Multi-Agent Event Recognition

Somboon Hongeng and Ramakant Nevatia
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089
{hongeng, nevatia}@iris.usc.edu

Abstract

This paper presents a new approach to recognizing multi-agent events observed by a static camera. To track objects robustly, knowledge about the ground plane and the events is used. An event is considered as composed of action threads, each thread being executed by a single actor. A single thread of action is recognized from the characteristics of the trajectory and moving blob of the actor using Bayesian methods. A multi-agent event is represented by a number of action threads related by temporal constraints. Multi-agent events are recognized by propagating the constraints and likelihoods of event threads in a temporal logic network.

1 Introduction

Large scale event recognition by computer involves the analysis of the spatio-temporal interaction among the trajectories of moving objects [7, 3, 4]. Robust detection and tracking of moving objects from an image sequence is therefore an important key to a reliable event recognition. In the case of a static camera, detection of moving regions is relatively easy to perform, often based on background modeling and foreground segmentation. However, noise, shadows and reflections often arise in real sequences, causing detection to be unstable. For example, moving regions belonging to the same object may not connect or may merge with some unrelated regions. Tracking moving objects involves making hypotheses about the shapes of the objects from such unstable moving regions and track them correctly in the presence of partial or total occlusions. If some knowledge about the objects being tracked or about the scene is available, tracking can be simplified [11]. Otherwise, correspondence between regions must be established based on pixel level information such as shape and texture [5]. Such auxiliary knowledge other than the sensed data is called **context**. In a video surveillance application, a large amount of context (e.g. the knowledge of the events that occur) is often available [10]. In this paper, we demonstrate the use of the ground plane information and the event context as constraints to achieve robust tracking.

Once the trajectories and moving blobs of mobile objects are obtained, the gap between these numerical image features and a high level abstract activity description must be bridged. This leads to a spectrum of approaches [6], most of which are for developing specific systems for recognizing a single activity (or a small set of them) in a constrained environment. Bayesian networks have been used to recognize static postures (e.g. *"standing close to a car"*) or simple events (e.g. *"sitting"*) that are recognizable from visual evidence gathered during one video frame [4]. Hidden Markov Models (HMMs) have been used to recognize simple durative events by computing the probability that the model produces the visual observation sequence [9]. A few extensions of these techniques have been further investigated to recognize more complex events with a single actor [2]. There is only a limited amount of research on multi-agent events as the tracking of multiple objects in a natural scene is difficult and it is difficult to maintain the parameters of the fine temporal granularity of the event models such as HMMs. In [7], a complicated Bayesian network is defined together with specific functions to evaluate a few binary temporal relationships among events to recognize actions involving multiple agents tracked manually in a football match. Generalizing this system for other tasks than those of a football match may require substantial development.

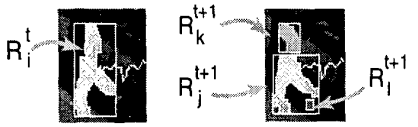
Aiming for a more systematic and effective event recognition system, we propose an alternative approach of event analysis. In a previous paper [3], we introduced an extendible hierarchical activity representation that allows the recognition of a series of actions performed by a single mobile object. The goal of this paper is to generalize this representation to recognize a multi-agent event.

2 Using Context for Robust Tracking

Robust tracking often requires an object model and a sophisticated optimization process [8]. In the case that a model is not available or the size of the image of an object is too small, tracking must rely on the spatial and temporal correspondence between low level image features of moving regions. However, the same moving regions at different times may split in several parts or merge with some other objects

0-7695-1143-0/01 $10.00 © 2001 IEEE

nearby due to noise, occlusion and low contrast. Figure 1 illustrates one of these problems, where the moving region $R_i{}^t$ at time t (a human shape) splits into two smaller regions $R_j{}^{t+1}$ and $R_k{}^{t+1}$ and noise detected at time $t+1$. Spatial correspondence between the moving region $R_i{}^t$ and $R_j{}^{t+1}$ or $R_k{}^{t+1}$ by itself is often low and creates an incorrect trajectory. Filtering moving regions is therefore an important step of a reliable trajectory computation.



(a) frame 74 (b) frame 75

Figure 1: *Splitting of moving regions and noise.*

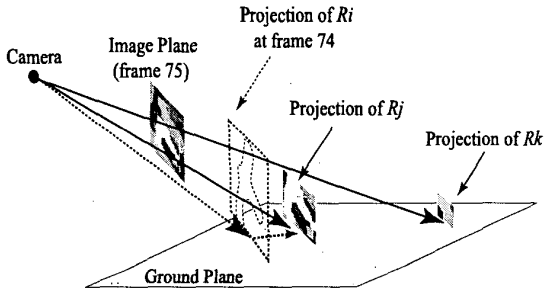


Figure 2: *Projection of the bottom points of moving regions on to the ground plane.*

2.1 Ground Plane Assumption for Filtering

Let us assume that objects move only along a known ground plane. An estimate of the ground plane location of the lowest point of a moving region can be used as a spatial constraint to find the best candidate region that corresponds to $R_i{}^t$. This is illustrated in figure 2. The dotted line shows the projection of moving region $R_i{}^t$ of frame 74 on the ground plane, whose correspondence to be found at frame 75. The solid lines show the projection of two moving regions detected at frame 75. Given that objects do not move in the air, $R_k{}^{t+1}$ is unlikely to correspond to $R_i{}^t$ (the head blob) as they would be located too far apart on the ground plane. The best candidate region $R_j{}^{t+1}$ (the body blob) can then be selected as the most likely region correspondence accordingly. Other moving blobs that appear to be above the ground as no correspondence is made, are tracked by their relationship to the body blob.

To accurately compute the world coordinates of a point on an image plane, we need a perspective camera model and its parameters. However, if we only need to estimate the 3D locations of moving regions standing on a ground plane, we can choose the world coordinate system such that $Z = 0$ as we are only interested in (X, Y). This reduces the 3x4 perspective camera transformation matrix to a 3x3 projective transformation matrix as follows:

$$\begin{pmatrix} x1 \\ x2 \\ x3 \end{pmatrix} = \begin{bmatrix} h_{11}h_{12}h_{13} \\ h_{21}h_{22}h_{23} \\ h_{31}h_{32}h_{33} \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

Since the non-singular homogeneous matrix H has 8 degrees of freedom, four or more points correspondence (x, y) to (X, Y) are enough to determine it uniquely, where $x = x_1/x_3$ and $y = x_2/x_3$. We have collected 8 points correspondence and solve for H with an error margin of $15cm$.

2.2 Merging Regions Using K-S Statistics

The best region candidate chosen based on a ground plane location may not be of the correct shape as shown by R_j. It needs to be merged with other sub-regions subject to some similarity measure. The merging process is iterated until no further merging occurs. In the final step, regions that are too small to represent valid objects can be disregarded.

The merging process between the best candidate sub-region $R_j{}^{t+1}$ and sub-region $R_k{}^{t+1}$ is performed as follows. We first make a hypothesis that both regions belong to the same object represented by $R_{j,k}^{t+1}$ and their pixel data are drawn from the same distribution as those of $R_i{}^t$. To test this hypothesis, we compute the spatial similarity between $R_{j,k}^{t+1}$ and $R_i{}^t$. We base the test on the distribution of gray intensity. The Kolmogorov-Smirnov (K-S) statistics provides a simple measure of the overall difference between two cumulative distribution functions. Let us assume that the intensity distribution of a region R_m is modeled by a Gaussian distribution ($N(x, \mu_m, \sigma_m)$). K-S statistics (D) of two regions R_m and R_n can be computed as follows:

$$D(R_m, R_n) = \max_{0 \le x \le 255} \left| \int_0^x \frac{1}{\sqrt{2\pi}\sigma_m} e^{-\frac{(x-\mu_m)^2}{2\sigma_m^2}} dx \right| - \left| \int_0^x \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{(x-\mu_n)^2}{2\sigma_n^2}} dx \right| \quad (1)$$

The significance level of D is then computed as $Q_{ks}([N_e + 0.12 + 0.11/\sqrt{N_e}]D)$, where N_e is the effective area of the regions ($N_e = \frac{size(R_m)*size(R_n)}{size(R_m)+size(R_n)}$) and

$$Q_{ks}(\lambda) = 2\Sigma_{j=1}^{\infty}(-1)^{j-1}e^{-2j^2\lambda^2} \quad (2)$$

Regions are merged if the significance level and the K-S statistics of $R_i{}^t$ and $R_{j,k}^{t+1}$ are lower than those of $R_i{}^t$ and $R_j{}^{t+1}$. If the type of objects are known (e.g. a human), a constraint on the size of the moving regions after merging can also be used as a criteria to reject incorrect hypotheses.

Figure 3 shows the detection of moving regions of the "stealing" sequence at different times. To track objects in the sequence, a graph representation is used [5]. Figure 4

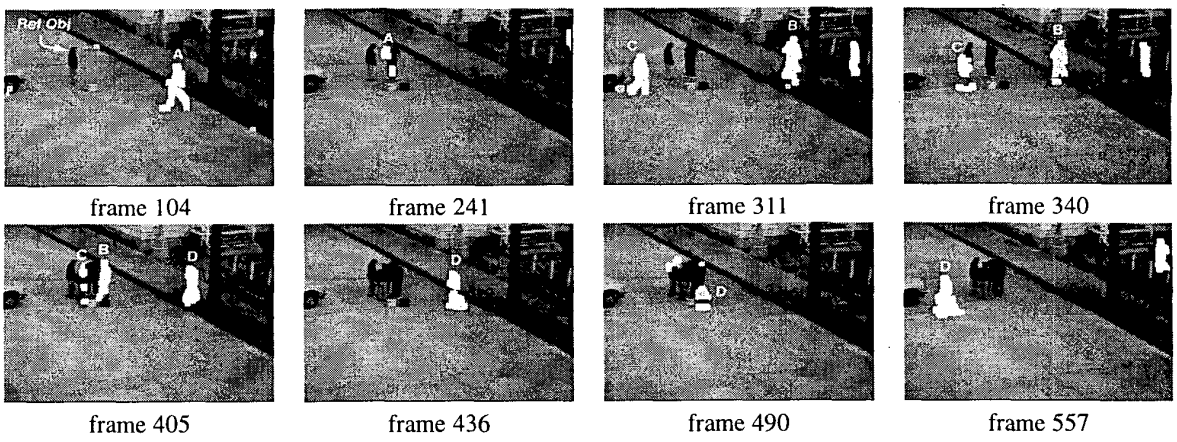
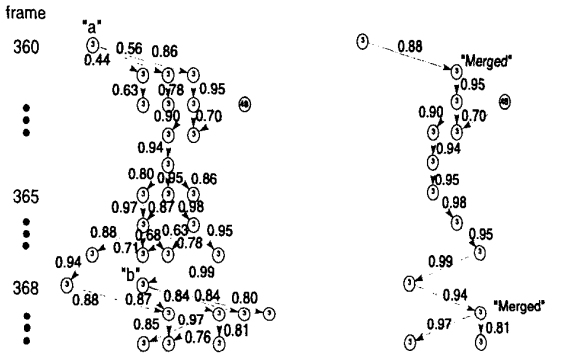


Figure 3: *The "Stealing" sequence. "A" approaches a reference object (a person standing in the middle with his belongings on the ground). "B" and "C" then approach and block the view of "A" and the reference person from their belongings. In the mean time, "D" comes in and takes the belongings away.*



(a) Before (b) After

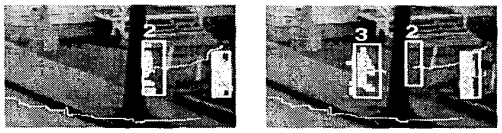
Figure 4: *A graph representation of the possible tracks of object "D". (a) Without using the ground plane knowledge, several hypotheses can be made about the possible tracks of the object. (b) After filtering, regions are merged or disregarded, decreasing the ambiguity.*

shows the graphs used for tracking object "D" before and after applying our filtering process. The nodes at each layer represent the moving regions detected at one video frame. An edge indicates a hypothesized correspondence between two regions of different frames. The nodes that are linked from the same parent represent the moving regions detected within the neighborhood of the parent node. The shaded node in the figure shows a moving region of another object being tracked nearby. The numbers associated with the edges indicate the similarity of the region distribution based on K-S test ranging from 0 to 1; the higher, the more similar. In figure 4(a), several hypotheses can be made about the possible tracks of "D" as indicated by numerous branching edges along the path from frame 360. However, none of these represents a correct track since most of the edges coming from a single parent node or going to a single child node are the results of the splitting and merging of moving regions as shown in figure 1. For example, at frame 360 and 368, "a" and "b" split into three sub-regions at the following frames. Figure 4 (b) shows the tracking graph after filtering regions based on the ground plane locations and K-S statistics. At frame 361 and 369, moving regions are merged and produce a higher similarity level to the moving region of the previous frame. At frame 365, 366 and 367, some moving regions are disregarded noise as they contradict with the ground plane assumption or when merged with the best candidate region, the similarity level decreases. After filtering, some tracking ambiguity may still remain in the graph. For example, the node at frame 362 can be associated with two regions in the following frame, one of which is also associated to the moving region of another object nearby. Such ambiguity can be removed based on other criteria such as event context.

2.3 Event Context for Tracking

The discontinuity of tracks of moving objects in an outdoor scene often arises when moving objects are not detected for a few frames due to problems such as total occlusion or when all regions do not satisfy the ground plan assumption. This is shown in figure 5. In this case, hypotheses about connections of fragments of tracks must be made. High level constraints such as the smoothness of the trajectory are often applied to verify a hypothesis. In the case of multiple moving objects or when movements of moving objects are performed based on a specific purpose such as to accomplish a task, a higher level constraint may also become

useful. We consider the use of event context (i.e. an event that objects are engaged in) to verify a hypothesis.



(a) Object 2 is totally occluded by the tree.

(b) Discontinuing tracks between Object 5 and 6.

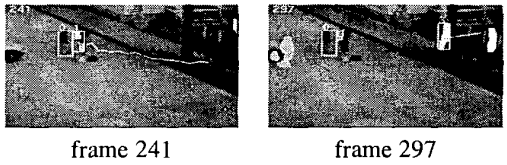
Figure 5: *(a) Object 3 is considered a different object from object 2 due to the total occlusion. (b) Due to the reflection on a car, the moving region of object 5 is larger than that of object 6, causing the discontinuity of the trajectory.*

Let us defer the discussion of event recognition to section 4 and assume that we have a system that can recognize events such as *"approaching a person"* and *"walking along a pedestrian path"*. Object 2 and 3, in fact, correspond to "B" and object 5 and 6 correspond to "C" in figure 3. They are both approaching "A". In figure 5 (a), several scenarios can be hypothesized after object 2 is lost as it can correspond to any objects nearby. Since these scenarios are ambiguous when viewed in a short time frame, reinterpretation of the tracks after being connected in a long term can help select the correct hypothesis. In this case, track correspondence can be verified based on the similarity of the average intensity distribution of the moving blobs and the consistency of object behavior. For example, when object 2 and 3 are connected and analyzed in a long term, *"approaching A"* is more likely than *"walking along the pedestrian path"* hypothesized when object 2 is merged with the human on the right. The hypothesis that object 5 and 6 are the same object can also be verified similarly.

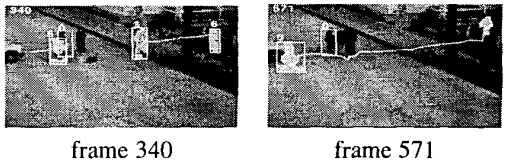
Figure 6 shows the final results of our tracking approach, where the correct shapes of moving objects are recovered and the confusion of possible multiple tracks is reduced.

3 Event Classification and Representation

To bridge the gap between a high level event description and the pixel level information, a hierarchy of **entities** proposed in [3] is used. Figure 7 shows an example of this representation for the event *"converse"*. First, **image features** (e.g. bounding boxes) are defined for the moving regions. **Mobile object properties** are then defined for the mobile objects based on spatio-temporal characteristics of the corresponding moving blobs (shown in light blue) and their trajectories (shown in dark blue). **Scenarios** correspond to long-term activities (or events) of mobile objects.



frame 241 frame 297



frame 340 frame 571

Figure 6: *Trajectories of the objects tracked using the ground plane knowledge, K-S statistics and event context.*

They are defined from a set of properties or a set of subscenarios. The structure of a scenario is thus hierarchical. Scenario events are viewed as consisting of *single* or *multiple* **threads**. In a **single thread event**, relevant actions occur along a linear time scale (as is the case when only one actor is present). In **multiple thread events**, multiple actors may be present. A single thread event may be further categorized into a simple or complex event. We describe in detail how these events are defined and modeled in the following.

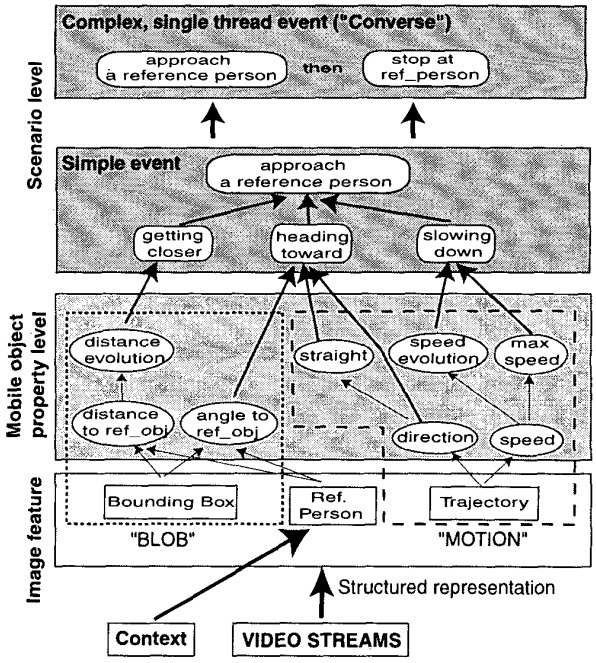


Figure 7: *A representation of complex event "Converse".*

- **Simple, single thread events** (or simple events) are defined as a single coherent unit of movement described

by a set of sub-events (other simple events) or directly from a set of mobile object properties. For example, in figure 7 simple event *"a person is approaching a reference person"* is described by the occurrence of three sub-events: *"getting closer to the reference person"*, *"heading toward it"*, and *"slowing down"*. These sub-events are also defined in a similar fashion. The representation of a simple event can be viewed as an instantiation of a Bayesian network.

- **Complex, single thread events** (or complex events) correspond to a linearly ordered time sequence of simple events (or other complex events). For example, complex event *"converse"* is described as a linear occurrence of two consecutive simple events: *"a person approaches the reference person"*, then *"stops at that person"*. We propose to use a finite state automaton to represent a complex event as it allows a natural description [3]. Figure 8 shows an automaton representation of *"converse"*, where each automaton state represents a sub-event.

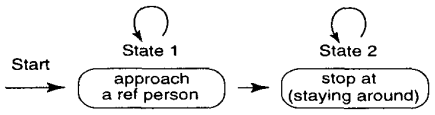


Figure 8: *A finite-state automaton that represents the complex event "Converse".*

- **Multiple thread events** correspond to two or more single thread events with some logical and time relations between them. Each composite thread in a multiple thread event may be performed by different actors. We propose to use the binary interval-to-interval relations, first defined by Allen [1], such as "before", "meets", "during" and "overlap", to describe temporal relations between sub-events of a multi-thread event. For example, *"event A must occur before event B"* and *"event A or event D occurs"*.

A multi-thread event can be represented by an event graph similar to interval algebra networks [1]. One particular specification of an event that we call *"stealing"* may be composed of several threads of complex events performed by four actors, as described in Table 1. The temporal constraints among these events are that

1) *"converse"* occurs before *"approach1"* and *"approach2"*, 2) *"blocking"* starts right after or some time after *approach1* or *approach2*, and 3) *"taking_object"* occurs during *"blocking"*. Figure 9 shows a graphical representation of these events. The symbols "b", "d" and "m" stand for interval temporal constraints "bofore", "during" and "meets" respectively.

Description of sub-events of *"stealing"*	
converse	$actor_1$ approaches his friend and drops a suitcase on the ground.
approach1	$actor_2$ approaches and stops between $actor_1$ and the suitcases.
approach2	$actor_3$ approaches and stops between $actor_1$'s friend and the suitcases.
blocking	$actor_2$ or $actor_3$ are blocking the view of the suitcases.
taking_object	$actor_4$ approaches and takes the suitcases away.

Table 1: *Description of a Multiple Thread Event "stealing".*

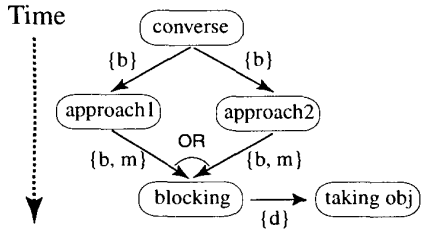


Figure 9: *A graphical description of a multi-thread event.*

4 Event Recognition

To recognize which event model matches the video images the best, the entities in the representation must be computed. Computations at each level are inherently uncertain, hence a formal uncertainty reasoning mechanism is needed. We propose the use of Bayesian methods for making an inference about all events. Suppose we have a set of competing event scenarios $S = S_1, ..., S_N$. Given an observation sequence $O = O_1 O_2 ... O_t$ of moving objects, where O_i is composed of a set of mobile object properties computed at time frame i, we want to compute $\forall i, P(S_i, O)$ and find the event with the maximal value. The event recognition process starts with the computation of the likelihoods of single-thread events from the properties defined for the moving object of interest. The probabilities of these events are then combined to verify a multi-agent event. We now define the major components of our approach:

4.1 Recognition of Single Thread Events

We present in this section an overview of single-thread event recognition. The details of the algorithms can be found in [3].

Simple Event Recognition Let us consider the representation of a simple event in figure 7. If we define a simple event such that its sub-events or mobile object properties

are conditionally independent of each other, the event hierarchy becomes an instantiation of a Bayesian network without introducing any hidden nodes. The evidence gathered from the mobile object properties computed from a single video frame (or their averaged values) can then be used to infer the likelihoods of simple events defined at a higher layer based on a Bayes' Rule. The parameters of the networks are learned by statistical methods from a set of training image sequences. In most cases, this task is simple as nodes in the networks in our case are transparent. Parameters can be determined by observing the values directly.

Complex Event Recognition A complex event is represented by a finite state automaton as shown in figure 8 for the event *"converse"*, where state i either advances to state $i+1$ or remains in the same state. Both sub-events of *"converse"* are simple events. As the occurrence of sub-events are uncertain, the decision on the transition between states also becomes uncertain. We define the probability of a multi-state complex event (MS) as $P(MS^*|O)$: *the probability that the complete automaton state sequence of MS occurs with the most likely state transition timing given the sequence of observations O.* This can be computed as follows:

$$P(MS^*|O) = \max_{\forall(t_1,t_2,...,t_N)} P(S_{1_{(t_1,t_2-1)}} S_{2_{(t_2,t_3-1)}} ... S_{N_{(t_N,t)}}|O), \quad (3)$$

where t_i refers to the time that the transition to state i from state $i-1$ occurs and $S_{i_{(t_i,t_{i+1}-1)}}$ means that scenario i occurs during t_i and t_{i+1} - 1.

The direct computation of $P(MS^*|O)$ over T frames from the probabilities of the occurrence of sub-events using eq. 3 requires an exploration of all possible transition timings. This is an operation of $O(T^N)$ complexity since there are $O(T^N)$ combination of values of $t_1, t_2, ..., t_N$. We have applied a more efficient recursive algorithm ($O(NT)$) based on dynamic programming similar to the Viterbi algorithm used in HMMs. Since the networks we propose to use are transparent, the initializing and learning of parameters is much simpler than for HMMs.

4.2 Recognition of Multi-Thread Events

Multi-thread events are recognized by evaluating the likelihood values of the temporal and logical relations among event threads. Constraint satisfaction and propagation techniques have been used in the past [1] based on the assumption that events and their durations are deterministic. We present in this section an algorithm to verify and propagate temporal constraints when events are uncertain.

4.2.1 Evaluation of Temporal Relations

Suppose that we have two complex events: *event A* and *event B*, each with a likelihood distribution similar to the one shown in figure 10. Computing the likelihood of these

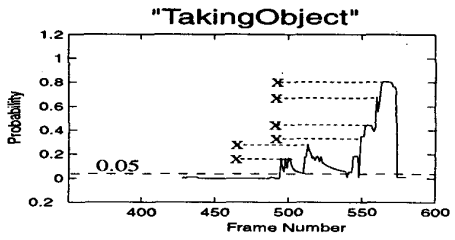


Figure 10: *The likelihood of the complex event inferred by a probabilistic finite state automaton. The event, at different times, may have different likely start times as illustrated by "x", depending on the most likely transition timings between states considered up to the current frame.*

events satisfying a temporal constraint requires an exploration of the combination of all possible event intervals of both events that may occur during time frame 1 and the current frame. For example, to compute *"A before B"*, we need to find a time t' frame such that *event A* started and ended before t' and *event B* started after t' and may still occur. The event intervals of A and B that give the maximal combined likelihood define the occurrence of the multi-thread event.

In our case, the interval of an event can be inferred from the finite state automation by keeping track of the transition timings between states and by making an assumption that the end point of an event can be determined when the likelihood of the event becomes very low. For example, in figure 10, assuming that an event ends when the likelihood becomes lower than 0.05, we have three possible end times of this event at frame 511, 542, and 575. From our experiment, the number of possible end points of a complex event is relatively small compared to the length of video data. Suppose that event threads are independent of each other. After the constraint on the interval of two events are verified, we can compute the likelihood of the multi-thread event as the product of the probabilities of them. This computation requires a search operation of $O(k^2M)$ complexity if there is an average of k possible starting points for an event model during the time we observe the video images and M number of temporal relations.

Other temporal relations can also be computed similarly. For example, to compute *"A during B"*, we first find all possible event intervals of A and B. We then search for a combination of these events that produces the maximal likelihood subject to a constraint that the start and end times of A must be during the interval of B.

Event threads that are defined using a logical constraint such as "and" and "or" can be combined much easier than a temporal constraint, as we do not need to verify a temporal relation. For *"A and B"*, we compute the product of the event likelihood. For *"A or B"*, we choose the event with a higher likelihood.

4.2.2 Inferring a Multi-Thread Event

A multi-thread event that is described by more than two event threads can be inferred by propagating the temporal constraints and the likelihood degrees of sub-events along the event graph. For example, suppose that we have two event constraints: *"A before B"* and *"B before C"*. To evaluate *"B before C"*, we need to consider an extra constraint that "event B" occur after "event A".

5 Results and Conclusions

To validate our approach, we have constructed about ten simple event models, five complex event models and four multi-thread event models, with the most complicated event model being the *"stealing"* scenario. The parameters of the networks that recognize single-thread events are learned from about 600 data samples containing half positive and half negative examples. We present in this section some event analysis results of one of the test video sequences (*"stealing"*) shown in figure 3. Figure 11 (a), (b), (c) and (d) show the analysis results of single-thread events evaluated on for object A, B, C and D respectively. The recognition of the most significant complex event is shown on the left plot and the analysis of its sub-events is shown on the right plot. For example, object A is recognized to be performing *"converse"* (the left plot) after it approaches (fine dotted) and stops (solid) at the reference person. The coarse dotted line in (a) show a weak recognition of *"walking along the pedestrian path"* that object A did not perform. The results of such unrecognized events are omitted for clarity in other plots. In (b) and (c), object B and C has approached then stopped between the reference person and his belongings. These events are an instantiation of *"approach1"* and *"approach2"* in figure 9. Figure 11 (d) shows that object D has performed *"taking objects"*. Figure 11 (e) shows the recognition of the temporal relations defined in *"stealing"* together with the assignment of the actors. Finally, figure 11 (f) shows the probability of two instantiations of event *"stealing"*. The solid and dotted lines show the probabilities of event *"stealing"* where the sub-event "approach" is performed by object B and object A respectively. We note that as the complexity of the event model increases, the probability of the event tends to decrease. This is due to the introduction of more parameters, and hence the ambiguity arises. To accurately select a rare but complex scenario, an alternative model selection criteria that takes into consideration of the model complexity is necessary.

Still, our system has been tested only on a limited set of examples. Further development of the temporal and logical constraints may be required to represent a more sophisticated scenario model for a real application. For example, numerical constraints such as *"A occurs before B at least an hour ago"* or additional logical constraints such as *"no other events should happen between A and B"* may be allowed to enhance the characterization of a scenario. These enhancements will complicate the representation and recognition algorithm. However, we believe that our approach can be extended in such a way that the $O(NT)$ bound of the complexity of the inference process can be maintained.

Acknowledgements

This research was supported in part by the Advanced Research and Development Agency under contract MDA 908-00-C-0036.

References

[1] J. F. Allen and G. Ferguson. Actions and events in temporal logic. *Journal of Logic and Computation*, 4(5):531–579, 1994.

[2] A. Bobick and Y. A. Ivanov. Action recognition using probabilistic parsing. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, Santa Barbara, CA, 1998.

[3] S. Hongeng, F. Bremond and R. Nevatia. Representation and optimal recognition of human activities. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, Hilton Head Island, SC, June 2000.

[4] H. Buxton and S. Gong. Visual surveillance in a dynamic and uncertain world. *Artificial Intelligence*, 78(1-2):431–459, 1995.

[5] I. Cohen and G Médioni. Detecting and tracking moving objects for video surveillance. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, Fort Collins, CO, 1999.

[6] R. T. Colins et. al. Special section on video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), August 2000.

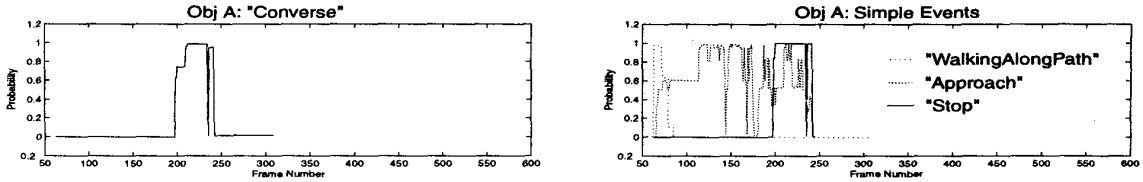
[7] S. Intille and A. Bobick. Visual recognition of multi-agent action using binary temporal relations. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, Fort Collins, CO, June 1999.

[8] H. SidenBladh, M.J.Black and D.J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *Proceedings of the European Conference on Computer Vision*, 2000.

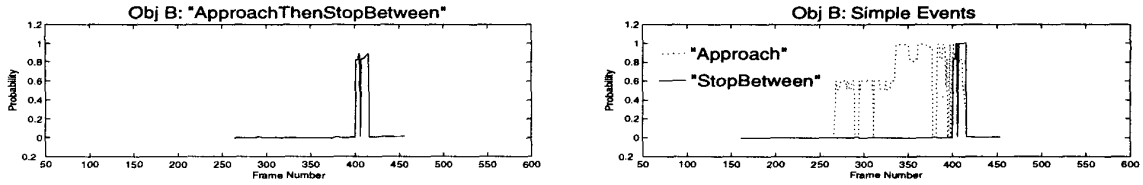
[9] P. Remagnino, J. Orwell and G. A. Jones. Visual interpretation of people and vehicle behaviours using a society of agents. In *Italian Association for Artificial Intelligence*, pages 333–342, 1999.

[10] N. Rota and M. Thonnat. Video sequence interpretation for visual surveillance. In *IEEE Workshop on Visual Surveillance*, 2000.

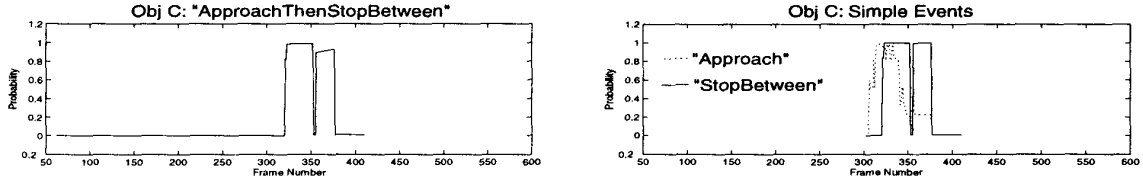
[11] M. Yamamoto, , and K. Yagishita. Scene constraints-aided tracking of human body. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, pages 151–156, 2000.



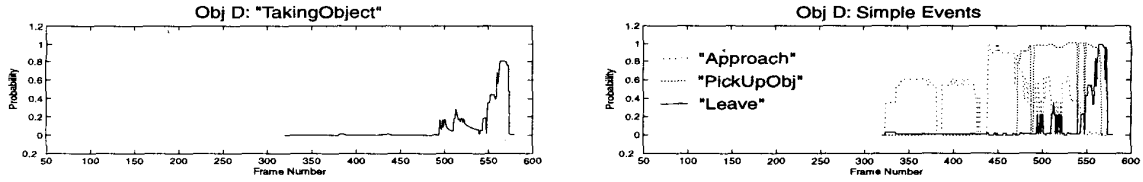
(a) Analysis of single-thread events for objA.



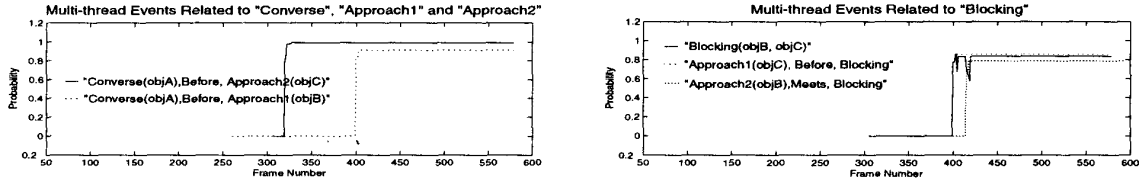
(b) Analysis of single-thread events for objB.



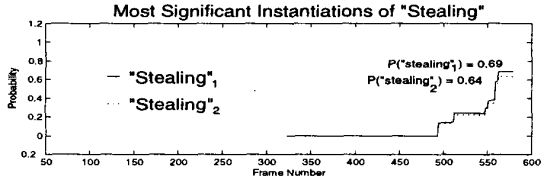
(c) Analysis of single-thread events for objC.



(d) Analysis of single-thread events for objD.



(e) Evaluation of temporal relations among single-thread events and the corresponding assignment of actors.



(f) "Two most significant recognition results of "stealing", each with different actor and temporal event combinations.

Figure 11: *Event analysis results of the sequence "stealing".*