram, and D. Zhang, “A fully automated content-based video search engine supporting spatiotemporal queries,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 5, pp. 602–615, Sept. 1998.
- [6] E. Sahouria and A. Zakhor, “Content analysis of video using principal components,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1290–1298, Dec. 1999.
- [7] L. Xie, P. Xu, S.-F. Chang, A. Divakaran, and H. Sun, “Structure analysis of soccer video with domain knowledge and hidden Markov models,” *Pattern Recognition Letters*, vol. 25, no. 7, pp. 767–775, May 2004.
- [8] Z. Liu, J. Huang, and Y. Wang, “Classification of TV programs based on audio information using hidden Markov model,” in *Proc. of 1998 IEEE Second Workshop on Multimedia Signal Processing (MMSP’98)*, Redonda Beach, CA, Dec. 1998, pp. 27–31.
- [9] Z. Xiong, R. Radhakrishnan, A. Divakaran, and T.S. Huang, “Audio events detection based highlights

- extraction from baseball, golf and soccer games in a unified framework,” in *Proc. of 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Hong Kong, China, Apr. 2003, pp. V – 632–5.
- [10] M.R. Naphade and T.S. Huang, “Extracting semantics from audiovisual content: the final frontier in multimedia retrieval,” *IEEE Transactions on Neural Networks*, vol. 13, no. 4, pp. 793–810, July 2002.
- [11] T.-W. Lee and M.S. Lewicki, “Unsupervised image classification, segmentation, and enhancement using ICA mixture models,” *IEEE Trans. on Image Processing*, vol. 11, no. 3, pp. 270–279, Mar. 2002.
- [12] L.R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
- [13] L.A. Liporace, “Maximum likelihood estimation for multivariate observations of Markov sources,” *IEEE Trans. Inform. Theory*, vol. 28, no. 5, pp. 729–734, Sept. 1982.
- [14] B.-H. Juang, S. Levinson S., and M. Sondhi, “Maximum likelihood estimation for multivariate mixture observations of Markov chains,” *IEEE Trans. Inform. Theory*, vol. 32, no. 2, pp. 307–309, Mar. 1986.
- [15] M. Girolami, “An alternative perspective on adaptive independent component analysis algorithms,” *Neural Computation*, vol. 10, no. 8, pp. 2103–2114, 1998.
- [16] T.-W. Lee, M. Girolami, and T. Sejnowski, “Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources,” *Neural Computation*, vol. 11, no. 2, pp. 609–633, 1999.
- [17] L.E. Baum and G.R. Sell, “Growth functions for transformations on manifolds,” *Pac. J. Math.*, vol. 27, no. 2, pp. 211–227, 1968.
- [18] M. A. T. Figueiredo and A. K. Jain, “Unsupervised learning of finite mixture models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, Mar. 2002.
- [19] M. H. C. Law, M. A. T. Figueiredo, and A. K. Jain, “Simultaneous feature selection and clustering using mixture models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1154–1166, Sept. 2004.
- [20] M.S. Drew, J. Wei, and Z.-N. Li, “Illumination-invariant color object recognition via compressed chromaticity histograms of color-channel-normalize images,” in *ICCV’98*, 1998, pp. 533–540.
- [21] J. Zhou and X.-P. Zhang, “Video shot boundary detection using independent component analysis,” in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, Philadelphia, PA, USA, Mar. 2005.