A Framework for Analysis of Surveillance Videos

Ayesha Choudhary[1] Santanu Chaudhury[2] Subhashis Banerjee[1]
[1]*Dept. of Computer Science and Engineering,* [2]*Dept. of Electrical Engneering*
Indian Institute of Technology, Delhi
[1]*ayesha, suban@cse.iitd.ernet.in,* [2]*santanuc@ee.iitd.ac.in*

Abstract

In this paper, we propose a novel framework for automated analysis of surveillance videos. By analysis, we imply summarizing and mining of the information in the video for learning usual patterns and discovering unusual ones. We approach this video analysis problem by acknowledging that a video contains information at multiple levels and in multiple attributes. Each such component and co-occurrences of these component values play an important role in characterizing an event as usual or unusual. Therefore, we cluster the video data at multiple levels of abstraction and in multiple attributes and view these clusters as a summary of the information in the video. We apply cluster algebra to mine this summary from multiple perspectives and to adapt association learning for automated selection of components because of which the event is unusual. We also propose a novel incremental clustering algorithm.

1. Introduction

In this paper, we propose a framework for automated analysis of surveillance videos to learn the usual patterns of events and discover unusual ones. We approach this analysis of surveillance videos by acknowledging that a video contains information at multiple levels (e.g., object, frame, group of frames, etc.) and in multiple attributes/components (such as time, size, shape, position of objects, etc.). Each such component may play an important role in characterizing an activity as usual or unusual. We cluster in each of these components independently to learn the patterns of that component. We view these clusters as a summary of the information in that component and present them in a visual form. Thus, clustering on all components independent of each other leads to a summary of the information in the video. We use cluster algebra [2] to mine the various combinations of patterns from these component summaries to learn the usual patterns and discover unusual ones. Although for our experiments and explanations, we have used features like color correlogram, size and position of objects, our framework does not depend on these feature but is flexible and complex features can also be incorporated.

An underlying strength of our framework is the ability to summarize information from multiple perspectives depending on the requirement of the system. This is necessary because an unusual event can also be termed as usual based on the circumstances. We give an example to illustrate our point of view. In a mall, summarizing on the information about the paths that people take can provide information of the scene from multiple perspectives, for example, "which are the common paths during certain time periods?", "which path is rarely taken at any time of the day?". A usual path taken after the mall is closed will be unusual but if someone goes on that path every day at the same time, then it is not a suspicious activity. The paths also give information about the correlations between the landmarks (shops). "Do people who visit shop A also visit shop B?", which shops are most often visited, during which time periods. These correlations help in zoning the mall, which is necessary from crowd management perspective as well as retailers desire to attract more buyers. Egress planning can also be facilitated with such an analyis, for example, shops that attract a large number of customers should be placed near the exits for easy evacuations in case of fire. Facility management is another issue in mall management. "What are the "hot spots" in the mall?", advertisers/ retailors would benefit from advertising in these locations. Thus, combination of different aspects of the same content can provide information relevant for decision making in that environment.

Such a framework is essential, because it is not possible for a human operator to continuously watch hours of video, either online through a webcam or offline and analyze the video from multiple perspectives. A human operator would benefit from visualizing the clusters to learn the normal patterns and would only need to check the events marked as suspicious. Moreover, the operator has the ability to choose and mine various combinations of clusters of components that he/ she thinks are relevant.

In our framework, we represent a video as a tuple of var-

ious components and apply component based clustering on it. We use cluster algebra and modify *usualness* measure [2] to adapt association learning for discovering unusual events and for automated selection of components of the tuple because of which the event is labeled as unusual. We also extend the cluster algebra to the temporal domain and provide a rough set based interpretation of the cluster algebra.

To cluster the data as a video is parsed, an incremental clustering algorithm is required. Since we cluster in various components of the video data, we develop an incremental clustering algorithm which can be used with any data type (numerical or symbolic) and is independent of predefining the number of clusters and cluster radii. It also helps in dealing with the large volume of data in case of offline analysis of stored videos. A semi-supervised version of our incremental clustering algorithm can be used to leverage on labeled data, in case available.

In the next section, we discuss the related work. In Section 3, we describe the video representation scheme and component based clustering. In Section 4, the incremental clustering algorithm is presented. The cluster algebra and automated association selection is given in Section 5, while the rough set based interpretation of cluster algebra is presented in Section 6. We discuss the results in Section 7 and conclude in Section 8.

2. Related Work

Detecting unusual activities is important for automated surveillance. Rule based methods which apply HMMs and FSMs are popular methods for activity analysis [3, 9]. The drawback of these methods is that they require predefinition of usual activities to be able to discover unusual ones. Statistical methods are better for surveillance as they automatically learn what is usual and what is not but require a lot of training data. However, unlike our technique these methods do not provide a tool to analyze the video from multiple perspectives. We view the clusters in each of the component spaces as a summary of the information in that component which we present in a visual form. This gives the advantage of knowing at a glance as to which patterns are usual and which are not.

In the literature, video summary or video synopsis means compressing the time axis of the original video to produce a shorter video [11]. Although it removes the temporal or spatio-temporal redundancies, it also leads to loss of chronological consistency. However, chronological consistency is necessary for event analysis. In general, video mining systems are based on detection of known patterns [10, 13] in the video data. However, in our framework we mine the clusters and combinations of clusters for the purpose of finding unusual events and learning usual ones.

Interest in incremental clustering algorithms [5, 4] has been on the rise because it is useful in clustering dynamic data and uses less memory since only the cluster representatives have to be kept in the memory. Most incremental clustering algorithms require the number of clusters to be pre-defined while some also require that the complete data be present *a priori*. The main contribution of our incremental clustering algorithm is that it is independent of setting thresholds on distance measures, predefinition of the number of clusters in which the data should be partitioned and works for symbolic as well as numerical data.

3. Video Representation Scheme

We represent a video as a tuple,

$$V =< v_1, v_2, \ldots, v_k >, \qquad k \geq 2 \tag{1}$$

where, the components of the tuple $v_i, 1 \leq i \leq k$ represent the low-level features like position of objects, size of objects, color correlogram of objects, etc., extracted from the frames of the video or high level semantically meaningful concepts like object category, object trajectory etc. For example, a frame from a video is represented as the tuple,

$$V =<\text{no. of objects}, \text{color correlogram of objects}, \\ \text{position of objects}, \ \text{size of objects} > \tag{2}$$

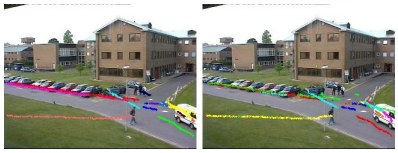
Each component of the video tuple defines an n-dimensional space, $n \geq 1$ such that $n <$ the dimension of the video tuple taken as a monolithic vector.

The information contained in the video exists at multiple layers of granularity. For example, at the frame level, information such as number of objects, size of objects, position of objects, color correlogram, etc. can be extracted. At the next level, higher level semantically meaningful information such as object category, object trajectory, time interval of presence of object in the scene, etc. can be gathered from a sequence of frames. At an even higher level, grouping together information from the lower layers gives us multiple information such as common trajectories, trajectories of pedestrians, trajectories during a certain time interval and uncommon trajectories. Thus, the tupular representation scheme can be applied at any level. For example, a video can be represented as a tuple at the level of a frame or at the level of a shot or a combination of features from various levels. Therefore, representation of a video as a tuple is a flexible method to represent the multi-layered structure of video data, where the components can be defined based on the class of the video.

Moreover, in the representation (see equation 2), the components are of different data types. For instance, the no. of objects is a number while size is a 2D vector of width and height of the bounding rectangle of the foreground objects.

Therefore, treated as a monolithic vector, the video tuple represents high dimensional heterogeneous data.

We use the concept of component based clustering and cluster in each of the component spaces of the video tuple. It enables us to tackle the heterogeneity of the tuple by focusing on the homogeneity of the components. Moreover, clustering in each component gives the data distribution of that component independently, which would be lost if the tuple is clustered as a monolithic vector. The component based clustering method allows defining a similarity measure for each of the components based on the data type and the semantics of the component. It also allows summarizing the information in each of the components independently. We apply our incremental clustering algorithm, given in the next section, to cluster in each component space. As seen



(a) (b)

Figure 1. The trajectories are the clusters across frames in the component spaces(a) *position of the objects*. **(b)** *size of objects*. **Each color indicates a cluster. Color coding of the two images are independent.**

from Figs. 1(a) and (b), clusters (across frames) in the component spaces *position of objects* and *size of objects* have different distributions. The color scheme in both the figures are independent of each other. No tracking algorithm is used on this video. The trajectories are the clusters across frames in the two independent component spaces.

4. Our Incremental Clustering Algorithm

In our incremental clustering algorithm (Algo. 1), unlike Leader's algorithm similarity is not defined by using a threshold on the distance between two points but *closeness* (or similarity) of two points is decided on the basis of a distance decay function. Let d be a well-defined distance measure and $d(x, x')$ be the distance between any two points x and x'. Then, $w(x, x') = e^{-d(x,x')}$ is the distance decay function, used as a weight to calculate the point $y = w(x, x') * x + (1 - w(x, x')) * x'$. If x is *close* to x', then the condition $d(x, y) < d(x', y)$ is true, since the weight $w(x, x')$ is high, implying that the distance is less. This allows working without setting thresholds on the distance measure and allows clustering complex (non-circular) geometries as can be seen in Fig. 2. For the discernibility of the distance decay function, the data should be mapped to a predefined interval, where the scale can be defined based on the data domain.

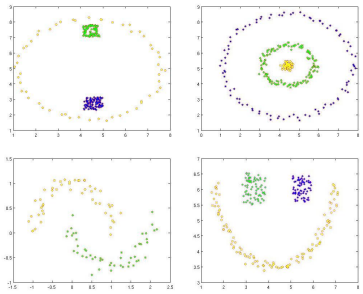


Figure 2. Results of our incremental clustering algorithm on synthetic datasets.

Figure 3. Representative images of object categories.

Moreover, to mitigate the effect of order in which the data points are processed, a buffer is used as proposed in [12]. To ensure that no point remains in the buffer forever, the points in the buffer are clustered by considering $min(d_1, d_3)$, (see Algo.1). As clusters grow, they may overlap or become adjacent. If majority of points from one cluster are *close* to the points of the other one in the overlapping region, then the two clusters are merged. The cluster representative of the merged cluster is calculated as the weighted average of the two representatives where the weight is the normalized size of the larger cluster. Applying our algorithm on the component *object size* gives a summary of different categories of objects in the scene, as shown in Fig. 3.

4.1. Temporal Incremental Clustering

Our incremental clustering algorithm to can also be used to cluster data in the temporal domain. To define the similarity measures in this domain, we use Allen's [1] temporal interval algebra. The similarity measures are shown in Fig. 4. Containment also takes into consideration *start together* and *end together* relations. Clustering in the temporal domain using standard distance metrics clusters the data

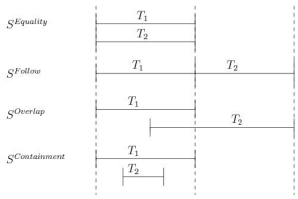


Figure 4. Temporal similarity measures defined using temporal interval algebra.

Figure 5. Trajectories clustered using (a) spatial and (b) spatio-temporal similarity. The two yellow tracks in (b) occurred together in time but the third yellow track in (a) is only spatially similar to them.

Figure 6. $C_{labeled}$**(green),** $C_{unlabeled}$ **(red)**

spatially and does not capture the temporal semantics associated with the data. Clusters created using the temporal similarity measures depict the actual temporal relations in the data. For example, if we cluster the trajectories spatially, the clusters show the different trajectories taken. Clustering in both the spatial and temporal components and composing the clusters gives a summary of which trajectories were taken in the same time period and which were not, as shown in Fig. 5.

4.2. Semi-Supervised Incremental Clustering

If labeled data is present *a priori*, instead of $\Omega = \phi$ in line 7 of Algo. 1, start with $\Omega = \{\text{set of labeled clusters}\}$. In case, labeled and unlabeled data is mixed, the incremental clustering algorithm remains the same as in Algo. 1. In both the cases, if a cluster has labeled data then all the unlabeled elements in that cluster are assigned the same label. We call such a cluster *labeled cluster*. The clusters formed give the distribution of data along with the number of elements in the labeled clusters. We give a small example to show how labeled data can be used for summarizing over specific component values. In a parking video, labeled data denoting correct parking is provided for components *object size* and *object position*. Semi-supervised incremental clustering and composition of clusters from these component spaces, gives two clusters $C_{labeled}$ and $C_{unlabeled}$. The elements of $C_{labeled}$ are indicative of number of times correct parking took place. $C_{unlabeled}$ shows a parking style which is deviant in this context. Fig. 6 shows two elements of $C_{labeled}$ and the only element of $C_{unlabeled}$.

5. Cluster Algebra and Automated Association Selection

We extend the algebraic operations defined in [2] to the temporal domain. These operations are possible if, while clustering in one of the component spaces of a tuple, the values of the other components in the tuple are are also stored.

5.1. Algebraic operations on clusters

Let (X, Y) be a tuple, then $X \times Y$ is a 2D space. Suppose $S_x(x) = x^*$ implies that x and x^* are *close* (or similar) based on the distance measure used on the X-component. Then, $C_{x^*} = \{(x, y)|S_x(x) = x^*\}$ is the cluster for the value $x^* \in X$ and $C_{y^*} = \{(x, y)|S_y(y) = y^*\}$ be the cluster for the value $y^* \in Y$. The algebraic operations on C_{x^*} and C_{y^*} are defined as:

1. Composition:

$$C_{x^* \otimes y^*} = \{(x, y)|S_x(x) = x^*, (x, y) \in C_{y^*} \text{ and } S_y(y) = y^*, (x, y) \in C_{x^*}\} \quad (3)$$

2. Intersection:

$$C_{x^* \cap y^*} = \{(x, y)|S_x(x) = x^*, (x, y) \in C_{x^*} \text{ and } S_y(y) = y^*, (x, y) \in C_{y^*}\} \quad (4)$$

3. Union: Union of clusters is defined for clusters from the same component space. Let C_{x^*} and $C_{x'}$ be two clusters from the X-component space. Then,

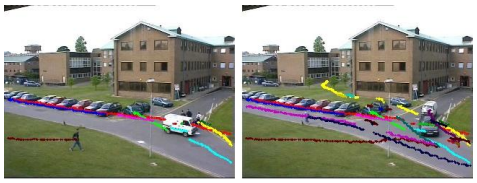
$$C_{x^* \cup x'} = \{(x, y)|S_x(x) = x^*, (x, y) \in C_{x^*} \text{ or } S_x(x) = x', (x, y) \in C_{x'}\} \quad (5)$$

4. Temporal Composition: Let $C^R_{t^*} = \{(x, t)|S^R_t(t) = t^*\}$ be a cluster in the temporal domain, where, R is any one of the temporal similarity measures: equality,

follow, containment or overlap. Then, if all points in C_{x^*} occur during the time interval t^*, the clusters C_{x^*} and $C_{x'}$ are temporally composed as:

$$C_{(x^* \cup x') \otimes t^*} = \{(x,t) | S_x(x) = x', (x,t) \in C^R_{t^*} \text{ and } S_x(x) = x^*, (x,t) \in C^{Equality}_{t^*}\} \quad (6)$$

Clusters in the high dimensional spaces are created using these algebraic operations on the component clusters. For



Frame no. 900 Frame no. 2340

Figure 7. Trajectories of objects are found by composition of clusters from different component spaces.

example, in the PETS2001 video, our incremental clustering algorithm is applied to each of the components:*(s)ize, (c)olor correlogram, (p)osition and (t)ime of presence of the objects*. Then, the composed cluster is given as:

$$C_{s \otimes p \otimes c \otimes t} = \{(s_i, p_i, c_i, t_i) | S_s(s_i) = s, (s_i, p_i, c_i, t_i) \in C_{p \otimes c \otimes t} \text{ and } S_p(p_i) = p, (s_i, p_i, c_i, t_i) \in C_{s \otimes c \otimes t} \text{ and } S_c(c_i) = c, (s_i, p_i, c_i, t_i) \in C_{s \otimes p \otimes t} \text{ and } S^{Follow}_t(t_i) = t, (s_i, p_i, c_i, t_i) \in C^{Follow}_{s \otimes p \otimes c}\} \quad (7)$$

Composition of clusters from these spaces give the trajectories of all the objects. We observe that the object trajectories are found in spite of occlusion, as can be seen in Fig. 7. The yellow trajectory shows that despite occlusion continuity and similarity are established.

5.2. Usualness measure associated with a cluster

Any event, usual or unusual has a certain time period. Therefore, any event that occurs for a single frame is treated as noise. To eliminate this noise from unusual event analysis, we modify the *usualness* measure of a cluster C given in [2]. Let Ω be the set of all clusters and $C \in \Omega$ be a cluster of size x, where size of a cluster is the number of elements in the cluster. The *usualness* measure of a cluster C is defined as the function:

$$p(C) = \begin{cases} -1 & x < t_1 \\ e^{-(x-t_2)^2/(2*\frac{(t_2-t_1)^2}{3})} & t_1 \leq x \leq t_2 \\ 1 & x > t_2 \end{cases} \quad (8)$$

where,
t_1 and t_2 are thresholds on the rate of growth of the *usualness* of a cluster.
If the *usualness* measure is -1, the cluster is treated as noise. We use this measure to discover the usual as well as unusual cluster values in component spaces as well as in the higher dimensional spaces of composed clusters.

Algorithm 1 Our Incremental Clustering Algorithm

Notation:
  Ω: Set of all clusters
  n: current number of clusters
  C_i: i^{th} cluster
  c_i: i^{th} cluster representative
  p_i: i^{th} data point
Initially, $\Omega = \phi$ and $n = 0$
$C_1 = \{p_1\}$, $c_1 = p_1$, $n = 1$
for $i = 2$ to ...
  $distance_k = \min_{1 \leq j \leq n} \{d(c_j, p_i)\}$
  $wt = e^{-distance_k}$
  $cnew_k = wt * c_k + (1 - wt) * p_i$
  $mid\_pt = (c_k + p_i)/2.0$
  Calculate
    $d_1 = d(c_k, cnew_k)$
    $d_2 = d(mid\_pt, cnew_k)$
    $d_3 = d(cnew_k, p_i)$
  if $d_1 = min(d_1, d_2, d_3)$
    $C_k = C_k \cup \{p_i\}$
    $c_k = cnew_k$
  else if $d_2 = min(d_1, d_2, d_3)$
    put the point in a buffer.
  else if $d_3 = min(d_1, d_2, d_3)$
    $C_{n+1} = \{p_i\}$
    $c_{n+1} = \{p_i\}$
    $n = n + 1$
    cluster all elements in the buffer.
    $\forall$ existing clusters
    merge overlapping or adjacent clusters.
  end if
end for
if size of buffer > 0
  cluster all elements in the buffer.
  merge overlapping or adjacent clusters
end if

5.3. Automated association selection

Suppose the video is represented as a tuple $T =< A, B, C, D, E >$. At time t, the tuple has values, say $T_t =< a, b, c, d, e >$ which belong to component clusters, say, $< A_1, B_1, C_1, D_1, E_1 >$. Therefore, the tuple T_t becomes a *cluster tuple*, that is, a tuple of component clusters. Incrementally clustering the *cluster tuples* with component-wise equality as the similarity measure and calculating its *usualness* measure, gives the usualness of the event. If the *cluster tuple* is usual, the event is usual. In case, *usualness* measure of any of the component clusters is lower than a pre-specified value, then the event is unusual. No further compositions need to be considered in the above two cases. If each of the component clusters is *usual* but the *cluster tuple* is unusual, then the event is unusual because of co-occurrence of some component values. If required, its subtuples can be searched hierarchically to find the *unusual* subtuple.

For example, the yellow track in Fig. 8 is an unusual event. It is usual in each of the components of tuple in Eq. 15 but composition of the clusters is unusual. Association selection shows that it is unusual because of the subtuple $<$ object size, object position, time $>$. On considering it's subtuples, it is found that it is unusual because of the co-occurrences of *object position* and *time*. This is consistent as the yellow trajectory is the only one during that time interval although in general this path is a usual one (see Fig. 10(a)).

Figure 8. Yellow trajectory is an unusual event for this time of day.

6. Rough set based interpretation of cluster algebraic operations

We apply rough set theory to show that the component based clusters are a good approximation to the high dimensional clusters. Let $C_{(x^*,y^*)}$ denote the cluster in the high dimensional space which is found by directly clustering in that space,and (x^*, y^*) be its cluster representative. Let $C_{x^* \otimes y^*}$ denote the composition of clusters C_{x^*} and C_{y^*}, from X and Y component spaces, respectively. In general, it is not necessary that $C_{x^* \otimes y^*}$ will be the same as $C_{(x^*,y^*)}$. The clusters that we get by applying the cluster algebraic operations are approximations of the high dimensional clusters that are formed by clustering directly in the high dimensional space. This holds when the similarity measure used in the high dimensional space is a monotonicity preserving function of the similarity measures used in the component spaces. Let $\overline{C}_{(x^*,y^*)}$ be the upper approximation of $C_{(x^*,y^*)}$ while $\underline{C}_{(x^*,y^*)}$ be the lower approximation of the cluster $C_{(x^*,y^*)}$. Then,

$$\begin{aligned}\overline{C}_{(x^*,y^*)} = \{(x,y) | & 0 \leq S_x(x,x^*) \leq 1, (x,y) \in C_{y^*} \\ & \text{and } 0 \leq S_y(y,y^*) \leq 1, (x,y) \in C_{x^*} \text{ such that} \\ & \text{at least one of } S_x(x,x^*) \text{ and } S_y(y,y^*) \text{ is non-zero.}\} \\ \implies \overline{C}_{(x^*,y^*)} &= C_{x^* \cap y^*} \end{aligned} \tag{9}$$

and,

$$\begin{aligned}\underline{C}_{(x^*,y^*)} = \{(x,y) | & S_x(x,x^*) = 1, (x,y) \in C_{y^*} \\ & \text{and } S_y(y,y^*) = 1, (x,y) \in C_{x^*}\} \\ \implies \underline{C}(x^*,y^*) &= C_{x^* \otimes y^*} \end{aligned} \tag{10}$$

where, $0 \leq S_x(x,x^*) \leq 1$ is the extent of similarity of x with x^*. It is defined as:

$$S_x(x,x^*) = \begin{cases} 1 & x \in C_{x^*} \\ e^{-d(x,x')} & x \notin C_{x^*} \end{cases} \tag{11}$$

where, d is a well-defined distance measure and m is the size of the cluster C_{x^*}.

$$d(x,x') = \min_{1 \leq i \leq m} \{d(x,x_i) | x_i \in C_{x^*}\} \tag{12}$$

By definition, $\underline{C}_{(x^*,y^*)} \subseteq \overline{C}_{(x^*,y^*)}$. We define the rough membership of elements of a 2D cluster as:

$$R(x,y) = \frac{S_x(x,x^*) + S_y(y,y^*)}{2} \tag{13}$$

Thus, $0 \leq R(x,y) \leq 1$. The rough membership of all elements in the cluster $\underline{C}_{(x^*,y^*)} = 1$ and that of the elements in $\overline{C}_{(x^*,y^*)}$ is in the interval $(0,1]$. Although here we have defined the extent of similarity for a 1D cluster and rough membership for a 2D cluster, they are extendible to any higher dimension. We define the degree of approximation as

$$DoA_{(x^*,y^*)} = 1 - \frac{\sum_{i=1}^{N} R(x_i,y_i)}{N} \tag{14}$$

where, (x_i, y_i) are elements of the 2D cluster and N is the size of that cluster. If the degree of approximation is 0 then $C_{(x^*,y^*)} = C_{x^* \otimes y^*}$. In this case, $\overline{C}_{(x^*,y^*)} = \underline{C}_{(x^*,y^*)}$.

In Fig. 9, the blue points are the cluster C_{x^*} in X-space, while the magenta points belong to the cluster C_{y^*} in Y-space. The green points belong to the cluster $C_{(x^*,y^*)}$, while the yellow points belong to both $C_{(x^*,y^*)}$ and $C_{x^*\otimes y^*}$. Here, $DoA_{(x^*,y^*)} = 0.0022$. In the

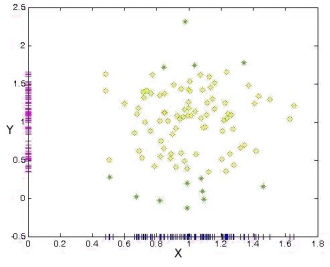
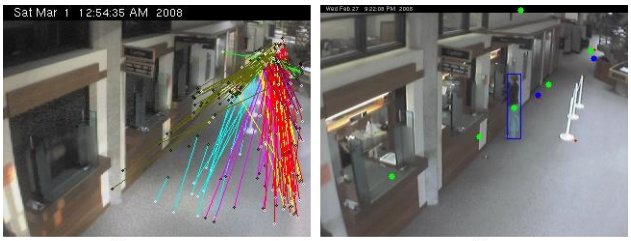


Figure 9. The yellow points belong to the composed cluster $C_{x^*\otimes y^*}$, while the green points belong to $C_{(x^*,y^*)}$. The composed cluster is a very close approximation of the actual 2D cluster.



(a) (b)

Figure 10. (a) Increntally clustering the tracks to find common paths, (b) Usual entry (blue), exit (red) and wait (green) locations.

PETS2001 video, we cluster the trajectories by clustering in the high dimensional space as well as clustering in component spaces and then composing the clusters. For the largest cluster, the degree of approximation is 0.0318, showing that the composed cluster is a good approximation of the high dimensional cluster.

7. Results

We have worked with videos where the frames are processed as they arrive via the Internet from webcams placed somewhere in the world. The video representation used for these are at the level of an object:

$$V =< size, color\ correlogram, position, time > \quad (15)$$

The trajectories are the clusters obtained by temporal composition of clusters from each of the component spaces of tuple in Eq. 15. The trajectories are represented as:

$$V =< start\ loc., end\ loc., start\ time, end\ time > \quad (16)$$

The trajectories are clustered in each of these components and composition of those clusters give the required trajectory clusters.

7.1. Online video 1

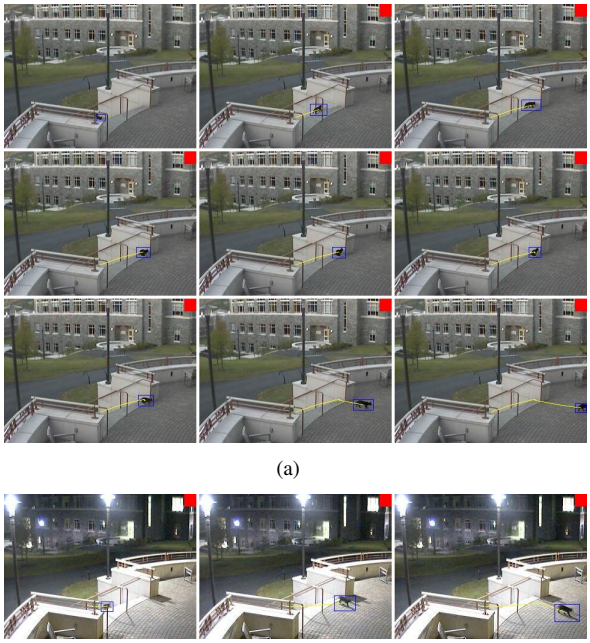
We have worked with 24 hours of online video from [8]. Object level summarization is generated by clustering across frames in the components of Eq. 15 and composing them. These composed clusters give the trajectories of each object that comes into the scene. Then, the trajectories are incrementally clustered in the components of Eq. 16 and composition of these clusters give a summary of the usual paths in the scene. The thresholds for the *usualness* measure are set to $T_1 = 10$ and $T_2 = 50$. Fig. 10(a) shows the usual paths of movement in the scene. The white point represents starting of the track while the black point represents the end. The average rate of arrival of frames through the Internet is 1 in 2-3 seconds. Therefore, many a time objects are detected for the first time in the middle of the scene or they suddenly vanish from the middle of the scene. This leads to detection of entry and exit location in the middle of the scene. In spite of this, our system is able to locate the actual exits and entrances, since they are most usual. Fig. 10(b) shows the usual entry (blue), exit (red) locations, locations where objects are static (green). The visualization of these clusters at any time t, gives a summary of the events that have occurred till that point in time.

7.2. Online video 2

We have analysed 24 hours of video from [6]. Object trajectories are obtained as defined above. At a higher level, objects are also clustered on the basis of their trajectory, time of presence and size. In this case, we find that most objects are similar, which is true since either individuals or groups of people walk around. We find a few instances of deviant objects/events, two of which are a cat prowling in the area, once in late evening Fig 11(a) and again at night Fig. 11(b). The usual paths in the scene are shown in Fig. 12

7.3. Online video 3

We experimented on another 24 hour video from a webcam at a parking lot [7]. As we get the frames from the webcam through the Internet, we incrementally cluster them to summarize the information in the video on two aspects: locations where objects are static for a long period of time and



(a)

(b)

Figure 11. The cat is detected as a deviant object in the scene

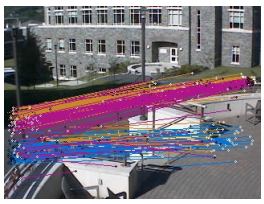


Figure 12. The usual paths in the scene.

the trajectories. As above, trajectories are found by clustering in the components of the tuple in Eq. 15 and then trajectory clusters are obtained as compositions of clusters in the component spaces of Eq. 16. Fig.13(a) shows the parking slots discovered in an unsupervised manner. Fig.13(b) shows two clusters of trajectories which are of objects of different sizes moving in the same direction.

8. Conclusion

We have shown that a summary of the information in a video can be extracted using clustering as a key tool. We have proposed an incremental clustering algorithm and used component based clustering and cluster algebra for summarization as well as automatic selection of component clusters to discover unusual patterns in a surveillance video. A

(a) (b)

Figure 13. (a) The green dots denote the parking locations, found by clustering on the component *position of objects*. (b) Each color denotes a cluster of trajectories, where similarity is based on *size of objects* to which these trajectories belong.

rough set based analysis of the cluster algebra is also presented. We conclude that summarizing a video on multiple components and at multiple layers of abstraction provides greater insight into the events that occur in the scene.

References

[1] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic, 1997. in O. Stock (ed.) "Spatial and Temporal Reasoning", Kluwer Academic Publishers.

[2] A. Choudhary, S. Chaudhury, and S. Banerjee. Unusual activity analysis in video sequences. *In Proceedings RSFDGrC, Lecture Notes in Artificial Intelligence*, 4482:443–450, 2007.

[3] N. P. Cuntoor, B. Yegnanarayana, and R. Chellapa. Activity modeling using event probability sequences. *In IEEE Trans. Image Processing*, 17(4):594–607, April 2008.

[4] D. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 1987.

[5] J. A. Hartigan. Clustering algorithms. *John Wiley New York*, 1975.

[6] http://149.43.156.107/view/index.shtml.

[7] http://152.3.114.19/view/view.shtml.

[8] http://161.28.134.223/view/index.shtml.

[9] P. Natrajan and R. Nevatia. Edf: A framework for semantic annotation of video. *Workshop on Semantic Knowledge in Computer Vision*, 2005.

[10] J. Oh and B. Bandi. Multimedia data mining framework for raw video sequence. *In Proceedings of International Workshop Multimedia Data Management*, 2002.

[11] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg. Webcam synopsis: Peeking around the world. *In Proc. ICCV*, pages 1–8, 2007.

[12] J. Roure and L. Talavera. Robust incremental clustering with bad instance orderings: A new strategy. *In IBERAMIA*, pages 136–147, 1998.

[13] P. K. Turaga, A. Veeraraghavan, and R. Chellappa. From videos to verbs: mining videos for activities using a cascade of dynamical systems, 2007.