An Extended Grammar System for Learning and Recognizing Complex Visual Events

Zhang Zhang, *Member*, *IEEE*, Tieniu Tan, *Fellow*, *IEEE*, and Kaiqi Huang, *Senior Member*, *IEEE*

Abstract—For a grammar-based approach to the recognition of visual events, there are two major limitations that prevent it from real application. One is that the event rules are predefined by domain experts, which means huge manual cost. The other is that the commonly used grammar can only handle sequential relations between subevents, which is inadequate to recognize more complex events involving parallel subevents. To solve these problems, we propose an extended grammar approach to modeling and recognizing complex visual events. First, motion trajectories as original features are transformed into a set of basic motion patterns of a single moving object, namely, primitives (terminals) in the grammar system. Then, a Minimum Description Length (MDL) based rule induction algorithm is performed to discover the hidden temporal structures in primitive stream, where Stochastic Context-Free Grammar (SCFG) is extended by Allen's temporal logic to model the complex temporal relations between subevents. Finally, a Multithread Parsing (MTP) algorithm is adopted to recognize interesting complex events in a given primitive stream, where a Viterbi-like error recovery strategy is also proposed to handle large-scale errors, e.g., insertion and deletion errors. Extensive experiments, including gymnastic exercises, traffic light events, and multi-agent interactions, have been executed to validate the effectiveness of the proposed approach.

Index Terms—Rule induction, parsing, event recognition.

1 Introduction

Recently, visual event recognition has become one of the most active topics in computer vision, due to its wide application prospects in video surveillance, video retrieval, and human-computer interaction [1], [2]. Much work has been reported for recognizing a wide range of events from short-term, single-agent actions, e.g., run and walk [4], [5], [6], to long-term, complex activities, e.g., complex operating procedures and multi-agent interactions [10], [12], [13], [14], [23], [25], [27].

The focus of this paper is to recognize long-term complex events that include the interactions between multiple moving objects and/or the interactions between moving objects and static references in scenes. As shown in Fig. 1, gymnastic exercises can be considered as the collaboration of moving feet and hands. The traffic light events at a crossroad consist of the passings of multiple vehicles at different lanes with different directions. And multi-agent interactions can also be represented as the combination of individuals' relative motions.

Motivated by a cognitive intuition that a complex event can be perceived as a number of components and their relations, we adopt a hierarchical approach based on some syntactic techniques [3]. As Stochastic Context-Free Grammar (SCFG) has a powerful capability to recover hierarchical structures in a input string, many researchers have used it to recognize fairly complex visual events, such as hand gesture [20], human-vehicle interactions [21], operating procedures [22], [23], [27], [28], vehicle behaviors [26], human interactions [25], etc. Commonly, the low-level image features are first transformed into a set of terminal symbols. The transformation is completed by some prior knowledge, e.g., motion detection in some special areas. Then, the symbol stream is fed into a grammar parser to recover the hidden structure and recognize the interesting events.

However, previous approaches have two major limitations that prevent stochastic grammar from real applications. One is that all event rules (grammar productions) are predefined by experts, which means huge manual cost. The other is that the grammar parser can only handle sequential relations between subevents, while parallel temporal relations often exist in complex events, such as "*raising both hands overhead concurrently*."

Therefore, for the above problems, we propose an extended grammar approach to recognize complex events. Fig. 2 presents the flowchart of our approach. As shown in the figure, the motion trajectories of moving objects are extracted as the original representation of complex events. Then, these trajectories are transformed into a set of primitives which describe some basic spatial relationships of a single moving object to a reference in the scene. In the following, the primitive stream is fed into the module of event rule induction, where a Minimum Description Length (MDL) based rule induction algorithm is performed to discover the hidden temporal structures in primitive stream. As results, a number of event rules are learned automatically, where the traditional SCFG production is extended by Allen's temporal logic [43] so as to describe the complex temporal relations between subevents. In testing, a Multithread Parsing (MTP) algorithm is executed to parse a given

• *The authors are with the National Lab of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, 95# Zhongguancun East Road, Haidian District, Beijing 100190, P.R. China.*
E-mail: {zzhang, tnt, kqhuang}@nlpr.ia.ac.cn.

Manuscript received 25 Apr. 2009; revised 7 Jan. 2010; accepted 12 Jan. 2010; published online 25 Feb. 2010.
Recommended for acceptance by S. Belongie.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-04-0258.
Digital Object Identifier no. 10.1109/TPAMI.2010.60.

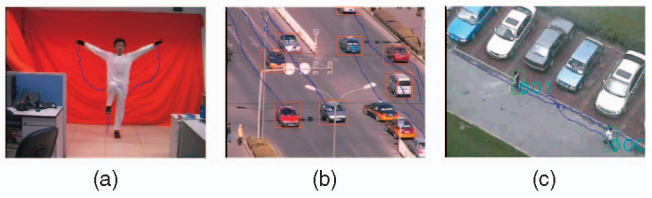


Fig. 1. Examples of multi-agent events involving complex temporal relations. (a) Gymnastic exercises. (b) Traffic light events. (c) Multiagent interactions.

primitive stream, further recognizing the interesting complex events that may include parallel temporal relations.

Compared with previous work, the main contributions of this paper can be summarized as follows:

- The traditional SCFG production is extended with Allen's temporal logic [43], where a temporal matrix is used to describe the temporal relations.
- In previous grammar-based work, the event rules are predefined manually. Here, based on an MDL-based rule induction algorithm, we try to learn the event rules automatically.
- Instead of the common parser that only handles sequential relations between subevents, a multi-thread parsing algorithm is adopted to recognize complex events with parallel temporal relations.
- Considering the large-scale errors, e.g., insertion and deletion errors, a Viterbi-like error recovery strategy is designed to enhance the robustness of the proposed parser.

This paper is an enhanced combination of our previous conference papers [40] and [41] which focus on rule learning and event parsing, respectively. Here, we integrate them into a full-fledged system. More experimental results with variant parameter settings are presented to evaluate the proposed algorithm. Furthermore, new experiments on recognizing multi-agent interactions are shown in this journal paper.

In activity recognition, most popular public data sets, such as the KTH action data set [6], Weizmann action data set [8], and INRIA data set [9], are adopted to recognize simple actions of single person, e.g., run and walk, where the person does not interact with other moving objects or references. To evaluate higher level complex activities recognition algorithms, very few public data sets exist [2]. Thus, our goal in experiments is to give some empirical proofs of the proposed approach, rather than trying to improve the performance in standard databases. Here, extensive experiments involving gymnastic exercises, traffic light events, and multi-agent interactions are performed to validate the effectiveness of our approach.

The remainder of this paper is organized as follows: Section 2 describes related work on visual event recognition. Section 3 introduces the feature representation and primitive modeling. As the core content, the MDL-based rule induction and the multithread parsing are presented in Sections 4 and 5, respectively. Section 6 gives the experimental results. Finally, we conclude this work in Section 7.

2 Related Work

As introduced in [2], visual events can be divided into two major classes, namely, actions and activities. Actions denote those short-term, single-agent events, such as walk and run. For actions, researchers mainly aim to develop effective spatiotemporal descriptors at image level, e.g., the spatial-temporal interesting points [4], [5], [6] and 3D spatiotemporal volume-based descriptors [7]. High recognition rates on some public databases, e.g., the KTH actions data set [6], Weizmann action data set [8], and INRIA action data set [9], have been reported.

However, classification of simple actions is not enough to interpret what happens in a long duration with contextual meanings. The events occurring in a long duration usually consist of several simple subevents. In [2], these events are denoted as "activities." To recognize

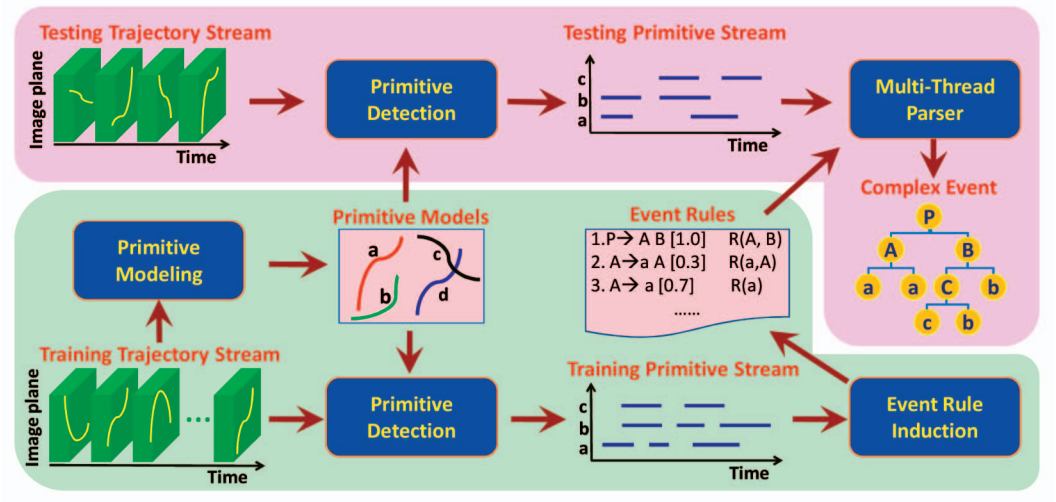
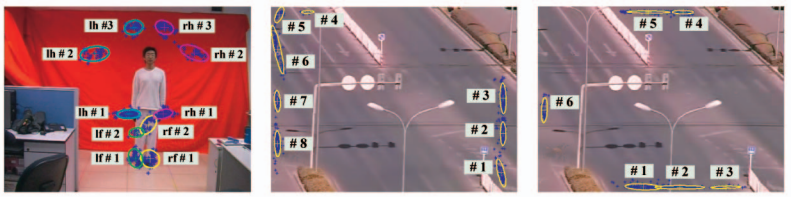


Fig. 2. The flowchart of our approach. First, motion trajectories (the dark green boxes denote the spatiotemporal scopes of trajectories) are mapped to a set of primitives that serve as the terminals in the grammar. Then, from the primitive stream, an event rule induction algorithm is adopted to learn a set of event rules. In testing, an MTP algorithm is adopted to recover the temporal structure (parse tree) in a given primitive stream, further recognizing the interesting complex events. The pink and green backgrounds denote the recognizing process and the learning process, respectively.



(a) (b) (c)

Fig. 3. Illustrations on the references in the scenes. (a) The stop points in the gymnastic exercises, where lf = left foot, rf = right foot, lh = left hand, and rh = right hand. (b) and (c) The entries and exits in the traffic scene, respectively.

complex events, the key problem is how to model the intrinsic spatiotemporal structures.

For modeling the temporal structures in complex events, a great deal of work has also been done at various levels. Some researchers worked at the signal level, in which the extracted feature sequence was directly feed into some probabilistic graphical models, e.g., Hidden Markov Model (HMM) and Dynamic Bayesian Network (DBN), where the model topology encodes the event structure [11], [12], [13], [14], [15]. Some well-studied inference approaches, e.g., the EM algorithm [19], have been developed for parameter estimation. Probabilistic graph models have the advantage of handling the uncertainties in low-level image processing or tracking. However, along with the increasing event complexity, such as the number of moving agents involved, the performance seriously relies on an appropriate model topology which would be too difficult to learn from little training data. In most cases, the model topology needs to be predefined manually.

On the other hand, some researchers perform event recognition at the symbol level. Some basic events (primitives) are first detected from low-level features. Then, complex events are represented as the combination of several subevents with certain spatial, temporal, or logical constraints [31], [33]. As mentioned above, SCFG has been adopted in much work [21], [22], [23], [24], [25], [26], [27]. However, two major limitations exist, i.e., rule learning and complex temporal relation modeling.

For rule learning, some methods were proposed. Needham et al. [29] set up an autonomous cognitive agent system to find the game protocols with Inductive Logic Programming (ILP). Hamid et al. [30] adopted n-grams to discover the meaningful structures in a long-time event streams, where the transitions between adjacent events were counted. Toshev et al. [34] used an a priori-based mining algorithm to find the frequent temporal patterns. In [32], Fleischman et al. also discovered some hierarchical frequent patterns. Then a Support Vector Machine (SVM) is adopted to classify events.

For parallel relation modeling, some work has also been done. The authors extended the traditional grammar by introducing the Temporal Relation Events (TRE) in [28]. And simple temporal reasoning is appended in grammar parsing. However, the temporal reasoning is only performed between two subactivities belonging to different agents. In [25], with a context-free grammar-based representation, Ryoo and Aggarwal used Allen's temporal logic [43] to model the parallel relations in human interactions. And the recognizing problem was turned into a constraint satisfaction problem. In [44], a multidimensional parsing algorithm was also proposed to handle parallel temporal relations in multimodal human-computer interaction. Nevertheless, the above methods have a lack of error handling; the parsing may fail due to the errors in primitive detection.

In our work, motion trajectories are adopted as original features, as trajectory is one of the most useful pieces of information to represent the behavior of moving object. Much effort has been done on trajectory analysis. In [16], a codebook of prototype is generated using online vector quantization. Then, different motion paths in scenes were obtained by the co-occurrence statistics of the prototypes within single trajectory. Porikli and Haga [17] proposed an HMM distance to measure the similarity between two trajectories, then eigenvalue decomposition was applied to find the usual clusters, where the number of clusters can be estimated by counting the number of eigenvectors used to span the feature similarity space. Hu et al. [18] also developed a hierarchical clustering system to learn a single vehicle's motion patterns. Based on the learned clusters, behavior prediction and anomaly detection have been realized. However, most previous work only focuses on modeling the motion patterns of single trajectory. In this work, we also try to discover the temporal structures hidden in a trajectory stream.

3 Feature Representation and Primitive Modeling

The grammar-based method works at symbol level. Thus, the low-level features should be transformed into a set of primitives (atoms) serving as the terminals in a grammar.

Referring to [33], a primitive is defined as a single, coherent unit of movement performed by one agent, which describes some aspects of the spatial relationship of the moving agent to a reference object. In our work, each primitive corresponds to one trajectory segment that describes the evolution of spatial position of one moving agent to other references in the scene.

As shown in Fig. 3, the references are obtained by clustering some meaningful points in trajectories, e.g., the stop points in the exercises or the entering/existing points in the crossroad.

Then, the trajectories occurring between each pair of references are considered as one basic motion pattern of single object. For example, the motion pattern in Fig. 4a indicates "*left foot moves from the reference $lf\#1$ to the reference $lf\#2$*" shown in Fig. 3. Finally, for each motion pattern, the HMM is trained as the primitive detector.

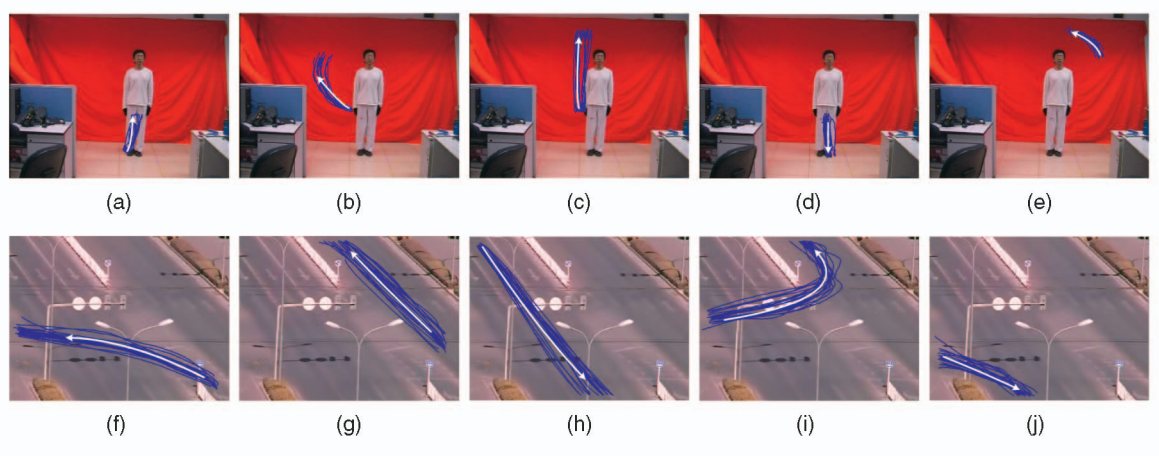


Fig. 4. Some basic motion patterns in our experiments. (a)-(e) Some basic patterns in the gymnastic exercises. (f)-(j) Some basic patterns in the crossroad scene. The white arrows denote the moving directions. (a) lf_1_2. (b) lh_1_2. (c) lh_1_3. (d) rf_2_1. (e) rh_2_3. (f) v_1_6. (g) v_2_5. (h) v_5_2. (i) v_7_5. (j) v_8_1.

Then, each primitive is represented as a 5-tuple:

$$\{e\_type, agent\_id, t\_range, s\_range, likelihood\}, \quad (1)$$

where e_type denotes event type, $agent\_id$ indicates the agent identity of the current event, t_range is the time interval $(start\_point, end\_point)$ of the event, s_range is represented as $\{(x_{min}, x_{max}), (y_{min}, y_{max}), (o_{min}, o_{max})\}$, which records the primitive's spread range in the x-axis and y-axis as well as the spread range in motion orientation, and *likelihood* is the likelihood probability of the trajectory belonging to the given primitive model. Finally, a primitive stream can be obtained in terms of the end_point of each primitive ascendingly.

Note that primitive modeling is not the focus of this paper. The primitive modeling process is independent of the following rule induction and parsing algorithms. The current approach to detecting primitives can thereby be replaced by other ones according to different application domains, as long as an interval-based primitive stream can be obtained.

4 MDL-Based Event Rule Induction

4.1 Definition of Event Rule

In this study, based on the traditional SCFG production, we use Allen's temporal logic [43] to describe the relation between two events. An event rule is defined as follows:

$$E \rightarrow \ s \ \{R\} \quad [P], \quad (2)$$

where E is the leftmost nonterminal, s is a symbol string where each symbol denotes a subevent of E, R is the temporal matrix where element r_{ij} denotes the temporal relation between the ith subevent and the jth subevent in s. P is the conditional probability of the production being chosen, given the event E. An example of event rule is shown in Fig. 5.

4.2 Basic Rule Induction Algorithm

Referring to [36], an MDL-based rule induction algorithm is adopted to learn event rules. As shown in Fig. 6, a number of rule candidates are generated by two operators $construct$ and $merge$. For each event pair (A, B) in current training data and rule set, the $construct$ operator generates a set of candidates $P \rightarrow A \ \ B \ \{R_j\}$, where R_j belongs to all possible temporal relations, whereas $merge$ operator creates a rule candidate $P \rightarrow A \mid B$, which means A and B can be replaced by P.

Then, in the MDL-based evaluation, a given rule candidate first updates the old rule set. For a *construct* candidate, the old rule set is updated by adding this candidate. For a *merge* candidate, the symbols on the right hand will be replaced by the symbol on the left hand. Then, we check whether identical rules exist in the current set (they should share the same left hand and right hand as well as the same temporal relationships). If true, all identical rules are merged into one rule, where the probability is also added together. Then, for each rule in the updated set, we replace the instances (right hand) in the training event stream with the left-hand symbol.

For each candidate, we compute the description lengths (DL) before and after the updating operation, respectively (see below), where the difference between the old DL and

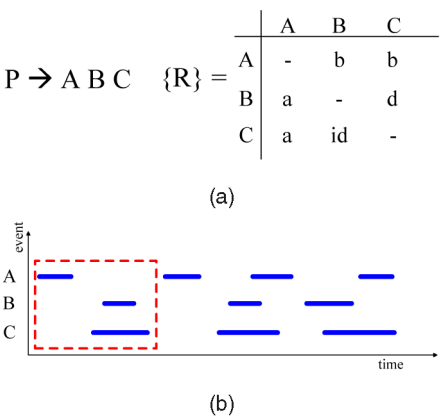


Fig. 5. Illustration of event rule. (a) An event rule where $P \rightarrow ABC$ is the traditional grammar production, R is the temporal matrix (abbreviations: a = after, b = before, d = during, id = inverse during). (b) The part enclosed by the dashed red lines illustrates an instance of this production in event stream.

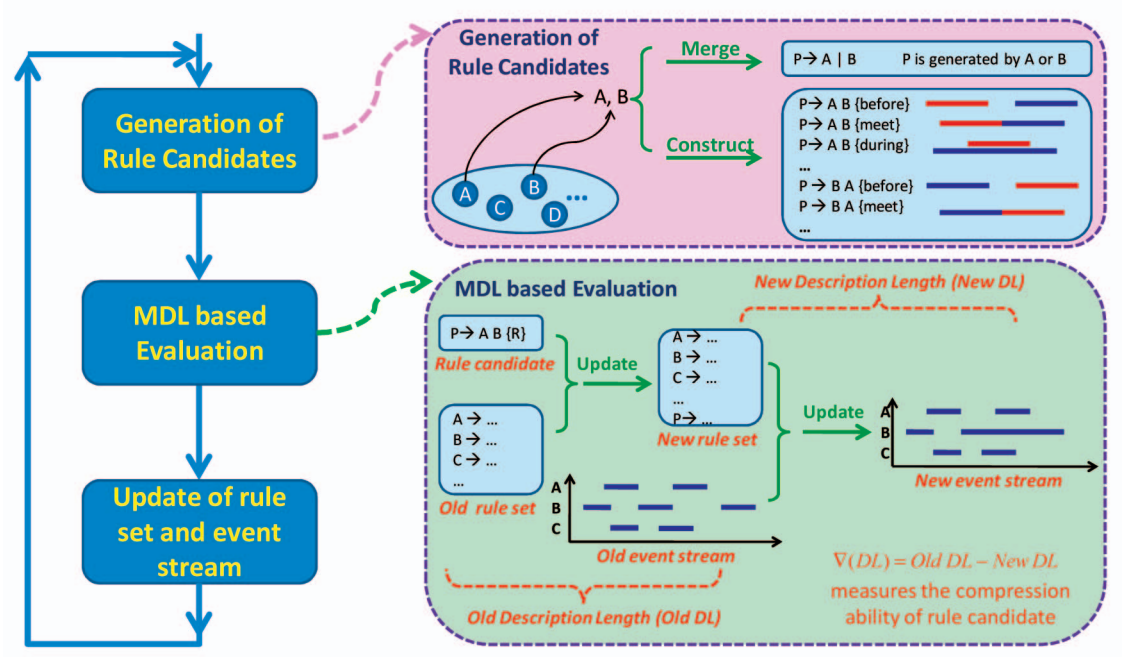


Fig. 6. The basic rule induction algorithm. First, a number of rule candidates are generated by the $construct$ and $merge$ operator. Then, the MDL principle is used to find the rule candidate which maximally decreases the DL of the event stream and rule set. Subsequently, the event stream is updated by the selected rule candidate. The process will be repeated until the decrease of DL is smaller than a given threshold or the maximum iteration number is reached.

the new DL evaluates the compression ability of the candidate. After the evaluations over all candidates, we chose the candidate which maximally decreases the DL as the new rule in current iteration. Further, we update the rule set and the training event stream with the new rule. If the decrease of DL is smaller than a given threshold or the maximum iteration number is reached, the induction process will be halted.

At the computation of DL, in terms of [38], the DL is computed as $L(G,S) = L(S \mid G) + L(G)$, where G is the learned grammar, S is the input event stream. $L(S \mid G)$ is the number of bits needed to encode the event stream with the help of the grammar, $L(G)$ stands for that to encode the grammar. Similarly to the definition of event rule, the event stream can also be described as a symbol string and the corresponding temporal matrix. Therefore, $L(S \mid G)$ is computed as follows.

4.2.1 Encoding the Symbol String

Suppose the symbol string of current event stream is $e_1e_2 \ldots e_n$ and the set of unique symbols is $\{E_1, E_2, \ldots, E_c\}$, where $e_i = E_k$ results from replacing the substream sub_s in the original event stream with the current rule set. To encode the symbol string, the number of bits S_{bits} is computed, according to [39]:

$$\begin{aligned} S_{bits} &= -\sum_{i=1}^{n} logP(sub\_s, e_i) \\ &= -\sum_{i=1}^{n} log(P(sub\_s|e_i) * P(e_i)) \\ &= -\sum_{i=1}^{n} log(lik\_e_i * P(e_i)), \end{aligned} \tag{3}$$

where $P(e_i) = P(E_k)$ is the normalized *support* value of E_k in the event stream, which ensures $\sum_{k=1}^{c} P(E_k) = 1$; lik_e_i is the likelihood probability of e_i. Here, the likelihood of event e is computed as soon as it is generated by replacing the substream $e'_1e'_2 \ldots e'_m$ with the event rule $E \rightarrow E'_1E'_2 \ldots E'_m$.

$$lik\_e = \left(\prod_{j=1}^{m} lik\_e'_j\right) * P(E \rightarrow E'_1E'_2 \ldots E'_m), \tag{4}$$

where $P(E \rightarrow E'_1E'_2 \ldots E'_m)$ is the probability of the production and e'_j is the instance of E'_j.

As in the above description, the likelihoods of subevents are embedded into the new events. And the uncertainties in primitive detection are also considered in the encoding process.

4.2.2 Encoding the Temporal Matrix

To encode the temporal matrix, the number of bits $Rbits$ is computed with an enumerative encoding strategy.

Since the temporal matrix is an anti-symmetry-like matrix, only encoding the upper triangular matrix is enough. There are only eight possible temporal relations $\{before, meet, overlap, start, during, finish, equal, inverse finish\}$ in the upper triangular matrix. Moreover, it is explicit that most of the values in the ith row of the upper triangular matrix are $before$, when $i << N$, where N is the length of the temporal matrix.

We assume that the numbers of seven relations (except for $before$) existing in the ith row of the temporal matrix are v_j, $j \in \{1, 2, \ldots, 7\}$. To encode all possible placements, a feasible scheme is that each placement has equal probability of occurrence. So, we need

$$r_i = \log \prod_{j=1}^{7} \binom{v_j}{N - i - \sum_{k=0}^{j-1} v_k} \tag{5}$$

bits to encode the positions of eight relations in the ith row, where $v_0 = 0$.

Then, encoding the value of v_j and the value of i requires $(7 \log u_i + \log N)$ bits, where $u_i = \max_j\{v_j\}$, N is the size of event stream. Thus, the total number of bits to encode the temporal matrix is

$$R_{bits} = \sum_{i=1}^{N} [r_i + 7 \log u_i + \log N]. \tag{6}$$

Finally, given the current grammar G, $L(S|G) = S_{bits} + R_{bits}$ bits are needed to encode the whole event stream.

A similar scheme is adopted to encode each event rule. $L(G)$ is the sum of the bits required for each event rule.

4.3 Multilevel Rule Induction Strategy

With the exhausted strategy of candidates generation in the above algorithm, too many candidates will be generated, which increases the evaluation costs. Furthermore, some rule candidates that can indeed compress the primitive stream, but without any semantic meanings, may disturb the subsequent induction process.

Thus, we use a heuristic cue to filter out the redundant candidates and obtain more meaningful rules. That is, people would like to combine one object together with its most similar objects to form a new concept. For instance, due to the close evolutionary relationship, *monkey*, *orangutan*, and *human* can be summarized as *primate* in biology. In our work, the similarity between two visual events A and B is measured by their spatiotemporal distance, which is defined as follows:

$$Dist(A, B) = \alpha * Dist_s(A, B) + (1 - \alpha) * Dist_t(A, B), \tag{7}$$

that is, the sum of the weighted spatial distance $Dist_s(A, B)$ and temporal distance $Dist_t(A, B)$. The value of α is set empirically. $Dist_s(A, B)$ is defined as follows:

$$Dist_s(A, B) = 2 - \frac{vol(\widehat{cube}_A) + vol(\widehat{cube}_B)}{vol(union(\widehat{cube}_A, \widehat{cube}_B))}, \tag{8}$$

where $\widehat{cube}_A$ denotes the average s_range (shown in Section 3) of all instances of event A, which is represented as a three-dimensional cube with x, y-axis in image coordinate and the orientation axis. $vol(\widehat{cube}_A)$ is the volume of $\widehat{cube}_A$. $union(\widehat{cube}_A, \widehat{cube}_B)$ returns the smallest cube that contains the $\widehat{cube}_A$ and $\widehat{cube}_B$.

$Dist_t(A, B)$ is computed as the average of temporal distance between two instances:

$$Dist_t(A, B) = \frac{1}{n_A + n_B} \left\{ \sum_{i=1}^{n_A} \min_{j=1,\ldots,n_B} \{dist_t(a_i, b_j)\} + \sum_{i=1}^{n_B} \min_{j=1,\ldots,n_A} \{dist_t(b_i, a_j)\} \right\}, \tag{9}$$

where

$$dist_t(a_i, b_j) = 2 - \frac{len(t_{a_i}) + len(t_{b_j})}{len(union(t_{a_i}, t_{b_j}))}, \tag{10}$$

where n_A is the total number of A's instances in the current event stream, $t_{a_i} = (start\_point_{a_i}, end\_point_{a_i})$, that denotes the temporal interval of the instance a_i, $len(t_{a_i})$ is the length of t_{a_i}, $union(t_{a_i}, t_{b_j})$ returns the smallest temporal interval that contains t_{a_i} and t_{b_i}.

With the spatial-temporal distance, a multilevel rule induction strategy is proposed, as shown in Algorithm 1. At first, the events (event classes) in the current event stream Cur_Str are clustered into several event sets E_Set by agglomerative hierarchical clustering [48]. Based on the obtained dendrogram, the event clusters can be easily determined by hand. Certainly, the number of clusters can be estimated by some clustering validity indices, e.g., the $Dunn$ index [49]. Then, the original induction algorithm $RuleInduction()$ is performed in each event set, where only the events in the same cluster can be constructed or merged. The new learned rules New_Rule are added into the rule set $Rule\_Set$. Subsequently, for each new rule, all instances in the current event stream are replaced by the leftmost nonterminal of the rule. The process is called $Compress()$. The *clustering-induction* process will be repeated until the event set cannot be further separated (all events are very close to each other). Then, the rule induction will be conducted in the final set for the last time.

Algorithm 1. Multilevel rule induction strategy

```
Rule_set = ∅ ; Cur_Str = Init_Str;
E_Set = EventClustering(Cur_Str);
while num(E_Set) > 1 do
  for i = 1 : num(E_Set) do
    New_Rule = RuleInduction(Cur_Str, E_Set(i));
    Rule_Set = Update(Rule_set, New_Rule);
  end for
  Cur_Str = Compress(Cur_Str, Rule_Set);
  E_Set = EventClustering(Cur_Str);
end while
New_Rule = RuleInduction(Cur_Str, E_Set(1));
Rule_Set = Update(Rule_set, New_Rule);
return Rule_Set;
```

An example of the proposed multilevel rule induction process for gymnastic exercise $E1$ is illustrated in Fig. 7. The left part shows the dendrogram of hierarchical clustering at different levels. Based on the dendrogram, we choose the appropriate partition by examining the semantic meanings of each clusters manually. In the figure, the events that are linked to each other below the dashed line will belong to the same cluster. The learned rules are listed on the right side of Fig. 7. After the training stream is replaced by the learned rules, the induction process will be carried at the next level. In this example, the rule induction process is over through three levels. In the learned rules, the hidden structure of the gymnastic exercise can be captured effectively. For example, $P3$ denotes "*raising both hands to the shoulder level simultaneously*" (referring to the primitives in Fig. 4). Finally, the exercise $E1$ is indicated as the nonterminal $P36$.

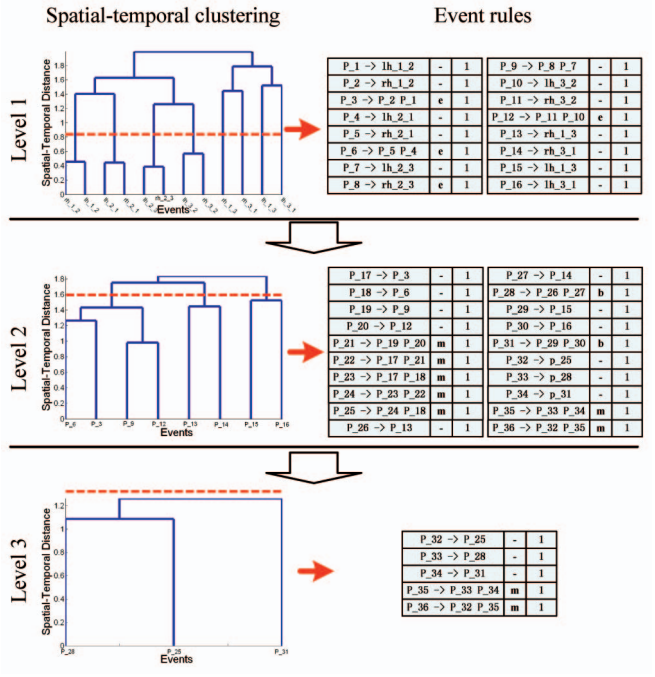


Fig. 7. The multilevel induction process for gymnastic exercise $E1$. Abbreviations of temporal relations: b = before, m = meet, e = equal [43]. $E1$ is denoted as the nonterminal $P36$.

Here, a frequent pattern mining algorithm [37] is also utilized to filter out the infrequent rule candidates: A construct candidate can be generated only if its *support* exceeds a threshold $supp_{min}$, where the *support* denotes how often a candidate occurs in the event stream.

The *support* of a rule candidate is also used to compute the production's probability. Suppose there are n productions that share the same leftmost nonterminal E, the probability of ith production is obtained as follows:

$$P(E \rightarrow s(i)) = \frac{support_i}{\sum_{j=1}^{n} support_j}. \quad (11)$$

For other nonterminals extended by a unique production, their probabilities are set to one.

5 Multithread Parsing

Provided the learned event rules, the task of parsing is to find out the most possible derivation (parse tree) T to interpret a given primitive stream S, further recognizing the interesting events. In this work, for each interesting complex event A, we learned a set of rules G_A. In terms of Maximum Likelihood criterion, the event recognition can be described as follows:

$$<A_d, T_d> = \arg \max_{<A,T>} P(S, T|G_A), \quad (12)$$

where A_d is the final decision on the type of complex event, T_d is the recovered parse tree, and $P(S, T|G_A)$ is computed as the product of the probabilities of the rules used in the parse tree.

5.1 Parsing Algorithm

Referring to [44], we propose the multithread parsing algorithm. The parsing algorithm executes three operations,

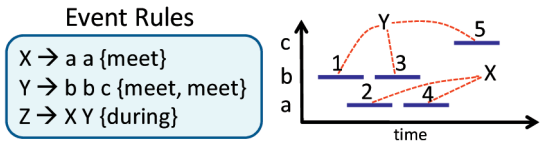


Fig. 8. Illustration of the *ID set* constraint. The left box shows a set of event rules, where the relation between subevent X and Y is $during$. Due to the parallel relation, the *ID set*s of subevents may be intersected. In this case, the *ID set* of Y is $(1, 3, 5)$, while the *ID set* of X is $(2, 4)$. The continuous *ID set* constraint should thereby be relaxed to an unordered one.

i.e., *scanning*, *completion*, and *prediction*, iteratively. In the end, the algorithm examines whether the root symbol is completed or not. If completed, the corresponding parse tree can be recovered by backtracking.

Here, the state in the parsing algorithm is represented as follows:

$$I : X \rightarrow \lambda \cdot Y\mu \quad [v], \quad (13)$$

where I is the *ID set* that indicates the state's constituents in the input primitive stream, the *dot* marker is the current parsing position, which expresses that the subevents λ have been observed and the next symbol to be scanned is Y, μ is the unobserved string, and v is the Viterbi probability which corresponds to the maximum possible derivation of the state. Additionally, the temporal property of the state is also saved, e.g., *start point* and *end point*.

The state's *ID set* in common parsing algorithms, e.g., Earley-Stolcke parser [46], can only involve a set of consecutive primitives. Under the strong constraint, a primitive string corresponding to one complex event cannot be inserted by the primitives belonging to other events. However, due to the possible parallel relations in our case, the *ID set*s of two subevents maybe overlapped. As shown in Fig. 8, the input primitives are arranged as $\{b\ a\ b\ a\ c\}$ by their end-times, where the numbers indicate the corresponding IDs. The left box of Fig. 8 shows the event rules. To recognize the event Z, the subevent Y is comprised of the first, third, and fifth primitives, the ID set should be $(1, 3, 5)$, whereas the ID set of another subevent X is $(2, 4)$. The *ID set* constraint should thereby be relaxed so that multiple parsing states overlapping in time can exist in the current state set. In [44], Johnston has relaxed the constraint in an unification chart parsing algorithm for multimodal human-computer interaction. Here, we adopt it in the form of Earley-Stolcke algorithm [46] and further supply an error recovery strategy.

Given the current $StateSet(i)$ and primitive, the following three steps will be performed.

5.1.1 Scanning

In our work, for each primitive, say d, we add a pre-non-terminal rule $D \rightarrow d$ where the probability is one so that the role of $Scanning$ is to accept the current primitive with the predicted state of the pre-non-terminal rule. And the likelihood of the detected primitive will be multiplied by the Viterbi probability of the predicted state. The process can be described as follows:

$$\begin{cases} I_d : D \rightarrow \cdot d \ \ [1] \\ j : d \ \ [lik_d] \end{cases} \Rightarrow I'_d : D \rightarrow d \cdot \ \ [lik_d], \quad (14)$$

where I_d is null set, $I'_d = \{j\}$, j is the ID of the primitive in the input stream, and lik_d is the likelihood of primitive d.

5.1.2 Completion

For a completed state in $StateSet(i)$, suppose $I'' : Y \rightarrow \omega \cdot [v'']$ that denotes event Y has been recognized; the state S_j in the last state set $StateSet(i-1)$ will be examined with the following conditions:

- Y is one of the unobserved subevents of S_j.
- $I'' \cap I_{S_j} = \phi$, the *ID set* of the completed state I'' does not have the same primitives as that of S_j.
- The temporal relation between Y and the observed subevents of S_j is consistent with the rule definition. The temporal relation is computed by a similar fuzzy method in [47].

For the state satisfying the above conditions, we examine whether Y is the first unobserved subevent (the symbol following the *dot*) or not. If it is not, the unobserved subevents that are prior to Y are treated as deletion error candidates, which will be handled in Section 5.2, else S_j can be assumed as $I : X \rightarrow \lambda \cdot Y\mu[v]$, a new state is generated:

$$\begin{cases} I : X \rightarrow \lambda \cdot Y\mu \ \ [v] \\ I'' : Y \rightarrow \omega \cdot \ \ [v''] \end{cases} \Rightarrow I' : X \rightarrow \lambda Y \cdot \mu \ \ [v'], \tag{15}$$

where $I' = I \bigcup I''$ and $v' = vv''$.

In the current state set, if another state identical to the new state has existed, the Viterbi probability v_c that the identical state will be modified is $v_c = \max\{v_c, v'\}$, else the new state will be added into the state set.

5.1.3 Prediction

As the next symbol may belong to another parsing thread, all of the uncompleted states in the last state set will be put into the current state set in *prediction*. Note that all of the nonterminals should be predicted in initialization.

5.2 Error Recovery Strategy

Commonly, there are three types of errors in practice: insertion, deletion, and substitution errors. Insertion errors mean the spurious detection of primitives that do not actually happen. Deletion errors are the missing detections of primitives that actually occur. Substitution errors denote the misclassification between primitives.

In [21], insertion errors are accepted by the extended $skip$ productions; nevertheless the deletion errors cannot be handled by such $skip$ productions. In [22], three hypotheses, corresponding to the three kinds of errors (insertion, deletion, and substitution), are generated when the parsing fails to accept the current primitive. However, an error may not lead to immediate failure, but to failure in the next parsing iteration.

Here, referring to the idea in [45], a number of error hypotheses will be generated along with the parsing process. Finally, the Viterbi-like backtracking will determine the maximum possible error occurrences. Since a substitution error can be seen as a pair of one insertion error and one deletion error, only insertion and deletion errors are considered in the following.

5.2.1 Insertion Error

Due to the relaxed *ID set* constraint in which the *identifiers* may be disconnected, the insertion errors are considered naturally. At the end of parsing, for each completed root state $I_f : 0 \rightarrow S \cdot [v_f]$, the primitives that are not contained in I_f will be treated as insertion errors of this derivation. The penalties will be added as follows:

$$v = v_f \prod_{i \in I'_f} \rho_i, \tag{16}$$

where ρ_i is the penalty of the ith insertion error that is a small value; I'_f is the set including all insertion errors.

5.2.2 Deletion Error

As presented in Section 5.1, deletion error candidates maybe generated in a *completion* operation. Supposing a state $I'' : X \rightarrow \lambda \cdot Y_1 Y_2 \ldots Y_n Y \mu[v'']$ and a completed event Y, Algorithm 2 will be performed to transform the state into a new one $I_e : X \rightarrow \lambda Y_1 Y_2 \ldots Y_n \cdot Y\mu[v_e]$, where the events $Y_1 Y_2 \ldots Y_n$ are assumed to be completed by deletion error hypotheses. Here, $An\_s = I'' : X \rightarrow \lambda \cdot Y_1 Y_2 \ldots Y_n Y \mu[v'']$, $I' = I \cup I''$, where I is the *ID set* of the completed state of event Y, $e\_position$ is the position where Y locates at in An_s, s_set is the last state set.

Algorithm 2. Recovery(An_s,I',$e\_postion$,s_set)

```
if An_s.predict is pre-non-terminal then
    z = Error_Hypothesize(An_s.predict);
    re_s = scanning(An_s, z);
else
    re_s = Max_Ex(An_s.predict, I', s_set);
    while re_s.dot < size(re_s.rule) do
        Recovery(re_s, I' ∪ I_re_s, re_s.dot + 1, s_set);
        re_s = Max_Ex(An_s.predict, I', s_set);
    end while
end if
new_s = completion(An_s, re_s);
if new_s.dot < e_position then
    Recovery(new_s, I' ∪ I_new_s, e_position, s_set);
else
    Return;
end if
```

Concretely, $Y_i = An\_s.predict$ is the symbol just behind the *dot* of An_s. If Y_i can only be completed by pre-nonterminal rule $Y_i \rightarrow z$ (z is a terminal), z will be recovered as an error hypothesis. The likelihood of z is assigned to a small value as penalty. Then, a state re_s will be generated by *scanning*.

Else if Y_i is a nonterminal, Max_Ex is performed to find out the state $re\_s = Y_i \rightarrow \lambda' \cdot Z\mu'$ in s_set, where the state re_s has the maximum Viterbi probability to complete Y_i. Then, $Recovery$ and Max_Ex are performed repeatedly until re_s becomes a completed state.

Then, An_s is combined with re_s to form a state new_s with *completion*. Finally, we examine whether the dot position of new_s reaches $e\_position$; if true, the recovery of An_s is over, else recover the next subevent.

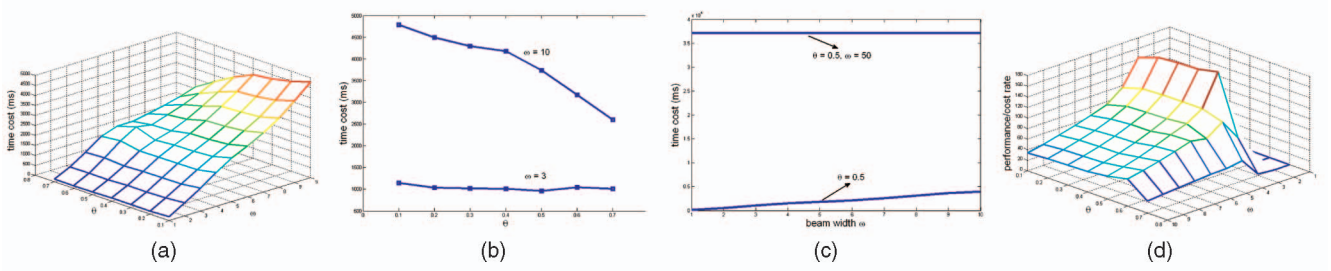


Fig. 9. Time costs of the parsing algorithm as different parameter settings. (a) Average time costs; (b) the effect of *maximum error constraint* to save time cost; (c) the effect of *beam-width constraint* to save time cost; (d) the performance/cost rate.

5.3 Additional Constraints

Due to the relaxed *ID set* constraint and the generated *error hypotheses*, there may be redundant states that are generated through the combinations of different primitives. Therefore, we propose two constraints to control the number of states in parsing process.

5.3.1 Beam-Width Constraint

Here, we define two states are *isomorphic*, if and only if they share the same rule, the same *dot* position, but different *ID set*. In terms of the *beam-width constraint*, we prefer the one with more primitives in *ID set* between two *isomorphic* states.

Assuming $\{S_1, S_2, \ldots, S_N\}$ is an isomorphic state set where each pair of the states in this set is isomorphic, we compute the rank score for each state as follows:

$$Likeli(S_i) = \nu(S_i) + \rho\left(\left|\bigcup_{i=1}^{N} I_{S_i}\right| - |I_{S_i}|\right), \qquad (17)$$

where $\nu(S_i)$ is the Viterbi probability of state S_i, ρ is the penalty of insertion error, presented in Section 5.2, $|\bigcup_{i=1}^{N} I_{S_i}|$ denotes the number of primitives in the union set of all *ID sets* of the isomorphic states, $|I_{S_i}|$ is the number of primitives in state S_i. Note that the hypothesized deletion errors are not counted in $|I_{S_i}|$.

Then, the *beam-width constraint* can be presented as: In an isomorphic state set, only the first ω states can be saved in terms of the rank score. The larger the ω is, the more states will be saved in the state set.

5.3.2 Maximum Errors Constraint

Here, we assume that deletion errors just take a small proportion in a state. Thus, the *maximum errors constraint* is proposed to prune the states with too many error hypotheses. An exponential distribution is used to model the number of deletion errors. It is written as $e^{-\theta n^{\frac{1}{2}}}$, where θ is a control parameter, n is the size of *ID set* of the state. For a given state with m deletion errors, if $\frac{m}{n} > e^{-\theta n^{\frac{1}{2}}}$, the state will be pruned.

6 Experimental Results

In this section, extensive experimental results in different scenes are reported to validate our approach.

6.1 Gymnastic Exercises

Three exercises called $E1$, $E2$, and $E3$ are chosen to validate our method. Twenty-nine sequences are collected. The numbers of sequences are 9, 10, and 10, respectively.

The goal of this experiment is to validate the effectiveness of the proposed approach to modeling and recovering the temporal relationships in the multiple moving agents. Thus, the gymnastic exercises are not considered as single-agent events, but multi-agent events where hands and feet are the moving agents. The motion trajectories of hands and feet need to be extracted as the original feature. We track hands and feet using some prior color and spatial information (more sophisticated tracking techniques may be useful, but this is not our focus here). After primitive modeling, 18 primitives are obtained. Some of them are shown in Fig. 4. Finally, each exercise includes around 23 primitives. To learn event rules, we take the first five sequences as training data for each exercise.

Based on the learned grammar, we first investigate the time costs of the parsing algorithm as different parameter settings. Twelve sequences (each exercise has four sequences) are chosen for the experiments. As the control parameters are set as different values, we perform the multithread parsing to classify the exercise in a given primitive stream.

The results of average time cost are presented in Fig. 9a, where ω and θ correspond to the *beam-width constraint* and *maximum errors constraint*, respectively. From this figure, we can see that the time cost is in direct proportion to the ω value. However, the effect of θ on time cost is not clear when ω is small. The inverse proportion between θ and time cost can be observed clearly until ω is larger than 7. The main reason is that a state with many deletion errors not only has a very little Viterbi probability due to the penalties for deletion errors, but also a low rank score in its isomorphic set. Therefore, when ω is small, a state with deletion errors may also be pruned due to the *beam-width constraint*, even if it satisfies the *maximum errors constraint* as θ is small. Fig. 9b illustrates this case clearly. The top curve ($\omega = 10$) is approximate to an inverse proportion line between θ value and time cost, while the bottom curve ($\omega = 3$) is approximate to a horizontal line.

Fig. 9c illustrates the effect of *beam-width constraint* on saving the time cost. The bottom curve shows the time costs with $\theta = 0.5$, as different ω settings. The top line presents the time cost as $\omega = 50, \theta = 0.5$, which is the approximation for the case without using *beam-width constraint*. We can see that *beam-width constraint* can reduce the time cost effectively.

TABLE 1
CCRs on Gymnastic Exercises Recognition

Event	MTP			HMM	CHMM
	θ = 0.7	θ = 0.5	θ = 0.2		
$E1$	9	9	9	9	9
$E2$	7	10	10	10	10
$E3$	9	10	10	8	9
Total	26	29	29	27	28
CCR	89.7%	100%	100%	93.1%	96.6%

Fig. 9d shows the performance/cost rates with different parameter configurations. The performance/cost rate is computed as $\frac{\exp(CCR)}{t}$, where CCR is the correct classification rate, t is the average time cost. Because we prefer a good recognition result even with a little high time cost, the CCR is enlarged by an exponential function. As shown in the figure, our method achieves the best performance/cost rate, as $\omega = 3, \theta = 0.5$.

Then, we validate the performance to event recognition. Here, HMM and Coupled Hidden Markov Model (CHMM) are chosen for comparison because they can be trained with little human intervention, while other DBN-based methods usually require manual construction of model topology in terms of different events. For HMM, the input is an eight-dimensional vector sequence formed by the four trajectories of hands and feet. For CHMM, the four trajectories are divided into two parts for the input of each chain (one chain includes the trajectories of left hand and left foot, the other chain is comprised of right hand and right foot). We also take the first five sequences for training parameters and all sequences for test, which is the same as the proposed grammar approach. The results are presented in Table 1, as the control parameter ω in MTP method is 3. As shown in Table 1, as θ is less than 0.5, our system can recognize all of the sequences correctly, whereas HMM misclassifies three sequences and CHMM misclassifies one.

To further validate the robustness of the MTP algorithm, three kinds of synthetic errors are randomly added into the testing trajectories as follows:

- A deletion error is added by replacing a motion trajectory segment that corresponds to a primitive with a still trajectory that does not correspond to any primitive.
- An insertion error is added by replacing a still trajectory segment with a motion trajectory segment that corresponds to a random primitive.
- A substitution error is added by replacing a motion trajectory segment with another segment that corresponds to a different primitive.

After various amounts of large timescale errors are added, we do event classification with the original trained models or rules. The performance is shown in Table 2. As six additional errors are added (one substitution error is equivalent to a pair of one insertion error and one deletion error, so there are over 25 percent errors in the primitive stream) as $\omega = 3, \theta = 0.2$ and $\omega = 5, \theta = 0.2$, the MTP still acquires a satisfying result of 96.6 percent due to the discriminative rules and the effective error recovery strategy, while the performances of HMM and CHMM obviously decrease as the number of errors increases. And we also can see that as $\omega = 3, \theta = 0.5$, the performance of MTP will decrease rapidly when the number of errors exceeds 2. That is because the maximum tolerance of the deletion errors is $23 * \exp(-1 * 0.5 * \sqrt{23}) \approx 2$ (23 is the average length of a primitive stream), according to the *maximum errors constraint*.

From the above comparisons, the effectiveness and robustness of our approach have been validated. Moreover, by the parsing process, a parsing tree can be obtained to express the hierarchical structure in the complex event explicitly. An example of the whole parsing process is shown in Fig. 10, where Fig. 10a shows a sequence of exercise $E3$ and the detected results of primitives. Fig. 10b shows the parse tree recovered by the multithread parsing. According to the parse tree, each input primitive has two possible afflictions. One is that the primitive is accepted by the parse tree. The other is that the primitive is identified as an insertion error. Thereafter, the metric *overall correct rate (OCR)* is adopted to measure the parsing accuracy, which can be defined as $\frac{NA+NI}{NP}$, where NA is the number of correct acceptance of primitives in the parsing tree, NI is the number of correct detection of insertion errors, and NP is the total number of primitives.

Table 3 presents the parsing accuracies (OCR) with original data as well as various additional errors. As shown in the table, most primitives are accepted by the parse tree, while the parsing accuracies decrease with the increase of the added errors. As ω becomes bigger and/or θ becomes smaller, more parsing states can be saved so that it is easier to find a globe-optimal parsing tree and a higher accuracy can be achieved.

TABLE 2
CCRs on Event Recognition with Additional Errors

Number of Errors	MTP			HMM	CHMM
	θ=0.5, ω=3	θ=0.2, ω=3	θ=0.2, ω=5		
1	93.1%	100%	100%	86.2%	89.7%
2	82.8%	100%	100%	82.8%	89.7%
3	72.4%	100%	100%	75.9%	86.2%
4	62.1%	100%	100%	69%	79.3%
5	55.2%	100%	100%	55.2%	75.9%
6	41.4%	96.6%	96.6%	51.7%	75.9%

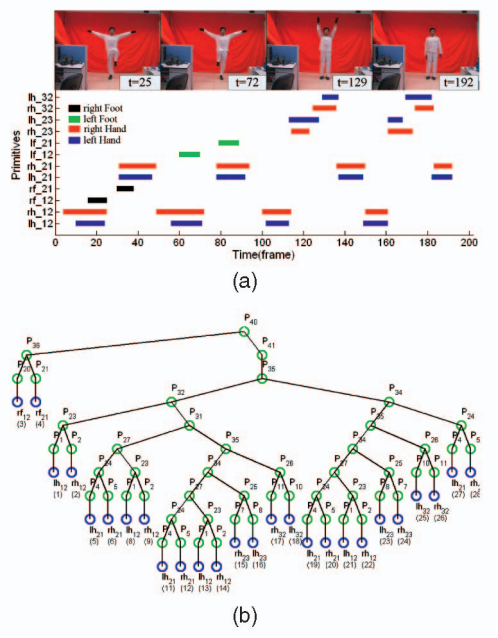


Fig. 10. An example of the recognition process of the exercise $E3$. (a) The detected primitives along with the time axis. (b) The corresponding parse tree.

6.2 Traffic Events in Crossroad

A traffic light event (one traffic cycle) consists of three subevents occurring sequentially, i.e., "*passing straight on the main road*," "*turning left from the main road to the side road*," and "*turning left from the side road to the main road*," as shown in Fig. 12. Further, "*passing straight in the main road*" can be divided into two parallel ongoing subevents, "*passing straight on the left side of the main road*" and "*passing straight on the right side of the main road*." Eventually, each event consists of a number of primitives corresponding to the passing of single vehicles.

We use a tracking system developed by Yang et al. [42] to acquire vehicle trajectories continuously in a rush hour. The trajectory stream lasts for about 90 minutes, which includes 45 traffic signal cycles. Due to the serious occlusions, many broken trajectories exist in the original trajectory set. For this problem, a post filtering is performed to remove the trajectories that start/end at the central zone of the scene.

After primitive modeling, 17 primitives are obtained. Some of them are illustrated in Fig. 4. Then, the spatial anomaly trajectories that do not belong to any primitive models are filtered out. Finally, a primitive stream with the size of 1,852 is obtained for the following rule induction and event recognition.

TABLE 3
Parsing Accuracy in Recognizing Gymnastic Exercise

Errors number	θ=0.5, ω=3	θ=0.2, ω=3	θ=0.2, ω=5
#e=0	87.0%	87.6%	89.7%
#e=1	83.4%	86.1%	88.8%
#e=2	79.4%	85.9%	88.3%
#e=3	76.3%	83.3%	87.2%
#e=4	73.4%	81.3%	85.1%
#e=5	73.4%	81.2%	83.2%
#e=6	73.7%	76.8%	80.7%

Here, $\#e=0$ means the ordinal data, $\#e=1$ is the data with one synthetic error, and so on.

ID	Rules	Temporal relation	Semantic Description
1	P57 → P53 P58 [1.0]	meet	Pass through the crossroad alternately following the traffic signal rule. The representative event is "P57".
2	P53 → P46 P47 [1.0]	meet	
3	P58 → P51 [0.719]	-	
4	P58 → P52 [0.281]	-	
5	P52 → P49 P50 [1.0]	during	
6	P51 →P49 P50 [1.0]	equal	
7	P46 →P46 P46 [0.442]	meet	Turn left from the main road to the sid road. The representative event is "P46".
8	P46 →v_1_6 [0.558]	-	
9	P47 →P47 P47 [0.094]	meet	Turn left from the side road to the main road. The representative event is "P47".
10	P47 → v_7_5 [0.680]	-	
11	P47 →v_7_4 [0.226]	-	
12	P49 →P49 P26 [0.041]	meet	Go straight over the crossroad at the left way of the main road. The representative event is "P49".
13	P49 →P49 P49 [0.093]	meet	
14	P49 →P49 P49 [0.029]	equal	
15	P49 →P26 P49 [0.10]	meet	
16	P49 → v_6_2 [0.119]	-	
17	P26 → v_6_1 [1.0]	-	
18	P49 → v_5_3 [0.166]	-	
19	P49 → v_5_2 [0.260]	-	
20	P49 → v_4_3 [0.191]	-	
21	P50 → P18 [0.242]	-	Go straight over the crossroad at the right way of the main road. The representative event is "P50".
22	P18 →P17 P50 [0.571]	meet	
23	P18 →P50 P50 [0.423]	meet	
24	P50 → v_3_5 [0.181]	-	
25	P50 → v_3_4 [0.281]	-	
26	P50 → v_2_5 [0.296]	-	
27	P17 → v_2_4 [0.735]	-	
28	P17 → v_1_5 [0.265]	-	

Fig. 11. The learned rules in traffic light events. v_i_j is the primitive where $i \in \{1,2,\ldots 8\}, j \in \{1,2,\ldots,6\}$, which represents the basic motion pattern of "*moving from the ith entry to the jth exit*." For the meaning of entries and exits, refer to Fig. 3.

However, we find that the whole traffic light rules in the scene still cannot be acquired correctly with the original training data due to the distortion of some unrelated events. For instance, the event "*turning right from the side road to the main road*," referred as v_8_1 in Fig. 4j, is totally unrelated to the traffic light rules. That is because, in this traffic crossroad, v_8_1 can occur at any time, whatever the traffic light is. The distortion event may be combined with another useful event to form a meaningless event which will disturb the subsequent induction process. So, four kinds of distortion events, e.g., v_8_1, are manually picked out from the original 17 trajectory clusters. After removing such unrelated trajectories, the whole traffic light rule in the scene (the top six of these rules in Fig. 11) can be obtained correctly in the final rule set where the whole traffic light event is denoted as $P57$. The other rules and the corresponding semantic meanings are also presented in Fig. 11.

Twenty-five traffic cycles are used for rule learning. Fig. 11 presents the learned rules. We find four main traffic events in the crossroad are learned correctly. They are "*turning left from the main road to the side road*," "*turning left from the side road to the main road*," "*passing straight on the left side of the main road*," and "*passing straight on the right side of the main road*," which are denoted as $P46$, $P47$, $P49$, and $P50$, respectively. As shown in the figure, a number of recursive rules, such as rule #7 and rule #9, are also obtained to describe the continuous vehicle passings in green light duration.

Then, 20 traffic cycles are used to test the parsing algorithm. We test the parsing algorithm with different parameter settings. As $\omega = 1, \theta = 0.5$, two of 20 traffic cycles cannot be recovered. As $\omega > 1$, all cycles can be recognized correctly. Furthermore, we examine the four main subevents ($P46$, $P47$, $P49$, and $P50$ in Fig. 11) and the

TABLE 4
Parsing Accuracies on Recognizing Traffic Events

Events	θ=0.5, ω=1	θ=0.5, ω=2	θ=0.5, ω=3	θ=0.5, ω=4	θ=0.5, ω=5
P46	85%	92.2%	92.2%	92.2%	100%
P47	77%	93.3%	93.3%	93.3%	100%
P49	80.5%	88.2%	91.3%	91.3%	100%
P50	73%	90.3%	94%	94%	97.8%
P57	76%	90.3%	93.3%	93.3%	98.6%

whole traffic light events ($P57$, in Fig. 11). The number of primitives belonging to each subevents is counted. Then, the parsing accuracies are computed in terms of the definition in Section 6.1.

The parsing accuracies are presented in Table 4. We can find that the parsing accuracies will be higher, as ω value is larger. We also find that in some traffic cycles, one of the main subevents does not occur at all. For instance, in the fourth cycle, no vehicles turn left from the main road to the side road ($P46$ is absent). Such absences of subevents will be considered as deletion errors in our parsing process. In summary, five traffic cycles with the absence of main subevent (4th, 6th, 7th, 13th, and 20th cycles) are all recognized successfully when $\omega > 1$.

A parsing example in the traffic scene is shown in Fig. 12. Fig. 12a presents the temporal relations of the four main subevents and Fig. 12b shows the parse tree.

We also find that some vehicles conflict with the traffic light rule in some traffic cycles, i.e., "*passing a red traffic light*." Our parsing algorithm has correctly decided them as insertion errors. It suggests the MTP algorithm is able to detect abnormal behaviors that conflict with other events in temporal relations.

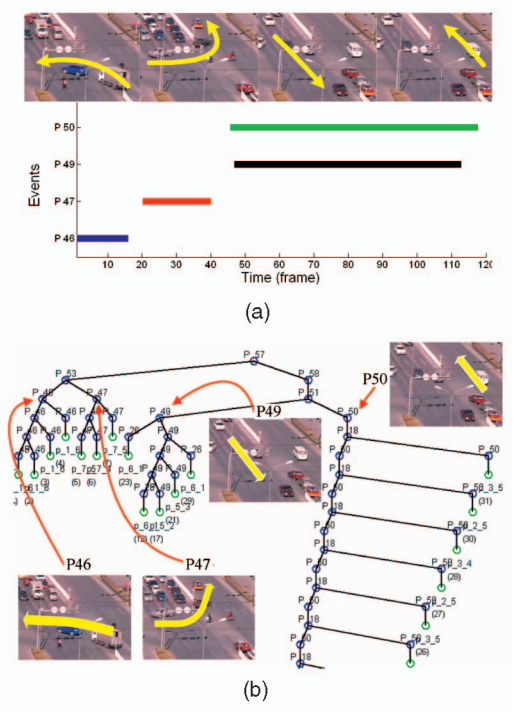


Fig. 12. An example of recognizing a traffic cycle. (a) The main subevents in a traffic cycle. (b) The corresponding parse tree of the traffic cycle.

6.3 Multi-Agent Interactions

Furthermore, we perform experiments on a new multi-agent interaction database that is a subset of the CASIA action database [50]. There are five kinds of two person interactions, including:

- $I1$: Two people pass through the scene in the same direction. One person follows the other.
- $I2$: Two people pass through the scene in the same direction. One of them first follows the other, then speeds up and goes beyond the other.
- $I3$: Two people pass through the scene. One first follows the other, then speeds up to reach the other, finally they keep walking together.
- $I4$: Two people enter the scene from opposite directions. They approach and stop to chat, then they depart along their original directions.
- $I5$: Two people enter the scene from opposite directions. They meet to chat, then one person changes his direction and walks with the other one.

Since the interactions may occur everywhere in the scene, we represent these interactions in relative local coordinates. For each moving object, we construct one local coordinate; where the original point is the moving object, the y-axis is aligned with the motion direction. The trajectory of the other moving object is projected onto this local coordinate. Fig. 13 presents an example on the interaction "*two people meet, then depart*."

Then, to transform the relative trajectories into a symbol stream, we cluster the possible positions into a set of interaction primitives (IPs) in local coordinate. An appropriate number of clusters is chosen, depending on the desired representative granularity. Here, nine clusters are obtained by k-means clustering. Fig. 14a shows the learned clusters, which describe different spatial relations between the reference and the other moving object.

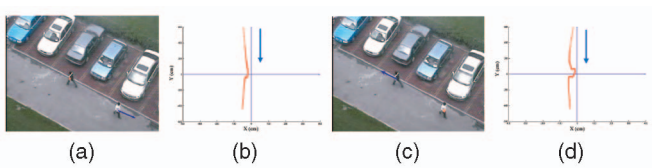


Fig. 13. The representation of the interaction "*two people meet, then depart*" in local coordinates. (a) Person 0, indicated by the blue arrow, serves as the reference; his motion direction is aligned with the y-axis. (b) The motion trajectory of person 1 in person 0's local coordinate, where the relative motion direction is denoted by the blue arrow. (c) Person 1 serves as the reference. (d) The motion trajectory of person 0 in person 1's local coordinate.

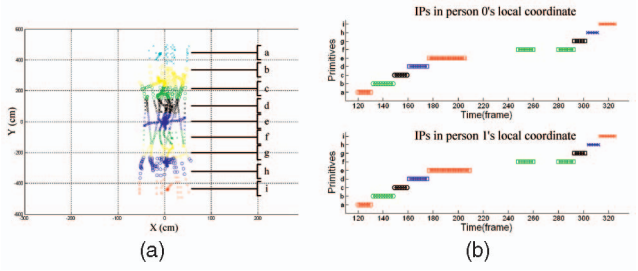


Fig. 14. The representation of multi-agent interaction at the symbol level. (a) The learned IPs describing the spatial relations between the moving object and the reference (the original point in the coordinate). (b) The IP stream for the interaction "*two people meet, then depart.*"

ID	Rules		Temporal relation	ID	Rules		Temporal relation
1	P37→P36 (2) P39 (2)	[1.0]	meet	14	P21 → P22 (0) P22 (1)	[1.0]	equal
2	P36→P32 (2) P39 (2)	[1.0]	meet	15	P18 → P18 (0) P18 (0)	[0.0001]	meet
3	P39→ P30 (2)	[0.49]	-	16	P10 → P6 (1) P10 (0)	[0.0002]	meet
4	P39→P27 (2)	[0.35]	-	17	P4 → a (2)	[0.49]	-
5	P39→P31 (2)	[0.16]	-	18	P4 → b (2)	[0.51]	-
6	P32 →P24 (2) P25 (2)	[1.0]	meet	19	P22 → h (2)	[0.5]	-
7	P30→ P28 (2) P29 (2)	[1.0]	meet	20	P22 → i (2)	[0.5]	-
8	P27→ P31 (0) P31 (1)	[1.0]	equal	21	P6 → c (2)	[1.0]	-
9	P24→ P3 (2) P3 (2)	[1.0]	meet	22	P31 → e (2)	[1.0]	-
10	P25 → P10 (1) P10 (0)	[1.0]	equal	23	P10 → d (2)	[0.9998]	-
11	P28 → P18 (1) P18 (0)	[1.0]	equal	24	P18 → f (2)	[0.5867]	-
12	P29 → P21 (2) P21 (2)	[1.0]	meet	25	P18 → g (2)	[0.4132]	-
13	P3 → P4 (0) P4 (1)	[1.0]	equal				

Fig. 15. The learned rules on the interaction "*two people meet, then depart.*" Here, $P32$ indicates "*two people approach close to each other,*" while $P39$ implies "*two people depart from each other.*" The whole interaction is denoted as $P37$.

Usually, a person moves continuously. Thus, if the person falls in the same primitive for a few consecutive frames, these primitives will be merged into one primitive interval. Therefore, each interaction can be represented by an interval-based primitive stream. Fig. 14b shows the primitive stream of "*two people meet, then depart.*"

Then, the MDL-based rule induction algorithm is performed to learn the event rules. Here, considering the role of one person in interactions, we extend the rule with the *subject* property. For example, a rule is represented as $P \rightarrow X(0)Y(1)$, where 0 and 1 are the subjects of X and Y. If some IPs become an instance of the rule, a consistent mapping of the subjects in these IPs to that in the rule should be found. In this case, both $X(0)Y(1)$ and $X(1)Y(0)$ are the instances of the rule, whereas $X(0)Y(0)$ is not.

The rules for "*two people meet, then depart*" are shown in Fig. 15. The numbers in parentheses denote the *subject* properties of the subevents, where $subject = 2$ means any *subject* values can match with it. Here, $P32$ indicates "*two people approach close to each other,*" while $P39$ implies "*two people depart from each other.*" The whole interaction is denoted as $P37$.

TABLE 5
Recognition Accuracies of HMM, CHMM, VLMM, and MTP in Recognizing Multiagent Interactions

Interaction	HMM	CHMM	VLMM	MTP
$I1$	83.3%	75%	83.3%	75%
$I2$	50%	58.3%	66.7%	75%
$I3$	75%	75%	83.3%	91.7%
$I4$	66.7%	83.3%	100%	100%
$I5$	91.7%	100%	100%	100%
Average	73.3%	78.3%	86.7%	88.3%

For each interaction, 12 sequences are collected. Then, three-fold cross validation is performed to validate the proposed method. HMM, CHMM, and Variable Length Markov Model (VLMM) are chosen for comparison. For HMM and CHMM, as presented in [11], we compute the four -dimensional feature vectors (relative distance, derivative of relative distance, velocity magnitude, and direction alignment) as input. The number of hidden states for each chain of CHMM are set as 2 due to the small training data set and 3 in the case of HMM. For VLMM, we adopt the method in [35]. The confusion matrices of HMM, CHMM, VLMM, and the proposed MTP are presented in Fig. 16. As shown in the figure, $I1$, $I2$, and $I3$ are easy to misclassify because, when the two people are close to merging, our tracking system tracks the entire group so that the two people's positions cannot be distinguished. As shown in Table 5, we find the proposed MTP method achieves the best recognition performance, due to the compact representation at symbol level.

6.4 Remark

The above has presented the performance of our system in both indoor and outdoor scenes. First, event rules containing hidden temporal structure can be learned by the MDL-based rule induction algorithm effectively. Then, we validate the effectiveness and accuracy of the MTP algorithm in gymnastic exercises experiments, traffic events recognition, and multi-agent interactions recognition.

In the experiments, some limitations of our system are also shown:

- In rule induction, the distortions of unrelated events cannot be removed automatically. To acquire meaningful rules, a manual intervention is needed to delete the distortion events.

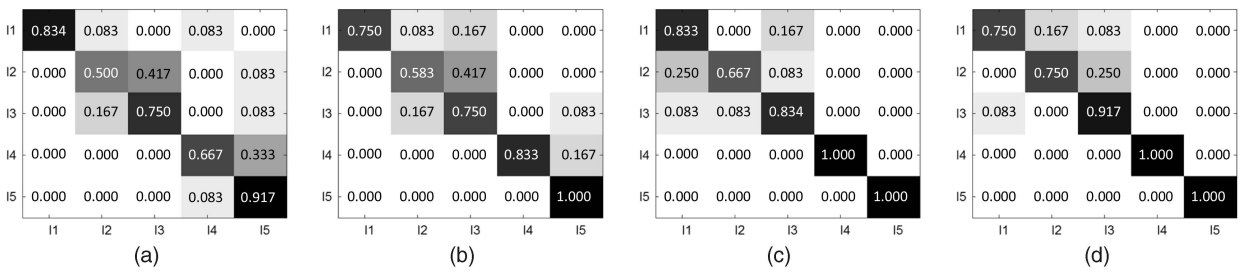


Fig. 16. Confusion matrices with different approaches in recognizing multi-agent interactions. (a) HMM. (b) CHMM. (c) VLMM. (d) MTP.

TABLE 6
Qualitative Comparison with Previous Work on Event Recognition

Comparison of Performance		Trajectory analysis		DBN based		Rule based		
		Stauffer [16]	Hu [18]	Shi [13]	Laxton [12]	Ivanov [21]	Nevitia [33]	Ours
Event Model	(a)	Yes	Yes	Yes	Yes	No	No	Yes
	(b)	No	No	No	Yes	-	-	Yes
Event type	(c)	single	single	multiple	multiple	multiple	multiple	multiple
	(d)	-	-	No	No	Yes	Yes	Yes
	(e)	-	-	Parallel	Parallel	Sequential	Parallel	Parallel
	(f)	No	No	No	No	Yes	Yes	Yes
Anomaly detection	(g)	Yes	Yes	No	No	No	No	Yes
	(h)	No	No	No	No	No	No	Yes

(a):whether the event model is constructed by learning; (b): if (a) is yes, whether the structure of event model is learnt; (c): single agent event or multiple agent event; (d): if (c) is multiple agent event, whether the number of agents should be unknown in the event model; (e): whether the event involves parallel sub-events; (f): whether the hierarchical structure exists in the sub-events; (g): whether the spatial anomaly can be detected; (h): whether the temporal anomaly can be detected.

- In parsing, the decision of temporal relation is based on an appropriate threshold, which is inflexible to handle the uncertainties in practice.
- Although two additional constraints can speed up the parsing process efficiently, the MTP algorithm also has a somewhat high computing cost.

7 Discussion and Conclusion

We have presented an extended grammar system for complex visual event recognition, based on rule induction and multithread parsing. In Table 6, we compare the proposed system with previous work on complex event recognition. There are some desirable properties in our system. In contrast with the trajectory analysis-based work [16], [18], our method not only learns the simple motion patterns of single trajectory, but also the temporal structure in trajectory stream. As a result, not only spatial anomaly, e.g., "*driving in the wrong lane*," can be detected by our method, but the temporal anomaly, e.g., "*passing a red traffic light*," can also be detected. Compared with the DBN-based work [12], [13], our training algorithm does not need manual intervention on the number of moving objects and model topology. Furthermore, because our system runs at the event level, the hidden event structure can be clearly recovered by the parsing algorithm. In comparison with the rule-based method [21], [33], we extend the common SCFG for recognizing complex events with parallel temporal relations, and propose an MDL-based rule induction algorithm to save the cost of the manual definition of event rules.

In summary, we have addressed three important issues left in previous grammar-based work, including: First, traditional SCFG is extended to represent parallel relations in complex event; second, an MDL-based rule induction procedure is proposed to learn the event rules; third, a multithread parsing algorithm is proposed to recognize the interesting event in the given primitive stream, where error recovery strategy is also embedded for robust parsing. Extensive experiments have demonstrated the effectiveness of our system.

In the future, we will focus on the following aspects:

- We will try to remove the distortion of unrelated events in rule induction with some cognitive theory.
- We will adopt some probabilistic methods to decide the temporal relation for handling the uncertainties.
- We will put more effort into optimizing the current parsing mechanism and do experiments in extensive realistic scenes.

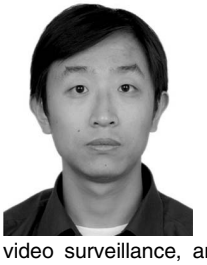
Acknowledgements

The authors thank the anonymous reviewers for their insightful comments that significantly improved the quality of this paper. This work is partly funded by research grants from the National Basic Research Program of China (No. 2004CB318110). The authors also thank Dr. Xuelong Li at Birkbeck College, University of London, and Dr. Dacheng Tao at Nanyang Technological University for their valuable suggestions, and also thank Liangsheng Wang, Daoliang Tan, Shiquan Wang at the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, and Lixin Duan at Nanyang Technological University for their help on trajectory extractions and experimental evaluations.

References

[1] T. Syeda Mahmood, I. Haritaoglu, and T. Huang, "Special Issue on Event Detection in Video," *Computer Vision and Image Understanding*, vol. 96, 2004.
[2] P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea, "Machine Recognition of Human Activities: A Survey," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473-1488, Nov. 2008.
[3] K.-S. Fu, *Syntactic Pattern Recognition and Applications.* Prentice-Hall, 1982.
[4] J.C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words," *Int'l J. Computer Vision*, vol. 79, no. 3, pp. 299-318, 2008.
[5] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior Recognition via Sparse Spatio-Temporal Features," *Proc. Joint IEEE Int'l Workshop Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65-72, 2005.
[6] I. Laptev and T. Lindeberg, "Space-Time Interest Points," *Proc. IEEE Int'l Conf. Computer Vision*, 2003.
[7] A. Yilmaz and M. Shah, "A Differential Geometric Approach to Representing the Human Actions," *Computer Vision and Image Understanding*, vol. 109, no. 3, pp. 335-351, 2008.
[8] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247-2253, Dec. 2007.

[9] D. Weinland, R. Ronfard, and E. Boyer, "Free Viewpoint Action Recognition Using Motion History Volumes," *Computer Vision and Image Understanding,* vol. 104, no. 2, pp. 249-257, 2006.
[10] X. Wang, X. Ma, and E. Grimson, "Unsupervised Activity Perception by Hierarchical Bayesian Models," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2007.
[11] N. Oliver, B. Rosario, and A. Pentland, "A Bayesian Computer Vision System for Modeling Human Interactions," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 831-843, Aug. 2000.
[12] B. Laxton, J. Lim, and D. Kriegman, "Leveraging Temporal, Contextual and Ordering Constraints for Recognizing Complex Activities in Video," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2007.
[13] Y. Shi, Y. Huang, D. Minnen, A. Bobick, and I. Essa, "Propagation Networks for Recognition of Partially Ordered Sequential Action," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2004.
[14] N.T. Nguyen, D.Q. Phung, S. Venkatesh, and H. Bui, "Learning and Detecting Activities from Movement Trajectories Using the Hierarchical Hidden Markov Model," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2005.
[15] S. Gong and T. Xiang, "Recognition of Group Activities Using Dynamic Probabilistic Networks," *Proc. IEEE Int'l Conf. Computer Vision,* 2003.
[16] C. Stauffer and W.E.L. Grimson, "Learning Patterns of Activity Using Real-Time Tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 747-757, Aug. 2000.
[17] F. Porikli and T. Haga, "Event Detection by Eigenvector Decomposition Using Object and Frame Features," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition Workshop,* 2004.
[18] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A System for Learning Statistical Motion Patterns," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 9, pp. 1450-1464, Sept. 2006.
[19] J. Bilmes, "A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models," Technical Report TR-97-021, Int'l Computer Science Inst., 1997.
[20] C. Qing, N.D. Georganas, and E.M. Petriu, "Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar," *IEEE Trans. Instrumentation and Measurement,* vol. 57, no. 8, pp. 1562-1571, Aug. 2008.
[21] Y.A. Ivanov and A.F. Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 852-872, Aug. 2000.
[22] D. Moore and I. Essa, "Recognizing Multitasked Activities from Video Using Stochastic Context-Free Grammar," *Proc. Am. Assoc. Artificial Intelligence,* 2002.
[23] D. Minnen, I. Essa, and T. Starner, "Expectation Grammars: Leveraging High-Level Expectations for Activity Recognition," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 626-632, 2003.
[24] K.M. Kitani, Y. Sato, and A. Sugimoto, "Deleted Interpolation Using a Hierarchical Bayesian Grammar Network for Recognizing Human Activity," *Proc. IEEE Int'l Workshop Visual Surveillance and Performance Evaluation of Tracking and Surveillance,* 2005.
[25] M.S. Ryoo and J.K. Aggarwal, "Recognition of Composite Human Activities through Context-Free Grammar Based Representation," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2006.
[26] S. Joo and R. Chellappa, "Attribute Grammar-Based Event Recognition and Anomaly Detection," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition Workshop,* 2006.
[27] M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato, "Bayesian Classification of Task-Oriented Actions Based on Stochastic Context-Free Grammar," *Proc. Int'l Conf. Automatic Face and Gesture Recognition,* 2006.
[28] Z. Zhang, K. Huang, and T. Tan, "Complex Activity Representation and Recognition by Extended Stochastic Grammar," *Proc. Asian Conf. Computer Vision,* 2006.
[29] C.J. Needham, P.E. Santos, D.R. Magee, V. Devin, D.C. Hogg, and A.G. Cohn, "Protocols from Perceptual Observations," *Artificial Intelligence,* vol. 167, pp. 103-136, 2005.
[30] R. Hamid, S. Maddi, A. Johnson, A. Bobick, and I. Essa, "Discovery and Characterization of Activities from Event-Streams," *Proc. Conf. Uncertainty in Artificial Intelligence,* 2005.
[31] A. Hakeem and M. Shah, "Learning, Detection and Representation of Multi-Agent Events in Videos," *Artificial Intelligence,* vol. 171, nos. 8-9, pp. 586-605, 2007.
[32] M. Fleischman, P. Decamp, and D. Roy, "Mining Temporal Patterns of Movement for Video Content Classification," *Proc. ACM Int'l Workshop Multimedia Information Retrieval,* 2006.
[33] R. Nevatia, T. Zhao, and S. Hongeng, "Hierarchical Language-Based Representation of Events in Video Streams," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition Workshop Event Mining,* 2003.
[34] A. Toshev, F. Bremond, and M. Thonnat, "An Apriori-Based Method for Frequent Composite Event Discovery in Videos," *Proc. IEEE Int'l Conf. Computer Vision Systems,* 2006.
[35] A. Galata, A. Cohn, D. Magee, and D. Hogg, "Modeling Interaction Using Learned Qualitative Spatio-Temporal Relations and Variable Length Markov Models," *Proc. 15th European Conf. Artificial Intelligence,* 2002.
[36] P. Grunwald, "A Minimum Description Length Approach to Grammar Inference," *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing,* pp. 203-216, Springer Verlag, 1996.
[37] F. Hoppner and F. Klawonn, "Finding Informative Rules in Interval Sequences," *Proc. Fourth Int'l Conf. Advances in Intelligent Data Analysis,* pp. 123-132, 2001.
[38] J. Rissanen, *Stochastic Complexity in Statistical Inquiry.* World Scientific Publishing Company, 1989.
[39] C.E. Shannon, *A Mathematical Theory of Communication.* Univ. of Illinois Press, 1949.
[40] Z. Zhang, K.Q. Huang, T.N. Tan, and L.S. Wang, "Trajectory Series Analysis Based Event Rule Induction for Visual Surveillance," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2007.
[41] Z. Zhang, K.Q. Huang, and T.N. Tan, "Multi-Thread Parsing for Recognizing Complex Events in Videos," *Proc. European Conf. Computer Vision,* 2008.
[42] T. Yang, S. Li, Q. Pan, and J. Li, "Real-Time Multiple Objects Tracking with Occlusion Handling in Dynamic Scenes," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* 2005.
[43] J.F. Allen and F. Ferguson, "Actions and Events in Interval Temporal Logical," *J. Logic and Computation,* vol. 4, no. 5, pp. 531-579, 1994.
[44] M. Johnston, "Unification-Based Multimodal Parsing," *Proc. 36th Ann. Meeting of the Assoc. for Computational Linguistics and 17th Int'l Conf. Computational Linguistics,* pp. 624-630, 1998.
[45] J.C. Amengual and E. Vidal, "Efficient Error-Correcting Viterbi Parsing," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 10, pp. 1109-1116, Oct. 1998.
[46] A. Stolcke, "An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities," *Computational Linguistics,* vol. 21, no. 2, pp. 165-201, 1995.
[47] C.G.M. Snoek and M. Worring, "Multimedia Event-Based Video Indexing Using Time Intervals," *IEEE Trans. Multimedia,* vol. 7, no. 4, pp. 638-647, Aug. 2005.
[48] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data Clustering: A Review," *ACM Computing Surveys,* vol. 31, pp. 264-323, 1999.
[49] J.C. Dunn, "Well-Separated Clusters and the Optimal Fuzzy Partitions," *J. Cybernetics,* vol. 4, pp. 95-104, 1974.
[50] CASIA action database, http://www.cbsr.ia.ac.cn/english/Actionpercent20Databases%20EN.asp, 2009.



Zhang Zhang received the BS degree in computer science and technology from Hebei University of Technology, Tianjin, China, in 2002, and the PhD degree in pattern recognition and intelligent systems from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2008. Currently, he is a postdoctoral research fellow at Nanyang Technological University. His research interests include activity recognition, video surveillance, and sequential data mining. He has published a number of papers in top computer vision conferences such as CVPR and ECCV. He is a member of the IEEE.

Tieniu Tan received the BSc degree in electronic engineering from Xian Jiaotong University, China, in 1984, and the MSc and PhD degrees in electronic engineering from Imperial College of Science, Technology, and Medicine, London, United Kingdom, in 1986 and 1989, respectively. In October 1989, he joined the Computational Vision Group in the Department of Computer Science, the University of Reading, United Kingdom, where he worked as a research fellow, a senior research fellow, and a lecturer. In January 1998, he returned to China to join the National Laboratory of Pattern Recognition (NLPR), Institute of Automation of the Chinese Academy of Sciences (CAS), Beijing, China, where he is currently a professor and the director of the NLPR, and a former director general of the Institute (2000-2007). He is also a deputy secretary general of the CAS and the head of the Department of Automation, the University of Science and Technology of China (USTC). He has published more than 290 research papers in refereed journals and conferences in the areas of image processing, computer vision, and pattern recognition. His current research interests include biometrics, image and video understanding, information hiding, and information forensics. He is a fellow of the IEEE and the International Association of Pattern Recognition (IAPR). He has served as the chair or a program committee member for many major national and international conferences. He has once served as an associate editor or a member of the editorial board of many leading international journals including the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Automation Science and Engineering*, *IEEE Transactions on Information Forensics and Security*, *IEEE Transactions on Circuits and Systems for Video Technology*, *Pattern Recognition*, *Pattern Recognition Letters*, *Image and Vision Computing*, etc. He is the editor-in-chief of the *International Journal of Automation and Computing* and *Acta Automatica Sinica*. He is the chair of the IAPR Technical Committee on Biometrics, the founding chair of the IAPR/IEEE International Conference on Biometrics (ICB), and the IEEE International Workshop on Visual Surveillance. He currently serves as the executive vice president of the Chinese Society of Image and Graphics.

Kaiqi Huang received the MS degree in electrical engineering from Nanjing University of Science and Technology and the PhD degree in signal and information processing from Southeast University. Then, he joins in National Laboratory of Pattern Recognition, Chinese Academy of Sciences and now he has been an associate professor. His interests are visual surveillance, image and video analysis, human vision and cognition, computer vision, etc. He has published more than 50 papers in international journals and conferences such as *IEEE Transactions on Image Processing*, *IEEE Transactions on SMC-B*, *Computer Vision and Image Understanding*, *Pattern Recognition*, CVPR, ECCV, and so on. He is a senior member of the IEEE and the deputy secretary-general of the IEEE Beijing section and serves as a member of PC or chair of more than 30 international conferences.