Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video

Benjamin Laxton[†] Jongwoo Lim[‡] David Kriegman[†]
blaxton@cs.ucsd.edu jlim@honda-ri.com kriegman@cs.ucsd.edu
[†] Department of Computer Science, University of California at San Diego
[‡] Honda Research Institute USA inc.

Abstract

We present a scalable approach to recognizing and describing complex activities in video sequences. We are interested in long-term, sequential activities that may have several parallel streams of action. Our approach integrates temporal, contextual and ordering constraints with output from low-level visual detectors to recognize complex, long-term activities. We argue that a hierarchical, object-oriented design lends our solution to be scalable in that higher-level reasoning components are independent from the particular low-level detector implementation and that recognition of additional activities and actions can easily be added. Three major components to realize this design are: a dynamic Bayesian network structure for representing activities comprised of partially ordered sub-actions, an object-oriented action hierarchy for building arbitrarily complex action detectors and an approximate Viterbi-like algorithm for inferring the most likely observed sequence of actions. Additionally, this study proposes the Erlang distribution as a comprehensive model of idle time between actions and frequency of observing new actions. We show results for our approach on real video sequences containing complex activities.

1. Introduction

Automatically monitoring and recognizing human activities is a long sought-after goal in the computer vision community. Successful implementation of a vision system with these capabilities would enable new methods for automatic surveillance monitoring, aware environments and computer interfaces. Motivated by this goal, researchers have developed systems that attempt to automatically recognize and in some cases label when actions and activities are occurring. Significant progress has been made in recognizing short-term actions [1, 2, 4, 5, 9, 14, 20] and, to a lesser extent, monitoring and recognizing longer-term activities [7, 15, 16, 18]; however, many challenges remain before activity recognition systems can be deployed in general settings.

This work furthers the idea that an understanding of the semantics and temporal constraints of complex activities is necessary for recognition and monitoring systems. This work presents methods for representing and efficiently reasoning about these types of activity constraints. Scalable methods for representing complex action and activity models are also explored.

In this paper a method for accurately monitoring, recognizing and labeling complex activities in video that involve several sub-actions and objects and typically last several minutes is developed. The proposed solution consists of three major components: action detectors, a Dynamic Bayesian Network (DBN) that encodes prior knowledge of action ordering constraints, and an approximate Viterbi-based inference algorithm that maintains an estimate of the most likely activity state given a DBN and the output of a set of detectors applied to an input video.

Throughout this work, the term *action* is used to refer to a simple short-term motion with a specific purpose, and *activity* is used to refer to a sequence of actions to achieve a more complex and meaningful goal. Actions are atomic components used to define an activity, which may be composed of several sub-activities. The system is not limited to instantaneous atomic actions, but instead uses a comprehensive temporal model for actions. Three important concepts used throughout the design to achieve activity monitoring and detection are introduced: *Object Orientation*, *Hierarchy of Objects, Actions and Activities* and *Contextual Disambiguation*. They provide a principled way to include contextual information and modify or add to representations of actions and activities. Finally, decoupled detection layers allow for clear, modular and reusable representations of activities. The following section contains a literature review of recent work in the area. An overall description of the system is given in Section 3. Details of our method are described in Sections 4 and 5, and results on long video sequences are shown in Section 6.

1-4244-1180-7/07/$25.00 ©2007 IEEE

2. Related Work

Recently, many approaches to detecting short-duration actions in video have been proposed. These approaches develop some statistical feature, computed over the space-time volume defined by a video segment, to characterize specific actions [1, 2, 4, 5, 9, 14, 20]. Typically, a modest number of action classes are used and a classifier is learned by clustering the features computed for training sequences. New examples are then discriminatively classified by computing the features and matching to the nearest action using some distance metric. These methods show high recognition rates on short term, often periodic actions, however, there is no notion of semantic meanings which are necessary to develop an interpretive context.

Motivated by success in natural language processing, stochastic context free grammars (SCFGs) have been proposed by several researchers as a method of including an interpretive context for more complex activities that typically last tens of seconds to minutes, are comprised of several sub-actions and may involve interaction with several objects [8, 11, 15]. CFG production rules define a grammar over activities specifying expected orders of sub-actions. Parsing mechanisms are used to translate underlying action detections into a coherent activity description. While these approaches show promising results on fairly complex scenes, there is no notion of a temporal model and, furthermore, only single-threaded activities are addressed.

Other researchers have applied Hidden Markov Models (HMMs) to video streams for recognizing activities [7, 10, 12]. Several variant HMM topologies have been proposed. Moore et al. define several HMMs for distinct actions in conjunction with an object detection system to exploit relationships between specific objects and actions [10]. Oliver et al. propose layering HMMs as a way to deal with model complexity and show results for activities in an office setting [12]. Hongeng and Nevatia make use of semi-HMMs that relax the Markovian assumption [7]. In general, HMM techniques suffer when an activity consists of concurrent or partially ordered streams of action, which is often the case for interesting activities. In these cases an HMM must enumerate an exponentially large state space.

A related class of approaches use dynamic Bayesian networks (DBNs) to model activities [3, 6, 13, 16]. Whereas an HMM representation must consider the entire temporal state space, DBNs divide the state space into constituent variables and are able to leverage sparseness in the variable relationships. Pinhanez and Bobick capture relative temporal relationships, 'past', 'now', 'future', in a DBN framework for activity detection [13]. Gong and Xiang explore several DBN topologies and report results on an aircraft loading scenario [6]. DBNs have been shown to improve low-level object trackers by leveraging rich event ordering constraints to deal with missing, spurious or fragmented tracks [3]. Shi et al. introduce a DBN framework that makes use of event ordering constraints and provides some simple modeling of event duration within the network [16]. They show how the DBN can detect activities that may consist of concurrent streams of action. DBN techniques show promise for complex activity recognition tasks, but there is room for improvement in terms of modeling sub-action relationships, scaling to large numbers of activities and choosing appropriate models of action duration.

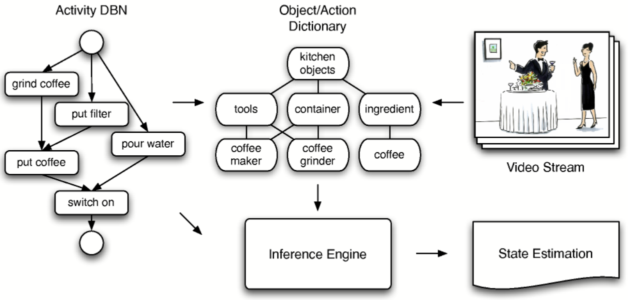

Figure 1. An overall schematic of our system for activity recognition.

3. Methodology

We use a hierarchical, modular approach to represent, monitor and recognize activities developed through the design concepts of *Object Orientation*, *Hierarchy of Objects, Actions and Activities* and *Contextual Disambiguation* each of which are described in terms of our approach. An overall schematic of the proposed system is shown in Figure 1. The three primary components of the system are the object-action dictionary, the activity DBN and the inference module. The object-action dictionary defines the action detectors and initially processes the video frames. This component makes use of the 'Object Orientation' and 'Hierarchy of Objects, Actions and Activities' design principles. Objects are defined rather loosely and may include inanimate objects such as cups, books, computers etc., people present in the scene or even locations such as a doorway, hallway, table etc. Each object may have associated attributes such as current state, action descriptor, location, etc.. Objects inherit the attributes and visual descriptors defined for their parent objects and may add new attributes and additional contextual information to refine the interpretation of the visual descriptors.

The activity DBN (ADBN) component encodes prior information about an activity domain such as the expected actions and ordering constraints between the actions. The ADBN formulation allows activities to be defined hierarchically; activities may be comprised of sub-activities. This allows for modular design of arbitrarily complex activities. Additionally, the ADBN makes use of 'Contextual Disambiguation' by enforcing temporal and ordering constraints that provide powerful contextual cues for activity recognition. The contextual cues capture the observation that although many actions may be observed in a video sequence, and detectors for the actions may be noisy, observing the actions in the correct order with correct temporal properties is unlikely unless the specified activity is being performed. Looking at the problem from the other way, given a sequence of observed actions that closely follows the ordering constraints of an activity, if a single action is missing it is probably due to detector error and the context can allow for recovery.

The third component of the system is the inference mechanism. Here, the ordering and temporal constraints imposed by the ADBN are combined with output from the action detectors defined by the object-action dictionary to generate an estimate of the current state, and the sequence of states leading up to it. The modular hierarchical, object-oriented nature of the system design affords great flexibility in terms of underlying detector implementation and supported activity domains.

4. Dynamic Bayesian Networks for Activity Recognition

We use a constrained form of Dynamic Bayesian Networks (DBNs) to encode the temporal structure and ordering constraints found in many activities. DBNs are Bayesian networks that, in addition to representing dependencies between states, also represent a temporal probability model over the states. The DBN representation takes advantage of sparseness in the dependencies both between states in the same time slice and across time. Additionally, DBNs can represent arbitrary distributions in the state-space. These properties make them a good choice for modeling real-world scenarios. A general DBN, N, is defined in Equation 1.

$$N = \{V, E, S\} : \begin{cases} V & \text{Set of state variables} \\ E & \text{Set of evidence variables} \\ S & \text{Set of edges} \end{cases} \qquad (1)$$

An edge in $s_i \in S$ may be between two state variables, which defines a transition constraint, or between some $e_i \in E$ and some $v_i \in V$, which defines links between states and observations. The state variables may be either discrete or continuous. The operation of the DBN is dependent upon a prior distribution over the states, a sensor model and a transition model.

4.1. Representing Activities with DBNs: ADBNs

An *activity* has already been defined as a sequence of actions required to achieve some meaningful goal. Activities are defined hierarchically such that each activity is made up of one or more sub-activities and each sub-activity is itself made up of sub-activities or a sequence of atomic actions. Atomic actions are the simplest representations in the framework and are tied directly to underlying detection mechanisms. This conceptual hierarchy allows for modular construction of complex activities as shown in Figure 2.

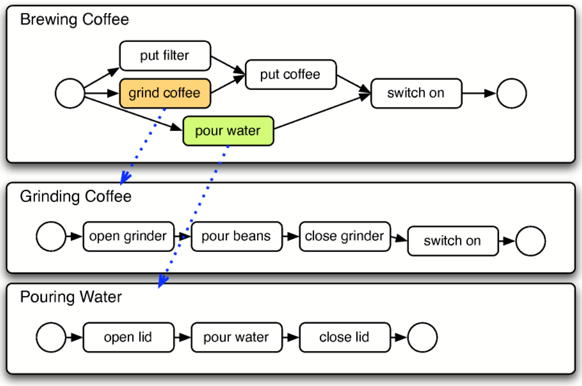

Figure 2. An example of how activities are represented hierarchically. Each node may either be atomic or hierarchically composed of a sub-activitiy.

We propose a special form of a DBN called an Activity Dynamic Bayesian Network (ADBN) defined as follows:

- V: Set of atomic actions that make up an activity.
- E: Evidence nodes that incorporate probabilities from visual detectors and a temporal model. There is one $e_i \in E$ for each $v_i \in V$.
- S: Edges between state nodes enforce ordering constraints. So if $s_i = (v_k, v_{k+1})$, v_k must occur before v_{k+1}. Additionally, there is a single edge between each v_i and its associated e_i.

Each $v_i \in V \rightarrow \langle\text{'waiting'}, \text{'active'}, \text{'finished'}\rangle$. These states correspond to the intuitive notion of the flow of action through a step-wise process. While a sub-action has not yet occurred it is labeled as *'waiting'*. When it is currently happening it should be labeled *'active'* and after it has occurred it should be labeled *'finished'*.

The information contained by the graph $N = \{V, E, S\}$ defines a single slice in time for the ADBN. To form the semi-infinite ADBN required for reasoning over the temporal dimension, this layer is replicated for each time slice. Inter-temporal edges are added from each node to its corresponding node in the next time slice as well as to all nodes it is connected to within the slice. Intuitively, this specifies that when in some state at time t, at time $t + 1$ it is possible to remain in the same state or go to the next ordered state defined by S. An example activity and a partial expansion over time of its associated ADBN are shown in Figure 3.

This representation bears resemblance to causality models developed in previous works such as PNF networks [13] and PNETs [16]. We adopt two constraints introduced in the work on PNETs, partial-ordering and one-time activation. The partial ordering constraint enforces the rule that a parent node must not be activated after any child node has been activated. The one-time activation constraint only allows for a node to be activated for one time span, although

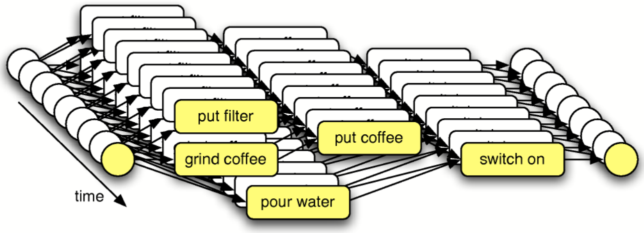

Figure 3. An expansion of a ADBN defining the activity of making coffee with intra and inter temporal links shown.

the span may last many frames. These constraints are flexible enough to allow for modeling many activities, but allow the number of search paths to be significantly pruned by the inference procedure.

Using this formulation, several desirable properties emerge. First, the high-level description of an activity given by the ADBN is in some sense independent from the underlying world state detectors. The only requirement is that the detectors can provide some probabilistic estimate of the likelihood of each sub-action occurring. Second, the ADBN formulation provides a natural way to hierarchically model activities. Ultimately, each ADBN is comprised of a sequence of atomic actions, however, sub-activities can themselves be represented by ADBNs and plugged in as prespecified components. Lastly, ADBNs can represent multiple parallel streams of action and can efficiently encode all valid partial orderings of parallel action streams.

4.2. Modeling Temporal Extent

When describing activities that happen over time, temporal frequency and duration can be a powerful cue. Although ADBNs can incorporate arbitrary probabilistic descriptions of temporal behavior as evidence variables, most researchers have either ignored temporal modeling [10, 12, 8] or used very simple models like Gaussians [7, 16]. When Gaussians are used to model temporal extent for activity recognition they must either be learned for each sub action in the model, or generalized for a set of disparate actions. Learning these distributions requires time consuming labeling and even then a Gaussian may not provide a meaningful temporal model for the action since the semantic description of actions is often independent of whether it is performed for a long or short duration or interrupted for some indefinite period. Finally, this model does not incorporate any information about temporal relationships between actions such as occurrence rate and idle time. Our approach does not explicitly model the duration of each action. For many cases a single distribution cannot meaningfully capture the variation in how the action is performed. The duration can vary greatly depending on the situation and it is unrealistic to expect to have a general duration model for many actions.

Instead of modeling the duration of each action we suggest modeling the duration of 'no action': the time period between the end of an action and the beginning of the next action. We call this the 'idle time model'. Thus we have only one type of event to model, and we can assume each occurrence to be independent. To address the problem of modeling temporal relationships we propose the Erlang Distribution, which is closely related to the Poisson distribution. The Poisson distribution is a probabilistic model for the number of events occuring over some time period whereas the Erlang distribution models the probability of an elapsed time occurring between k independent events.

The general form of the Erlang distribution is given in Equation 2.

$$f(t:k,\lambda) = \frac{\lambda^k t^{k-1} e^{-\lambda t}}{(k-1)!} \text{for } t > 0 \qquad (2)$$

Here, k is called the shape parameter and corresponds to the number of events being modeled. λ is called the rate parameter and represents the expected number of events during a unit time. The function is defined over temporal values $t \geq 0$. Since we are only interested in modeling a single event $k = 1$. This special case of the Erlang distribution is the exponential distribution given in Equation 3.

$$f(t;\lambda) = \begin{cases} \lambda e^{-\lambda t} & ,t \geq 0 \\ 0 & ,t < 0 \end{cases} \qquad (3)$$

Only a single event is modeled and as a result we only need to specify a single parameter, λ, for an entire ADBN. This takes advantage of the powerful constraints offered by temporal modeling without running into the problem of representing heterogeneous actions with a simplified distribution or learning separate distributions for each action. Although we do not explicitly model action duration, our method does not exclude an action duration model. When such a model exists, it can be easily included as an augmenting component of an action detector by weighting the observation likelihood of the action. Section 4.3 demonstrates the power of this temporal model in pruning search paths for activity recognition.

4.3. Inference Procedure

Given a ADBN that defines some activity, the goal of an inference procedure is to produce the most likely explanation for the underlying evidence. This corresponds to choosing a label from $\langle$'waiting', 'active', 'finished'$\rangle$ for each action node in V during each frame that maximizes the probability given the past states of the ADBN, the current probabilistic measures of the world captured by the nodes $e_i \in E$, and the constraints defined by the edges $s_i \in S$. Using the simplifying assumption that each action happens only once in the course of the activity, an equivalent formulation is to maintain a sample that assigns each node $v_i \in V$ a start-time, $start(v_i)$ and an end-time, $end(v_i)$ that maximizes the probability given the evidence.

Exact inference in a ADBN is problematic because of the huge state-space. Notice that an action node beginning its activation at time t and another at time $t-1$ represent two distinct samples. In a ADBN with n independent nodes the number of possible assignments grows as $O(t^{2n})$ where t is the temporal window. Even though the intra-slice connections may limit the number of truly independent nodes the state-space grows exponentially in t. As a result, only approximate inference methods can be efficient.

In general, filtering methods can be used to estimate the posterior distribution over the current state of a ADBN at time t given all previous observations. However, an estimate of the posterior distribution is unnecessary. We are only interested in the sequence of state assignments to a ADBN that obey the constraints and best explain the evidence. The Viterbi algorithm is often used to find the globally optimal assignment of states for an HMM. This is infeasible for ADBNs in general so we develop an approximate Viterbi-like algorithm.

First we set up our notation: $\mathbf{X}_t$ is the random variable for a time-slice of the ADBN at time t, $\mathbf{x}_{1:t}$ is a sequence of state assignments leading up to the current time, $\mathbf{o}_t$ is the observation at time t, and $\mathbf{o}_{1:t}$ is a sequence of observations leading up to the current time. The goal of our approximate Viterbi algorithm is to find the state assignments, $\mathbf{x}^*_{1:t}$ with maximal probability given all evidence. We can derive the recursive relationship shown in Equation 4 and use it for inference.

$$\max_{\mathbf{x}_{1:t}} \mathbf{P}(\mathbf{x}_{1:t}, \mathbf{X}_{t+1}|\mathbf{o}_{1:t+1}) = \alpha \mathbf{P}(\mathbf{o}_{t+1}|\mathbf{X}_{t+1}) \times \max_{\mathbf{x}_t}\left(\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) \max_{\mathbf{x}_{1:t-1}} P(\mathbf{x}_{1:t-1}, \mathbf{x}_t|\mathbf{o}_{1:t})\right) \quad (4)$$

Each sample has a specific state assignment and the associated probability as given in the Equation 4. When the observation at time $t+1$ is available, we propagate the sample according to the transition model and update the probability. Since it is infeasible to maintain all samples, we keep a set of high probability assignments and discard the rest. In practice, the probability of a sample $P(\mathbf{x}_{1:t-1}, \mathbf{x}_t|\mathbf{o}_{1:t})$ is updated as follows:

$$P(\mathbf{x}_{1:t-1}, \mathbf{x}_t|\mathbf{o}_{1:t}) = P(\mathbf{o}_t|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{x}_{t-1})P(\mathbf{x}_{1:t-1}|\mathbf{o}_{1:t-1})$$

The first term on the right side of the equation is the observation probability measured by detectors. The second term is the transition probability specified by a ADBN, and the last is the probabilty of the path through the ADBN leading up to the sample i.e. the probability of the generating sample. $P(\mathbf{x}_t|\mathbf{x}_{t-1})$ reflects the idle time model (and action duration model if used), probability of missed detections and the connectivity of nodes in the ADBN.

The prior model, shown in Equation 5, specifies that initially the ADBN is in a default start state.

$$P(\mathbf{X}_0) = \begin{cases} 1 & : \mathbf{x}_0 = \langle \text{finished,waiting,...waiting} \rangle \\ 0 & : \text{for all other assignments to } \mathbf{x}_0 \end{cases} \quad (5)$$

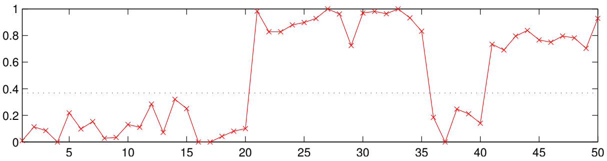

Figure 4. An example sequence of observation probability for an action A, compared to the unit idle time penalty $e^{-\lambda}$ (blue dotted line; $\lambda = 1$ is used here).

The exponential distribution model for the idle time between actions allows particles to be systematically pruned. As an example, consider Figure 4, which shows the observation likelihoods of an action A. There are four possible cases:

1. $A =$ *waiting* and $o_t^A \leq e^{-\lambda}$
2. $A =$ *waiting* and $o_t^A > e^{-\lambda}$
3. $A =$ *active* and $o_t^A \leq e^{-\lambda}$
4. $A =$ *active* and $o_t^A > e^{-\lambda}$

In case 1, there is no reason to change A's state to *active*, since the resulting probability of the change will be always smaller than that of no-change for all possible future state assignments. In cases 2 and 3, both choices of assignment change must be explored since, depending on future observations, either choice can have higher probability. In case 4, the observation probability of the child nodes of A is also considered. If all of the child observation probabilities are smaller than the threshold $e^{-\lambda}$, A will keep the state of *active*, since this gives the maximal probability. Otherwise, all possibilities are investigated in the next time step. For relatively accurate detectors (especially low false-positive detectors), this propagation rule substantially prunes the number of expansions. This pruning procedure may not be applicable to filtering methods as it could distort the posterior estimate.

5. Underlying Action Representations

Although the ADBN formulation for activity recognition is decoupled from the specific underlying sensor implementation, it does ultimately require input from the visual signal. Specifically, it needs a probabilistic detector for each sub-action that comprises the activity. In general it is not immediately clear how to do this in domains where a large number actions may be observed and each action may exhibit significant variation. We propose an object-oriented framework for building probabilistic action detectors from the visual signal.

Our framework for creating action detectors is called an *'Object-Oriented Activity Knowledge Base'* . Like object-oriented programming, in this framework objects are the building blocks. This object-centric view allows composition and inheritance to be used to build arbitrarily complex and descriptive action models of a domain. Objects

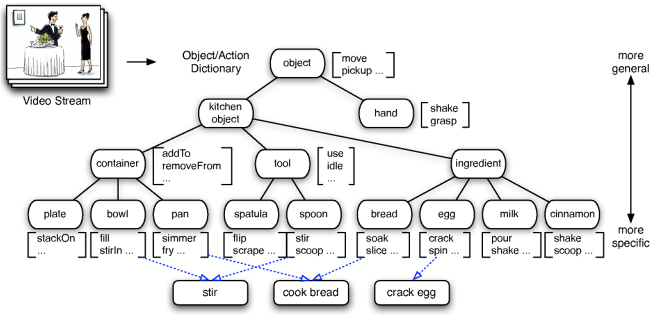

Figure 5. A partial object-oriented action knowledge base for a kitchen domain.

are arranged hierarchically according to 'is-a' relationships where more general classes of objects are represented earlier in the hierarchy. A set of possible actions is associated with each object. The object is responsible for implementing a detector that outputs a probability for the particular action given the input signals. The outputs of these detectors are cascaded to child objects that can apply more filtering to build detectors for their own associated actions.

An activity knowledge base for the kitchen domain is shown in Figure 5. The video stream is fed as input to the object hierarchy. Each object in the hierarchy can be thought of as a filter that acts upon the probabilistic outputs of its ancestors and any new features calculated from the input video stream. Detection probabilities from multiple objects can be combined as a single probability for input to a ADBN as shown in 'CrackEgg' and 'CookBread' nodes. New objects and their associated actions can easily be added by inserting a node into the hierarchy and defining the relevant detectors. This does not require modifying existing objects and their actions – making the solution scalable.

Although the action detectors are object-centric, purely motion based detectors are not precluded from the framework. For example, the related work on motion descriptors described in Section 2 is largely concerned with motion signatures of the human body and do not consider interactions with objects in the world. This type of detector could be incorporated into the framework by including a 'person' object and defining all the motion descriptors as detectors for the possible actions. In this way the implementation is independent of the specific types of features calculated over the input video making it flexible and applicable to a wide array of domains.

6. Experimental Results

To demonstrate the utility of our activity recognition approach we have tested the framework on several real video sequences of activities including a cooking sequence involving 30 distinct actions and lasting several minutes. The results highlight the challenges of long-term activity recognition and strengths of our approach. We provide a detailed analysis of our system's performance on recognizing cooking activities. Additional results included in the supplemental materials are multiple cooking sequences and manipulation of a toddler learning toy.

6.1. Cooking Sequence

Our goal is to simultaneously recognize activities and accurately label when atomic actions are occuring. We show results for recognition on a challenging cooking sequence in which french toast is prepaired. The ADBN that encodes prior information about action ordering constraints for the french toast activity is shown in Figure 6. Notice there are several parallel paths in the ADBN encoding partial ordering. The ADBN intuitively and efficiently models both independence and dependence among the steps. The french toast activity lasts several minutes and involves 30 sub-actions and 10 objects, making it a difficult scenario that shows the strength of our approach.

6.2. Implementation details

A list of objects for this scenario is shown in Figure 5 where the leaves of the object-oriented action knowledge base represent the objects of interest in the scene. The input video frames are taken from an rgb+depth camera setup. Given a sequence of video frames, three basic image statistics are computed for each object using the color and depth information: a probability of the object being removed, occluded or present. Additionally, we implemented a hand tracking module that maintains an estimate of the current position and velocity of hand movement.

The sub-actions we are interested in detecting and labeling are the nodes, $v_i \in V$, of the ADBN shown in Figure 6. Each of these nodes takes input from the underlying action detector of the associated object in the form of a probabilistic measurement. The temporal model specifies a distribution over the idle times between actions. Then, the inference procedure is used to combine the detector and temporal-modeling information with knowledge of past states to provide an estimate of the current activity state.

There are only two tunable parameters for the ADBN and inference implementation; the number of samples maintained by the inference procedure and a λ that specifies the rate for the Erlang distribution described in Section 4.2. The implementation is written entirely in Matlab and runs in ≈5 frames/sec depending on the number of samples maintained.

6.3. Results

Some sample output frames of the implementation run over a french toast sequence are shown in Figure 7. The graph shown under each frame mimics the structure of Figure 6 and darkened nodes represent the current best state estimate output by the inference procedure. The output for the 1194 frame sequence is included in the supplemental materials.

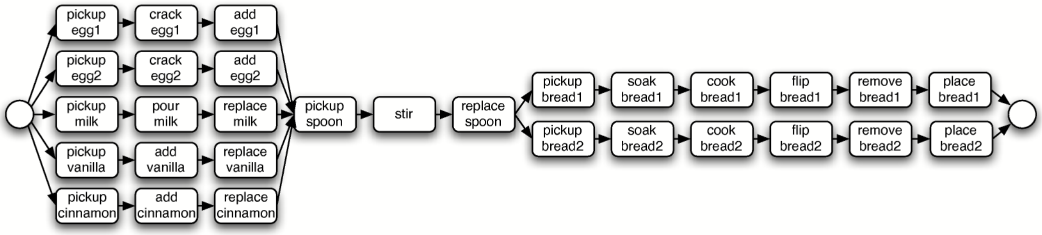

Figure 6. An ADBN that defines prior contextual ordering information for cooking french toast.

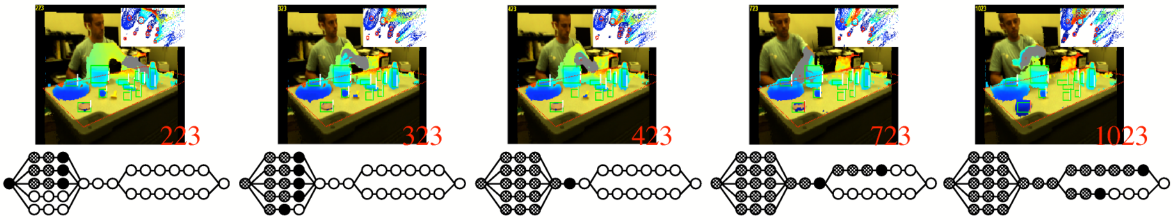

Figure 7. Some sample output frames of our ADBN with approximate Viterbi inference.

An activation plot showing typical output of our method run over a french toast video sequence is shown in Figure 9. The atomic actions are listed on the y axis and frame numbers on the x axis. The output probabilities of the action detectors are shown in gray scale values on the plot. The hand-labeled ground truth appear as green triangles and the output labels of the ADBN with approximate Viterbi inference are shown as red circles. Notice that the underlying action detectors exhibit significant noise. Relying on these alone to label the sequence would result in very poor results. In contrast, our method's labels closely follow the ground truth. No actions are labeled out of order and only 3, 'remove_bread1' 'remove_bread2 and 'place_bread2' , are mislabeled in time. Furthermore, our method recovers from the labeling error of 'remove_bread1' for following actions.

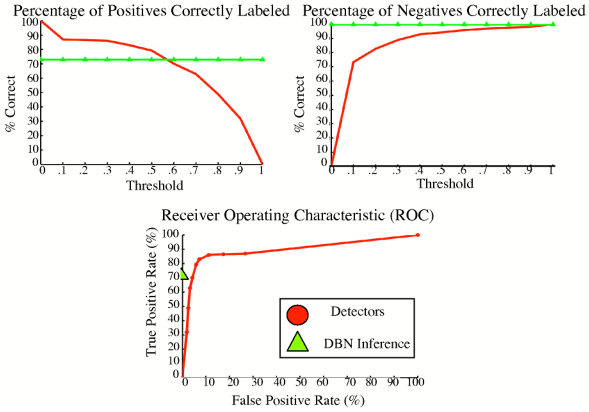

Figure 8. Performance characteristics of our ADBN inference method versus raw action detectors.

Performance characteristics on frame by frame labeling of individual actions for our method compared to the probabilistic action detectors are shown in Figure 8. The top distribution shows the percentage of positive examples correctly labeled by our method, the green horizontal line, versus the raw detectors for threshold values from 0 to 1. Similarly, the middle distribution shows performance for labeling negative frames correctly. These plots show that our method always outperforms the detectors in terms of correctly labeling negative frames. The detectors do label more of the positive frames correctly for some thresholds at the cost of significant error on the negatives (false positives). This is further illustrated by the bottom plot showing the ROC for the detectors and the *(true-positive, false-positive)* point for our method. Our method correctly labeled 77% of all positive frames while maintaining an extremely low false-positive rate of .03%.

7. Discussion and Future Work

In this paper we have developed an efficient method for monitoring recognizing and labeling complex activities in video sequences. Furthermore, our design incorporated hierarchical modeling of activities, object-centric design of the underlying action detectors and the use of contextual information making it a flexible, scalable approach. Experiments on long video sequences illustrate the power of this approach.

The current system requires the topology of the ADBN to be specified by hand. We plan to explore approaches for automatically learning models that encode the structure inherent in activities. We are also exploring different underlying action detectors. We are interested in applying our ADBN and approximate Viterbi inference procedure on top of a motion descriptor model of action to describe long-term

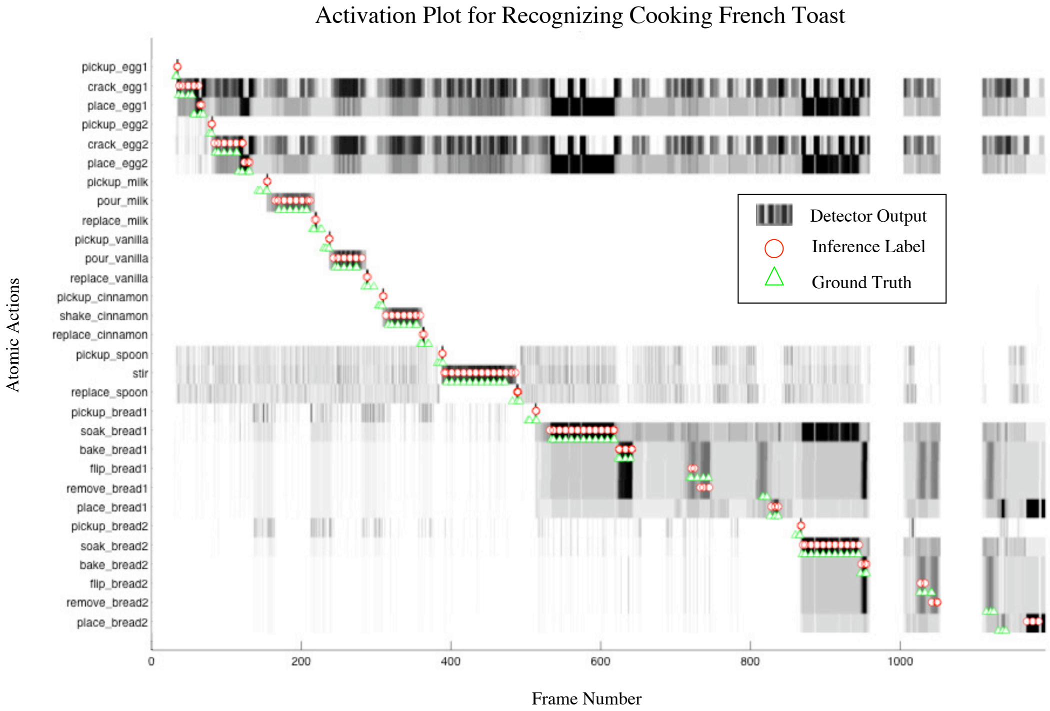

Figure 9. Here we show a comparison of the output of individual action detectors with the labels assigned by our method and ground truth. The gray-scale values represent the probabilities output by each action detector where darker represents higher probability.

motions.

Acknowledgments

This work was partially funded under NSF IGERT Grant DGE-0333451 and a grant from the Honda Research Institute.

References

[1] M. Blank, L. Gorelick, E. Shechtman, M. Irani, R. Basri. Actions as Space-time Shapes. In *ICCV*, 2005.

[2] A. Bobick, J. Davis. The Recognition of Human Movement Using Temporal Templates. In *PAMI*, vol. 23(3), March 2001.

[3] M. Chan, A. Hoogs, R. Bhotika, A. Perera. Joint Recognition of Complex Events and Track Matching. In *CVPR*, 2006.

[4] P. Dollar, V. Rabaud, G. Cottrell, S. Belongie. Behavior Recognition via Sparse Spatio-Temporal Features. In *ICCV*, 2005.

[5] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

[6] S. Gong, T. Xiang. Recognition of group activities using dynamic probabilistic networks. In *ICCV*, 2003.

[7] S. Hongeng, R. Nevatia. Large-scale event detection using semi-hidden markov models. In *ICCV*, 2003.

[8] Y. Ivanov, A. Bobick. Recognition of Visual Activities and Interactions by Stochastic Parsing. In *PAMI*, vol. 22(8), August 2000.

[9] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[10] D. Moore, I. Essa, and M. Hayes. Exploiting human actions and object context for recognition tasks. In *ICCV*, 1999.

[11] D. Moore and I.Essa. Recognizing multitasked activities using stochastic context-free grammar from video. In *Proceedings of AAAI Conference*, 2002.

[12] N. Oliver, E. Horvitz, and A. Garg. Layered representations for human activity recognition. In *ICMI*, 2003.

[13] C. Pinhanez, A. Bobick. Human action detection using PNF propagation of temporal constraints. In *CVPR*, 1998.

[14] D. Ramanan, D.A. Forsyth. Automatic Annotation of Everyday Movements. In *NIPS*, 2003.

[15] M.S. Ryoo, J.K. Aggarwal. Recognition of Composite Human Activities through Context-Free Grammar based Representation. In *CVPR*, 2006.

[16] Y. Shi, Y. Huang, D. Minnen, A. Bobick, and I. Essa. Propagation networks for recognition of partially ordered sequential action. In *CVPR*, 2004.

[17] Y. Shi, A. Bobick, and I. Essa. Learning temporal sequence model from partially labeled data. In *CVPR*, 2006.

[18] V.T. Vu, F. Bremond and M. Thonnat. Automatic Video Interpretation: A Novel Algorithm for Temporal Scenario Recognition. In *IJCAI*, 2003.

[19] J. Ward, P. Lukowicz, G. Tröster, and T. Starner. Activity recognition of assembly tasks using body-worn microphones and accelerometers. In *PAMI*, vol. 28(10), October 2006.

[20] D. Weinland, R. Ronfard, E. Boyer. Automatic Discovery of Action Taxonomies from Multiple Views. In *CVPR*, 2006.