

# Activity Based Surveillance Video Content Modelling

Tao Xiang\* and Shaogang Gong

Department of Computer Science

Queen Mary, University of London, London E1 4NS, UK

{txiang,sgg}@dcs.qmul.ac.uk

## Abstract

This paper tackles the problem of surveillance video content modelling. Given a set of surveillance videos, the aims of our work are two-fold: firstly a continuous video is segmented according to the activities captured in the video; secondly a model is constructed for the video content, based on which an unseen activity pattern can be recognised and any unusual activities can be detected. To segment a video based on activity, we propose a semantic meaningful video content representation method and two segmentation algorithms, one being offline offering high accuracy in segmentation, and the other being online enabling real-time performance. Our video content representation method is based on automatically detected visual events (i.e. ‘what is happening in the scene’). This is in contrast to most previous approaches which represent video content at the signal level using image features such as colour, motion and texture. Our segmentation algorithms are based on detecting breakpoints on a high-dimensional video content trajectory. This differs from most previous approaches which are based on shot change detection and shot grouping. Having segmented continuous surveillance videos based on activity, the activity patterns contained in the video segments are grouped into activity classes and a composite video content model is constructed which is capable of generalising from a small training set to accommodate variations in unseen activity patterns. A run-time accumulative unusual activity measure is introduced to detect unusual behaviour while usual activity patterns are recognised based on an online Likelihood Ratio Test (LRT) method. This ensures robust and reliable activity recognition and unusual activity detection at the shortest possible time once sufficient visual evidence has become available. Comparative experiments have been carried out using over 10 hours of challenging outdoor surveillance video footages to evaluate the proposed segmentation algorithms and modelling approach.

**Keywords:** Video content analysis, activity recognition, surveillance video segmentation, Dynamic Bayesian Networks, dynamic scene modelling, unusual activity detection.

---

\*Corresponding author. Tel: (+44)-(0)20-7882-8020; Fax: (+44)-(0)20-8980-6533

# 1 Introduction

The rapid increase in the amount of CCTV surveillance video data generated has led to the urgent demand for automated analysis of video content. Content analysis for surveillance videos is more challenging than that for broadcasting videos such as news and sports programmes because the latter are more constrained, well-structured, and of better quality. This paper aims to address a number of key issues of surveillance video content analysis:

1. How to construct a representation of video content which is informative, concise, and able to bridge the gap between the low level visual features embedded in the video data and the high level semantic concepts used by human to describe the video content.
2. How to segment a continuous surveillance video temporally into activity patterns according to changes in video content.
3. Given a training video dataset, how to construct a model for the video content which can accommodate the variations in the unseen activity patterns both in terms of duration and temporal ordering?
4. Given a video content model and an unseen video, how to perform online activity recognition and unusual activity detection?

To this end, we first propose in this paper a semantic meaningful representation based on automatically detected discrete visual events in a video and two segmentation algorithms. One of the proposed algorithms is offline offering better accuracy in segmentation but is computationally more demanding, while the other is online enabling real-time performance. We then develop a generative video content model based on unsupervised learning. Using this model an unseen activity pattern can be recognised into different classes if similar patterns are included in the training dataset and unusual activities can also be detected.

## 1.1 Video Segmentation

For activity based video segmentation, we propose to represent surveillance video content holistically in space and over time based on visual events detected automatically in the scene. This is in contrast to most previous approaches which represent video content at the signal level using image features such as colour, motion and texture [19, 28, 31, 1, 26, 37]. The essence of our method is to represent video content based on ‘what is happening’ rather than ‘what is present’ in the scene. ‘What is happening’ in the scene is reflected

through activities captured in the video which most likely involve multiple objects interacting or co-existing in a shared common space. An activity is composed of groups of co-occurring events which are defined as significant visual changes detected in image frames *over time*. Events are detected and classified by unsupervised clustering using Gaussian Mixture Model (GMM) with automatic model selection based on Schwarz's Bayesian Information Criterion (BIC) [32, 42]. Video content is then represented as temporally correlated events automatically labelled into different classes. By doing so changes in the presence of and temporal correlations among different classes of events can indicate video content changes therefore providing vital cues for activity based video segmentation.

Video segmentation has been studied extensively in the past two decades. Traditionally, a four-layer hierarchical structure is adopted for video structure analysis which consists of a frame layer, a shot layer, a scene layer and a video layer [1]. At the bottom of the structure, continuous image frames taken by a single camera are grouped into shots. A series of related shots are then grouped into a scene. Shot change detection is performed as the first step for video segmentation by most previous approaches [19, 28, 31, 1]. The segmentation of shots and scenes heavily relies on a well-defined feature space usually dominated by colour and motion. For example, in [19], image frames were represented using a holistic colour histogram and the frame difference was exploited to detect shots. This structure is in general valid for constrained, well-structured broadcast videos of news and sports programmes. However, for a surveillance video which is taken continuously by a fixed camera *without* script-driven panning and zooming, global colour and motion information is either highly unreliable or unavailable [11]. More importantly, there is only one shot in a surveillance video and any shot-change detection based segmentation approach would be unsuitable.

Recently, Dementhon et al. [5] proposed to represent a video as a high dimensional temporal trajectory based on colour histogram and treat video segmentation as a trajectory breakpoint detection problem. Compared to the thresholding based segmentation algorithms adopted by most previous video segmentation approaches [19, 28], a trajectory breakpoint detection based approach is more robust to local noise at individual frames because segmentation is performed holistically over the whole duration of the video. Various approaches have been proposed to segment a continuous trajectory into segments through breakpoint detection for time-series data segmentation [20, 23, 14]. Most of them are based on Piecewise Linear Approximation (PLA) or probabilistic graphical models such as Hidden Markov Models (HMMs) [9]. PLA refers to finding the best approximation of a trajectory using straight lines by either Linear Interpolation or Linear Regression. However, the computational cost of PLA is nontrivial especially when the dimensionality of the trajectory space is high [14], resulting in most of the existing PLA segmentation algorithms only

being applied to trajectories in a space with a dimensionality no bigger than 3. On the other hand, HMMs have the potential to be robust to noise and are capable of dynamic time warping. However, a large number of parameters are needed to describe a HMM when the dimensionality of the trajectory space is high. This makes a HMM vulnerable to over-fitting when training data are insufficient. To solve this problem, we propose a Multi-Observation Hidden Markov Model (MOHMM) which requires less parameters compared to a conventional HMM. It is thus more suitable for high dimensional video content trajectory segmentation.

Most existing segmentation algorithms are offline. For the purpose of video segmentation, an online algorithm has its distinctive advantage due to the huge amount of surveillance video data to be processed and more importantly the real-time nature of some surveillance applications. For instance, if a video content change is detected online and in real-time, it can be used for alerting CCTV control room operators to act accordingly. One of the most popular online segmentation algorithms for temporal trajectories is the Sliding Window (SW) algorithm based on the Sliding Window principle [21] and local Piecewise Linear Approximation. However, the Sliding Window algorithm tends to over-segment [20, 21, 33]. One possible explanation is that it lacks a global view of the data since it only ‘looks backward’ without ‘looking forward’. Keogh et al. [20] attempted to solve the problem by combining the bottom-up offline algorithm [21] with the Sliding Window principle. Nevertheless, their algorithm only works on trajectories with very short segments. It is thus impossible for the algorithm to run in real-time on a typical surveillance video sequence which comprises segments lasting over hours. In this paper, we propose a novel Forward-Backward Relevance (FBR) algorithm. Compared to a conventional Sliding Window algorithm, FBR is less sensitive to noise and more importantly, can be run in real-time.

## **1.2 Video Content Modelling for Activity Recognition and Unusual Activity detection**

Using the proposed segmentation algorithms, continuously recorded video or online CCTV input can be segmented into activity patterns. Given these activity patterns, the goal of video content modelling is to learn a model that is capable of detecting unusual activity patterns whilst recognising novel instances of expected usual activity patterns. In this context, we define an unusual activity as an atypical activity pattern that is not represented by sufficient samples in a training dataset but critically it satisfies the specificity constraint to an unusual pattern. This is because one of the main challenges for the model is to differentiate unusual activity from outliers caused by noisy visual features used for activity representation. The effectiveness of an video content modelling approach shall be measured by (1) how well unusual activities can be detected (i.e. measuring specificity to expected patterns of activity) and (2) how accurately and robustly different

classes of usual activity patterns can be recognised (i.e. maximising between-class discrimination).

To solve the problem, we develop a novel framework for fully unsupervised video content modelling and online unusual activity detection. Our framework has the following key components:

1. Discovering natural groupings of activity patterns using a activity affinity matrix. A number of affinity matrix based clustering techniques have been proposed recently [38, 35, 43]. However, these approaches require known number of clusters. Given an unlabelled dataset, the number of activity classes are unknown in our case. To automatically determine the number of clusters, a recently proposed spectral clustering algorithm [40] is deployed.
2. A composite generative video content model using a mixture of DBNs. The advantages of the such a generative video content model are two-fold: (a) It can accommodate well the variations in the unseen and usual activity patterns both in terms of duration and temporal ordering by generalising from a training set of limited number of samples. This is important because in reality the same usual activity can be executed in many different usual ways. These variations cannot possibly be captured in a limited training dataset and need to be dealt with by a learned video content model. (b) Such a model is robust to errors in activity representation. This is because that a mixture of DBNs can cope with errors occurred at individual frames and is also able to distinguish an error corrupted usual activity pattern from an unusual one.
3. Online unusual activity detection using a run-time accumulative unusual activity measure and usual activity recognition using an online Likelihood Ratio Test (LRT) method. A run-time accumulative measure is introduced to determine how usual/unusual an unseen activity pattern is on-the-fly. The activity pattern is then recognised as one of the usual activity classes if detected as being usual. Normal activity recognition is carried out using an online LRT method which holds the decision on recognition until sufficient visual features have become available. This is in order to overcome any ambiguity among different activity classes observed online due to insufficient visual evidence at a given time instance. By doing so, robust activity recognition and unusual activity detection are ensured at the shortest possible time, as opposed to previous work such as [4, 11, 29] which requires completed activity patterns being observed. Our online LRT based activity recognition approach is also advantageous over previous ones based on the *Maximum Likelihood* (ML) method [44, 11, 29]. A ML based approach makes a forced decision on activity recognition at each time instance without considering the reliability and sufficiency of the accumulated visual evidence. Consequently, it can

be error prone.

The rest of the paper is structured as follows: in Section 2, we describe an event based surveillance video content representation approach. In Section 3, we address the problem of surveillance video segmentation. Two novel segmentation algorithms, MOHMM and FBR are introduced in Sections 3.1 and 3.2 respectively. Comparative experiments are conducted using over 10 hours of challenging outdoor surveillance video footages and the results are presented in Section 3.3. The pros and cons of both algorithms are analysed. The advantage of our event based video content representation over the traditional image feature based representation is also made clear by our experiment results. The proposed video content modelling approach is described in Section 4, where experiments are also presented to evaluate the effectiveness and robustness of the approach. A conclusion is drawn in Section 5.

## 2 Semantic Video Content Representation

We consider an activity based video content representation. Visual events are detected and classified automatically in the scene. The semantics of video content are considered to be best encoded in the occurrence of such events and the temporal correlations among them.

### 2.1 An Example Scenario

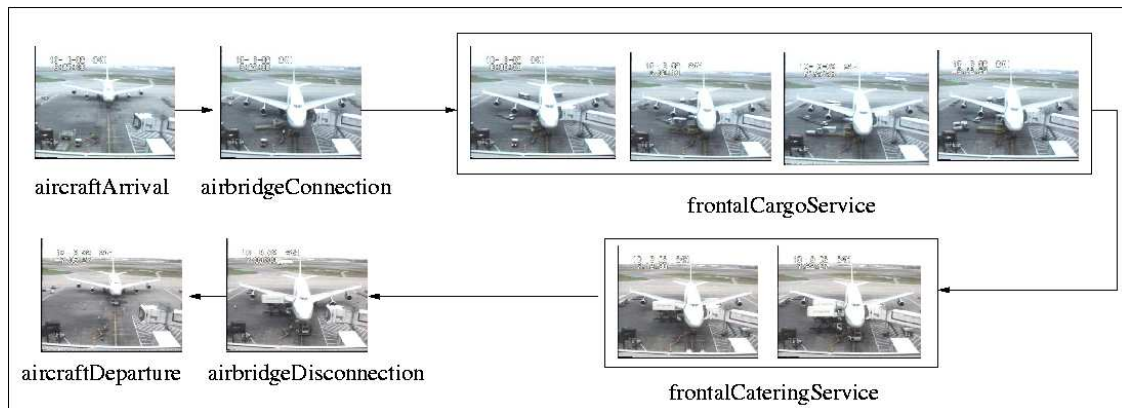


Figure 1: An example structure of a complete aircraft docking video. Representative frames of different activities occurred in the scene are also shown.

For clarity and concreteness, we shall illustrate our approach using surveillance videos monitoring aircraft docking operations at an airport. Around 15 different activities take place in the scene. Among them, only six activities are visually detectable. Figure 1 shows an example activity based structure of a complete

aircraft docking operation. One activity can be followed by either another activity immediately or a period of inactivity when no meaningful visual changes can be detected. The durations of activities and inactivity gaps vary hugely during a single aircraft docking operation and across different operations. There are also variations in the temporal order of activities and inactivity gaps. For example, the temporal order of `frontalCargoService` and `frontalCateringService` appears to be arbitrary.

It is noted that some activities such as `airCraftArrival` involve the movement of a single object while other activities such as `frontalCargoService` and `frontalCateringService` consist of the movement of multiple objects which may leave and re-appear in the scene. For the latter case, there often exist a number of short inactivity break-ups within an activity which are different from a long inactivity gap between two activities only by their durations. All these characteristics make analysis of surveillance video content very challenging.

## **2.2 Detecting and Classifying Visual Events**

We define events as significant scene changes characterised by the location, shape and direction of the changes. They are object-independent and location specific. We also consider that these events are autonomous, meaning that both the number of these events and their whereabouts in a scene are determined automatically bottom-up without top-down manual labelling using predefined hypotheses.

### **2.2.1 Seeding Event: Measuring Pixel Change History**

Adaptive mixture background models are commonly used to memorise and maintain the background pixel distribution of a dynamic scene [25, 27, 36]. The major strength of such a model is its potential to cope with persistent movements of background objects such as waving tree leaves given appropriate model parameter setting. However, an adaptive mixture background model cannot differentiate, although may still be able to detect the presence of, pixel-level changes of different temporal scales. In general, a pixel-level change of different temporal scales can have different significance in its semantics:

1. A short term change is most likely to be caused by instant moving objects (e.g. passing-by people or vehicles).
2. A medium term change is most likely to be caused by the localised moving objects (e.g. a group of people standing and talking to each other).

3. A long term change is most likely to be caused by either the introduction of novel static objects into the scene, or the removal of existing objects from the scene (e.g a piece of furniture is moved in the background or a car is parked in a carpark).

We seek a single, unified multi-scale temporal representation that can capture and differentiate changes of such different rates/scales at the pixel level. Temporal wavelets were adopted for such a multi-scale analysis [34]. However, the computational cost for multi-scale temporal wavelets at the pixel level is very expensive. They are therefore unsuitable for real-time performance. Alternatively, Motion History Image (MHI) is less expensive to compute by keeping a history of temporal changes at each pixel location which then decays over time. MHI has been used to build holistic motion templates for the recognition of human movements [3] and moving object tracking [30]. An advantage of MHI is that although it is a representation of the history of pixel-level changes, only one previous frame needs to be stored. However, at each pixel location, explicit information about its past is also lost in MHI when current change are updated to the model with their corresponding MHI values ‘jumping’ to the maximal value. To overcome this problem, Pixel Signal Energy was introduced to measure the mean magnitude of pixel-level temporal energy over a period of time defined by a backward window [27]. The size of the backward window determines the number of frames (history) to be stored. However, this approach suffers from its sensitivity to noise and also being expensive to compute.

Here we propose to use Pixel Change History (PCH) [41] for measuring multi-scale temporal changes at each pixel. The PCH of a pixel is defined as:

$$P_{\varsigma,\tau}(x, y, t) = \begin{cases} \min\left(P_{\varsigma,\tau}(x, y, t-1) + \frac{255}{\varsigma}, 255\right) & \text{if } D(x, y, t) = 1 \\ \max\left(P_{\varsigma,\tau}(x, y, t-1) - \frac{255}{\tau}, 0\right) & \text{otherwise} \end{cases} \quad (1)$$

where  $P_{\varsigma,\tau}(x, y, t)$  is the PCH for a pixel at  $(x, y)$ ,  $D(x, y, t)$  is a binary image indicating the foreground region,  $\varsigma$  is an accumulation factor and  $\tau$  is a decay factor. When  $D(x, y, t) = 1$ , instead of jumping to the maximum value, the value of a PCH increases gradually according to the accumulation factor. When no significant pixel-level visual change is detected at a particular location  $(x, y)$  in the current frame, pixel  $(x, y)$  will be treated as part of the background and the corresponding pixel change history starts to decay. The speed of decay is controlled by a decay factor  $\varsigma$ . The accumulation factor and the decay factor give us the flexibility of characterising pixel-level changes over time. In particular, large values of  $\varsigma$  and  $\tau$  imply that the history of visual change at  $(x, y)$  is considered over a longer backward temporal window. In the meantime, the ratio between  $\varsigma$  and  $\tau$  determines how much weight is put on the recent change.

If the binary image  $D(x, y, t)$  in Equation (1) is determined by the temporal difference between the current frame and the dynamic background maintained by an adaptive mixture model, a PCH based foreground model can be introduced to detect the medium and long term pixel changes. Specifically, we detect those pixels that are associated with medium term changes by the following condition:

$$|I(x, y, t) - I(x, y, t - 1)| > T_M \quad (2)$$

where  $T_M$  is a threshold. Pixel level changes that do not satisfy the above condition are caused by long term changes such as the introduction of static novel objects into the scene or the removal of existing objects from the scene. Note that Equation 2 is used for distinguishing the detected foreground pixels according to the nature of the visual changes. It is not used for detecting foreground pixels using background subtraction

### 2.2.2 From Pixel Groups to Unsupervised Clustering and Classification of Events

Given detected pixel changes in each image frame, we aim to form discrete events. The connected component method is adopted to group those changed pixels. Small groups are then removed by a size filter and the rest groups with an average PCH (of the PCHs for all the pixels within each group) larger than a threshold  $T_B$  are referred to as salient pixel groups and considered as events. An event is represented by a 7-dimensional feature vector

$$\mathbf{v} = [\bar{x}, \bar{y}, w, h, R_m, M_px, M_py] \quad (3)$$

where  $(\bar{x}, \bar{y})$  is the centroid of the salient pixel group,  $(w, h)$  are the width and height of the salient pixel group,  $R_m$  represents the percentage of those pixels in the group that satisfy Condition (2), and  $(M_px, M_py)$  are a pair of first order moments of the PCH image within the salient pixel group. Among these features,  $(\bar{x}, \bar{y})$  are location features,  $(w, h)$  are shape features,  $R_m$  is visual change type feature and  $(M_px, M_py)$  are motion features capturing the direction of object motion direction<sup>1</sup>. Note that in our approach, salient pixel groups are defined within each image frame. Alternatively, salient groups can be defined in a spatio-temporal volume which could in theory lead to better clustering. One could adopt a method such as the one proposed by Greenspan et. al [12]. Alternatively, we have also developed an approach for salient event detection over a spatio-temporal volume using multi-scale entropy ratio over space and time presented elsewhere [18]. However, such a spatio-temporal volume based events detection and recognition approach is always computationally expensive, and may not be tractable given the complexity of the activities captured in video

<sup>1</sup>Similar to the MHI (see [3]), PCH implicitly represents the direction of movement. First order moments based on PCH value distribution within the bounding box is thus capable of measuring the direction of movement quantitatively.

footages. In order to achieve real-time performance, we decided to make a compromise between the speed and performance of the event detection and recognition algorithm by defining the salient groups within each image frame.

Salient pixel groups are clustered and classified unsupervised into different events in the 7-D feature space using a Gaussian Mixture Model (GMM). The GMM is estimated using Expectation-Maximisation (EM) [2] and the model order of the GMM is determined using the Bayesian Information Criterion (BIC) [32]. An example of event detection and classification is shown in Figure 2.

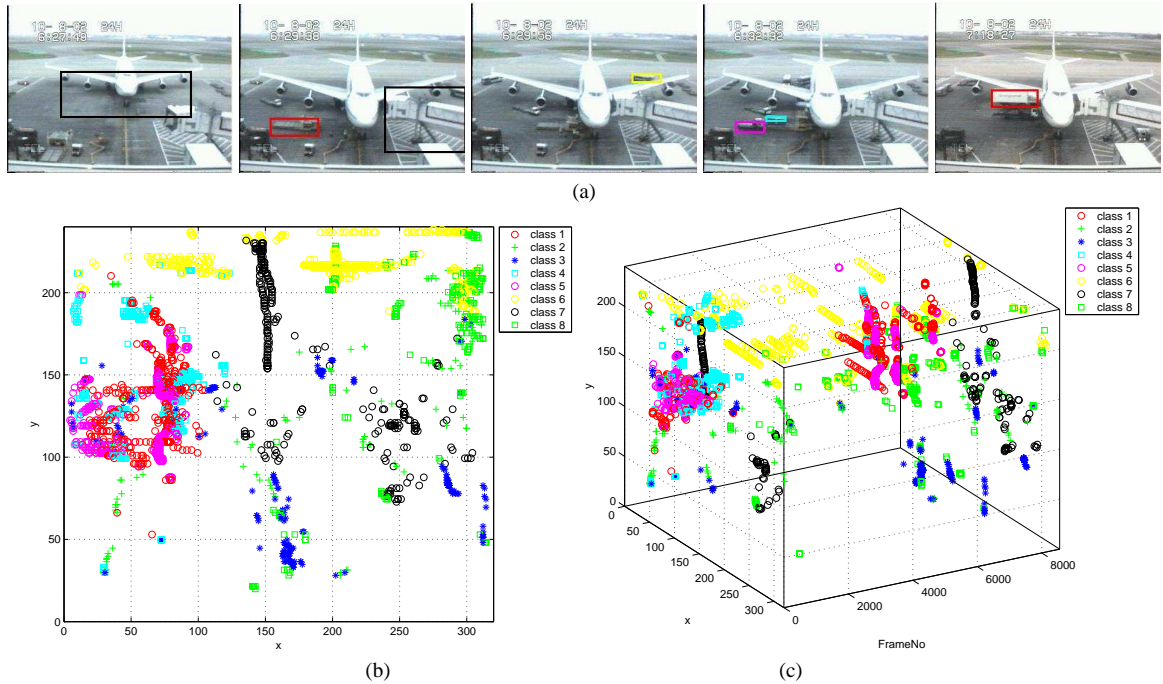


Figure 2: Event detection and classification in an aircraft docking scene video. Eight classes of events are detected automatically, each of which corresponds to different movement patterns in the image frame. For example, event class 7 corresponds to the movements of the aircraft. Event classes 4, 5, and 6 correspond to the movements of objects involved in activities `frontalCargoService` and `frontalCateringService`. (a) Events are detected and classified into different classes which are highlighted using bounding boxes in different colours. The spatial and temporal distribution of events of different classes are illustrated in (b) and (c) respectively, with centroids of different classes of events depicted using different colours.

### 2.3 Constructing a Scene Vector

Classified events can be considered as ‘snapshots’ of activities captured in a scene. To hide the image feature information and focus on the semantic video content, a scene vector is constructed for each image frame of

a video. A scene vector  $\mathbf{sv}_t$  for a video frame  $t$  is defined as:

$$\mathbf{sv}_t = [s_t^1, \dots, s_t^k, \dots, s_t^{K_e}] \quad (4)$$

where  $K_e$  is the number of event classes automatically determined using BIC. The value of  $s_t^k$  is the number of events of the  $k^{th}$  event class detected in the frame  $t$ .

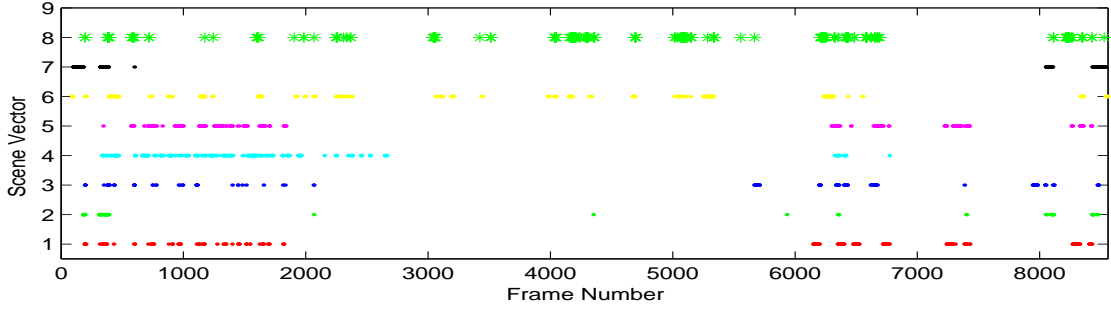


Figure 3: The example video shown in Figure 2 is represented by scene vectors evolving over time. We have  $K_e = 8$  for this video.  $s_t^k$  is depicted by a dot in colour when  $s_t^k > 0$ . There are frequent but short inactivity break-ups within activities (see between frame 1 and frame 2000) and a long inactivity gaps between activities (between frame 3000 and frame 6000).

A scene vector gives a description of ‘what is happening’ in the scene through the class labels of the detected events. It is thus a concise representation of the video content at the semantic level (see Figure 3 for an example). However, directly using this scene vector to detect video content changes can cause problems for video segmentation. Specifically, the value of a scene vector  $\mathbf{sv}_t$  can become  $\mathbf{0}$  (i.e. absent of any event at a given frame) frequently throughout the video sequence (see Figure 3). This can be caused by either frequent but short inactivity break-ups within activities or long inactivities between activities. Each ‘coming to zero’ is reflected as a dramatic change in the value of  $\mathbf{sv}_t$  due to the discrete nature of  $s_t^k$ . Those changes that correspond to real changes of video content can thus easily be overwhelmed by changes caused by the inactivity break-ups within activities, which makes temporal segmentation of the video difficult.

## 2.4 Representing Video Content Over Time

To overcome this problem, let us now consider representing the video content using a cumulative scene vector computed at frame  $t$  using  $\mathbf{sv}_t$  from frame 1 to the frame  $t$ . More specifically, the  $k^{th}$  element of the cumulative scene vector (denoted as  $\tilde{\mathbf{sv}}_t$ ) is computed as:

$$\tilde{s}_t^k = \sum_{i=1}^t s_i^k \quad (5)$$

The value of each element of  $\widetilde{\mathbf{sv}}_t$  will increase monotonically with time (see Figure 4). Compared to the scene vector representation  $\mathbf{sv}_t$  (see Figure 3), the short inactivity break-ups at individual frames have little impact on the values of the cumulative scene vector evolved over time. It thus becomes easier to detect breakpoints that correspond to significant changes in video content.

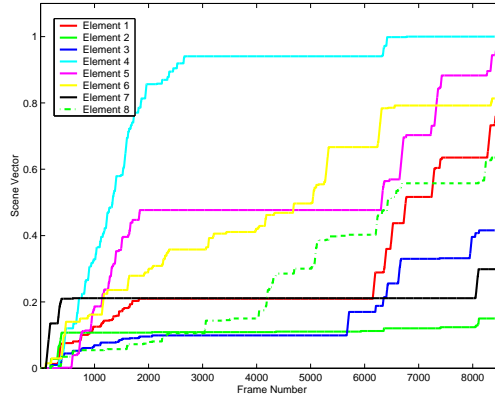


Figure 4: The 8 elements of  $\widetilde{\mathbf{sv}}_t$  over time for the example video shown in Figure 2. Each element of  $\widetilde{\mathbf{sv}}_t$  is normalised to have a value range of  $[0, 1]$ .

### 3 Temporal Segmentation of Surveillance Videos

It has been shown in the preceding section that a cumulative scene vector can represent the video content at the semantic level over time and is capable of capturing video content changes despite variations in activity durations and occurrences of inactivity break-ups within activities. After mapping a video sequence into a cumulative scene vector trajectory<sup>2</sup>, the breakpoints on the trajectory correspond to the video content change points. We thus consider the video segmentation problem as a temporal video content trajectory breakpoint detection problem. To solve the problem, one offline and one online algorithms are proposed in this section.

#### 3.1 Multi-Observation Hidden Markov Model (MOHMM)

We first consider detecting breakpoints on a temporal video content trajectory using a Dynamic Bayesian Network (DBN), more specifically, a Multi-Observation Hidden Markov Model (MOHMM). Dynamic Bayesian Networks (DBNs) are Bayesian Belief Networks (BBNs<sup>3</sup>) that have been extended to model time series data [10, 15]. A DBN  $\mathbf{B}$  is described by two sets of parameters  $(\mathbf{m}, \Theta)$ . The first set  $\mathbf{m}$  represent

<sup>2</sup>More precisely, it is a video polyline due to the discrete nature of the image frames.

<sup>3</sup>BBNs are also known as Bayesian Networks, Belief Networks or Directed Acyclic Graphical (DAG) Models. They are special cases of graphical models which combine probability theory and graph theory to address two important issues in data modelling: uncertainty and complexity.

the structure of the DBN which include the number of hidden variables and observation variables per time instance, the number of states for each hidden variable and the topology of the network (set of directed arcs connecting nodes). The  $i$ th hidden variable and the  $j$ th observation variable at time instance  $t$  are denoted as  $S_t^{(i)}$  and  $O_t^{(j)}$  respectively where  $i \in \{1, \dots, N_h\}$  and  $j \in \{1, \dots, N_o\}$  and  $N_h$  and  $N_o$  are the number of hidden variables and observation variables respectively. The second set of parameters  $\Theta$  quantify the state transition models  $P(S_t^{(i)}|Pa(S_t^{(i)}))$ , the observation models  $P(O_t^{(j)}|Pa(O_t^{(j)}))$  and the initial state distributions  $P(S_1^{(i)})$  where  $Pa(S_t^{(i)})$  are the parents of  $S_t^{(i)}$  and similarly,  $Pa(O_t^{(j)})$  for observations. In this paper, unless otherwise stated,  $S_t^{(i)}$  are discrete and  $O_t^{(j)}$  are continuous random variables. Each observation variable has only hidden variables as parents and the conditional probability distributions (CPDs) of each observation variable are Gaussian for each state of its parent nodes.

When a DBN is adopted for video content modelling, the observation variables in the network correspond to the cumulative scene vector  $\widetilde{\mathbf{sv}}_t$  and hidden states in the network correspond to the video content. After model parameter estimation and hidden state inference, changes in hidden states should reflect the video content changes and thus can be used for video temporal segmentation. DBNs of different topologies can be employed for video structure modelling. We can choose the DBNs with the simplest topology, the Hidden Markov Models (HMMs) (Fig. 5 (a)). In a HMM, the observation variable at each time instance corresponds to  $\widetilde{\mathbf{sv}}_t$ , which is of dimension  $K$ , i.e. the number of event classes. However, a drawback of using HMMs for modelling complex temporal process is that a large number of parameters are needed to describe the model when the observation variables are of high dimension. This may result in poor model learning given insufficient training data. To address this problem, various topological extensions to the standard HMMs can be considered to factorise the state and/or observation space by introducing multiple hidden variables and multiple observation variables. In our case, we need to detect video content changes based on hidden state changes. Because the hidden states of the single hidden variable at each time instance corresponds clearly to the video content segments, the factorisation in the states space is unnecessary. Therefore we only perform factorisation in the observation space which results in a Multi-Observation Hidden Markov Model (MOHMM) shown in Fig. 5 (b). Specifically,  $O_t^{(j)}$  ( $j \in \{1, \dots, K\}$ ) in Fig. 5 (b) is a 1D random variable corresponding to one of the  $K$  elements of  $\widetilde{\mathbf{sv}}_t$ . This is to factorise the observational space based on the assumption that the occurrence of events of different event classes is independent from each other. Consequently, the number of parameters for describing a MOHMM is much lower than that for a HMM ( $2KN_h + N_h^2 - 1$  for a MOHMM and  $(K^2 + 3K)N_h/2 + N_h^2 - 1$  for a HMM).

The remaining problem is to automatically determine  $N_h$ , the number of hidden states which corre-

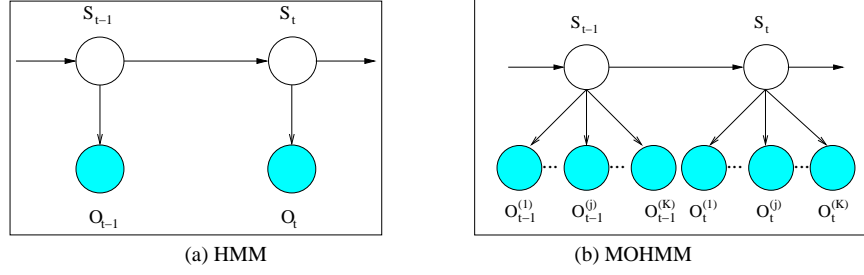


Figure 5: HMM and MOHMM with the observation nodes illustrated as shaded circles and hidden nodes as clear circles.

sponds to the number of video segments. To this end, we adopt Schwarz’s Bayesian Information Criterion (BIC) [32] to automatically determine  $N_h$ . For a model  $\mathbf{m}_i$  parameterised by a  $C_i$ -dimensional vector  $\Theta_{\mathbf{m}_i}$ , the BIC is defined as:

$$BIC = -2 \log L(\Theta_{\mathbf{m}_i}) + C_i \log N \quad (6)$$

where  $L(\Theta_{\mathbf{m}_i})$  is the maximal likelihood under  $\mathbf{m}_i$ ,  $C_i$  is the dimensionality of the parameters of  $\mathbf{m}_i$  and  $N$  is the size of the dataset. For the MOHMM,  $L(\Theta_{\mathbf{m}_i})$  can be written as:

$$-2 \log \left\{ \sum_{S_t} \left\{ P(S_1) \prod_{t=2}^T P(S_t | Pa(S_t)) \prod_{t=1}^T \prod_{j=1}^K P(O_t^{(j)} | Pa(O_t^{(j)})) \right\} \right\}$$

where  $Pa(S_t)$  are the parents of  $S_t$  and similarly,  $Pa(O_t^{(i)})$  for observations. The search for the optimal  $N_h$  that produces the minimal BIC value involves parameter learning. More specifically, for each candidate state number, the corresponding parameters are learned iteratively using the Expectation-Maximisation (EM) algorithm. The E step, which involves the inference of hidden states given the parameters estimated in the last M step, can be implemented using an exact inference algorithm such as the junction tree algorithm [17]. After parameter learning, the BIC value can be computed using Equation (6) where  $L(\Theta_{\mathbf{m}_i})$  has been obtained from the final M step of EM for parameter learning.

### 3.2 Forward-Backward Relevance (FBR)

The inference and learning of a MOHMM is computational expensive especially when a video content trajectory is long and of a high dimension. For computational efficiency and real-time performance, a novel online video segmentation algorithm is proposed which is based on a Forward-Backward Relevance (FBR) principle. More specifically, at each time instance, the relevance of one vertex on the video trajectory is measured with respect to the starting point of the current video segment (looking backward) and a certain

distance ahead on the trajectory (looking forward). The relevance of this vertex is minimal when the three vertices sit on a straight line in the high dimensional trajectory space. When the video trajectory turns drastically at the vertex, indicating a video content change, a high relevance value will be obtained. This relevance measure can thus be used for breakpoint detection on a video trajectory. Our algorithm is motivated by the offline Discrete Curve Evolution (DCE) segmentation algorithm and aims to overcome the over-segment tendency of the conventional online Sliding Window (SW) algorithm.

DCE was originally proposed for 2-D shape decomposition [23]. It can readily be extended for detecting breakpoints of high-dimensional video content trajectories. The algorithm starts with the finest possible segmentation (i.e.  $T - 1$  segments for a trajectory of length  $T$ ) and progresses with two segments being merged at each step. Specifically, the cost of merging all adjacent pairs of segments are computed and the pair with the minimal cost is merged. This process is performed iteratively until a stopping criterion is met. DCE computes the merging cost as a relevance measure  $R$  which is computed for every vertex  $v$  and depends on  $v$  and its two neighbour vertices  $u$  and  $w$ :

$$R(u, v, w) = d(v, u) + d(v, w) - d(u, w) \quad (7)$$

where function  $d(\cdot)$  can be any distance or similarity measures such as the Euclidean distance. In each iteration, the vertex with minimal  $R$  is deleted until the minimal  $R$  exceeds certain threshold or only a user-specified number of vertices are left in the trajectory. The remaining vertices correspond to the breakpoints on the video trajectory. Although DCE is similar in spirit to the bottom-up segmentation algorithm based on Piecewise Linear Approximation (PLA) in time series data analysis [20], it is much faster compared to the bottom-up algorithm using either Linear Interpolation or Linear Regression, especially when the dimensionality of the trajectory space is high.

In our online segmentation algorithm, instead of computing the relevance of each vertex iteratively, the relevance of only one vertex is computed at each time instance when a new image frame has arrived. The algorithm is outlined in Figure 6:

Two parameters,  $L_{min}$  and  $Th$  need to be determined. In this paper, they are learned from a training dataset. Specifically, we perform DCE on the training data and  $L_{min}$  is set to half of the length of the shortest segment in the training data and  $Th$  is computed as the average relevance value of the breakpoints detected by DCE.

There are a number of important characteristics of the algorithm that distinguish it from previous ones:

```

input : A  $K$  dimensional trajectory of length  $T$ :  $\widetilde{\mathbf{sv}}_t$ 
output: Detected breakpoints on the trajectory:  $breakPoint[]$ 

1  $breakpointNo = 0$ ;
2 while  $t < T$  do
3   if  $t > (breakPoint[breakpointNo] + L_{min})$  then
4     compute the relevance of the  $t - L_{min}/2$ -th vertex as:
        $R(t - L_{min}/2) = R(breakPoint[breakpointNo], t - L_{min}/2, t)$ ;
5     if  $R(t - L_{min}/2) > Th$  then
6        $breakpointNo = breakpointNo + 1$ ;
7        $breakPoint[breakpointNo] = t - L_{min}/2$ ;
8     end
9   end
10 end

```

Figure 6: An online video segmentation algorithm based on FBR. In Line 4,  $R(t - L_{min}/2)$  is computed using Equation 7 with the function  $d(\cdot)$  defined as the Euclidean distance.

1. For each new data point, instead of fitting the trajectory to a straight line, FBR computes a relevance measure of a single vertex which involves computing only 3 Euclidean distances. The computation complexity is *independent* of the segment length and much lower than the Sliding Window algorithm and its variations.
2. At each time instance, only the coordinates of  $L_{min}/2 + 1$  vertices need to be stored for computation. This makes our algorithm suitable for long videos such as 24/7 surveillance videos where it could be hours before any video content changes take place.
3. Compared to the Sliding Window algorithm and its variations, our FBR algorithm is more robust to noise because it has a global view of the trajectory by looking both forward and backward.
4. The FBR algorithm has a  $L_{min}/2$  delay in detecting breakpoints which in practice is a very small number compared to the length of a typical video segment. This is the price one has to pay to achieve robustness in detection.

### 3.3 Experiments

Experiments were conducted on the representation and segmentation of CCTV surveillance videos monitoring an aircraft ramp area. A fixed CCTV analogue camera took continuous recordings. After digitisation from VHS tapes, the final video sequences have a frame rate of 25Hz. Note that it is not uncommon to have such an extremely low frame rate for CCTV surveillance videos, which makes the video segmentation problem even more challenging. Each image frame has a size of  $320 \times 240$  pixels. Our database for the experiments consists of 7 videos of aircraft docking lasting from 6470 to 17262 frames per sequence (around 50 to 140 minutes of recording), giving in total 72776 frames of video data (around 10 hours in duration). These video data cover different times of different days under changing lighting conditions, from early morning, midday to late afternoon. They are referred to as video1 to video 7 respectively. Among the 7 videos, videos 1 to 6 follow the typical video structure with cargo services performed before catering services while video 7 has a different video structure with cargo services performed after catering services. Around 15 different activities take place in the scene. Among them, 6 activities are visually detectable (see Figure 1). In the following we present results on (1) comparative performance evaluation on video segmentation using the offline MOHMM and DCE algorithms, and the online FBR algorithm; and (2) comparing our cumulative scene vector based video content representation with a colour histogram based one.

### 3.3.1 Video Content Representation

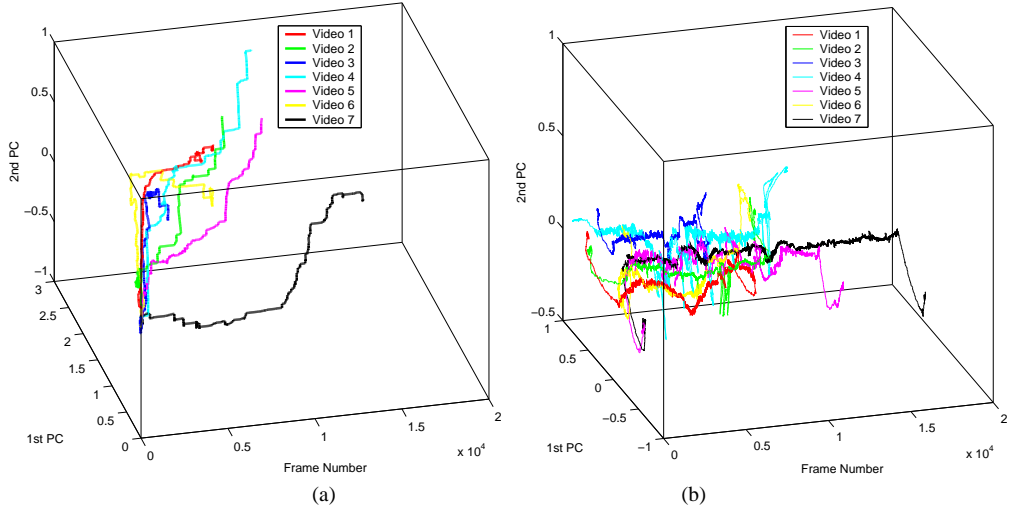


Figure 7: (a) and (b) show the first 2 principal components of  $\widetilde{\mathbf{sv}}_t$  and colour histogram evolving over time for the 7 aircraft docking videos respectively. Note that PCA was used only for visualisation here. Each element of  $\widetilde{\mathbf{sv}}_t$  and colour histogram is normalised to have a value range of  $[0, 1]$ .

Eight classes of events were automatically detected and classified from the aircraft docking video sequences. They were caused by the movements of the docking aircraft and various ground vehicles such as catering trucks, cargo trucks and cargo lifts (see Figure 2 for an example). A cumulative scene vector  $\widetilde{\mathbf{sv}}_t$  (Equation (5)) was used to represent the video content. Figure 7 (a) shows the evolution of the first two principal components of  $\widetilde{\mathbf{sv}}_t$  for the seven different sequences<sup>4</sup>. It can be seen that the video content trajectory of video 7 is distinctively different from those of the other 6 videos. For comparison, the same video sequences were also represented using colour histograms. We defined 4 histogram bins for each of the three colour channels of the RGB colour space. Separate RGB colour histograms were concatenated into a single one with 12 components in each frame. Figure 7 (b) illustrates the evolution of the first two principal components of the colour histogram representation over time from the same seven sequences as Figure 7 (a). It is evident from Figure 7 (b) that this colour histogram based video content trajectories are less smooth compared to the video content trajectories based on the cumulative scene vector. It is also noted that only the arrival and departure of the aircraft are captured well by the colour histogram changes (see the beginning and ending part of the trajectories). It can be seen from Figure 7 (b) that the trajectory corresponding to video 7 is similar to other trajectories under the colour histogram representation.

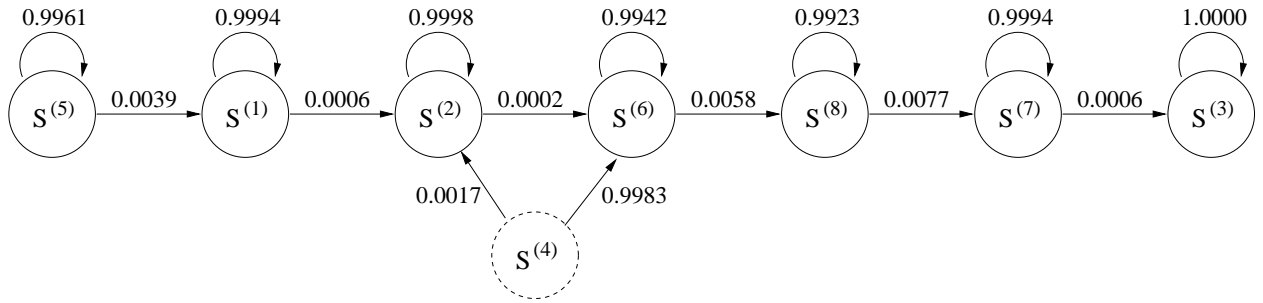
<sup>4</sup>Note that Principal Component Analysis (PCA) was performed only for visualisation of the high dimensional video trajectory. Video segmentation was carried out in the original 8-dimensional video trajectory space.

### 3.3.2 Video Segmentation

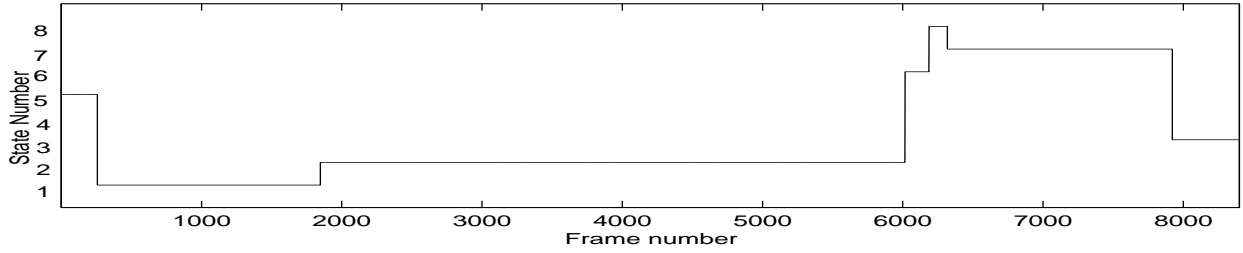
The 7 videos were first manually segmented into activities and inactivity gaps to give the ground truth of the breakpoints. The number of segments in each sequence was between 6 and 11. The videos were then divided into training and test sets with 3 videos in the training set and 4 in the testing set. This was repeated 10 times and in each trial a different training set of 3 videos were randomly chosen in order to avoid any bias in comparing the performance of different segmentation algorithms. We compared the performance of the offline Discrete Curve Evolution (DCE) and Multi-Observation Hidden Markov Model (MOHMM) algorithms and our online Forward-Backward Relevance (FBR) algorithm on the videos represented by scene vector  $\widetilde{\mathbf{v}}_t$  (Figure 7 (a)). Note that in our experiments, training data were needed by both DCE and FBR for determining the number of segments. However, they are not required directly by MOHMM because it automatically determines the number of video segments through model selection (see Section 3.1).

An example of video segmentation using MOHMM is shown in Figure 8. A MOHMM was trained for video 1. The number of observation variables in each time slice was 8, which was automatically determined by the number of event classes. The number of hidden states was automatically determined to be 8 using the BIC. All the model parameters are initialised randomly. The learned state transition probabilities are illustrated in Figure 8(a). The inferred states of MOHMM are shown in Figure 8(b). The video was then segmented based on changes on the inferred states (see Figure 8(c)). A manual segmentation with a short description of activities occurred in each segment is shown in Figure 8(d). Comparing Figure 8(c) with Figure 8(d), we can see that most video content changes have been detected correctly. It is noted that a breakpoint between `aircraftArrival` and `airbridgeConnection` was not detected. This was due to the overlapping in time between this two activities. It is also observed that some sparsely and randomly appearing events corresponding to a number of passing-by vehicles in the aircraft ramp area (see Fig. 3 between frame 2000 and 6000) were not regarded as being significant and were thus not picked up by our model as video content changes, thanks to the cumulative scene vector based video content representation.

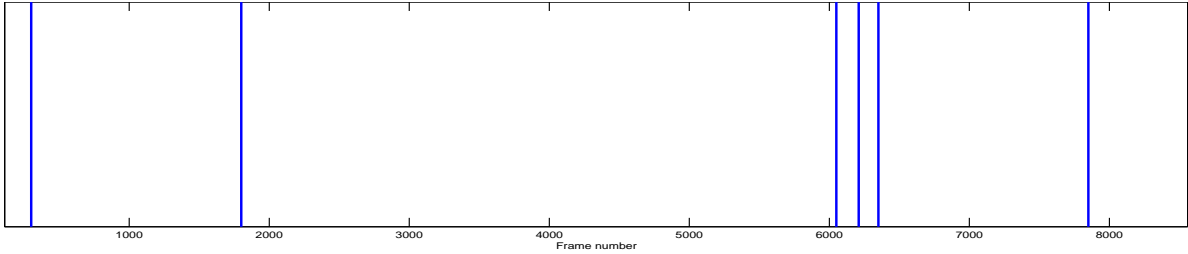
Figure 9 shows an example of breakpoint detection using the FBR algorithm. Figure 9(b) shows video 2 represented by the cumulative scene vector  $\widetilde{\mathbf{sv}}_t$  (illustrated in a 3D PCA space) with the 7 detected breakpoints overlapped on the video content trajectory (the trajectory evolves from left to right). The top row of Figure 9 shows the frames corresponding to the detected breakpoints. Compared with the manual segmentation result (Figure 9(c)), it can be seen that the fourth breakpoint is a false positive which was caused by a long inactivity break-up during the frontal cargo service. A breakpoint between the fifth and sixth detected



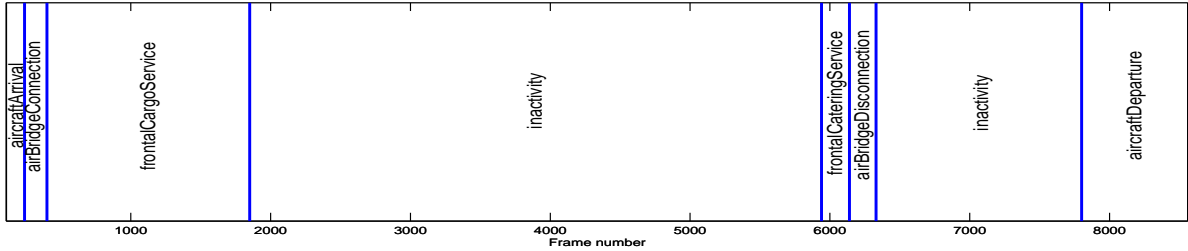
(a) Transition probabilities of hidden states of the MOHMM trained for video 1



(b) The inferred states of MOHMM



(c) MOHMM segmentation



(d) Manual Segmentation

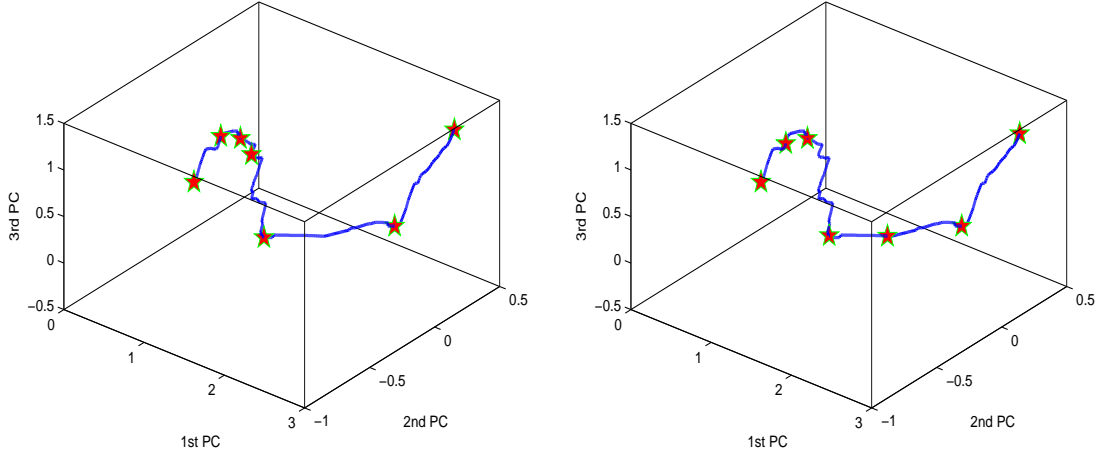
Figure 8: Temporal segmentation of video 1 using MOHMM. In (a), the inferred hidden state 4 is a virtual state that does not model data (the probability of staying in state 4 is zero).

breakpoints was missed which corresponds the ending point of a frontal catering service. This was caused by the quick departure of the catering truck and the low frame rate of the video.

The performance of the three algorithms over 10 trials was measured by the detection rate and false positive rate of breakpoint detection. A breakpoint detected within 100 frames of the true breakpoint was regarded as being correctly detected. In total, there were 334 true breakpoints in the testing sets over 10 trials. The distance measure in Equation (7) was Euclidean for both DCE and FBR. Table 1 shows that satisfactory results were obtained by all three algorithms considering the challenging nature of the problem and the noisy data used in the experiments. The FBR algorithm achieved slightly worse detection rate



(a)



(b)

(c)

Figure 9: An example breakpoint detection result by the FBR algorithm. (a) The frames corresponding to the detected breakpoints. (b) Detected breakpoints overlapped on the video content trajectory. (c) Manual segmentation result. Notice that although the video are illustrated in a 3D PCA space in (b) and (c), it was segmented in an 8-dimensional video trajectory space.

	DCE	MOHMM	FBR
Detection Rate	78.1 %	82.3 %	75.6%
False Positives	0.0236 %	0.0188 %	0.0388 %

Table 1: Segmentation results on cumulative scene vector trajectories.

compared to the two offline algorithms and tended to over-segment the video sequences. It is also noted that the starting and ending points of the frontal catering service were often miss-detected by all three algorithms. In particular, when a catering truck moved into position very quickly during the frontal catering service, the resultant change in the video content trajectory was similar to those caused by passing vehicles which were not involved in any activity. This type of changes would not be picked up by the segmentation algorithms. On the other hand, the long inactivity break-ups caused by the lack of movements during the frontal catering service were the main reason for false positives. These issues could be partially addressed by employing a more sophisticated background model.

The algorithms were implemented on a Xeon 3GHz platform. The computational cost for background modelling and event detection was 0.11 second per frame. Given the constructed cumulative scene vector, the computational cost of the three segmentation algorithms are compared in Table 2. It can be seen that the computational cost for FBR is much low than those of the other two. Since FBR is an online algorithm,

our online segmentation algorithm can run at a near real-time speed (9Hz) without any code optimisation (i.e. the speed is mainly determined by the background modelling and event detection).

	DCE	MOHMM	FBR
Computational cost (second per frame)	0.023	0.098	0.000032

Table 2: Comparing the computational costs of the three segmentation algorithms.

### 3.3.3 Comparison with Colour Histogram based Representation

	DCE	MOHMM	FBR
Detection Rate	31.4%	26.7%	28.4%
False Positives	0.0624%	0.0430%	0.0539%

Table 3: Segmentation results on colour histogram trajectories.

We also performed DCE, MOHMM and FBR on the colour histogram based video content representation following the same experimental procedure. It is shown in Table 3 that all three algorithms yielded much inferior results on the same 7 aircraft docking videos. In particular, it is noted that only the arrival and departure of the aircraft were detected reliably. The changes caused by the start and end of other activities were easily overwhelmed by image noise and illumination changes frequently encountered in a typical outdoor environment. This resulted in the lower detection rates and higher false positive rates (comparing Table 3 with Table 1).

### 3.3.4 Discussions

The key findings of our experiments are summarised and discussed as below:

1. Comparing the two proposed segmentation algorithms, MOHMM achieved higher detection rate and lower false alarm rate. Using MOHMM, video content is modelled explicitly by hidden states which need to be inferred from the observations. This makes MOHMM more robust to representation noise. However, it is also computationally more expensive than FBR. Also importantly, FBR is an online algorithm that can run in real-time together the background modelling and event detection. This makes FBR attractive for applications for which real-time performance is the key priority.

2. Our results show that a semantically meaningful video representation is crucial for segmentation of surveillance videos. This is related to the problem of ‘bridging the semantic gap’ which has received much attention recently in multimedia content analysis [13, 26, 7]. In our case, there is a gap between the low level image features (e.g. colour and motion) and semantic meaningful segmentation results (i.e. each segment corresponds to an activity which can be described using semantic concepts). To bridge this gap, we introduce the event-based video content representation with each class of events being constituents of different activities. When a new activity takes place, the changes in the event classes or/and the temporal correlations among them can be picked up by our segmentation algorithm for video content breakpoint detection.
3. Constructing a scene vector  $\mathbf{sv}_t$  is essentially a Vector Quantization process which is useful for concise representation. However, using the scene vector directly is not appropriate for segmentation because of the discreteness of the scene vector. The introduction of the cumulative scene vector  $\widetilde{\mathbf{sv}}_t$  is critical to make the representation less sensitive to short inactivity break-ups within activities and noise in event detection.

## 4 Video Content Modelling for Activity Recognition and Unusual Activity Detection

Using the approaches described in the preceding section, continuous surveillance videos in a training dataset are segmented into  $N$  video segments, ideally each video segment now containing a single activity. Taking an unsupervised approach, the task now is to discover the natural grouping of the activity patterns (i.e. clustering), upon which a video content model is built for the  $N$  video segments. Given an unseen activity, the model is then employed to detect if the activity pattern is different from what has been seen before (i.e. an unusual activity), and recognise it into different activity classes if a similar pattern has been observed previously.

### 4.1 Activity Representation

The activity pattern captured in the  $n$ th video segment  $\mathbf{v}_n$  is represented as a feature vector  $\mathbf{P}_n$ , given as

$$\mathbf{P}_n = [\mathbf{p}_{n1}, \dots, \mathbf{p}_{nt}, \dots, \mathbf{p}_{nT_n}], \quad (8)$$

where  $T_n$  is the length of the  $n$ th video segment and the  $t$ th element of  $\mathbf{P}_n$  is a  $K_e$  dimensional variable:

$$\mathbf{p}_{nt} = [p_{nt}^1, \dots, p_{nt}^k, \dots, p_{nt}^{K_e}]. \quad (9)$$

$\mathbf{p}_{nt}$  corresponds to the  $t$ th image frame of  $\mathbf{v}_n$  where  $p_{nt}^k$  is the posterior probability that an event of the  $k$ th event class has occurred in the frame given the learned GMM. If an event of the  $k$ th class is detected in the  $t$ th image frame of  $\mathbf{v}_n$ , we have  $0 < p_{nt}^k \leq 1$ ; otherwise, we have  $p_{nt}^k = 0$ .

It is worth pointing out that: (1) An activity pattern is decomposed into temporally ordered, semantically meaningful scene-events. Instead of using low level image features such as location, shape, and motion (Eqn. (3)) directly for activity representation, we represent an activity pattern using the probabilities of different classes of event occurring in each frame. Consequently, the activity representation is compact and concise. This is critical for a model-based video content modelling approach because model construction based upon concise representation is more likely to be computationally tractable for complex activities. (2) Different types of activity patterns can differ either in the classes of events they are composed of, or in the temporal orders of the event occurrence. For instance, activity patterns A and B are deemed as being different if 1) A is composed of events of classes a, b, and d, while B is composed of events of classes a, c and e; or 2) Both A and B are composed of events of classes a, c and d; however, in A, event (class) a is followed by c, while in B, event (class) a is followed by d.

## 4.2 Activity Clustering

The video content modelling problem can now be defined formally. Consider a training dataset  $\mathbf{D}$  consisting of  $N$  feature vectors:

$$\mathbf{D} = \{\mathbf{P}_1, \dots, \mathbf{P}_n, \dots, \mathbf{P}_N\}, \quad (10)$$

where  $\mathbf{P}_n$  is defined in Eqn. (8) representing the activity pattern captured by the  $n$ th video segment. The problem to be addressed is to discover the natural grouping of the training activity patterns upon which a model for usual activity can be built. This is essentially a data clustering problem with the number of clusters unknown. There are a number of aspects that make this problem challenging: (1) Each feature vector  $\mathbf{P}_n$  can be of different lengths. Conventional clustering approaches such as K-means and mixture models require that each data sample is represented as a fixed length feature vector. These approaches thus cannot be applied directly. (2) A definition of a distance/affinity metric among these variable length feature

vectors is nontrivial. Measuring affinity between feature vectors of variable length often involves Dynamic Time Warping [22]. A standard dynamic time warping (DTW) method used in computer vision community would attempt to treat the feature vector  $\mathbf{P}_n$  as a  $K_e$  dimensional trajectory and measure the distance of two activity patterns by finding correspondence between discrete vertices on two trajectories. Since in our framework, an activity pattern is represented as a set of temporal correlated events, i.e. a stochastic process, a stochastic modelling based approach is more appropriate for distance measuring. (3) Model selection needs to be performed to determine the number of clusters. To overcome the above-mentioned difficulties, a recently proposed spectral clustering algorithm with feature and model selection is deployed [40].

To apply a spectral clustering algorithm, a data affinity matrix needs to be constructed. Here we only describe the procedure for constructing an affinity matrix for activity clustering and leave out the details of the spectral clustering algorithm which can be found in [40]. We utilise Dynamic Bayesian Networks (DBNs) to provide a dynamic representation of each activity pattern feature vector in order to both address the need for dynamic warping and provide a string similarity metric. More specifically, each activity pattern in the training set is modelled using a DBN. To measure the affinity between two activity patterns represented as  $\mathbf{P}_i$  and  $\mathbf{P}_j$ , two DBNs denoted as  $\mathbf{B}_i$  and  $\mathbf{B}_j$  are trained on  $\mathbf{P}_i$  and  $\mathbf{P}_j$  respectively using the EM algorithm [6, 10]. Similar to the synthetic data case (see Section 2), the affinity between  $\mathbf{P}_i$  and  $\mathbf{P}_j$  is then computed as:

$$A_{ij} = \frac{1}{2} \left\{ \frac{1}{T_j} \log P(\mathbf{P}_j | \mathbf{B}_i) + \frac{1}{T_i} \log P(\mathbf{P}_i | \mathbf{B}_j) \right\}, \quad (11)$$

where  $P(\mathbf{P}_j | \mathbf{B}_i)$  is the likelihood of observing  $\mathbf{P}_j$  given  $\mathbf{B}_i$ , and  $T_i$  and  $T_j$  are the lengths of  $\mathbf{P}_i$  and  $\mathbf{P}_j$  respectively.

DBNs of different topologies can be used. However, it is worth pointing out that since a DBN needs to be learned for every single activity pattern in the training dataset which could be short in duration, a DBN with less number of parameters is more desirable. To this end, we employ the Multi-Observation Hidden Markov Model (MOHMM) which is described in Section 3.1. Note that when applying MOHMM for modelling each activity pattern, the number of hidden states for the MOHMM is set to  $K_e$ , i.e. the number of event classes. This is reasonable because the value of  $N_s$  should reflect the complexity of an activity pattern, so should the value of  $K_e$ .

### 4.3 Modelling Video Content Using a Composite Model

Using the spectral clustering algorithm proposed in [40], the  $N$  activity patterns in the training dataset is grouped into  $K$  clusters with  $K$  being automatically determined via model order selection. To build a model for the observed/expected activities, we first model the  $k$ th activity class using a MOHMM  $\mathbf{B}_k$ . The parameters of  $\mathbf{B}_k$ ,  $\theta_{\mathbf{B}_k}$  are estimated using all the patterns in the training set that belong to the  $k$ th class. A activity model  $\mathbf{M}$  is then formulated as a mixture of the  $K$  MOHMMs. Given an unseen activity pattern, represented as an activity pattern feature vector  $\mathbf{P}$  as described in Section 4.1, the likelihood of observing  $\mathbf{P}$  given  $\mathbf{M}$  is

$$P(\mathbf{P}|\mathbf{M}) = \sum_{k=1}^K \frac{N_k}{N} P(\mathbf{P}|\mathbf{B}_k), \quad (12)$$

where  $N$  is the total number of training activity patterns and  $N_k$  is the number of patterns that belong to the  $k$ th activity class.

### 4.4 Online Activity Recognition and Unusual Activity Detection

Once constructed, the composite video content model  $\mathbf{M}$  can be used to detect whether an unseen activity pattern is usual using a run-time unusual activity measure. If detected as being usual, the activity pattern is then recognised as one of the  $K$  classes of usual activity patterns using an online Likelihood Ratio Test (LRT) method.

An unseen activity pattern of length  $T$  is represented as  $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_t, \dots, \mathbf{p}_T]$ . At the  $t$ th frame, the accumulated visual information for the activity pattern, represented as  $\mathbf{P}_t = [\mathbf{p}_1, \dots, \mathbf{p}_t]$ , is used for online reliable unusual activity detection. First the normalised log-likelihood of observing  $\mathbf{P}$  at the  $t$ th frame given the activity model  $\mathbf{M}$  is computed as

$$l_t = \frac{1}{t} \log P(\mathbf{P}_t|\mathbf{M}). \quad (13)$$

$l_t$  can be easily computed online using the forward-backward procedure [24]. Specifically, to compute  $l_t$ , the  $K_e$  forward probabilities at time  $t$  are computed using the  $K_e$  forward probabilities computed at time  $t - 1$  together with the observations at time  $t$  (see [24] for details). Note that the complexity of computing  $l_t$  is  $\mathcal{O}(K_e^2)$  and does not increase with  $t$ . We then measure how likely  $\mathbf{P}_t$  is an unusual activity using an

online unusual activity measure  $Q_t$ :

$$Q_t = \begin{cases} l_1 & \text{if } t = 1 \\ (1 - \alpha)Q_{t-1} + \alpha(l_t - l_{t-1}) & \text{otherwise} \end{cases} \quad (14)$$

where  $\alpha$  is an accumulating factor determining how important the visual information extracted from the current frame is for unusual activity detection. We have  $0 < \alpha \leq 1$ . Compared to  $l_t$  as an indicator of normality/anomaly,  $Q_t$  could add more weight to more recent observations. Unusual activity is detected at frame  $t$  if

$$Q_t < Th_A \quad (15)$$

where  $Th_A$  is the unusual activity detection threshold. The value of  $Th_A$  should be set according to the detection and false alarm rate required by each particular surveillance application. Note that it takes a time delay for  $Q_t$  to stabilise at the beginning of evaluating an activity pattern due to the nature of the forward-backward procedure. The length of this time period, denoted as  $T_w$  is related to the complexity of the MOHMM used for activity modelling. We thus set  $T_w = 3K_e$  in our experiments to be reported later in Section 4.5, i.e. the unusual activity of an activity pattern is only evaluated when  $t > T_w$ .

At each frame  $t$  an activity pattern needs to be recognised as one of the  $K$  activity classes when it is detected as being usual, i.e.  $Q_t > Th_A$ . This is achieved using an online Likelihood Ratio Test (LRT) method. More specifically, we consider a hypotheses test between:

$H_k$  :  $\mathbf{P}_t$  is from the hypothesised model  $\mathbf{B}_k$  and belongs to the  $k$ th usual activity class

$H_0$  :  $\mathbf{P}_t$  is from a model other than  $\mathbf{B}_k$  and does not belong to the  $k$ th unusual activity class

where  $H_0$  is called the alternative hypothesis. Using LRT, we compute the likelihood ratio of the accepting the two hypotheses as

$$r_k = \frac{P(\mathbf{P}_t; H_k)}{P(\mathbf{P}_t; H_0)}. \quad (16)$$

The hypothesis  $H_k$  can be represented by the model  $\mathbf{B}_k$  which has been learned in the activity profiling step. The key to LRT is thus to construct the alternative model which represents  $H_0$ . In a general case, the number of possible alternatives is unlimited;  $P(\mathbf{P}_t; H_0)$  can thus only be computed through approximation [39, 16]. Fortunately in our case, we have determined at the  $t$ th frame that  $\mathbf{P}_t$  is usual and can only be generated by one of the  $K$  usual activity classes. Therefore it is reasonable to construct the alternative model as a mixture

of the rest  $K - 1$  usual activity classes. In particular, Eqn. (16) is re-written as

$$r_k = \frac{P(\mathbf{P}_t|\mathbf{B}_k)}{\sum_{i \neq k} \frac{N_i}{N - N_k} P(\mathbf{P}_t|\mathbf{B}_i)}. \quad (17)$$

Note that  $r_k$  is a function of  $t$  and computed over time.  $\mathbf{P}_t$  is reliably recognised as the  $k$ th activity class only when  $1 \ll Th_r < r_k$ . In our experiments we found that  $Th_r = 10$  led to satisfactory results. When there are more than one  $r_k$  greater than  $Th_r$ , the activity pattern is recognised as the class with the largest  $r_k$ .

For comparison, the commonly used *Maximum Likelihood* (ML) method recognises  $\mathbf{P}_t$  as the  $k$ th activity class when  $k = \arg \max_k \{P(\mathbf{P}_t|\mathbf{B}_k)\}$ . Using the ML method, recognition has to be performed at each single frame without considering how reliable and sufficient the accumulated visual evidence is. This often causes errors especially when there are ambiguities between different classes (e.g. an activity pattern can be explained away equally well by multiple plausible activity at its early stage). Compared to the ML method, our online LRT method holds the decision on activity recognition until sufficient evidence has been accumulated to overcome ambiguities. The recognition results obtained using our approach are thus more reliable compared to those obtained by the ML method.

## 4.5 Experiments

The same datasets described in Section 3.3 was employed for validating our approach on video content modelling and online unusual activity detection. The videos were segmented automatically using the FBR online segmentation algorithm described in Section 3.2, giving 59 video segments of actual activity pattern instances. Each segment has on average 428 frames with the shortest 74 and longest 2841.

A1	Aircraft arrives	A2	Airbridge connected	A3	Frontal cargo service
A4	Frontal catering service	A5	Aircraft departs	A6	Airbridge disconnected

Table 4: Six classes of commonly occurred activity patterns in the airport scene.

**Model training** — A training set was created which consisted of 40 video segments and the remaining 19 were used for testing. This model training exercise was repeated 20 times and in each trial a different model was trained using a different random training set. This is in order to avoid any bias in the unusual activity detection and usual activity recognition results to be discussed later.

In each training session, the 40 training activity patterns were represented based on the event detection

results and clustered to build a composite activity model. The number of clusters for each training set was determined automatically as 6 in every trial using the spectral clustering algorithm proposed in [40]. By observation, each of the 6 discovered data clusters mainly contained samples corresponding to one of the 6 activity classes listed in Table 4.

Anomaly Detection Rate (%)	False Alarm Rate (%)
$79.2 \pm 8.3$	$5.1 \pm 3.9$

Table 5: The mean and standard deviation of the unusual activity detection rate and false alarm rates. The results were obtained over 20 trials with  $Th_A = -0.5$ .

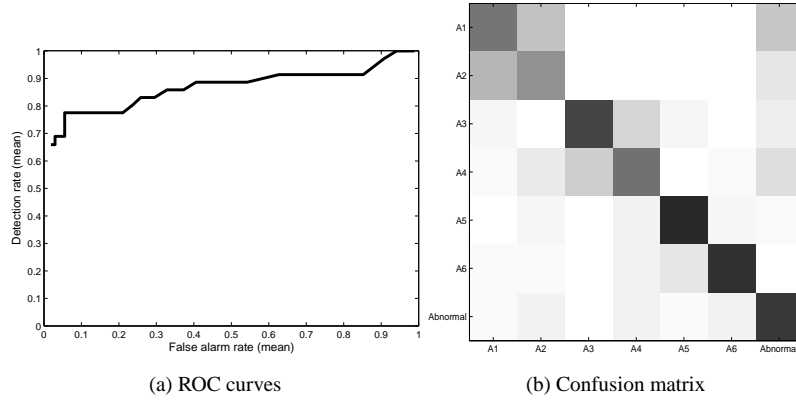


Figure 10: The performance of unusual activity detection using aircraft docking video content model. The mean ROC curves were obtained over 20 trials. (b) the performance of usual activity recognition obtained by averaging the results over 20 trials with  $Th_A = -0.5$ . Each row represents the probabilities of the corresponding class being confused with all the other classes averaged over 20 trials.

**Unusual activity detection** — Each activity pattern in a testing set was detected as being unusual when Eqn. (15) was satisfied at any time after  $T_w = 3K_e = 24$  frames. The accumulating factor  $\alpha$  for computing  $Q_t$  was set to 0.1. To measure the performance of the learned models on unusual activity detection, each activity pattern in the testing sets was manually labelled as usual if there were similar patterns in the corresponding training sets and unusual otherwise. We measure the performance of unusual activity detection using unusual activity detection rate and false alarm rate<sup>5</sup>, which are defined as:

$$\begin{aligned}
 \text{Unusual activity detection rate} &= \frac{\# \text{ True positives (unusual detected as unusual)}}{\# \text{ All positives (unusual patterns) in a dataset}} \\
 \text{False alarm rate} &= \frac{\# \text{ False positives (usual detected as unusual)}}{\# \text{ All negatives (usual patterns) in a dataset}}
 \end{aligned} \tag{18}$$

The detection rate and false alarm rate of unusual activity detection are shown in the form of a Receiver Operating Characteristic (ROC) curve by varying the unusual activity detection threshold  $Th_A$  (see Eqn. (15)).

<sup>5</sup>Detection rate and false alarm rate are also called true positive rate and false positive rate respectively in the literature. Note that the performance can also be measured using true negative rate and false negative rate. Since we have ‘true negative rate’=1-‘false alarm rate’ and ‘false negative rate’=1-‘detection rate’, showing only unusual activity detection rate and false alarm rate is adequate.

The result is shown in Fig. 10(a) and Table 5. Fig. 11(b)&(f) show examples of online reliable unusual activity detection using our run-time unusual activity measure.

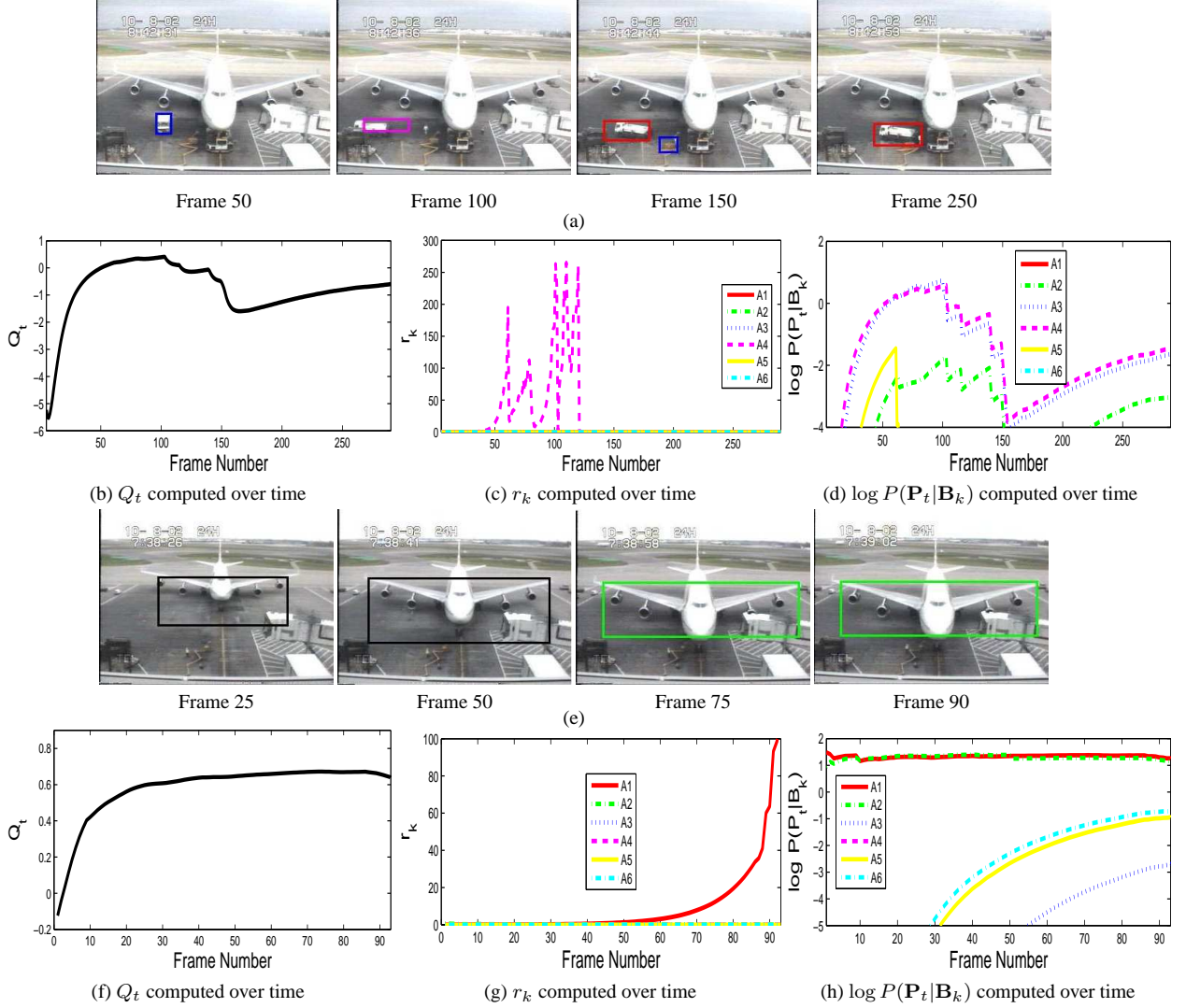


Figure 11: Compare our LRT method with ML method for online usual activity recognition in an aircraft docking scene. (a): An unusual activity pattern where a truck brought engineers to fix a ground power cable problem. It resembled A4 in the early stage. (b): It was detected as unusual activity from Frame 147 till the end based on  $Q_t$ . (c): The activity pattern between Frame 53 to 145 was recognised reliably as A4 using LRT before becoming unusual and being detected using  $Q_t$ . (d) The activity pattern was wrongly recognised as A3 between Frame 80 to 98 using the ML method. (e): An usual A1 activity pattern. (f): The activity pattern was detected as usual throughout based on  $Q_t$ . (g): It was recognised reliably as A1 from Frame 73 till the end using LRT. (h): It was wrongly recognised as A2 between Frame 12 to 49 using ML. In (a) and (e), detected events are illustrated using the same colour scheme as in Fig. 2.

**Recognition of usual activity patterns** — The usual activity recognition results are illustrated in Fig. 10(b). When  $Th_A$  was set to  $-0.5$ , the recognition rate averaged over 20 trials was 72.1%. Examples of online reliable activity recognition using our online LRT method are shown in Fig. 11 in comparison with the ML method. It can be seen that our LRT method is superior to the ML method in that usual activity patterns can be reliably and promptly recognised after sufficient visual evidence had become available to overcome the

ambiguities among different activity classes.

## 5 Conclusion

In conclusion, we have presented a novel approach for representing, segmenting, and modelling the content of CCTV surveillance videos according to the activities captured in the scene. The video content is represented by constructing a cumulative multi-event scene vector over time. An offline Multi-Observation Hidden Markov Model (MOHMM) based segmentation algorithm was introduced to deal with noise in video content representation. An online Forward-Backward Relevance (FBR) algorithm was also developed to detect breakpoints in the video content and segment a continuous video into activities on-the-fly. A framework for robust online activity recognition and unusual activity detection has been proposed which is based on constructing a generative composite video content model. Experiments were carried out to evaluate the effectiveness and robustness of our approach using over 10 hours of challenging outdoor surveillance video footages. Our experiments demonstrate that an event based video content representation is superior to a colour histogram based representation for surveillance video segmentation. Our experiments also suggest that an online LRT based method is preferred over the conventional ML based method for activity recognition.

It is worth pointing out that in our current MOHMM, each observational variable is one dimensional and assumed to be independent to each other. Since each variable corresponds to one class of event and causal relationships exist among different classes of events, this assumption is not valid. We are investigating the possibility of joining dependent observational variables together into combined variables or introducing arcs between dependent variables to improve the performance of our MOHMM based segmentation algorithm. Similar idea has been exploited in structure improvement of the static naive Bayes classifier [8].

## References

- [1] N. Babaguchi, Y. Kawai, and T. Kitahashi. Event based indexing of broadcasting sports video by intermodal collaboration. *IEEE Transactions on Multimedia*, 4(1):68–75, 2002.
- [2] C. Bishop. *Neural Networks for Pattern Recognition*. Cambridge University Press, 1995.

- [3] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.
- [4] O. Boiman and M. Irani. Detecting irregularities in images and in video. In *IEEE International Conference on Computer Vision*, pages 462–469, 2005.
- [5] D. DeMenthon, L. Latechi, A. Rosenfeld, and M. Stuckelberg. Relevance ranking of video data using hidden markov model distances and polygon simplification. In *Advances in Visual Information Systems*, 2000.
- [6] A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society B*, 39:1–38, 1977.
- [7] P. Enser and C. Sandom. Towards a comprehensive survey of the semantic gap in visual image retrieval. In *CIVR*, pages 291–299, 2003.
- [8] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.
- [9] X. Ge and P. Smyth. Segmental semi-markov models for endpoint detection in plasma etching. In *AEC/APC Symposium XII*, 2000.
- [10] Z. Ghahramani. Learning dynamic bayesian networks. In *Adaptive Processing of Sequences and Data Structures. Lecture Notes in AI*, pages 168–197, 1998.
- [11] S. Gong and T. Xiang. Recognition of group activities using dynamic probabilistic networks. In *IEEE International Conference on Computer Vision*, pages 742–749, 2003.
- [12] H. Greenspan, J. Goldberger, and A. Mayer. Probabilistic space-time video modelling via piecewise GMM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):384–396, 2004.
- [13] William I. Grosky and Rong Zhao. Negotiating the semantic gap: From feature maps to semantic landscapes. *Lecture Notes in Computer Science*, 2234:33–42, 2001.
- [14] P. Heckbert and M. Garland. Survey of ploygonal surface simplification algorithms. In *International Conference on Computer Graphics and Interactive Techniques*, 1997.
- [15] D. Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995.

- [16] A. Higgins, L. Bahler, and J. Porter. Speaker verification using randomized phrase prompting. *Digital Signal Processing*, pages 89–106, 1991.
- [17] C. Huang and A. Darwiche. Inference in belief networks: a procedural guide. *International Journal of Approximate Reasoning*, 15(3):225–263, 1996.
- [18] H. Hung and S. Gong. Quantifying temporal saliency. In *British Machine Vision Conference*, pages 727–736, 2004.
- [19] J.R. Kender and B.L. Yeo. Video scene segmentation via continuous video coherence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 367–373, 1998.
- [20] E. Keogh, S. Chu, D. Hart, and M. Pazzani. An online algorithm for segmenting time series. In *International Conference on Data Engineering*, pages 289–296, 2001.
- [21] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Mining and knowledge discovery*, 7(4):349–371, 2003.
- [22] J. Kruskal and M. Liberman. *The symmetric time-warping problem: From continuous to discrete*. Addison-Wesley, 1983.
- [23] L. Latecki and R. Lakamper. Convexity rule for shape decomposition based on discrete contour evolution. *Computer Vision and Image Understanding*, 73:441–454, 1999.
- [24] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [25] S. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking group of people. *Computer Vision and Image Understanding*, 80:42–56, 2000.
- [26] M.R. Naphade, I. Kozintsev, and T.S. Huang. A factor graph framework for semantic indexing and retrieval in video. *IEEE trans. on CAS for VT*, pages 40–52, 2002.
- [27] J. Ng and S. Gong. Learning pixel-wise signal energy for understanding semantics. In *British Machine Vision Conference*, pages 695–704, 2001.
- [28] C.W. Ngo, T.C. Pong, and H.J. Zhang. Motion-based video representation for scene change detection. *International Journal of Computer Vision*, 50(2):127–142, 1998.

- [29] N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modelling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, August 2000.
- [30] J.H. Piater and J.L. Crowley. Multi-modal tracking of interacting targets using gaussian approximation. In *Proceedings of 2nd IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, pages 141–147, 2001.
- [31] Y. Rui, S. Huang, and S. Mehrota. Exploring video structures beyond the shots. In *IEEE International Conference on Multimedia Computing and Systems*, pages 237–240, 1998.
- [32] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.
- [33] H. Shatkey and S. Zdonik. Approximate queries and representations for large data sequences. In *International Conference on Data Engineering*, pages 546–553, 1996.
- [34] J. Sherrah and S. Gong. Automated detection of localised visual events over varying temporal scales. In *Proc. European Workshop on Advanced Video-based Surveillance System*, 2001.
- [35] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [36] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–758, August 2000.
- [37] N. Vasconcelos and A. Lippman. Bayesian modeling of video editing and structure: Semantic features for video summarization and browsing. In *ICIP*, pages 550–555, 1998.
- [38] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *IEEE International Conference on Computer Vision*, pages 975–982, 1999.
- [39] J. Wilpon, L. Rabiner, C. Lee, and E. Goldman. Automatic recognition of keywords in unconstrained speech using hidden markov models. *IEEE Trans. Acoustic, Speech and Signal Processing*, pages 1870–1878, 1990.
- [40] T. Xiang and S. Gong. Video behaviour profiling and abnormality detection without manual labelling. In *IEEE International Conference on Computer Vision*, pages 1238–1245, 2005.
- [41] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67(1):21–51, 2006.

- [42] T. Xiang, S. Gong, and D. Parkinson. Autonomous visual events detection and classification without explicit object-centred segmentation and tracking. In *British Machine Vision Conference*, pages 233–242, 2002.
- [43] S. Yu and J. Shi. Multiclass spectral clustering. In *IEEE International Conference on Computer Vision*, pages 313–319, 2003.
- [44] L. Zelnik-Manor and M. Irani. Event-based video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 123–130, 2001.