Multiple Agent Event Detection and Representation in Videos

Asaad Hakeem, and Mubarak Shah
University of Central Florida, Orlando FL 32816
{ahakeem,shah}@cs.ucf.edu

Abstract

We propose a novel method to detect events involving multiple agents in a video and to learn their structure in terms of temporally related chain of sub-events. The proposed method has three significant contributions over existing frameworks. First, we present the concept of a *video event graph*, to learn the event structure from training videos. The video event graph is composed of temporally correlated sub-events, which is used to automatically encode the *event correlation graph*. The event correlation graph signifies the frequency of occurrence of conditionally dependent sub-events. Second, we pose the problem of event detection in novel videos as clustering the maximally correlated sub-events, and use normalized cuts to determine these clusters. The principal assumption made in this work is that the events are composed of highly correlated chain of sub-events, that have high weights (association) within the cluster and relatively low weights (disassociation) between clusters. Last, we recognize the importance of representing the variations (in the temporal order of sub-events) occurring in a event and encode the probabilities directly into our representation. We show results of our learning and detection of events for videos in the meeting, surveillance, and railroad monitoring domains.

Introduction

The world that we live in is a complex network of *agents* and their interactions which we term *events*. These interactions can be visualized in the form of a hierarchy of events and sub-events. An instance of an event is a composition of directly measurable low-level actions (which we term sub-events) having a temporal order. For example, a voting event is composed of a sequence of move, raise and lower hand sub-events. Also, the agents can act independently (e.g. voting) as well as collectively (e.g. touchdown in a football game) to perform certain events. Hence, in the enterprise of machine vision, the ability to detect and learn the observed events must be one of the ultimate goals. In literature, a variety of approaches have been proposed for the detection of events in video sequences. Most of these approaches can be arranged into three categories based on their approach to event detection. First, approaches where event models are pre-defined include force dynamics (Siskind 2000), stochastic context free grammars (Bobick and Ivanov 1998), state machines (Koller, Heinze, and Nagel 1991), and PNF Networks (Pinhanez and Bobick 1998). These approaches either manually encode the event models or provide constraints (grammar or rules) to detect events in novel videos. Second, approaches that learn the event models such as Hidden Markov Models (HMMs) (Ivanov and Bobick 2000, Brand and Kettnaker 2000), Coupled HMMs (Oliver, Rosario, and Pentland 1999), and Dynamic Bayesian Networks (Friedman, Murphy, and Russell 1998) have been widely used in the area of activity recognition. The above learning methods either model single person activities or require prior knowledge about the number of people involved in the events and variation in data may require complete re-training, so as to modify the model structure and parameters to accommodate those variations. Similarly, there is no straight-forward method of expanding the domain to other events, once training has been completed. Third, approaches that do not model the events, but utilize clustering methods for event detection include co-embedding prototypes (Zhong, Shi, Visontai 2004), and spatio-temporal derivatives (Zelnik-Manor and Irani 2001). These methods find event segments by spectral graph partitioning (e.g. normalized cut) of the weight (similarity) matrix. These methods assume maximum length of an event and are restricted to single person non-interactive event detection.

What is missing in these approaches is ability to model long complex events involving multiple agents performing multiple actions simultaneously. Can these approaches be used to automatically learn events involving unknown number of agents? Will the learnt event model still hold for a novel video, in case of interfering events from an independent agent? Can these approaches extend their abstract event model to representations related to human understanding of events? Can a human communicate his or her observation of an event to a computer or vice versa? These questions are addressed in this paper, where event models are learnt from training data, and are used for event detection in novel videos. *Event learning* is formulated in a probabilistic framework while *event detection* is treated as a graph-theoretic clustering problem. The primary objective of this work is to detect and learn the complex interactions of the multiple agents performing multiple actions in the form of domain events, without prior knowledge about the number

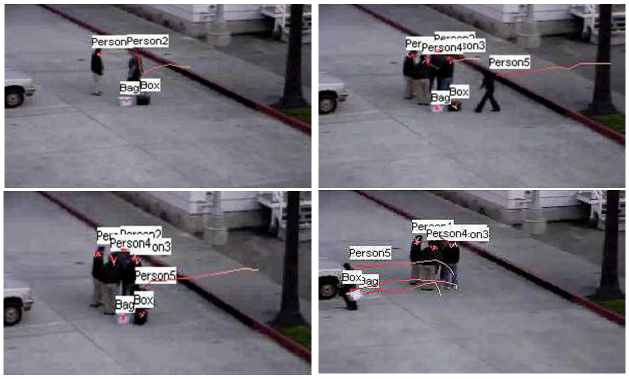

Figure 1: Automated detection of sub-events for stealing video. Using the tracked trajectories, the sub-events of each agent are detected, and frames 37, 119, 127, and 138 of the video are shown.

of agents involved in the interaction and length of the event. Another objective is to present a coherent representation of these domain events, as a means to encode the relationships between agents and objects participating in a domain event. Formally, a *domain event* is defined as a collection of actions performed by one or more agents. Also, we term these actions as *video events*, since they are directly measurable from the video (e.g. move, pick, enter, etc.). In this paper, events refer to *domain events*, and sub-events refer to *video events*, unless otherwise stated.

Although CASE^E (Hakeem, Sheikh, Shah 2004) is an existing multiple agent event representation, the proposed method caters for three of its shortcomings. Firstly, we automatically learn the domain event structure from training videos and encode the *domain event ontology*. This has a significant advantage, since the domain experts need not go through the tedious task of determining the structure of events by browsing all the videos in the domain. Secondly, we recognize the importance of representing the variations in the temporal order of the sub-events occurring in a domain event and encode it directly into our representation. These variations in the temporal order of sub-events occur due to the style of execution of events for different agents. Finally, we present the concept of a video event graph (instead of event-tree) for event detection in videos. The reason for departing from the temporal event-tree representation of the video is that it fails to detect events when there are interfering sub-events from an independent agent, present in the tree structure of the novel video, which were not present in the actual event tree structure. Also, it fails to represent the complete temporal order between sub-events, which can easily be represented by video event graphs.

For learning the domain events from training videos, firstly, we introduce the notion of video event graph, which is a *Directed Acyclic Graph* (DAG) for representing the temporal relationship of sub-events in a video. In the video event graph each *vertex* represents a sub-event and each *directed edge* provides the temporal relationship between two sub-events. These temporal relationships are based on the interval algebra in (Allen and Ferguson 1994), which is a more descriptive model of relationships compared to the low level abstract relationship model of HMMs. Secondly, using the video event graph, we determine the event correlation graph, which is an *Edge-Weighted Directed Hypergraph* (EWDH) representing the temporal conditional dependency between sub-events. The EWDH has a number of vertices representing the *unique* sub-events, hyperarcs that contain *ordered* subset of the vertices, and weights on the hyperarcs that denote the frequency of occurrence of the conditionally dependent sub-events. Intuitively, the event correlation graph encodes the frequency of conditionally dependent sub-events occurring in the video event graph. Also, the learnt event model is scalable to the number of agents involved in the event.

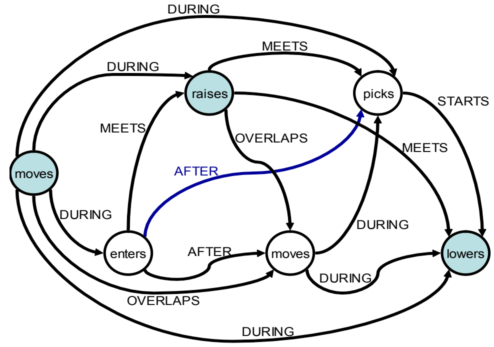

Figure 2: Partial video event graph for a sequence containing two agents performing actions simultaneously. The sub-events (actions) are the vertices and the temporal relationships between sub-events are shown as directed edges between vertices. Agent1's sub-events are greyed while Agent2's are white to provide a visual distinction between their actions.

For event detection in novel videos, we estimate a *Probabilistic Network* (PN) of sub-events, which is a pair $B = (G, \theta)$, where G is the video event graph, and θ are the weights obtained from the hyperarcs of the event correlation graph. This PN forms the probabilistic weight matrix used for spectral graph partitioning. Thus, normalized cut is applied recursively to this PN, to cluster the highly correlated sub-events. These clusters represent the domain events, and the *event structure* composed of sub-events and their temporal order is extracted using graph partitioning. Lastly, as an application of the framework, we modify CASE^E to represent the variations in temporal order of sub-events, occurring in an event. We also empirically demonstrate our framework for event detection in meeting, surveillance, and railroad monitoring domains.

Learning the Event Structure

In this section, we address some issues of learning the event structure from training videos. Let $f(\mathbf{p}, t)$ represent a continuous video signal, indexed by spatial and temporal coordinates respectively. Each object is represented in terms of its label and motion, e.g. $\{\mathsf{person}_a, \ \mathbf{u}_a\}$, where $\mathbf{u}_a$ = $\{\ (x_1, y_1), (x_2, y_2), \ldots (x_N, y_N)\ \}$ is the trajectory of person_a's centroid. Here it is assumed that the lower-level tasks of object detection, classification and tracking have been performed for a stationary camera. Also, it is important to note that since it is the *relative* concept of motion that we are interested in (e.g. where did agent_1 move to with respect to object_2?), two-dimensional projections of three-dimensional world trajectories are sufficient for event representation (barring some degenerate configurations). An example video of stealing event is shown in Figure 1. These sub-events are input into a system that represents them in terms of a video event graph described next.

Video Event Graph

The temporal structure of events in a video can be intuitively represented as a *Directed Acyclic Graph* (DAG), with each vertex corresponding to a sub-event, and each edge corresponding to the temporal relationship between two vertices (e.g. AFTER). The video event graph is directed since there is a temporal order between nodes and acyclic since time is monotonically increasing. An example DAG for a small voting sequence is shown in Figure 2. More formally, a video event graph is a DAG, $G = (V, E)$ where $V = \{v_1, v_2, ..., v_n\}$; $v_i \in C$, and C is the set of n automatically measured sub-events; $E = \{e_1, e_2, ..., e_m\}$, where $e_i \in T$ and e_i are directed edges, and T is the set of temporal variables in the interval algebra of (Allen and Ferguson 1994). A naive formulation of the problem would be to consider a complete (video event) graph, and estimate correlated chains of sub-events in order to determine the *Event Correlation Graph* (as detailed in the next section). The problem with the complete graph formulation is that sub-events are not dependent on *all* their predecessor sub-events, rather they are dependent on their proximal predecessor sub-events. For example, a person *raising* a hand at the start of the video has nothing to do with *picking* a book sub-event, occurring after a few minutes have passed. Thus *transitive reduction* based upon proximity x is applied to the video event graph. This does not imply that we constrain our events to be a maximum of x length, rather it denotes that the events are composed of x-1*th* order Markov chain of sub-events. That is, each sub-event is conditionally dependent upon (at most) x-1 parent sub-events, which is true for most of the events in the considered domains.

Event Correlation Graph

Given the proximity-based transitively reduced video event graph, we estimate the Event Correlation Graph (ECG). The ECG is an *Edge-Weighted Directed Hypergraph* (EWDH) estimated by determining the frequency of higher order Markov chains of sub-events in a video event graph. The reason for estimating higher order Markov chains instead of first order chains is that the sub-events are usually conditionally dependent upon more than one sub-event. More formally, EWDH is a hypergraph $G = (V, E, W)$ having a number of *vertices* $V = \{v_1, v_2, ..., v_n\}$ representing n *unique* sub-events, *hyperarcs* $E = \{e_1, e_2, ..., e_m\}$ are backward arcs (*B-arcs*), and *weights* $W = \{w_1, w_2, ..., w_m\}$ are the weights on each B-arc corresponding to the frequency of occurrence of conditionally dependent sub-events. Each B-arc is an *ordered* pair of vertices $e_i = (P_i, v_i)$ where $P_i \subseteq V$, and P_i is an ordered set representing the parent sub-events of v_i. Since all the hyperarcs of the EWDH are B-arcs, the EWDH can also be termed as a *B-graph*. An example of a partial ECG estimated from a sample $voting$ video is given in Figure 3. An ordinary graph is a 2-uniform hypergraph, where k-uniform represents that each hyperedge has a $cardinality$ of k vertices. We do not enforce a k-uniform hypergraph, rather we allow the hypergraph to have a maximum x edge cardinality (4 in our experiments). This allows the frequency encoding of a sub-event v_i, having a maximum of x-1 parent sub-events, for the given video

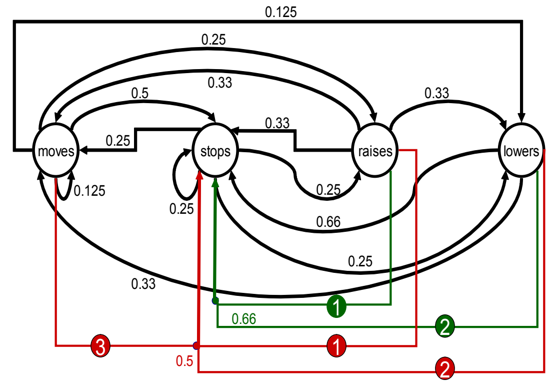

Figure 3: Partial event correlation graph for the sample video of *voting* events. The sub-events are the vertices, and the conditional probabilities between sub-events are represented by the weights on the hyperarcs. Note that a single example of hyperarcs with cardinality of 3 and 4 are shown respectively in green and red, so as to keep the figure comprehendible. Also, the circled number on the hyperarc represents the order index in P_i, e.g. the B-arc of cardinality 4 represents P(stops|moves,lowers,raises).

event graph. The equations for estimating the weights w_i on hyperarcs e_i for cardinality of $X \in \{2, 3, 4\}$ are respectively given by:

$$P(v_i^t | v_j^{t-1}) = \frac{P(v_i^t, v_j^{t-1})}{P(v_j^{t-1})}$$

$$P(v_i^t | v_j^{t-1}, v_k^{t-2}) = \frac{P(v_i^t, v_j^{t-1}, v_k^{t-2})}{P(v_j^{t-1} | v_k^{t-2}) P(v_k^{t-2})}$$

$$P(v_i^t | v_j^{t-1}, v_k^{t-2}, v_l^{t-3}) = \frac{P(v_i^t, v_j^{t-1}, v_k^{t-2}, v_l^{t-3})}{P(v_j^{t-1} | v_k^{t-2}, v_l^{t-3}) P(v_k^{t-2}, v_l^{t-3})}$$

where v_i^t represents a sub-event i occurring at index t, and $Agent(v_i^t) = Agent(v_a^b)$; $a \in \{j, k, l\}$, $b \in \{t\text{-}1, t\text{-}2, t\text{-}3\}$, which enforces the current and parent sub-events to be performed by the *same* agent. This is necessary since sub-events performed by different agents are not conditionally dependent on each other. Note that the ECG captures all the variations in temporal order of sub-events as well as the frequency of occurrence of the chain of sub-events in a video.

Event Detection in Novel Videos

After learning the set of events ξ_i in a supervised manner (as described above), *event detection* in novel video sequences is posed as clustering of highly correlated chain of sub-events in a Probabilistic Network (PN). A PN is formally denoted by $B = (G, \theta)$, where G is a graph, such as our video event graph, and θ are the parameters on the graph, which are determined by the likelihood estimates of the ECG parameters. The PN_p is constructed for each learnt event ξ_p and is then mapped to a weight matrix W_p. Finally, normalized cut is applied recursively to the estimated weight matrix, resulting in clusters of sub-events that represent the segmented events. This process is repeated for the all the learnt events, resulting in the extraction of the different events from the novel video.

Determining the Weight Matrix

In order to determine the weight matrix W for *Normalized cut*, we estimate the $PN_p = (G, \theta_p)$ for a specific event ξ_p, where G is the novel video event graph, and θ_p are the likelihood estimates obtained from the ECG parameters. Each

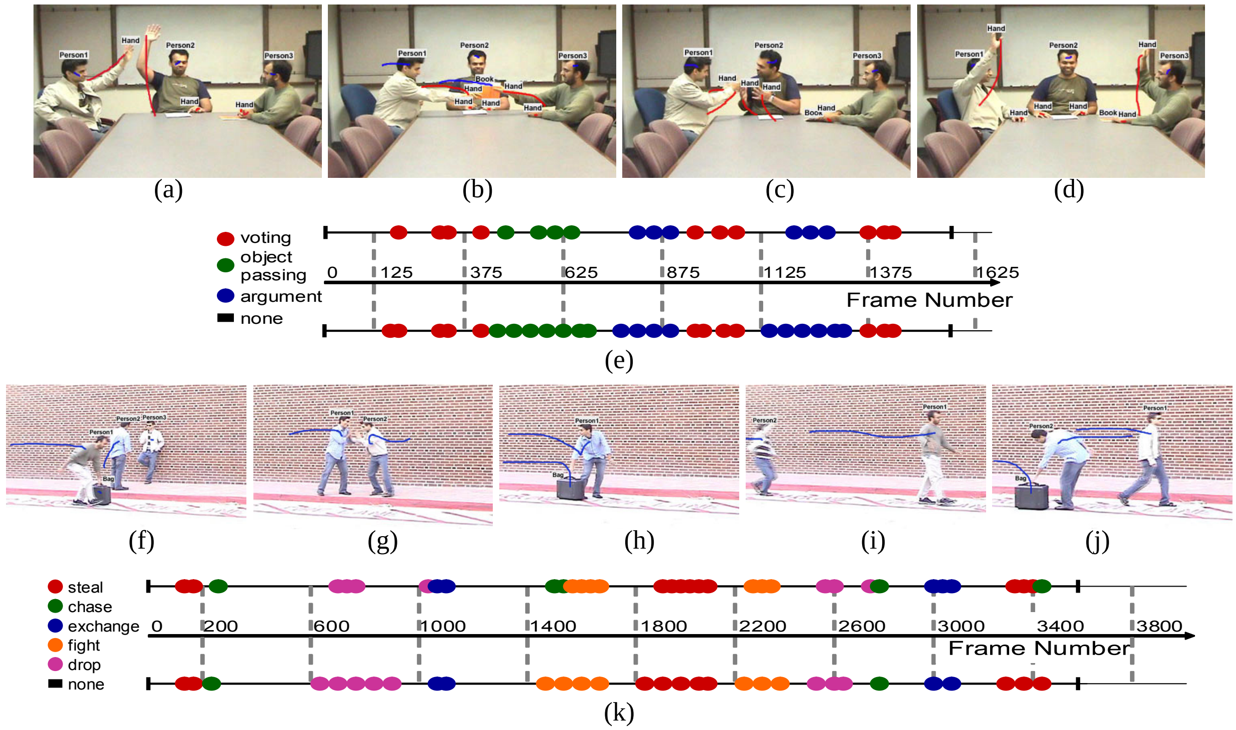

Figure 4: Event detection results using normalized cuts for meeting and surveillance domain test videos. (a)-(d) represent frame 328, 560, 755, and 1375 respectively of meeting video consisting of 1551 frames. (e) Time indexed clustering results for meeting video, where the top bar shows the actual event detection results and the bottom bar denotes the ground truth of the events. (f)-(j) represent frame 159, 2388, 2626, 2874, and 3125 respectively of surveillance video consisting of 3580 frames. (k) Time indexed clustering results for surveillance video.

weight w_{ij} of the weight matrix $\hat{W}_p$ is given by:

$$w_{\alpha\beta} = P(v_l^t | Pa(v_l^t)) = P(v_l^t | v_k^{t-1}, v_j^{t-2}, v_i^{t-3}) \quad (1)$$

where α is the index of sub-event v_l^t, and β is the index of $Pa(v_l^t)$ sub-event. $Pa(v_l^t)$ is the oldest parent sub-event that v_l^t conditionally depends upon, such that $Pa(v_l^t) \in \{v_k^{t-1}, v_j^{t-2}, v_i^{t-3}\}$. Note that a sub-event may be dependent upon one or two parent sub-events hence the estimates from hyperarcs of cardinality one and two are respectively inserted from the ECG to the weight matrix. Summarily, the above likelihood estimate assigns higher weights to the longer chain of sub-events, that occurred frequently in the video event graph of ξ_p. The final weight matrix $\hat{W}_p$ of the PN_p is upper triangle, since G is a directed acyclic graph. The weight matrix is made symmetric by $\tilde{W}_p = \hat{W}_p + \hat{W}_p^T$ (Ding 2004), where $\hat{W}_p^T$ is the transpose matrix of $\hat{W}_p$. The *Ncut* minimization function for weight matrices W_p and $\tilde{W}_p$ are equivalent and the proof is given in Appendix A.

Event Detection using Normalized Cut

Normalized cut (Shi and Malik 2000) is an unbiased method of partitioning a graph V into two (or more) segments A and B, since it uses a global criterion for graph segmentation, rather than focusing on the local features. The global criterion is given by:

$$Ncut(A,B) = \frac{cut(A,B)}{asso(A,V)} + \frac{cut(A,B)}{asso(B,V)} \quad (2)$$

where $cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$, $w(u,v)$ is the edge weight between vertices u and v, and $asso(A,V) = \sum_{u \in A, v \in V} w(u,v)$. If the *Ncut* criterion is minimized, then the graph is partitioned at the edges with the minimum cut weight, and the two partitions have maximum association within and minimum disassociation between their respective partitions. The minimization of the $Ncut$ criterion is achieved by finding the second smallest eigenvector of the generalized eigensystem:

$$(D - W)x = \lambda D x \quad (3)$$

where D is an $N \times N$ diagonal matrix with $d(i) = \sum_j w(i,j)$ as the diagonal elements, W is an $N \times N$ symmetric weight matrix, λ and x are the eigenvalues and eigenvectors respectively. The sub-event clustering algorithm using normalized cuts is summarized below:

1. Compute the weight matrix W and estimate the diagonal matrix D.
2. Solve $(D - W)x = \lambda D x$ to obtain the eigenvector with the second smallest eigenvalue, and use it to bipartition the graph by finding the splitting point such that the *Ncut* is minimized.
3. Decide if the current partition should be subdivided by checking that *Ncut* and *average edge weight* (that determines the association within a partition) are below their respective thresholds, and recursively repartition the segmented parts (if necessary).

The sub-event clusters determined by normalized cuts are the maximally correlated sub-events, given the likelihood estimates of the chain of sub-events. These segmented events have high weights between sub-events within the

	moves agent1	stops agent1	raises agent1	lowers agent1	raises agent2	moves agent2	lowers agent2	stops agent2	moves agent1	stops agent1
moves ag1	0	0.1826	0.1027	0.1598	0	0	0	0	0	0
stops ag1	0.1826	0	0.1141	0.0776	0	0	0	0	0	0
raises ag1	0.1027	0.1141	0	0.0717	0	0	0	0	0	0
lowers ag1	0.1598	0.0776	0.0717	0	0.013	0.1928	0	0.2029	0.1014	0
raises ag2	0	0	0	0.013	0	0.877	0.437	0.2087	0.163	0.0261
moves ag2	0	0	0	0.1928	0.877	0	0.4337	0.1826	0.1027	0.0457
lowers ag2	0	0	0	0	0.437	0.4337	0	0.2029	0.1014	0.0203
stops ag2	0	0	0	0.2029	0.2087	0.1826	0.2029	0	0.1141	0.0228
moves ag1	0	0	0	0.1014	0.163	0.1027	0.1014	0.1141	0	0.0457
stops ag1	0	0	0	0	0.0261	0.0457	0.0203	0.0228	0.0457	0

Figure 5: The *weight matrix* for a novel video containing two *voting* events. After application of *Normalized cut* algorithm, the two events are automatically segmented and are shown as red and blue patches.

cluster and relatively low weights between sub-events outside their clusters. An example of the voting *weight matrix* estimated using the ECG and the segmentation obtained after recursive application of *Ncut* is shown in Figure 5.

Experiments and Applications

We performed experiments for event detection in videos for the meeting, railroad monitoring, and surveillance domains. These videos contain multiple agents that act independently or interact with each other or objects. The videos in all domains (in our experiments) totalled 40977 frames, having 2673 sub-events and 157 events. We used three standard video datasets as well as other videos for training and testing the event detection framework. First, standard PETS dataset video was used as one of the training sequence for learning the voting event. Second, standard CAVIAR dataset videos were utilized as the training sequences for learning the object drop, and fighting events. Third, standard VACE dataset videos were adopted as the training and test sequences for the object drop, sneaking, stealing, loading, and unloading events. A total number of 57 videos were adopted for training 16 events. Initial object identification and labelling were performed manually and further tracking was attained using MEANSHIFT algorithm. Using the tracked trajectories, the temporally correlated sub-events were detected in real-time, that were further utilized for event learning. A list of all unique sub-events for the surveillance, railroad monitoring, and meeting domains, and the summary of results for event learning is provided in Table 1.

Using the learnt event models, event detection in novel video proceeded by estimating the weight matrix for each event model. Furthermore normalized cuts is applied to obtain event clusters, for a specific event model, in the novel video. The results for event detection using normalized cuts are summarized in Figure 4 for the meeting, surveillance, and railroad monitoring (not shown due to space limitation) domains. The precision and recall values for each test video is estimated using $Precision = \frac{\sum_{i,j}\psi(tde_i^j)}{\sum_{i,j}\psi(de_i^j)}$ and $Recall = \frac{\sum_{i,j}\psi(tde_i^j)}{\sum_{i,j}\psi(te_i^j)}$ respectively, where $\psi(tde_i^j)$ is the *true detected sub-events*, $\psi(de_i^j)$ is the *detected sub-events*, and $\psi(te_i^j)$ is the *true sub-events*, belonging to the i^{th} cluster of the j^{th} event. The summary of event detection results with precision and recall values are supplied in Table 2.

Sub-event list

Moves, Enters, Exits, Stops, Approaches, Extends, Holds, Passes, Blocks, Drops, Picks, Raises, Lowers, Sits, Stands, Pushes, Breaks, Collides, Switches, Hides, Emerges, Leaves, Crouches.

Event Name	Total Frames	No. of Videos	No. of Sub-events	No. of Events
voting	2938	5	221	26
argument	913	3	82	7
object passing	532	3	70	4
stealing	1386	4	129	4
chasing	680	3	55	3
fighting	2492	4	137	4
object exchange	1805	3	94	3
object drop	4484	4	81	4
loading	761	2	62	3
unloading	1485	1	38	6
sneaking	2259	3	77	3
railroad event1	2731	5	199	17
railroad event2	2314	4	85	6
railroad event3	1228	3	44	4
railroad event4	1577	6	131	10
railroad event5	1745	4	93	4

Table 1: Summary of results for different events in the training videos of the meeting, surveillance, and railroad monitoring domains.

Test Video	Total Frames	No. of Events	No. of Sub-events	Precision %	Recall %
Meeting	1551	15	224	92.3	85.7
Surveillance1	3580	13	335	92.1	86.7
Surveillance2	4256	12	209	81.3	87.2
Railroad	2260	9	307	80.2	72.3

Table 2: Summary of results for different events in the testing videos of the meeting, surveillance, and railroad monitoring domains.

Event Representation

A training video consists of sub-events that belong to a particular event, as well as sub-events detected due to noise or error in measurements. Thus the ECG captures the variations in temporal order of both types of sub-events in the training video of an event. We obtain an event representation that captures the temporal variations of only those sub-events that actually belong to the event, by applying Ncut to the training video and sub-event alignment of the segmented events. This alignment in sub-events for different instances of the same event is necessary to discover and encode the variations in the temporal relations between sub-events. Given the video event graph and the ECG for a training video of a particular event, we can estimate the *Probabilistic Network* (PN). Each vertex in the PN can be encoded with a complete case-frame (Hakeem, Sheikh and Shah 2004), rather than just the sub-event and the agent information. Once the events are segmented through normalized cuts of the PN, we pose the problem of aligning similar sub-events in segmented events, as a *maximum matching* of a bipartite graph. Given two graphs G_1 and G_2, which represent the instances $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ of the same event in the video event graph. By considering V_1

and V_2 as two disjoint sets of a bipartite graph G, we obtain weights w_{ab} between nodes V_1^a and V_2^b as a measure of similarity between case-frames. To that end, the Jaccard similarity measure is utilized which is defined and evaluated in (Hakeem, Sheikh and Shah 2004). The vertices are aligned through maximum matching of the bipartite graph such that vertices in V_1 have a one-to-one relationship to the vertices in V_2 of the other set. After alignment, the variation weights w_i^j in temporal relationships are computed using $w_i^j = \frac{\psi(T_i^j)}{\sum_k^n T_k^j}$, where w_i^j denotes the i^{th} weight for the j^{th} edge, $\psi(T_i^j)$ is the frequency of occurrence of the i^{th} temporal relationship for the j^{th} edge, and $\sum_k^n T_k$ is the normalizing factor representing all the n temporal relationships in the interval algebra for the j^{th} edge. The event representation is modified by introducing these temporal variation weights w_i^j on directed edges of the segmented event graph. An object exchange event representation example is shown in Figure 6, with the encoding of different styles of event execution in various videos, that may have alternate starting sub-events e.g. 'holds' can be before 'moves'. The ability to encode events with alternate starting sub-events is another advantage, lacking in previous representations.

Conclusion

The problem of detecting events in a video involving multiple agents and their interaction was identified. Event models were learnt from training videos having variations in the number of agents and the temporal order of sub-events. Event learning was formulated in a probabilistic framework, and the learnt event models were used for event detection in novel videos. Event detection was treated as a graph theoretic clustering of sub-events having high association within the event clusters and low association outside the clusters. We demonstrated our event detection framework on videos in the railroad monitoring, surveillance, and meeting domains. Domain event ontologies were automatically extracted from training videos, and an event representation was developed to cater for the temporal variations in the sub-events. We are interested in several future directions of this work including inference of causality in video sequences, and event-based retrieval of video.

Appendix A

Proof of equivalency for W and $\tilde{W}$ based minimizations

Given W, the global criterion for minimization of Ncut function is given by:

$$Ncut(A,B) = min[\frac{cut(A,B)}{asso(A,V)} + \frac{cut(A,B)}{asso(B,V)}]$$
$$= min[\frac{\sum_{i\in A,j\in B} P(v_j|v_i) + \sum_{i\in A,j\in B} P(v_i|v_j)}{\sum_{i\in A,k\in I} P(v_k|v_i)}$$
$$+ \frac{\sum_{i\in A,j\in B} P(v_j|v_i) + \sum_{i\in A,j\in B} P(v_i|v_j)}{\sum_{j\in B,k\in I} P(v_k|v_j)}]$$

where $I = A\bigcup B$, and since W is symmetric therefore $P(v_j|v_i) = P(v_i|v_j)$. Thus the above equation is equivalent to:

$$Ncut(A,B) = min[\frac{\sum_{i\in A,j\in B} 2P(v_j|v_i)}{\sum_{i\in A,k\in I} P(v_k|v_i)} + \frac{\sum_{i\in A,j\in B} 2P(v_j|v_i)}{\sum_{j\in B,k\in I} P(v_k|v_j)}]$$

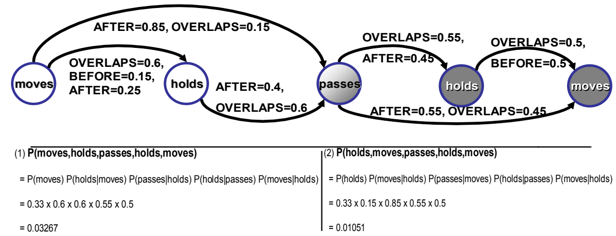

Figure 6: (top) *Object exchange* event representation for VACE videos. The weights on directed edges depict probability of occurrence of a specific temporal relationship between sub-events, while grey and white vertices represent sub-events of agents one and two respectively. (bottom) Two styles of object exchange, where (1) is more common than (2) in the VACE videos.

Similarly, given $\tilde{W}$, the global criterion for minimization of Ncut function is given by:

$$Ncut(A,B) = min[\frac{\sum_{i\in A,j\in B} 2P(v_j|v_i) + \sum_{i\in A,j\in B} 2P(v_i|v_j)}{\sum_{i\in A,k\in I} P(v_k|v_i) + \sum_{i\in A,k\in I} P(v_i|v_k)}$$
$$+ \frac{\sum_{i\in A,j\in B} 2P(v_j|v_i) + \sum_{i\in A,j\in B} 2P(v_i|v_j)}{\sum_{j\in B,k\in I} P(v_k|v_j) + \sum_{j\in B,k\in I} P(v_j|v_k)}]$$

and since $\tilde{W} = \hat{W} + \hat{W}^T$, where $\hat{W}$ is upper triangle matrix therefore $P(v_i|v_j) = P(v_i|v_k) = P(v_j|v_k) = 0$. Thus the above equation is reduced to:

$$Ncut(A,B) = min[\frac{\sum_{i\in A,j\in B} 2P(v_j|v_i)}{\sum_{i\in A,k\in I} P(v_k|v_i)} + \frac{\sum_{i\in A,j\in B} 2P(v_j|v_i)}{\sum_{j\in B,k\in I} P(v_k|v_j)}]$$

Since both the equations minimize the same function, thus it is equivalent to deal with W and $\tilde{W}$.

References

Allen, J. F., and Ferguson, G. 1994. Actions and Events in Interval Temporal Logic. In *JLC*, vol.4(5), pp.531-579.

Bobick, A. F., and Ivanov, Y. A. 1998. Action Recognition using Probabilistic Parsing. In *Proc. of CVPR*, pp.196-202.

Brand, M., and Kettnaker, V. 2000. Discovery and Segmentation of Activities in Video. In *PAMI*, vol.22(8), pp.844-851.

Ding, C. 2004. Tutorial on Spectral Clustering. In *International Conference on ML*.

Friedman, N., Murphy, K., and Russell, S., 1998. Learning the Structure of Dynamic Probabilistic Networks. In *Proc. Conf. on Uncertainty in AI (UAI)*, pp.139-147, Madison, WI.

Hakeem, A., Sheikh, Y., and Shah, M. 2004. CASE^E: A Hierarchical Event Representation for the Analysis of Videos. In *Proc. of AAAI*, pp.263-268.

Ivanov Y. A., and Bobick A. F. 2000. Recognition of Visual Activities and Interactions by Stochastic Parsing. In *IEEE Trans. on PAMI*, vol.22, pp.852-872.

Koller, D., Heinze, N., and Nagel, H. H. 1991. Algorithmic Characterization of Vehicle Trajectories from Image Sequences by Motion Verbs. In *Proc. of CVPR*, pp.90-95.

Oliver, N., Rosario, and B., Pentland, A., 1999. A Bayesian Computer Vision System for Modelling Human Interaction. In *Proc. of ICCVS*, pp.255-272.

Pinhanez, C., and Bobick, A. 1998. Human Action Detection Using PNF Propagation of Temporal Constraints. In *Proc. of CVPR*, pp.898-904.

Shi, J., and Malik, J. 2000. Normalized Cuts and Image Segmentation. In *IEEE Trans. on PAMI*, vol.22(8), pp.888-905.

Siskind, J. M., 2000. Visual Event Classification via Force Dynamics. In *Proc. of AAAI*, pp.149-155, Menlo Park, CA:AAAI Press.

Zelnik-Manor, L., and Irani, M. 2001. Event-based Analysis of Video. In *Proc. of CVPR*, pp.123-130.

Zhong, H., Shi, J., and Visontai, M. 2004. Detecting Unusual Activity in Video. In *Proc. of CVPR*, pp.819-826.