Expectation Grammars: Leveraging High-Level Expectations for Activity Recognition *

David Minnen, Irfan Essa, Thad Starner
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280 USA
{dminn, irfan, thad}@cc.gatech.edu

Abstract

Video-based recognition and prediction of a temporally extended activity can benefit from a detailed description of high-level expectations about the activity. Stochastic grammars allow for an efficient representation of such expectations and are well-suited for the specification of temporally well-ordered activities. In this paper, we extend stochastic grammars by adding event parameters, state checks, and sensitivity to an internal scene model. We present an implemented system that uses human-specified grammars to recognize a person performing the Towers of Hanoi task from a video sequence by analyzing object interaction events. Experimental results from several videos show robust recognition of the full task and its constituent sub-tasks even though no appearance models of the objects in the video are provided. These experiments include videos of the task performed with different shaped objects and with distracting and extraneous interactions.

1. Introduction

Humans often have strong prior expectations concerning the constituent actions of an activity. For example, one would expect a tennis player to hit a ball after it bounces on her side, and a cook will almost always stir his batter after adding the ingredients. In this paper, we present our approach for supplying and leveraging this kind of knowledge to support machine understanding and recognition of longer-term human activities. We employ stochastic grammars to represent such expectations and enhance this representation by adding event parameters, state checks, and sensitivity to an internal scene model.

Certainly, a full theory and specification of all human behaviors is well beyond the capabilities of current knowledge representation and reasoning systems. Specifying the relevant knowledge for a specific task in a reasonably constrained environment, however, is quite feasible. In such situations, an intelligent system can benefit from the constraints provided by the expectations associated with the task. These benefits include enhanced robustness, improved error recovery, and the capability to make predictions about future behavior at multiple levels of abstraction.

We present an implemented system capable of analyzing video of a person performing complicated tasks. At present, we focus on the Towers of Hanoi task since it provides a reasonably complex, non-deterministic task that still follows clear rules. Our approach uses manually-specified expectations in the form of a parameterized stochastic grammar in order to automatically generate a detailed annotation of the video. This means that the system can identify when the Towers of Hanoi task occurs and when each sub-activity (*e.g.,* movement of a disc from the first to the third peg) occurs. We present several experiments that show our system to be robust to image noise, imperfect foreground/background segmentation, distracting actions, and occlusions. Finally, we explore the boundaries implied by the assumptions of our system, discuss possible enhancements to our analysis algorithm, and outline future research directions for activity specification and recognition.

2. Related Work

Recently, there has been significant progress in the area of recognizing activities in video for surveillance, monitoring, and data annotation tasks. While much work on activity recognition research has focused on short-term, motion-based activities such as gesture, pose, and isolated action recognition (see [13] for examples of such contributions), there has also been progress in the recognition of extended sequences of actions.

A large body of work in the recognition of human activity has relied on recovering a sequence of states using stochastic, model-based approaches. For example, hidden Markov models (HMMs) have become very popular for rec-

*This material is based upon work supported under a National Science Foundation Graduate Research Fellowship, NSF ITR Grant #0121661, NSF Career Grant #0093291, and DARPA HumanID Grant #F49620-00-1-0376.

ognizing gestures [1, 12], sign language [17, 15], and actions [9, 2]. It has been shown, however, that the recognition of extended activities with predefined context or inherent semantics is difficult to accomplish with a purely probabilistic method unless augmented by additional structure. Examples of such activities include parking cars or dropping people off at the curb [7, 10], longer-term office and cooking activities [9], airborne surveillance tasks [3], observing simple repair tasks [2], and American Sign Language recognition [15].

Previous research has explored a variety of representations other than stochastic, state-based models including event-based predicate logic [4], deterministic action grammars [2], stochastic context-free grammars (SCFGs) [7, 10], past-now-future (PNF) networks [11], and per-frame correspondence graphs [8]. In addition, Fern and Siskind, *et al.* [4, 14] present a major departure from the motion and state-based paradigm by basing analysis on force dynamic interactions.

Stochastic grammars with context-sensitivity to an internal scene model are used in our system. This choice reflects an assessment of several relevant factors including (a) the temporal complexity of tasks like the Towers of Hanoi game, cooking a specific recipe, and monitoring complex medical regiments (*e.g.,* blood glucose monitoring), (b) the constraints imposed by the output of our segmentation and tracking system (discussed in detail in section 3), and (c) the ease of manually developing the activity specification.

The sub-tasks involved in solving the Towers of Hanoi problem are well-ordered (*i.e.,* no sub-tasks at the same level of abstraction will temporally overlap) but non-deterministic. This means that representations that allow more complex temporal structures, such as PNF or full interval algebra networks [11], are unnecessary for this domain. Furthermore, the uncertainty that arises due to the existence of multiple, valid options during certain parts of the activity suggest the need for probabilistic analysis. Moore's experiments [10] dealing with the recognition and parsing of blackjack show that stochastic grammars are well suited for this task. Ivanov and Bobick [7] discuss the computational equivalence of state-based event networks and context-free grammars when operating on finite (and thus enumerable) activities, but show that many common temporal structures are far easier to express within a grammar. Finally, Möller and Posch [8] and Fern *et al.* [4] both present sophisticated recognition systems for short actions, but do not explore the analysis of longer-term activities.

Systems that employ grammatical structure must eventually transform their input into the symbols that compose the alphabet of the grammar. In traditional parsing, this is accomplished directly by a lexer, and in typical speech and sign language recognition systems (*e.g.,* [6, 15]) segmentation at one level occurs implicitly during Viterbi decoding while each HMM provides a probabilistic symbol for a higher-level grammar.

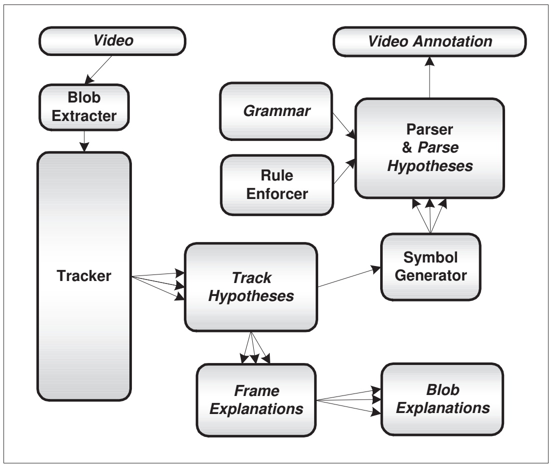

Figure 1: *System overview: The system analyzes activities in a video and generates detailed annotations.*

In our domain, however, the visual input to the system is not easily transformed into a symbol stream. This is largely due to the absence of a well-defined basic unit of motion analysis, analogous to the phoneme in speech. Without such a basic unit, the development of an accepted preprocessing stage, similar in purpose to the widely used Mel-Frequency Cepstral Coefficients (MFCCs) in speech processing, becomes very difficult. As an alternative, our approach extracts symbols based on object interactions using an event-based paradigm [10, 7, 2, 8]. The details of this transformation, along with other important aspects of the system, are discussed in the following section.

3. System Overview

Although a holistic approach to the analysis of each object's behavior is possible, the computational complexity and requisite training data is prohibitive. Instead, our system relies on interactions between blobs, which represent the objects in the scene. Three assumptions underlie our low-level vision system for blob extraction:

1. The system will *not*, in general, be able to segment the objects from each other in a static scene.
2. The system will *not*, in general, be able to recognize an object in a static scene, even if it is isolated and properly segmented.
3. The system *will*, however, be able to distinguish between a foreground object and the background, even if the foreground blob represents several objects.

The consequence of these assumptions is that our system is not dependent on the performance of any particular appear-

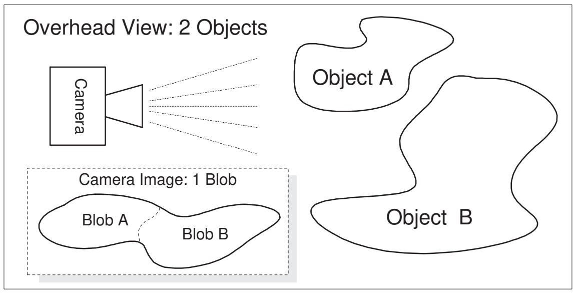

Figure 2: *Distinct objects will appear as a single, merged blob whenever one object partially occludes the other*

ance modeling or object recognition technique. Our framework, therefore, is designed to be independent of lighting or other incidental appearance changes of the objects and should perform equally well with easily discernable or very similar objects (*e.g.,* compare the objects in Figures 7 and 8) so long as the foreground/background segmentation assumption is not violated. Finally, it means that all objects must be identified based on how they *behave* rather than how they *appear*.

Briefly, our analysis system proceeds as follows:

1. Low-level blobs are extracted from the video.
2. Multiple, hypothetical interpretations are generated for each frame.
3. Inconsistent hypotheses are pruned.
4. Domain-general events are generated from the blob interactions for each frame.
5. The events are transformed into domain-specific symbols by the symbol generator and passed to the parser.
6. All valid parses are generated; each represents a different interpretation of the activity.
7. Each parse generates a private internal model used to detect future inconsistencies.
8. The most likely consistent parse is chosen as the correct interpretation of the activity.

In the following subsections we detail these steps and discuss how they interact in our analysis algorithm. Figure 1 provides a schematic outline of the system.

Object Detection: A stationary camera captures a visual record of each activity. Relevant objects are detected by finding connected groups of foreground pixels labeled by a background subtraction algorithm based on a simplified version of the system by Horprasert, *et al.* [5]. This algorithm provides consistent results even in the presence of lighting intensity changes caused by shadows cast by either the human participant or the objects.

Blob Labelling and Tracking: Each blob found during the object detection phase can represent zero, one, or several actual objects (see Figure 2). Due to this ambiguity and our assumption that static, appearance-based classification is not generally possible, a "blob explanation" is created that maps blobs to objects. A special *noise* label is used to mark a blob that does not correspond to a real object (see Figure 7), *distracter* is used for a blob representing a real object that is not actually part of the activity (see Figures 8 and 9), and *hand* is used to mark the blob corresponding to the participant. All other blob labels are task-dependent and, for the three-disc Towers of Hanoi task, are simply *Block A*, *Block B*, and *Block C*, representing each disc in ascending size order.

Since it is not possible to accurately label the blobs from a single frame, or even from a short sequence, our system maintains multiple "frame explanations" for each ambiguous time step. These frame explanations are composed of several blob explanations, which specify the identity and origin of each blob. Each frame explanation represents a different hypothetical interpretation of the scene. As the activity plays out, simple heuristics and probabilistic measures eliminate and rank the hypotheses. For example, only the *hand* is self-propelled, so if a hypothesis contains a blob labelled as *Block A* and that blob begins to move independently, then the hypothesis will become inconsistent and thus be pruned.

Probabilistic penalties help rank all of the self-consistent label hypotheses. Such penalties may arise heuristically (*e.g.,* a *hand* label is preferred to a *noise* label) or due to tracking. Tracking in our system is handled using an explanation based approach. The origin and disappearance of each blob in every frame is explained using the events *enter*, *exit*, *merge*, *split*, and *tracked*. Consistency is enforced through time (*e.g.,* a *tracked* object can not change identity) and within a single frame (*e.g.,* a blob can not *merge* with itself). The likelihood of each hypothesized explanation is influenced by a fixed penalty associated with the explanation and by the similarity between the blobs representing each *tracked* object. This similarity is computed as a weighted distance in a feature space composed of typical blob features such as low-order moments, mean color, bounding-box coordinates, and color variance.

Symbol Generation: Although domain-general events form the basis of our activity specifications, a domain-specific module transforms these events into context-sensitive symbols corresponding to the terminals of the current activity grammar. For example, in the Towers of Hanoi task, *merge* and *split* events are transformed into *touch_X*, *release_X*, *remove_X*, and *remove_last_X* symbols, where *X* represents one of the pegs in the task. This transformation is possible due to a domain-specific internal model of the

```
ToH -> Setup, enter(hand), Solve, exit(hand);
Setup -> TowerPlaced, exit(hand);
TowerPlaced -> enter(hand, block_A, block_B, block_C),
               Put_1(block_A, block_B, block_C);
Solve -> state(Tower = TowerStart), MakeMoves, state(Tower = TowerGoal);
MakeMoves -> Move(block) [0.1]  |  Move(block), MakeMoves [0.9];

Move -> Move_1-2 | Move_1-3 |  Move_2-1 | Move_2-3  |  Move_3-1 | Move_3-2;

Move_1-2 -> Grab_1, Put_2;
Move_1-3 -> Grab_1, Put_3;
Move_2-1 -> Grab_2, Put_1;
Move_2-3 -> Grab_2, Put_3;
Move_3-1 -> Grab_3, Put_1;
Move_3-2 -> Grab_3, Put_2;

Grab_1 -> touch_1, remove_1(hand, ~)  |  touch_1(~), remove_last_1(~);
Grab_2 -> touch_2, remove_2(hand, ~)  |  touch_2(~), remove_last_2(~);
Grab_3 -> touch_3, remove_3(hand, ~)  |  touch_3(~), remove_last_3(~);

Put_1 -> release_1(~) | touch_1, release_1;
Put_2 -> release_2(~) | touch_2, release_2;
Put_3 -> release_3(~) | touch_3, release_3;
```

Figure 3: *Stochastic grammar for the Towers of Hanoi task: The full grammar specification includes other declarations, for example that* block A *is a* block*. Also, the tilde ($\sim$) as a parameter means that the particular terminal should inherit arguments from its parent rule's invocation.*

scene and the ability to map blob bounding-box coordinates to a peg.

The transformation of low-level events into symbols is both local in time and relative to an internal model. The system keeps a different model for each track hypothesis and, as discussed in the following section, for each valid parse of that track (see Figure 4). That the transformation from events to symbols is local means that not all symbols will be correct. The system handles this by assigning a fixed probability to each symbol and by maintaining multiple parse hypotheses. As additional visual evidence becomes available, the system will choose one of the parses based on a discovered inconsistency or because one is more likely overall. Thus, local errors are reduced by pooling evidence through time.

4. Stochastic Parsing

The parameterized stochastic grammar used to represent the Towers of Hanoi is shown in Figure 3. The parsing algorithm used by our system is derived from the original work by Stolcke [16] and its subsequent application to computer vision by Ivanov and Bobick [7] and Moore and Essa [10]. Here we only discuss deviations in our system from this previous research.

Parameters: Parameterized grammars increase the specificity of a stochastic grammar without resorting to customized vision detectors as used in [10]. Our system permits both specific objects and object types as parameters and also allows the matched object to be bound to a name for later reference in the grammar. Thus, an activity specification could include *enter(block A)* to signify that *block A* must enter the scene at a particular time, or it could include *enter(block:FirstBlock)* to mean that any block can enter and that the particular block observed can later be referenced by the name "FirstBlock." Object types are defined in the grammar and are part of the activity specification, not the recognition system itself.

Scene Model Generation & Maintenance: Each valid parse maintains an internal model of the current scene. The model is an abstract, domain-specific structure. In the Towers of Hanoi task, it represents where each object is located and whether it is supported by the hand or the table. Initialization places all objects outside of the scene and each parsed event-symbol effects a change. For example, a *release_1* event would cause the block currently supported by the hand to move to the first peg in the model. Note that the model does not need to be probabilistic because each parse keeps a separate version. Since all valid parses are maintained simultaneously, an incorrect model will be naturally eliminated when the associated parse is pruned.

Activity States: Activity states ensure that a particular assertion about the scene holds at a given point during the activity. In the Towers of Hanoi task, for example, *state(Tower = TowerGoal)* signifies that the task has not completed until the tower is in the goal state. Note that this assertion is not checked directly in the current video frame, as this would require complex, static scene analysis. Instead, the assertion is verified by consulting the internal model of the scene. In the *state(Tower = TowerGoal)* case, the assertion is easily checked by verifying that all blocks are on the last peg.

Distracter Objects: Although the *noise* label can be applied to any blob that does not correspond to an object relevant to the current activity, it is inappropriate in some situations. During many activities, concurrent tasks may be taking place. Although the system does not have any expectations concerning these other tasks, they should not disrupt the recognition and analysis of the main activity.

To handle such situations, the system uses the *distracter* label. *Distracter* objects are ignored during parsing, thus making it possible for the system to correctly identify an activity even while unrelated object interactions occur in the scene. However, because there are no expectations associated with *distracter* objects, semantic inconsistencies can not be detected and thus misinterpretations involving the behavior of the *distracter* objects will not be pruned. Instead, the system ranks hypotheses by always preferring explanations with fewer *distracter* object interactions. This is a useful heuristic in practice, but it can be inappropriately applied in certain, ambiguous situations.

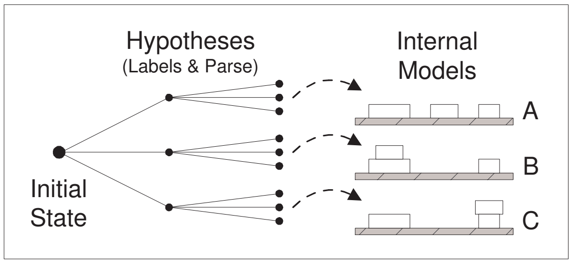

Figure 4: *Multiple parse hypotheses are maintained simultaneously; each keeps its own internal model of the scene. Note that model C is inconsistent with the rules of the Towers of Hanoi task and thus will be pruned.*

Symbol Error Handling: Three kinds of symbol errors can occur: insertion, deletion, and substitution errors. Substitution errors are treated as a simultaneous pair of one insertion and one deletion error. Insertion and deletion errors are handled by maintaining multiple activity parses. Whenever a new symbol is scanned, two parse states are created. The first represents the case in which the symbol is valid, while the second assumes that an insertion error has occurred. In the latter case, the parse is penalized according to the probability that the symbol is erroneous, namely $1 - p(Symbol)$. This allows the parser to ignore the current symbol but still scan an equivalent symbol later in the activity, albeit with a lower probability.

Deletion errors, on the other hand, are handled by "hallucinating" the missing symbols when it is necessary to form a complete parse. If a real event-symbol is later encountered, any equivalent hallucinations are eliminated. Also, the probability of a hallucinated symbol is set to be significantly lower than the typical probability of a valid observation, thus naturally leading the system to prefer parses with the most visual evidence.

5. Experiments

We performed several experiments to test our system. In the Towers of Hanoi domain, the experiments varied along four dimensions: the type and color of the objects representing the discs, the number of steps used to complete the game, the presence of a distracter object, and the presence of complete occlusions. We also developed components for another task modeled after the Simon game (see Figure 10). In this activity, the player places three blocks and then taps either a specified pattern (*e.g.,* ABC), a palindrome (*e.g.,* ABCBA), or a set of pairs (*e.g.,* AABBCC). The videos for each experiment and the grammar for the Simon game can be found on the web at http://www.cc.gatech.edu/cpl/projects/expectationGrammars.

Blocks of different colors and sizes were used as the discs in Experiment I (see Figure 5). The initial tower was

Figure 5: *Experiment I: Blobs can merge due to grasping or occlusion*

```
[113-739 (1-29) ToH . ]
  [113-739 (1-29) Setup, Solve, exit(hand) . ]
    [113-234 (1-4) TowerPlaced, exit(hand), enter(hand) . ]
      [113-169 (1-2) enter(hand, red, green, blue), Put_1(red, green, blue) . ]
        [169-169 (2-2) release_1 . ]
    [248-733 (5-28) state(Tower=TowerStart), MakeMoves, state(Tower=TowerGoal) . ]
      [248-733 (5-28) Move(block), MakeMoves . ]
        [248-315 (5-7) Move_1-3 . ]
          [248-315 (5-7) Grab_1, Put_3 . ]
            [248-270 (5-6) touch_1, remove_1(hand) . ]
            [315-315 (7-7) release_3 . ]
        [337-733 (8-28) Move(block), MakeMoves . ]
          [337-385 (8-10) Move_1-2 . ]
            [337-385 (8-10) Grab_1, Put_2 . ]
              [337-350 (8-9) touch_1, remove_1(hand) . ]
              [385-385 (10-10) release_2 . ]
          [396-733 (11-28) Move(block), MakeMoves . ]
            [396-442 (11-14) Move_3-2 . ]
              [396-442 (11-14) Grab_3, Put_2 . ]
                [396-396 (11-12) touch_3, remove_last_3 . ]
                [418-442 (13-14) touch_2, release_2 . ]
            [452-733 (15-28) Move(block), MakeMoves . ]
              [452-529 (15-17) Move_1-3 . ]
                [452-529 (15-17) Grab_1, Put_3 . ]
                  [452-452 (15-16) touch_1, remove_last_1 . ]
                  [529-529 (17-17) release_3 . ]
              [537-733 (18-28) Move(block), MakeMoves . ]
                [537-588 (18-20) Move_2-1 . ]
                  [537-588 (18-20) Grab_2, Put_1 . ]
                    [537-553 (18-19) touch_2, remove_2(hand) . ]
                    [588-588 (20-20) release_1 . ]
                [602-733 (21-28) Move(block), MakeMoves . ]
                  [602-654 (21-24) Move_2-3 . ]
                    [602-654 (21-24) Grab_2, Put_3 . ]
                      [602-602 (21-22) touch_2, remove_last_2 . ]
                      [632-654 (23-24) touch_3, release_3 . ]
                  [673-733 (25-28) Move(block) . ]
                    [673-733 (25-28) Move_1-3 . ]
                      [673-733 (25-28) Grab_1, Put_3 . ]
                        [673-673 (25-26) touch_1, remove_last_1 . ]
                        [709-733 (27-28) touch_3, release_3 . ]
```

Figure 6: *Experiment I: Annotation for a typical Towers of Hanoi video: Each line shows the time interval (in frames) followed by the symbol range and the actual label.*

placed on the first peg as part of the setup phase of the activity, and then the Towers of Hanoi game was played . The final parse for this video is shown in Figure 6, which represents the full annotation for a successfully recognized game.

Experiment II used plastic, torus-shaped objects for the discs, this time all of the same size and color (see Figure 7). The results from analysis are identical to similar experiments using the more easily distinguishable blocks. Note that the yellow pegs used in this video are considered part of the background and thus are not identified or analyzed by our system.

In Figure 8 you can see a frame from Experiment III that

Figure 7: *Experiment II: All of the discs have the same size and shape. Also, note that the dark shadow under the hand is labelled as* noise

Figure 8: *Experiment III: The small foreground block is a* distracter *object in this scene. The system must determine from behavior that it is irrelevant to the activity even when placed on a tower.*

includes a distracter object. The small, foreground block is not part of the Towers of Hanoi task, even though it looks similar to the other blocks and does occasionally move from one tower to another. During analysis, the system does not initially know how to classify this block. It will explore interpretations in which it is part of the task, but visual evidence later in the video will show these interpretations to be inconsistent. In some situations, a lack of visual evidence will permit multiple valid interpretations, and the system will be forced to choose between the possibilities. The parse and track likelihoods serve as a basis for this decision as the most likely parse is always preferred.

Finally, Experiment IV deals with full occlusion of part of the activity (see Figure 9). The system depends on the visual evidence from the first two pegs to substantiate a positive recognition of the task. Two kinds of unavoidable errors arise in this situation. First, since there is no direct evidence for the hidden events, there is no way to determine when, or even if, they actually occur. If enough indirect evidence exists, the system will assign an arbitrary time for the hallucinated events, corresponding to the time when the relevant object is first hidden or first reappears. The second problem deals with the fact that the system can hallucinate events indefinitely, although even the hallucinated symbols must be consistent with the internal scene model. In theory, every blob in a video could be labelled as *noise* or *distracter* objects and any activity can be hallucinated. In practice, however, such an interpretation will be highly improbable and a more realistic interpretation will be chosen. The system

Figure 9: *Experiment IV: In this frame, the book fully occludes the third peg. The system must infer the interactions that are hidden.*

Figure 10: *Screenshot from the Simon game demonstrating analysis of another domain.*

should detect, however, if the activity was not completed correctly rather than always hallucinating a valid interpretation. In this case, a simple threshold on the average symbol likelihood will distinguish between scenes that have enough visual evidence to substantiate the activity and those that do not.

6. Discussion & Future Work

Our approach to activity recognition has several limitations in both its ability to represent and track blob configurations and in its ability to represent expectations about an activity. We outline several such issues and propose methods of enhancing our system below.

Additional blob explanations: Several kinds of low-level vision errors can be accommodated by our framework but are not currently incorporated into the system. Although we correctly handle erroneous blobs by labelling them as *noise*, sometimes noise will cause divisions in legitimate foreground objects. In the presence of such noise, an object might disappear for several frames or might split into what appears to be two distinct objects. By increasing the blob explanation labels to include *partial-blob*, *partial-merge*, *noise-disappear*, and *noise-reappear*, we can explicitly detect and account for these low-level errors.

Appearance models for event disambiguation: In the description of our system, we discussed how our system does not use object appearance models. In some instances, however, even cursory knowledge about the appearance of an object can be useful. For example, if the hand passes behind a block, knowledge of the appearance of the block can be used to decrease the likelihood of a *grab* event since grasping will generally require the hand to partially occlude the block. More importantly, consider a situation in which two blocks are on a peg and the hand touches them. If a drastic change in the shape or color distribution of the blocks follows a subsequent *split* event, it is likely that the hand removed one of the blocks. If, however, no change is observed, then the hand probably touched or just passed by the blocks. In the current system, this ambiguity would not be resolved until a later interaction occurs, for example a *put* action on another tower.

Adaptive tracking via high-level feedback: Although our high-level parser can influence tracking results by deeming certain interpretations inconsistent with the specified grammar, we would like to build a more tightly coupled system. Consider an object that appears as two blobs due to noise, as discussed above. When the parser detects that a single object apparently split into two pieces, it could notify the tracker rather than immediately eliminating that interpretation. The tracker could then try an alternative foreground/background segmentation method or adjust its parameters in an effort to bring the low-level vision in line with the high-level interpretation.

Limitations of stochastic grammars: Finally, we would like to explore more expressive activity representations. Stochastic grammars can only implicitly represent concurrent actions (*e.g.,* stir the batter *while* holding the bowl), and are not able to express global rules except by explicit enumeration. For example, during the Towers of Hanoi task, the hand can always exit and then reenter the scene without disrupting the activity. Our system can only handle such occurrences by assuming that both the *enter* and *exit* events are insertion errors. To correctly represent this knowledge in a stochastic grammar, however, would require a modification that includes an optional *exit/enter* pair between every pair of "real" events. This is both cumbersome and unnecessarily complex. One of our goals for future research is to develop a more expressive activity specification that could represent such knowledge directly.

7 Conclusion

Stochastic grammars provide a useful representation for specifying high-level expectations about an activity. This representation is easily understood by humans and can be efficiently employed by a computational system. We show how a stochastic grammar can be used to pool evidence through time in order to recover from local errors and find a consistent overall interpretation of an activity. Our system demonstrates that static object recognition is unnecessary for the recognition of activities governed by strong expectations and contextual constraints. Finally, we discuss possible extensions to our system including enhanced blob explanations and a low-level feedback and adaptation scheme based on high-level constraints.

References

[1] A. F. Bobick and A. D. Wilson. A state based approach to the representation and recognition of gesture. *PAMI*, 19(12):1325–1337, December 1997.

[2] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *CVPR*, 1997.

[3] F. Bremond and G. Medioni. Scenario recognition in airborne video imagery. In *DARPA Image Understanding Workshop 1998*, pages 211–216, 1998.

[4] A. Fern, R. Givan, and J.M. Siskind. Learning temproal, relational, force-dynamic event definitions from video. In *AAAI*, pages 159–166, 2002.

[5] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtractiona nd shadow detection. In *ICCV FRAME-RATE Workshop*, 1999.

[6] X.D. Huang, Y. Ariki, and M. A. Jack. *Hidden Markov Models for Speech Recognition*. Edinburgh Univ. Press, 1990.

[7] Y.A. Ivanov and A.F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *PAMI*, 22(8):852–872, August 2000.

[8] B. Möller and S. Posch. Analysis of object interactions in dynamic scenes. In L. van Gool, editor, *Pattern Recognition, Proc. of 24th DAGM Symposium, Zurich, Switzerland*, LNCS 2449, pages 361–369. Springer, September 2002.

[9] D. Moore, I. Essa, and M. Hayes. Exploiting human actions and object context for recognition tasks. In *ICCV'99*, pages 80–86, 1999.

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

[11] C. Pinhanez. *Representation and Recognition of Action in Interactive Spaces*. PhD thesis, 1999.

[12] J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputs using hidden markov models. In *WACV94*, pages 187–194, 1994.

[13] M. Shah and R. Jain. *Motion Based Recognition*. Computational Imaging and Vision Series. Kluwer Academic Publisher, 1997.

[14] J.M. Siskind. Visual event classification via force dynamics. In *AAAI/IAAI*, pages 149–155, 2000.

[15] T. Starner, J. Weaver, and A.P. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *PAMI*, 20(12):1371–1375, December 1998.

[16] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2), 1995.

[17] C. Vogler and D. Metaxas. A framework for recognizing the simultaneous aspects of american sign language. *CVIU*, 81(3):358–384, March 2001.