# Trajectory Series Analysis based Event Rule Induction for Visual Surveillance

Zhang Zhang, Kaiqi Huang, Tieniu Tan, Liangsheng Wang
National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China, 100080
{zzhang, kqhuang, tnt, lswang} @nlpr.ia.ac.cn

## Abstract

*In this paper, a generic rule induction framework based on trajectory series analysis is proposed to learn the event rules. First the trajectories acquired by a tracking system are mapped into a set of primitive events that represent some basic motion patterns of moving object. Then a minimum description length (MDL) principle based grammar induction algorithm is adopted to infer the meaningful rules from the primitive event series. Compared with previous grammar rule based work on event recognition where the rules are all defined manually, our work aims to learn the event rules automatically. Experiments in a traffic crossroad have demonstrated the effectiveness of our methods. Shown in the experimental results, most of the grammar rules obtained by our algorithm are consistent with the actual traffic events in the crossroad. Furthermore the traffic lights rule in the crossroad can also be leaned correctly with the help of eliminating the irrelevant trajectories.*

## 1. Introduction

In visual surveillance, there has been increasing interest in recognizing object behaviors, interpreting the high level semantics of dynamic scenes. Trajectory that records the object's position from entering to exiting a scene is one of the most useful information to embed the behavior of moving objects. Much behavior understanding work has been done based on trajectory analysis [3] [4] [5]. However, most previous work focused on modeling the spatial distribution of single trajectory by some clustering techniques. The obtained clusters can be seen as the pathes commonly taken by moving objects. Then based on these pathes, some simple single agent event can be recognized. Nevertheless, the contextual information in the continuous trajectory series has not been studied for event recognition so far, which maybe imply more complex semantics in a longer duration.

In this paper, besides learning the spatial patterns of single trajectory, we proposed a rule induction framework to find the event rules embedded in a continuous trajectory series. The whole framework consists of three layers: trajectory extraction and segmentation layer, primitive event detection layer and event rule induction layer. The trajectory extraction and segmentation layer includes moving object tracking as well as trajectory segmentation that partitions a whole trajectory into several segments with basic semantic meanings. Then the primitive event detection layer maps the trajectory segments into a set of primitive events and forms a primitive event series. Finally in the temporal rule induction layer, a grammar induction algorithm is adopted to acquired a set of event rules. Compared with previous work, the main contributions of this paper include the following:

(1) Different from previous work on trajectory analysis for visual surveillance, our work not only models the spatial distribution of single trajectory, but also the temporal structure in a trajectory series.

(2) Our work is a grammar rule based approach to behavior understanding. Compared with previous grammar rule based work where the rules are all defined manually, we attempt to learn the event rules automatically.

(3) Within the grammar induction framework, frequency and attribute constraint are proposed to filter out the redundant rule candidates. With the constraints, the time cost is reduced vastly and more meaningful rules are also obtained.

The remainder of this paper is organized as follows: Section 2 describes related work on trajectory analysis, event recognition and grammar induction. Section 3 introduces the methods used for the trajectory extraction and segmentation layer. The techniques used in primitive event detection layer are outlined in Section 4. The proposed rule induction algorithm is presented in Section 5. In Section 6, we show the experimental results. Finally, we conclude this work.

## 2. Related work

Most work on trajectory analysis for visual surveillance focused on learning several statistical motion routes in a scene. Hu et al. [3] proposed a hierarchical self-organizing neural network model to learn the motion patterns of single moving object. Stauffer et al.[4]acquired a set of prototypes

1-4244-1180-7/07/$25.00 ©2007 IEEE

using a online vector quantization from trajectories and used hierarchically clustering with the co-occurrence statistics of the prototypes within single trajectory. Porikli [5] proposed a hidden markov model (HMM) based distance to measure the similarity between two trajectories, and a kind of spectral clustering algorithm is adopted to acquire several clusters of single object. In the above work only simple single agent event ("one agent goes from... to...", etc.) could be modeled by these motion routes.

For complex event recognition, a great deal of work has also been done at various levels. Some research focused on modeling the visual event at signal level, in which the observed feature sequence was directly feed into some trained probabilistic model, such as hidden markov model (HMM) and dynamic bayesian network (DBN) and the likelihood probability decided whether the corresponding event occurred or not [6][7]. Probabilistic model has the advantage of handling the uncertainty in the input observation and reasoning using uncertainty. However along with the increasing complexity of events such as the increasing number of agents involved in the event or a larger temporal scale, it will be very difficult to obtain a right model with little training data in a huge dimensional feature space.

On the other hand, there has been some work on modeling the complex activity at event level, where some primitive (atomic) events were first modeled directly from low level features, then based on these primitive events a composite event was represented as a set of sub-events and a set of temporal or logical relations between them. Among different methods, stochastic grammar is very convenient to represent the event structure in a natural way. In [9], Ivanov and Bobick used stochastic context-free grammar (SCFG) to recognize complex large scale events. A two-levels approach is proposed, at the lower level HMMs were used to recognize primitive events, then the primitive events sequence was feed into a grammar parser to identify the complex events. In [8], the Towers of Hanoi game was analyzed by a parameterized stochastic grammar. The experiments showed the high-level parser could recover from local errors and find a consistent overall interpretation of an activity. Recently, Joo and Chellappa [11] recognized normal events and detected abnormal events with attribute grammar which can describe constraints on attributes in addition to the syntactic structure. In [12], Ryoo and Aggarwal also used a context-free grammar (CFG) based representation scheme to represent two persons' interactions such as approach, depart, shake-hands, etc. However in the above work, the event rules were all defined manually. Domain experts were needed to provide all possible rules for all possible events, which seemed impossible in a practical application. Therefore it is necessary to develop a method for learning event rule automatically.

Grammar rule learning called grammar induction has been studied for several decades in artificial intelligence and natural language processing, which aimed to identify a set of grammar rules from a set of training sentences. In [16], Stolcke and Omohundro proposed a probabilistic context free grammar induction method by Bayesian Model Merging. Two operators(merging and chunking) were used to generate the candidates models and evaluate these candidates by maximum posterior probability (MAP) principle. Grunwald [13] has also developed a framework that used minimum description length (MDL) principle to guide search for 'partial' grammar with the similar operators (construct and merge) to generate candidates. In this paper, we adopt a similar grammar induction strategy to learn visual event rule. However the input of our work is not sentences, but trajectory series. Therefore the transformation from trajectory that is in signal level to symbol that is in event level is needed.

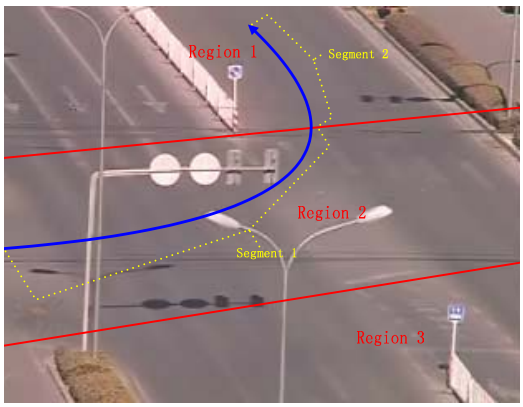



Figure 1. Illustration on semantic regions in a crossroad scene. The two horizontal red lines denote the top and bottom zebra lines respectively. The blue line denotes a trajectory, the arrow indicates the moving direction. The yellow dashed lines show the trajectory segments partitioned by the semantic regions.

## 3. Trajectory extraction and segmentation

We use a tracking system developed by Yang et al.[1] to obtain a large number of continuous trajectories in a long time. However the semantics of each trajectory is not explicit, which may lead to a blind investigation in the following process. So some preprocessing step is needed.

As presented in[10], the vehicles and pedestrians in a far field surveillance scene are all inherently intentional objects, therefore the visible component of their behavior can often be explained with reference to their goals. In this work,we define several semantic regions manually (as it is not the focus of this paper) in a surveillance scene whose borders are equivalent to the goals in [10]. Therefore a whole trajectory is divided into several segments by the semantic regions it passed. Figure 1 illustrates an example

of trajectory segmentation in a crossroad scene. Then each segment is deemed to coincide with a primitive event with the basic semantic meaning, such as "vehicle $A$ approaches a certain region border along $X$ way". "$X$ way" denotes the motion pattern of "vehicle $A$". And how to decide the motion pattern will be solved in the next section.

## 4. Primitive event detection

In this section, we simply outline how to transform a trajectory segment series into a primitive event series.

### 4.1. Similarity based clustering

In each semantic region, a similarity based clustering algorithm is performed to obtain a set of motion patterns of single trajectory segment. Here, only a small subset of the large dataset is needed to do the work, considering the computing cost.

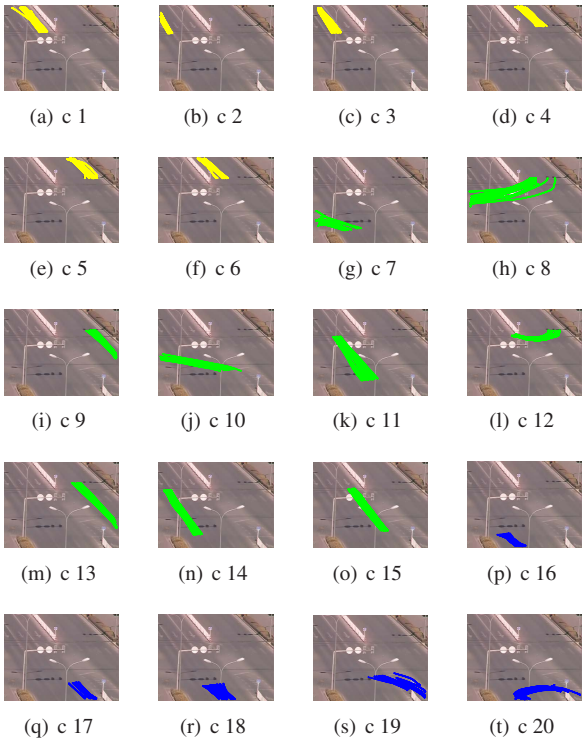
Due to the advantage for clustering task in outdoor surveillance scene [2], the PCA (Principle Components Analysis) +Euclidean distance is adopted to compute the similarity between two trajectory segments. After the similarity matrix is obtained, the spectral clustering algorithm proposed in [18] is used to partition the segments into several motion patterns. The clustering results in a traffic crossroad are shown in Figure 2. Each sub-figure shows one of the motion patterns. One can see that the trajectory classes in different driving lanes are separated correctly.

### 4.2. Primitive event detector

The motion patterns obtained by clustering are deemed as a set of primitive events. Then a hidden markov model (HMM) that takes account into the uncertainty of the low level processing is trained in each cluster. These HMMs are used as the detectors of primitive events. For a given trajectory segment, the HMM that yields the maximum likelihood is the recognition result.

### 4.3. Representation of event series

By far we have transformed a trajectory segment at signal level into a symbol at event level. However just a symbol is not enough to represent the event for the following rule induction. In our work, utilizing some trajectory descriptors, such as the bounding box that describes the spread spatial range of the trajectory and the motion orientation, we represent each event as a 5-tuple $\{e\_type, id, t\_range, s\_range, likelihood\}$, where $e\_type$ denotes which event occurred. $id$ indicates the agent identity of the current event. $t\_range$ means the the time interval of the event in the whole event series, which is represented as $(start\_point, end\_point)$. $s\_range$ denotes the event's spread range in image coordinates in addition to the motion orientation, which is represented as



(a) c 1 (b) c 2 (c) c 3 (d) c 4
(e) c 5 (f) c 6 (g) c 7 (h) c 8
(i) c 9 (j) c 10 (k) c 11 (l) c 12
(m) c 13 (n) c 14 (o) c 15 (p) c 16
(q) c 17 (r) c 18 (s) c 19 (t) c 20

Figure 2. The trajectory segments clusters corresponding to 20 primitive events. The clusters in different semantic regions are represented by different colors.

$\{(x_{min}, x_{max}), (y_{min}, y_{max}), (\theta_{min}, \theta_{max})\}$. *likelihood* is the likelihood probability of the observed data, given the event model. Finally an event series is formed by arranging all the events in terms of the $end\_point$ ascendingly.

## 5. MDL principle based event rule induction

In this section, we present the proposed event rule induction algorithm in details.

### 5.1. Definition of event rule

In this study, event rule comprises two parts: structure and attribute, in order to represent composite event.

The representation of structure is based on stochastic context-free grammar (SCFG) [9]. However the traditional grammar only can represent the sequential relation between two sub-events, whereas the event in our work is interval based representation which allows more complex temporal structure to exist in the sub-events. We refer to the representation of state interval patterns in [15], and also use Allen's temporal interval logic[14] to describe the relation between two interval events. Thus in our work, a production rule with size of $n$ is defined as follows:

$$E \rightarrow s \ \{R\} \quad [P] \tag{1}$$

where $E$ is the leftmost nonterminal, $s$ is a symbol string of size $n$, each symbol denotes an event. $R$ is the relation matrix, where the element $r_{ij}$ denotes the temporal relation between the $i_{th}$ event and the $j_{th}$ event in $s$. $P$ is the conditional probability of the production being chosen given the event $E$. An example of production rule and the corresponding instance in the event series is shown in Figure 3.

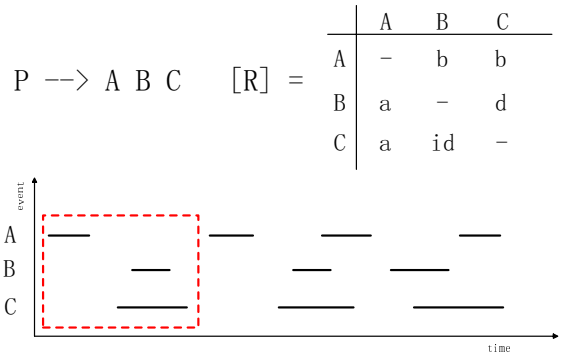



Figure 3. An example of production rule in our work. (abbreviations: a=after, b=before, d=during, id=inverse during). In the underside of the figure, the part enclosed by the red dashed line illustrates an instance of this production.

Besides the structure representation, two real-valued attributes associated with each symbol occurring in the production rule are used to characterize the event rule, which can weaken the possible ambiguity in the event rule set. one is temporal duration $\widehat{t\_length}$ that denotes the average temporal duration of the rule's instances and the other is spatial range $\widehat{s\_range}$ that is the average $s\_range$ of the rule's instances, where $s\_range$ has been presented in section 4.3

## 5.2. Rule induction framework

Referring to the work on grammar induction [13] [16] and structure discovery [17], we adopt a MDL principle based grammar induction algorithm to generate a set of event rules. The basic algorithm is shown in Alg.1.

As shown in Alg.1, a beam search strategy is selected to avoid local optimization. The second while loop is the core of our algorithm. In each run, for each item $\langle CurG, CurS\rangle$ that includes the current grammar and the compressed event series, the rule candidates are acquired by two operators $merge$ and $construct$. Then they are evaluated by the MDL principle. The best candidate that can compress the event series farthest results in a new item which is inserted in $ChildStack$. In the replacement, the attributes $\widehat{t\_length}$ and $\widehat{s\_range}$ of the rule are obtained by computing the average temporal duration and $s\_range$ of the rule's instances respectively. After all items in $ParentStack$ have been handled, the algorithm will switch $ParentStack$ and $ChildStack$, so that the next induction is to be performed.

**Algorithm 1** Grammar induction framework

```
1:  ParentStack = ∅; ChildStack = ∅;
2:  Result = ⟨initialG, S⟩; Depth = 0;
3:  Push ⟨initialG, S⟩ in ParentStack;
4:  while Depth < MaxDepth do
5:    while ParentStack ≠ ∅ do
6:      ⟨CurG, CurS⟩ = ParentStack.at(0);
7:      Delete ParentStack.at(0)
8:      Candi = GenerateCandi(⟨CurG, CurS⟩);
9:      for each rule r in candidates set Candi do
10:       [NewDL,⟨NewG, NewS⟩]
          =MDLEncoding(⟨CurG, CurS⟩, r);
11:       Insert ⟨NewG, NewS⟩ in ChildStack in
          terms of NewDL ascendingly;
12:       if length(ChildStack)>BeamWidth then
13:         Delete the item at the end of ChildStack;
14:       end if
15:     end for
16:   end while
17:   Depth = Depth + 1;
18:   if ChildStack ≠ ∅ then
19:     Result = ChildStack.at(0);
20:     Switch ChildStack and ParentStack;
21:   else
22:     break;
23:   end if
24: end while
25: return Result;
```

## 5.3. Rule candidates generation with constraints

Two operators called $construct$ and $merge$ presented in [13] are adopted to generate rule candidates.

In [13], candidates were generated for all pairs of nonterminals in the current "partial grammar". However too many candidates will be generated with such exhausted strategy, which increase the next evaluation load vastly. Furthermore some rule candidates that can compress the data but without any semantic meaning may disturb the subsequent induction process. Therefore two kinds of constraints are proposed to filter out the redundant candidates.

**Frequency Constraint**. The more frequent of the instances of a $construct$ rule exist in the event series, the more decrease of description length (DL) will be achieved by rule replacement. Therefore in $construct$ operation, a rule candidate can be generated only if its $support$ exceeds a threshold $supp_{min}$, where the $support$ denotes how often a candidate occurs in event series. And a frequent pattern mining algorithm [15] is applied to decide the $support$. The details of the mining algorithm can be found in [15]. In this work the maximal length of rule is set to 2 for saving computing time.

Here, the $support$ of each rule candidates is also used for computing the conditional probability of a production lately. In each iteration, suppose there are $n$ productions that share the identical leftmost non-terminal, the conditional probability of $i$th production is obtained as follows:

$$P(E \rightarrow s(i)) = \frac{supp_i}{\sum_{j=1}^{n} supp_j} \tag{2}$$

For other production that has the unique leftmost non-terminal, its probability is set to 1.

**Attribute Constraints**. By using three event attributes $id, t\_range, s\_range$, three constraints are proposed:

*a. Spatial constraint.* Two given events $A$ and $B$ in the event series can be constructed or merged, only if $spatial\_dis(A, B) < threshold_s$, where $threshold_s$ is a given threshold. The spatial distance between event $A$ and event $B$ is denoted by $spatial\_dis(A, B)$. As we have presented in section 4, the $s\_range$ of an event can be represented as a three-dimensional cube $cube_A$ with x, y axis in image coordinate and the orientation axis. So $spatial\_dis(A, B)$ is defined as:

$$spatial\_dis(A, B) = 1 - \frac{vol(cube_A) + vol(cube_B)}{vol(union(cube_A, cube_B))} \tag{3}$$

where $vol(cube_A)$ indicates the volume of $cube_A$. $union(cube_A, cube_B)$ returns the smallest cube that contains the $cube_A$ and $cube_B$. The distance ranges from -1 to 1. The smaller the distance, the nearer the two events. And the distance is scale independent, which is very important because the average spatial-temporal scale of event will become larger and larger in the induction process. Note, $union(cube_A, cube_B)$ is also the $spatial\_range$ of new event.

*b. Temporal constraint.* Two given events A and B can be constructed, only if $temporal\_dis(A, B) < threshold_t$. The constraint is only valid in $construct$ operation. Similar to the definition of $spatial\_dis$, the temporal distance is defined:

$$temporal\_dis(A, B) = 1 - \frac{len(t_A) + len(t_B)}{len(union(t_A, t_B))} \tag{4}$$

where $t_A$ is the temporal interval of event $A$ $(start\_point_A, end\_point_A)$. $len(t_A) = (end\_point_A - start\_point_A)$. $union(t_A, t_B)$ return the smallest temporal interval that contains the $t_A$ and $t_B$. It is also the temporal interval of the new event.

*c. Agent constraint.* If the constraint is valid, only the events whose $id$ are identical can be constructed. In our work, the agent constraint is valid in the beginning of the induction process to generate single agent event rules. When no one candidates satisfies the agent constraint, it will be terminated. Whereas the spatial and temporal constraints are valid throughout the rule induction process.

### 5.4. MDL encoding of event series

The MDL principle was proposed in its modern form by J.Rissanen [19]. According to the MDL principle, the best grammar is the one that minimizes $L(G, S) = L(S \mid G) + L(G)$, where $G$ is the learned grammar, $S$ is the input event series. $L(S \mid G)$ is the number of bits needed to encode the event series with the help of the grammar, $L(G)$ is that to encode the grammar.

Similar to the definition of event rule, the event series also can be described as a symbol string and the corresponding relation matrix. Therefore $L(S \mid G)$ is computed as two steps:

*a. Encoding the symbol string:* Suppose the symbol string of current event series is $e_1 e_2 ... e_n$ and the set of unique symbols is $\{E_1, E_2, ..., E_c\}$, where $e_i = E_k$ results from replacing the sub-series $sub\_s$ in the original event series by the current rule set. To encode the symbol string, the number of bits $S_{bits}$ is computed according to [20]:

$$\begin{aligned} S_{bits} &= -\sum_{i=1}^{n} logP(sub\_s, e_i) \\ &= -\sum_{i=1}^{n} log(P(sub\_s|e_i) * P(e_i)) \\ &= -\sum_{i=1}^{n} log(lik\_e_i * P(e_i)) \end{aligned} \tag{5}$$

where $P(e_i) = P(E_k)$ is the normalized support of $E_k$ in the event series, 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 sub-series $e'_1 e'_2 ... e'_m$ with the event rule $E \rightarrow E'_1 E'_2 ... E'_m$.

$$lik\_e = \left( \prod_{j=1}^{m} lik\_e'_j \right) * P(E \rightarrow E'_1 E'_2 ... E'_m) \tag{6}$$

where $P(E \rightarrow E'_1 E'_2 ... E'_m)$ is the probability of the production, $e'_j$ is the instance of $E'_j$.

As described in the above part, the likelihood of sub-event is embedded into the new event by replacement, therefore the uncertainty of primitive event detection is also considered in the encoding process.

*b. Encoding the relation matrix:* We compute the number of bits $Rbits$ needed to encode the relation matrix with an enumerative encoding strategy.

Because the relation matrix is an antisymmetry-like matrix, only encoding the upper triangular matrix is enough. And there are only 8 possible temporal relations $\{before, meet, overlap, start, during, finish, equal, i-finish\}$ in the upper triangular matrix. Moreover, it is explicit that most of the values in the $i$th row of the upper triangular matrix are "*before*", when $i << N$.

Therefore given the numbers of 7 relations(except for "*before*") occurred in the $i$th row $v_j, j \in \{1, 2, ..., 7\}$, there are $\prod_{j=1}^{7} \begin{pmatrix} v_j \\ N - i - \sum_{k=0}^{j-1} v_k \end{pmatrix}$ possible placements, where $v_0 = 0$. A feasible encoding scheme is assuming each placement has equal probability of occurrence. So we need

$$r_i = \log \prod_{j=1}^{7} \begin{pmatrix} v_j \\ N - i - \sum_{k=0}^{j-1} v_k \end{pmatrix} \quad (7)$$

bits to encode the positions of 8 relations in the $i$th row.

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 the event series. Thus the total number of bits for encoding the relation matrix is

$$R_{bits} = \sum_{i=1}^{N} [r_i + 7 \log u_i + \log N] \quad (8)$$

Finally we need $L(S|G) = S_{bits} + R_{bits}$ bits to encode the event series, given the current grammar G.

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

## 6. Experimental results

A crossroad scene shown in Figure 1 was chosen to validate our methods. A continuous vehicle trajectory series with the size of 2499 was obtained in a rush hour. The total time length was 90 minutes which included 45 traffic signal cycles. After trajectory segmentation, 4455 trajectory segments were obtained to form Dataset 1. 20 primitive events were acquired by the techniques presented in section 4, which have been shown in Figure 2. The following experiments verified the effectiveness of the proposed rule induction algorithm. We realized tests on a PC with WinXP, VC++6.0, P4 3.0GHz, 512M RAM.

### 6.1. Rule induction with original data

We performed the rule induction algorithm with different size of the events series in Dataset 1. In the experiments the beam width was set to 3, and the thresholds of attribute constraints were tuned by trial and error. For each event rule obtained in the final rule set, we examined whether the structure and attributes were reasonable or not manually, thereby its semantic meaning could be assessed. To draw a comparison, the experiments were also carried out with the original algorithm without any constraints. The number of meaningful rules and meaningless rules in these experiments are recorded in Table 1. Shown in this table, the proposed algorithm produces more meaningful rules effectively due to the attribute constraints, in contrast the result of original algorithm is poor. That is because the algorithm will prefer overgeneralization if no constraint is imposed on $merge$ operator, which leads to the confusion of different semantic events.

Table 1. The number of rules with semantic meaning (Num1) and meaningless rules (Num2) in the final rule set. $Avg. = \frac{Num1}{(Num1+Num2)}$ that indicates the effectiveness of an algorithm.

| Series Length | Original Alg. | | Proposed Alg. | |
|---|---|---|---|---|
| | Num 1 | Num 2 | Num 1 | Num2 |
| 1000 | 1 | 20 | 33 | 10 |
| 2000 | 2 | 20 | 34 | 10 |
| 3000 | 2 | 23 | 35 | 11 |
| 4000 | 2 | 23 | 34 | 10 |
| Avg. | 7.53% | | 76.8% | |

Several main traffic events in the crossroad are obtained successfully by the proposed algorithm, which includes "Go straight over the crossroad in the main road", "Turn left from the main road to the side road" and "Turn left from the side road to the main road". Figure 4 shows these event rules and the corresponding semantic meanings. As shown in Figure 4, not only are the single agent events represented correctly, but also the recursive rules are generated successfully which describe the continuous passing event in a green light period.

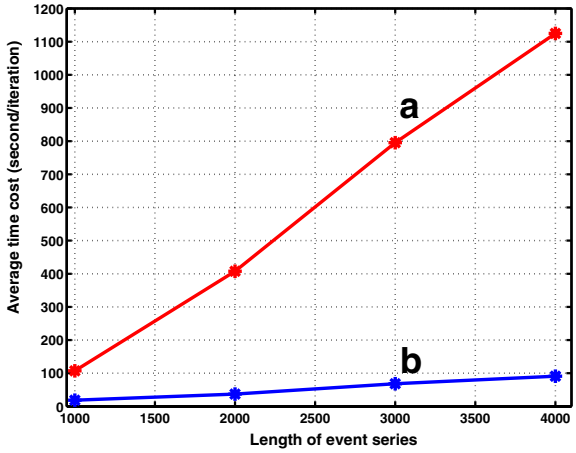



Figure 4. The time cost of (a) the algorithm without any constraints and (b) the proposed algorithm with frequency and attribute constraints.

The comparison result of time cost in the above experiments is presented in Figure 5. We find the time costs of the original algorithm and the proposed algorithm are all close to a liner function of the size of event series. However the time cost of the original algorithm is much huger than the proposed algorithm due to the redundant rule candidates.

| Event Rule | | | Semantics |
|---|---|---|---|
| S | R | P | |
| A[3.21] -> c13[3.21] | - | 0.533 | Go straight over the crossroad at the right way of the main road. |
| A[2.34] -> c9[2.34] | - | 0.467 | |
| B[4.66] -> A[2.33] c5[2.29] | m | 0.387 | |
| B[5.27] -> A[3.20] c4[2.05] | m | 0.433 | |
| B[5.47] -> A[3.41] c6[2.01] | m | 0.180 | |
| C[10.49] -> B[5.13] B[5.21] | b | 0.517 | |
| C[20.2] -> C[9.34] C[9.51] | b | 0.483 | |
| D[4.90]->c19[1.90] c10[2.9] | m | 1 | Turn left from the main road to the side road |
| E[9.56] -> D[4.95] D[5.03] | o | 0.582 | |
| E[19.3] -> E[9.45] E[9.57] | o | 0.418 | |
| F[3.00] -> c15[3.00] | - | 0.390 | Go straight over the crossroad at the left way of the main road. |
| F[3.17] -> c11[3.17] | - | 0.610 | |
| G[4.32] -> F[3.16] c18[1.12] | m | 0.816 | |
| G[4.41] -> F[3.18] c17[1.19] | m | 0.184 | |
| H[4.91] ->c14[3.81] c16[1.1] | m | 0.328 | |
| H[6.53] -> c3[2.22] G[4.28] | m | 0.273 | |
| I[5.20] -> c1[2.23] F[2.94] | m | 1 | |
| H[6.44] -> I[5.18] c17[1.22] | m | 0.248 | |
| H[6.77] -> c2[1.85] H[4.88] | m | 0.151 | |
| J[12.99] -> H[6.43] H[6.75] | o | 0.587 | |
| J[20.3] -> J[12.82] J[15.41] | o | 0.413 | |
| K[6.91] -> c8[4.12] c4[2.73] | m | 0.813 | Turn left from the side road to the main road |
| K[7.82] -> c8[4.92] c5[2.86] | m | 0.187 | |
| L[14.12] -> K[6.89] K[7.04] | b | 0.613 | |
| L[29.12] -> L[13.89] L[14.3] | b | 0.387 | |

Figure 5. Main meaningful rules obtained by the proposed algorithm in Dataset1. Abbreviations in R: b=before, m=meet, o=overlap. The capital letter denotes the inducted composite event. $c\ i, i \in \{1, 2, ...20\}$ is the primitive event that has presented in Figure 2. The numeric value in the bracket of every event symbol is the temporal duration attribute. The spatial range attribute is not shown in this figure, due to the space limitation.

### 6.2. Rule induction with manual intervention

In the above experiments, we have demonstrated the performance of the proposed methods. However we find the traffic light rules in the whole scene still can not be represented correctly, which can be described as: "In one traffic cycle, first is go straight in the main road, then turn left from the main road to the side road, finally turn left from the side road to the main road." The main reason is there are too many irrelevant trajectories existing in the dataset, which are unrelated to the traffic light rules at all. These trajectories mainly include "Turn right from the side road to the main road ($c7$ in the Figure 2)" and "Turn left between the left side and the right side of main road ($c12$, $c20$ in the Figure 2)". Their combinations with other relevant events are to form meaningless rules that may disturb the subsequent induction process. And there are not clear spatial and temporal constraints for their occurrences, therefore the simple attribute constraints are invalid to remove their distortions.

To further illustrate the effectiveness of our algorithm, we manually removed these unrelated trajectories ($c7$, $c12$ and $c20$) from Dataset 1, and tested the proposed algorithm in the new dataset (Dataset 2). Dataset 2 consists of 3478 trajectory segments. The experimental results show the traffic light rules are represented correctly in the final rule set by the proposed algorithm. Figure 6 shows a recovered structure in a traffic signal cycle. "$P$" means the traffic cycle event. "$M$" denotes "Go straight in the main road with two directions". "$N$" is the alternate event between event"Turn left from the main road to the side road"and event "Turn left from the side road to the main road". The "before","equal", etc. indicate the temporal relations between the two subevents. The meanings of other symbols can be found in Figure 2 and Figure 4.

## 7. Conclusion

We have presented a generic framework to learn the event rules based on trajectory series analysis. Compared with previous grammar rule based work on event recognition, our work focuses on learning the event rules automatically. First a set of primitive events are obtained by trajectory segment clustering. Then a MDL based grammar induction algorithm with frequency and attribute constraint is adopted to obtain a set of event rules. Experiments in surveillance scene have demonstrated that the proposed algorithm is effective and efficient to generate the meaningful event rules. Although the proposed algorithm may be disturbed by some unrelated trajectories, the problem can be solved with a little manual intervention. Compared with previous work, our methods can result in huge manual cost savings.

In this work, only some simple attribute constraints are adopted to filter out the redundant rule candidates. In the future, we plan to take some cognitive knowledge into account. Also, the insertion and deletion errors in event series should be handled.

## Acknowledge

The work reported in this paper was funded by research grants from the National Basic Research Program of China (No.2004CB318110), the National Natural Science Foundation of China (No.60605014, No.60335010 and No.2004DFA06900) and CASIA Innovation Fund for Young Scientists. The authors also thank the anonymous reviewers and the editor for their valuable comments.

## References

[1] Tao Yang, Stan Z.Li, Quan Pan and Jing Li, "Real Time Multiple Objects Tracking with Occlusion Handling in Dynamic Scenes", *In Proc. Conf. CVPR* vol.1 pp.970- 975 2005. 2

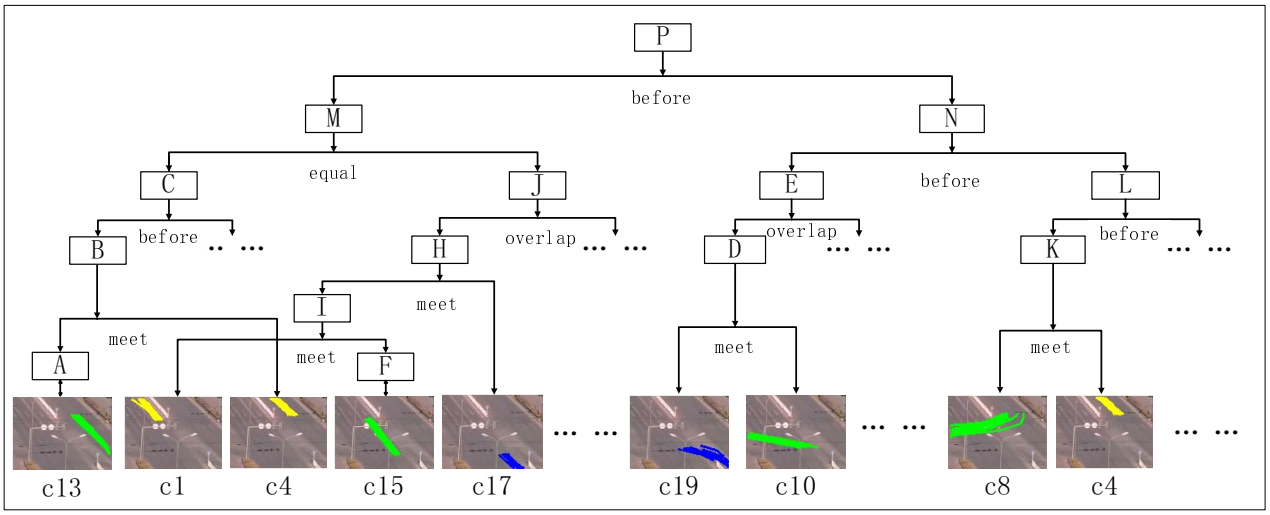



Figure 6. A traffic signal cycle recovered by our algorithm. Just a partial recovered structure is shown, due to the space limitation.


[2] Zhang Zhang, Kaiqi Huang, Tieniu Tan, "Comparison of Similarity measures for Trajectory Clustering in Outdoor Surveillance Scenes", *In Proc. Intl Conf. Pattern Recognition* pp.1135-1138 2006. 3

[3] Weiming Hu, Dan Xie, and Tieniu Tan, "A hierarchical self-organizing approach for learning the patterns of motion trajectories, *IEEE Transactions on Neural Networks*, Vol.15, pp.135-144, 2004. 1

[4] Chris Stauffer, Eric Grimson, "Learning Patterns of Activity Using Real-Time Tracking", *IEEE TRANS.PAMI*, 22(8):747-757, 2000. 1

[5] Porikli F.M. Haga T., "Event Detection by Eigenvector Decomposition Using Object and Frame Features", *in Proc. Conf. on CVPRW*, vol 7, pp 114-121, 2004. 1, 2

[6] S. Gong, T. Xiang, "Recognition of group activities using dynamic probabilistic networks" *in Proc. Intl. Conf. on Computer Vision*, vol.2, pp. 742-749, 2003. 2

[7] Y.Shi, Y.Huang, D.Minnen, A.Bobick, and I.Essa, "Propagation Networks for Recognition of Partially Ordered Sequential Action", *In Proc. Conf. CVPR* vol 2, pp.862-869, 2004. 2

[8] D.Minnen, I.Essa, T.Starner, "Expectation Grammars: Leveraging High-Level Expectations for Activity Recognition",*in Proc. Conf. CVPR*, vol.2.pp.626-632 2003. 2

[9] Y.A.Ivanov, A.F.Bobick, "Recognition of visual activities and interactions by stochastic parsing",*IEEE TRANS.PAMI*, vol.22.no8,pp.852-872, Aug 2000. 2, 3

[10] Hannah Dee and David Hogg, "Detecting Inexplicable Behaviour", *In Proc. British Machine Vision Conference 2004 (BMVC'04)*, 2004. 2

[11] Seong-Wook Joo, Rama Chellappa, "Attribute Grammar-Based Event Recognition and Anomaly Detection", *in Proc. Conf. on CVPRW*,pp.107-114, 2006. 2

[12] M.S.Ryoo, J.K.Aggarwal, "Recognition of Composite Human Activities through Context-Free Grammar Based Representation" *in Proc. Conf. on CVPR*, pp.1709-1718,2006. 2

[13] P. Grunwald. "A minimum description length approach to grammar inference" *Connectionist, Statistical and Symbolic Approaches to Learning for NaturalLanguage Processing* , volume 1004 of Lecture Notes in AI, pp 203-16 Berlin: Springer Verlag, 1996. 2, 4

[14] J F.Allen, F.Ferguson, "Actions and Events in Interval Temporal Logical", *J.Logic and Computation* Volume 4, Number 5, pp.531-579, Otc 1994. 3

[15] F. Hoppner, F. Klawonn, "Finding Informative Rules in Interval Sequences Advances in Intelligent Data Analysis", *in Proc. of the 4th International Symposium*, Lecture Notes in Computer Sciences 2189, pp.123-132, 2001 3, 4

[16] A.Stolcke, S.Omohundro, "Inducing Probablilistic Grammars by Bayesian Model Merging", *in Proc. Intl. Conf. on Grammar Induction(ICGI'94)*, 1994 2, 4

[17] D J.Cook, L B.Holder, "Substructure Discovery Using Minimum Description Length and Background Knowledge", *Journal of Artificial Intelligence Research*, pp 231-255, 1994 4

[18] M.Maila, J.Shi, "A Random Walks View of Spectral Segmentation" *AI and STATISTICS (AISTATS01)*, 2001 3

[19] J.Rissanen, *Stochastic Complexity in Statistical Inquiry*, World Scientific Publishing Company, 1989. 5

[20] Claude E. Shannon, *A Mathematical Theory of Communication*, University of Illinois Press, 1949. 5