Hierarchical Language-based Representation of Events in Video Streams *

Ram Nevatia Tao Zhao Somboon Hongeng
University of Southern California
Institute for Robotics and Intelligent Systems
Los Angeles CA 90089-0273
{nevatia|taozhao|hongeng}@iris.usc.edu

Abstract

We aim to define an event ontology that allows natural representation of complex spatio-temporal events common in the physical world by a composition of simpler events. The events are abstracted into three hierarchies. Primitive events are defined directly from the mobile object properties. Single-thread composite events are a number of primitive events with temporal sequencing. Multi-thread composite events are a number of single-thread events with temporal/spatial/logical relationships. This hierarchical event representation naturally leads to a language description of the events. We define an Event Recognition Language (ERL) which allows the users to define the events of interest conveniently without interacting with the low level processing in the program. We will also briefly mention some approaches to compute the proposed representation.

1 Introduction

Automated understanding of human activity is important for a number of tasks that can be used to augment human capability. Such ability may be used for surveillance to provide security in a home, office or industrial environment. Monitoring of children, elderly or sick people can allow more such individuals to be cared for at a high quality level at lower costs. Understanding human activity is also important for enhanced human-computer interaction including advanced video conferencing and for intelligent content-based retrieval of video from digital libraries.

In this paper, we aim to define an event ontology that allows natural representation of complex spatio-temporal events common in the physical world by a composition of simpler events. At the lowest level, primitive events are defined directly from the mobile object properties; single-thread composite events are defined as a number of primitive events with temporal sequencing, forming an event thread; at the highest level, multi-thread composite events are defined as a number of single-thread events with temporal/spatial/logical relationships, possibly involving multiple actors. In this way, a transparent event hierarchy is constructed. This hierarchical event representation naturally leads to a language description of the events. We define an Event Recognition Language (ERL) which allows the users to define the events of interest conveniently without interacting with the low level processing in the program. We will also briefly mention some approaches to computing the proposed representations, though this is not the focus of this paper.

[1]This research was supported, in part, by the Advanced Research and Development Activity of the U.S. Government under contract No. MDA-908-00-C- 0036.

2 Related Work

State-based representations have been employed to represent temporal trajectories. Hidden Markov Models (HMMs) are a popular state-based model. The number of states and the transition probabilities can be learned automatically from training samples for each event but such a representation is not transparent and not easily generalized for new events. Some variants of HMM have also been used in activity representation and recognition. Entropic-HMMs [3] are designed to simplify the structure of an HMM by minimizing the entropy of the data as well as that of the structure, making the learnt states closer to human concepts. Coupled-HMMs [10] are used to model the interaction of two agents by coupling the states of two HMMs. Parametric- HMMs [12] are used to model the gestures with underlying parameters (*e.g.*, the direction of a pointing gesture). State-based models can also represent higher-level behavior if the states have higher-level meanings. [2] used a state machine to represent and recognize a human's state (*e.g.*, "sitting", "talking on the phone", etc) in an office.

The work of [5] aims at higher-level behavior and employs a two- level event abstraction. At the lower level, simple events at the level similar to our defined primitive events are modeled by HMM or spatio-temporal properties. At the higher level, a stochastic context free grammar (SCFG) is constructed for the problem domain with the simple events as

terminals. The recognition of events defined in the SCFG is done by language parsing in the simple event stream. There are two major limitations of this approach. Firstly constructing a grammar for a large problem domain may not be very realistic since it is difficult for human to have a complete understanding of the domain and anticipate all possible events. Secondly the language parsing essentially can only handle input of a single temporal thread, therefore it is difficult to represent and recognize high-level temporal, spatial or logical relationship of the events possibly by multiple agents.

There has also been work on event inference using traditional AI approaches. A classical approach is based on interval temporal logic [1] for the temporal relations of events. We also use this logic in our representation, However, we do not assume that the temporal intervals are given *a priori*, rather they need to be inferred from the video data itself.

[6] attempted to translate human actions into sentences of natural language. The translation is based on concept hierarchy from more abstract concept to more concrete ones (*e.g.*, "move" v.s. "walk"). Their system only investigates a single human interacting with the environment and the hierarchy does not extend to temporal compositions of events. In the work of [11], the video sequence interpretation involves incremental recognition of states of the scene, events and scenarios. These elements are described in a declarative manner and the recognition problem is translated into a constraint-solving problem.

3 Hierarchical Event Representation

In order for the system to be more generic and suitable for wider range of applications, we aim for a generic and extendable representation of human activity that can be applied to a large variety of behaviors of different abstraction levels, different scales and different environments, and it has to follow a hierarchical way. We starts from structured scene representation where scene is represented by objects and their (time-varying) properties. The events are then defined on these properties with the temporal and spatial relationships.

3.1 Structured scene representation

The scene is represented in a structured way by its various aspects as well as the objects in it. The scene geometry can vary from as simple as a ground plane to as complex as a VRML model. The scene appearance can be maintained by a statistical background model.

The objects in the scene include scene objects, user specified regions, mobile objects and carried objects. Scene objects are the static objects in the scene possibly involved in events of interest by serving as reference objects or providing context for the events, *e.g.*, a telephone booth. User specified regions are virtual objects which are meaningful for events, *e.g.*, a region which trespassing is forbidden. Mobile objects, *e.g.*, humans and vehicles, are the most important objects since they are the actors of the events. Carried objects are objects carried in or out of the scene by the mobile objects, *e.g.*, a suitcase.

The objects are represented by their various properties. Properties of mobile objects are especially important since it is the change of their properties that makes interesting events. We define two groups of properties. **Object characteristic properties** correspond to the instantaneous characteristics of the moving blob that represents an object. Some mobile object properties are elementary (*e.g.*, width, height, color histogram, texture or principle axis) while the others can be complex (*e.g.*, a graph description of the shape of an object). **Trajectory related properties** are computed from the trajectory of the mobile objects (*e.g.*, the direction of the trajectory or the object speed).

3.2 Classification of Events

Different human activities may be analyzed at various scales, such as observing facial expressions, body gestures or a group behavior in a global environment. Here, we focus on large-scale events of rigid or articulated objects. We consider the events of the interactions of mobile objects with other (mobile) objects, in the environment by tracking the whole body in the case of a global event. The link between the characteristics of the tracks of mobile objects (*i.e.*, the whole body or a moving part of the body) and verbal space must be performed in a systematic way. We begin with the formalization of the basic spatio-temporal properties of the tracks possibly with respect to a related reference object, and extend such notions to categorize activities, in a natural way.

We classify events in physical world in broad categories of *primitive* and *composite* events. *Primitive events* are those that can be inferred as an entity without considering previous events; for example, a person walking towards a building, a car traveling on a highway or a person waving his arms. *Composite* events are defined by a composition of primitive events, such as a person walking up to an object, picking it up and walking away with it. *Sequencing* is the most common composition operator. Other temporal, logical and spatial relations between the sub-events may also be applied. Events whose composition can be ordered along a time scale are called *single-thread* events and usually correspond to actions of a single actor. Actions involving multiple actors are represented as *multi-thread* events, which include certain temporal, spatial, and logical relations between actions of individual actors. The three types of events are defined formally as the following.

A **primitive event** corresponds to a single, coherent unit of movement, which is performed by one agent (or a group

of agents acting in the same manner). Primitive events are described by a combination of a set of mobile object's properties describing different aspects of this agent or the spatial relationship of the mobile object to a reference object. The primitive events can be identified instantaneously without resorting to the history. For example, primitive event "a person is slowing down toward an object of reference" can be described by satisfying the following three constraints simultaneously: "the person is getting closer to the object of reference", "the person is heading toward the object of reference", and "the person is slowing down".

A **single-thread composite event** (or **single-thread event**) corresponds to a consecutive occurrence of multiple primitive events. For example, single-thread event "Contact" is described as a linear occurrence of three consecutive primitive events: "a person slows down towards another person", then "stops at that person", and then "turns around and walks away". Usually the primitive events involved are by the same actor, however, they can also by different actors if the sequencing property can hold. It should be noted this is only for the event of which the sub-events inherently follow each other. For those events whose sub-events only have time ordering but not necessarily strictly follow each other, we represent them with the events of the next level.

A **multiple-thread composite event** (or **multi-thread event**) is composition of multiple single-thread events with some logical, temporal or spatial relations between them. Each consituent thread in a multiple-thread event is possibly performed by a different actor. The temporal relationship specifies the temporal synchronization of different threads or the states of different threads. And the spatial relationship specifies the spatial constraints of different constituent events. The temporal and spatial relationships will be described next. The logical relationships include "and", "or", etc. For example, "Either event A or event D occurs". We do not incorporate a complete first-order logic in our representations; as we deal with a closed world (set of objects that may participate in actions is fixed prior to interpretation), propositional logic suffices to express the desired relations.

3.2.1 Temporal/spatial relationship

Temporal and spatial relationships are important for visual spatio-temporal events. Temporal relationships are used in single-thread events as sequencing and in multi-thread events in a more general fashion; spatial relationships are both used in primitive events as relationships between objects and in multi- thread events as relationships between events. There is substantial research work on temporal and spatial reasoning in the AI community. Part of the difficulty arises from the qualitative description of the entities. However, in the domain of visual event recognition, both the temporal and the spatial information we measure from the world are quantitative, though with certain amount of uncertainty. This has made the temporal and spatial representation/reasoning here easier.

Interval and temporal relationship Time is an one-dimensional quantity. The relationship of time instants is relatively simple. However, events usually occur for a period of time, *e.g.*, an interval, represented as $[start, end]$. The relationship of time intervals is needed to represent the temporal relationship of different events. We adapt the definition of interval relationship in Allen's interval temporal logic [1]. The possible relationships of two time intervals $i = [start_i, end_i]$ and $j = [start_j, end_j]$ are:

- $Before(i, j) \Leftrightarrow end_i < start_j$
- $Meets(i, j) \Leftrightarrow end_i = start_j$
- $Overlaps(i, j) \Leftrightarrow start_i < start_j < end_i$
- $Starts(i, j) \Leftrightarrow start_i = start_j$
- $During(i, j) \Leftrightarrow start_i > start_j$ and $end_i < end_j$
- $Finishes(i, j) \Leftrightarrow end_i = end_j$

The inverse (inv(relation(i,j)) = relation(j,i)) of the above relationship are "After"," MetBy", "OverlappedBy", "StartedBy", "Contains" and "FinishedBy" respectively. Some of the relationships are further augmented with a time quantity for more precise description. For example,

$$Before(i, j, 1min+) \Leftrightarrow end_i < start_j - 1min$$

means interval i is before j by at least 1 minute.

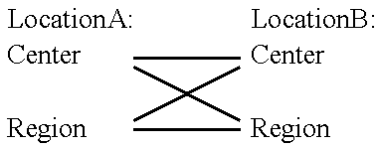
$$Overlap(i, j, 50\%+) \Leftrightarrow \frac{End_i - start_j}{End_i - start_i} > 50\%$$

means interval i and j have at least 50% overlap based on the length of interval i. Considering the inevitable noise associated low level processing and the fuzziness of the relationship themselves, soft constraints can be used instead of hard constraints. For example, "Before(i,j)" can be defined if $start_i < start_j$ and the overlap of the intervals are below 5% of the average of the two interval lengths. The interval of a composite event is usually the union of the intervals of the compositing sub-events.

Location and spatial relationship Compared to the one-dimensional time, the representation and the possible relationships of two (or three) dimensional locations are more complex. Location is a property of an object, an event or a user defined area of interest. We feel that representing location as a region R with a center C projected on the ground plane is usually sufficient for large-scale event scenarios to describe spatial relationship. For an object, the center is naturally its center of gravity and the region is its physical spatial occupancy. For an event, the region is the spatial area which the event takes place in. It needs to be defined by the

user if it is different from the union of regions of the participating objects. The locations of mobile objects keep on changing with respect to time. The region can be simplified into a bounding box or a polygonal approximation for computational efficiency.

The types of spatial relationship are as following:



"DistanceCC" is the most important Center-Center relationship and can be computed easily. Quantitative relationships on the distance can be specified easily. For example, "DistanceCC(MobileObjectA, RefObjectB) $<$ 1m". Qualitative relationships can also be specified, such as "Close(MobileObjectA, RefObjectB)". However, additional mechanism is needed to transform the qualitative description into quantitative form for computation, possibly according to the property of the objects and context or by learning. Region- Region relationship includes "Touch", "DistanceRR" which is defined to be the least distance of any two points on the boundary if they don't touch, 0 otherwise, etc. Center-Region relationship includes "InRegion", "DistanceCR" which is defined to be the least distance from the center to any of the points on the region boundary if "InRegion" is false, 0 otherwise, etc.

4 The Event Recognition Language

The hierarchical representation of activities described above naturally leads to a language description of activities. And on the other hand, a language is needed for the common users of practical event recognition systems to describe what events they want to be recognized without going into the program. Therefore, we designed the Event Recognition Language (ERL).

In terms of expressive power, the language is not intended to model all the possible events, but it should be generative enough to be able to define most of the events in daily life and especially the ones interested in surveillance applications. Therefore, the language should be able to describe from simple motion like "walking-in" to more complex events including interaction of multiple objects, environmental context, spatial and temporal relation, etc. In terms of convenience, the language should be easy to use. It is ideal to be similar to existing popular computer languages and the event definition should be close to human's understanding. What's more, the language should be designed to have good modularity so that the codes can be easily reused in the future.

4.1 Data types

The language includes the following data types:

- `Object`
  - `Mobile Object`
    - `Human`
    - `Vehicle`
  - `Scene Object`
  - `Carried Object`
- `Location`
- `Interval`
- `Numerical Value`

Each data type has a number of properties. The data types are defined in an object-oriented fashion. The more specific data type (*e.g.*, "Human") inherits properties from and is assignment compatible with more abstract data type (*e.g.*, "MobileObject") in the hierarchy. For convenience, unless ambiguity exists, the data type of the property can be filled by its host. For example, "SlowDownToward(MobileObject, Location)" can be instantiated with "SlowDownToward(HumanA, ReferenceObjectB)", with "ReferenceObjectB" automatically converted to its location property.

4.2 Syntax of definition

The definitions in ERL are similar to the function definitions of a computer programming language, which makes the user familiar with any programming language be able to write ERL code easily. The general syntax is as follows:

```
EventType EventName (ObjType ObjName, ...)
  Event descriptions;
```

It will return the probability of occurrence of the defined event as well as the parameter assignments.

We will show below the definitions of different types of events with examples. The implementation of mobile object properties is provided by the developers and the properties are available for constructing events in the form of "system functions". We use all capital names for functions provided by the system.

A primitive event "SlowDownToward" is defined as the following. "TRAJ_TOWARD" and "SPEED" are mobile properties computed by the system. The function "BAYES_COMB" stands for combining these two aspects using a Bayesian network with the parameters in the argument list.

```
PRIMITIVE SlowDownToward(human A, location O)
  BAYES_COMB(TRAJ_TOWARD(A, O),
             SPEED(A, LOWER), parameters);
```

Single-thread composite event is a consecutive occurrence of several primitive events of one object. A definition example is as follows. The function "SEQ_COMB" is to combine the three primitive events sequentially. The definitions of primitive event "BeingClose" and "GoingAway" are not shown here.

```
SINGLE_THREAD PassBy (human A, location O)
  SEQ_COMB(SlowDownToward(A O),
           BeingClose(A O),
           GoingAway(A O), parameters);
```

Multi-thread event is an event which consists of multiple actors and the temporal / spatial / logical relationship between them. It is defined by specifying all the events involved and all the constraints among them.

In specifying temporal relationships, sometimes we need to synchronize some certain state of a single-thread event (which is a primitive event) instead of the entire event thread. One state in a single-thread event is expressed as single_thread_event_name [state_number].

```
MULTI_THREAD PassByAfter
              (human A1, human A2, location O)
  AND(e1=PassBy(A1, O),
      e2=Passby(A2, O),
      AFTER(e1, e2));
```

In all the events defined, some serve as building block of other events only and others are to be recognized. In our framework, composite events are valid events to be recognized. In the case that a primitive event is interested to be recognized, it is automatically cast to a single-thread event of a single state. In the case a primitive event serves as a constituent event in a multi-thread event, it is also cast to a single-thread event. In ERL, we specify an event to be recognized using the key word "REC" before its definition.

```
REC PassByAfter(a1, a2, O);
```

4.3 Parameter binding

When we specify an event to be recognized, we can fix all the parameters, or we can leave some or all parameters unbound. In the latter case, all the possible binding of the unbound parameters (*e.g.*, mobile objects, reference object, locations, intervals) will be considered and the return will be a list of the combinations of parameter bindings with sufficiently large probabilities.

For example, "PassByAfter(MO01, MO02, RO01)" will check if the event PassByAfter has occurred with mobile object MO01, MO02 with respect to reference object RO01. MO01, MO02 and RO01 are all object IDs in the scene. "PassByAfter(MO001, x, RO001)" will return a list of mobile objects that made the probability of event "PassByAfter" sufficiently large. Of course it will return every one who passed by reference object RO01 after MO01 did. It is the user's responsibility to control the expected results. In this case, the user may specify an interested time interval by replacing "AFTER(e1, e2)" with "AFTER(e1, e2, 5min-)" in the definition of "PassByAfter".

4.4 Compiling of an example

The ERL source code is passed to a parser that we have implemented. During parsing, each defined event results in a node in a tree structure [1]. Each of the non-leaf nodes has a number of descendents of lower-level or same level events. A non-leaf node also includes information about the relationships of the descendants. For example, the descendents of a multi-thread event will have some temporal/spatial/logical relationships; the descendents of a single- thread event usually have sequencing relationship. These relationships are represented by a graph in each node. The root nodes of the trees are the events to be recognized and are passed to the event recognition module for inference. We show a more complex example "StealingByBlocking" and its compilation graph.

This multi-thread event includes 5 sub-events:

- A1 (the owner of the suitcase) approaches A0 (his friend, serving as a reference object) and drops a suitcase on the ground.
- A2 approaches and stops between A1 and the suitcase.
- A3 approaches and stops between A0 and the suitcase.
- A2 and A3 block the view of the suitcases.
- A4 (the thief) approaches and takes the suitcase away.

The ERL source code is as the following:

```
REC StealingByBlocking(Human00,a1,a2,a3,a4,o);

MULTI_THREAD StealingByBlocking(human A0,
  human A1,human A2,human A3,human A4,car_obj O)
  AND(s1 = Converse(A1, A0)
      s2 = ApproachStopBetw(A2, A1, O)
      s3 = ApproachStopBetw(A3, A0, O),
      s4 = Block(A2, A3, A1, A0, O),
      s5 = TakeAway(A4, 0),
      BEFORE(s2, s1),
      BEFORE(s3, s1),
      OR(AND(BEFORE(s2, s4),START(s2, s4)),
         AND(BEFORE(s3, s4),START(s3, s4))),
      DURING(s5, s4));

SINGLE_THREAD Converse(human A, location L)
```

[1] Different instances of the same event in the definition are treated as different events though they may be treated as one during recognition computation.

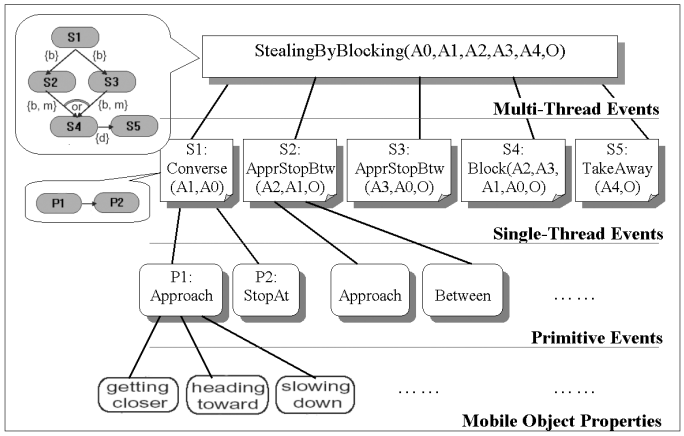


Figure 1: Example of the result of compiling the event definition of "StealingByBlocking". The graph is partially expanded due to the space limit.

```
  SEQ_COMB(Approach(A, L), StopAt(A, L));

SINGLE_THREAD ApproachStopBetw(human A,
                 location L1, location L2)
  SEQ_COMB(Approach(A,CENTER(L1,L2)),
           Between(A,L1,L2));

SINGLE_THREAD TakeAway(human A, car_obj O)
  SEQ_COMB(Approach(A,0), PickUp(A,O),
           LeaveWith(A,O));

PRIMITIVE Block(object A1, object A2,
              human A3, human A4, object O)
  AND(Between(A1, A3, O), Between(A2, A4, O));

PRIMITIVE Approach(object A, location L)
  BAYES_COMB(GETTING_CLOSER(A, L),
             HEADING_TOWARDS(A, L),
             SLOWING_DOWN(A));
```

Other involved primitive events "Between", "StopAt", "PickUp", "LeaveWith" are not listed due to the space limit. Part of the compilation graph is shown in Fig.1.

5 Event Recognition

The task of event recognition consists of making inferences about the predefined events taking place in a scene from the video images. As an example, the recognition involves computing the probability of the occurrence of event "StealingByBlocking" defined in the previous section. Although the compilation graph looks complex, the inferences are made bottom-up (from mobile object properties to primitive events to single-thread events and to multi-thread events) following the hierarchy.

The recognition approaches are relatively independent of the event representation proposed in the previous section. Different techniques may be used to make inferences for each class of events. In the following, we summarize the approaches developed in our group to indicate that our representation method can indeed be linked to recognition methods and to make possible a discussion of the computational complexity of the processes. For the following, we assume that mobile object detection and tracking have been performed and that properties of these objects and their trajectories have been computed using methods such as described in [4], [8], [13].

5.1 Primitive event recognition

Multiple constituent properties of a primitive event need to be combined to infer the likelihood of the primitive event. There are multiple choices to accomplish it (*i.e.*, weighed summation, neural network, Bayesian network, etc). We chose to use naive Bayesian network to combine the multiple cues. It is advantageous due to its optimality and transparency. The parameters (CPTs) can be filled in either by human knowledge or by learning. The computation of the probability of a primitive event is very efficient given the values of the object properties.

5.2 Single-thread composite event recognition

As mentioned above, the sub-events of a single-thread event happen one after another, which naturally leads to a state-based representation. Due to the uncertainty associated with each state, probabilistic inference is needed. Probabilistic finite-state machine is used to recognize a complex event. In [7], an Markov assumption is used and the inference is similar to that of the HMM. The complexity of the inference is $O(N^2T)$ where N is the number of states and T is the length of the sequence. However, the duration of the different constituent sub-events may have large variations compared to speech recognition. This motivated the use of semi-HMM which explicitly models the priors on the durations of the sub-events [9]. The computational complexity of the semi-HMM inference algorithm is $O(NTd)$ where d is the maximum time span of the event. This can be further reduced to $O(NT)$ if some approximations are made.

5.3 Multi-thread composite event recognition

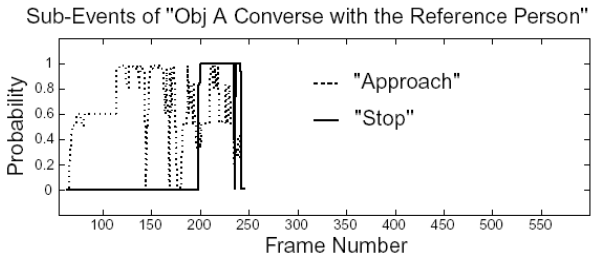
Multi-thread events are recognized by evaluating the logical constraints of the constituent event threads and their temporal/logical relationships. Logical relations can be easily computed. Computing the temporal relationship of different event threads is essentially a constraint satisfaction problem.

A number of event instances are generated from the continuous likelihood values for each event thread over time. The solution is sought by finding the combination of event instances which gives the highest combined likelihood from all the combinations which satisfy the temporal constraints [8] [9]. Denote k as the average number of instantiations per single-thread event, and R as the number of constraints. The computational complexity for the most general case is $O(k^R)$. However, when the constraints of different sub-events form a linear structure as in many cases when specifying temporal constraints, the complexity can be reduced to $O(k^2R)$. Furthermore, due to the hierarchical build, many constraints have been applied on the lower level events and the number of constraints at this granularity is usually small. Thus the expected computation is still not significant.

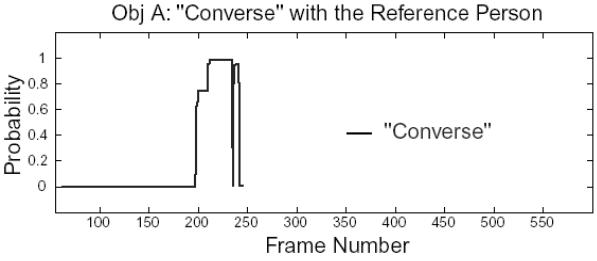
The computational complexities given in Sec.5.2 and Sec.5.3 are based on the event parameters being bound (*i.e.* the actors and reference objects are given). If these assignments are unknown and must be inferred, the computation complexity will increase as follows. Suppose there are K objects in the scene and the event contains N unbound parameters. The number of possible assignments is $\frac{K!}{(K-N)!}$. When there are a large number of people in the scene, this number can be large. Some pruning techniques may be needed to keep the computation within acceptable limits.

5.4 A multi-thread event recognition example

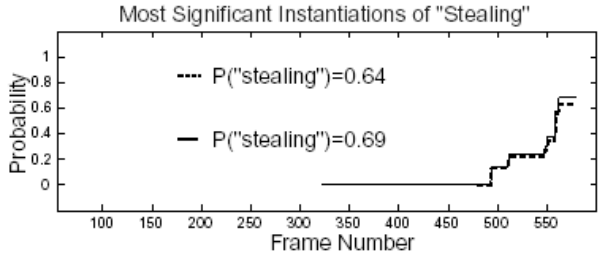
The multi-thread event "StealingByBlocking" as defined in Sec.4.4 is recognized in a video sequence (460 frames) with the approaches above. Fig.2. shows the probability histories of two most significant instantiations of "StealingBy-Blocking", the probability history of a single- thread event "Converse" of the most significant instantiation and the probability histories of the two primitive events "Approach" and "Stop" of "Converse". It also shows several frames of the data with the actor assignment of the most significant instantiation overlaid. The computation time of this example (also including the rejection of a multi-thread event "CooperativeObjectExchange" at the same time) is 222 sec (2 fps) on a Pentium III 1G Hz PC with un-optimized C++ code where the 4 moving humans are treated as unbound parameters.



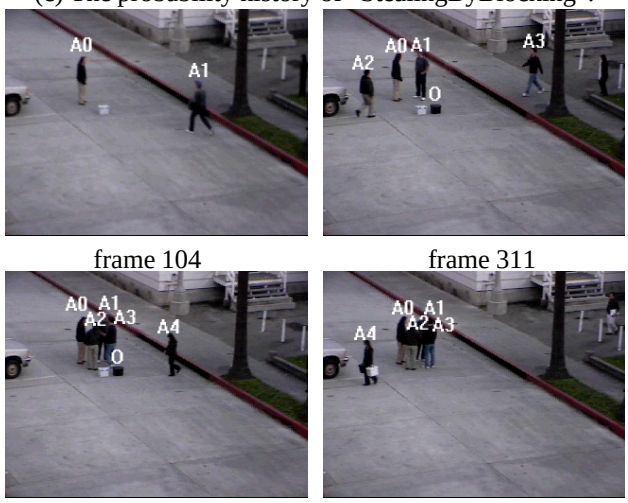
(a) The probability history of "Approach" and "StopAt".



(b) The probability history of "Converse".



(c) The probability history of "StealingByBlocking".



frame 104 frame 311

frame 436 frame 557

(d) Selected video frames of the result.

Figure 2: The recognition of multi-thread event "StealingBy-Blocking". The annotation is according to the initialization with the highest probability.

6 Conclusion

We have defined an event ontology that allows natural representation of complex spatio-temporal events common in the physical world by a composition of simpler (primitive) events. The composition from simper to more complex event naturally leads to a language description of the events. An event recognition language (ERL) is defined for the users to describe events of various complexities for the purposes of

event recognition and query. Although we only showed example in the context of video surveillance, the representations are abstract and is capable to model a wide variety of other events.

In the future, we are interested in exploring the following aspects:

- Validate the expressive power of ERL in various application domains.
- Augment the single-thread composite event representation in ERL with additional logical constructs.
- Integrate low-level processing, event recognition techniques and ERL to create systems for event recognition and query.

References

[1] James F. Allen and George Ferguson, Actions and Events in Interval Temporal Logic, *J. of Logic and Computation*, 4(5):531-579, 1994.

[2] D. Ayers and M. Shah, Monitoring Human Behavior from Video Taken in an Office Environment, *Image and Vision Computing*, vol.19, no.12, pp.833-846, 2001.

[3] Matthew Brand, Vera Kettnaker, Discovery and Segmentation of Activities in Video, *IEEE Trans. on PAMI*, vol. 22, no. 8, August 2000.

[4] Isaac Cohen and Gerard Medioni, Detecting and Tracking Moving Objects for Video Surveillance, *Proc. Int. Conf. on Computer Vision and Pattern Recognition*, 1999.

[5] Y.A. Ivanov, A.F. Bobick, Recognition of Visual Activities and Interactions by Stochastic Parsing, *IEEE Trans. on PAMI* no. 8, pp. 852-872, August 2000.

[6] A Kojima, T. Tamura and K. Fukunaga, Natural Language Description of Human Activities from Video Images Based on Concept Hierarchy of Actions, *Int J of Computer Vision*, vol.50, no.2, pp.171-184, 2002.

[7] Somboon Hongeng, Francois Bremond and Ramakant Nevatia, Representation and Optimal Recognition of Human Activities, *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2000.

[8] Somboon Hongeng and Ramakant Nevatia, Multi-Agent Event Recognition, *Proc. Int. Conf. on Computer Vision*, 2001.

[9] Somboon Hongeng, A Unified Bayesian and Logical Approach for Video-base Event Recognition, Phd thesis, University of Southern California, 2002.

[10] Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland, A Bayesian Computer Vision System for Modeling Human Interactions, *IEEE Trans. on PAMI*, vol. 22, no. 8, August 2000.

[11] Nathanael A. Rota and Monique Thonnat, Activity Recognition from Video Sequences using Declarative Models, *Proc. European Conf. on A.I.*, 2002.

[12] Andrew D. Wilson, Aaron F. Bobick, Parametric Hidden Markov Models for Gesture Recognition, *IEEE Trans. on PAMI*,Vol. 21, No. 9, September 1999.

[13] Tao Zhao, Ram Nevatia and Fengjun Lv, Segmentation and Tracking of Multiple Humans in Complex Situations, *Proc. Int. Conf. on Computer Vision and Pattern Recognition*, 2001.