REPRESENTATION AND RECOGNITION OF AGENT INTERACTIONS USING MARKING ANALYSIS IN GENERALIZED STOCHASTIC PETRI NETS

Artyom Borzin, Ehud Rivlin, and Michael Rudzsky

Computer Science Department, Technion-Israel Institute of Technology 32000, Haifa, Israel
email: artyom.borzin@tx.technion.ac.il , (ehudr, rudzsky)@cs.technion.ac.il

ABSTRACT

This paper presents a novel approach for video event representation and recognition of multi agent interactions. The proposed approach integrates behavior modeling techniques based on Generalized Stochastic Petri Nets (GSPN) and introduces Petri net marking analysis for better scene understanding. The GSPN model provides remarkable flexibility in representation of time dependent activities which usually coexist with logical, spatial and temporal relations in real life scenes. The nature of Petri net concept allows efficient modeling of the complex sequential and simultaneous activities but disregards the global scope of a given model. The proposed marking analysis creates a new model extension based on the global scene view and uses historical and training information for current and future state interpretations. The GSPN approach is evaluated using the developed surveillance system which can recognize events from videos and give a textual expression for the detected behavior. The experimental results illustrate the ability of the system to create complex spatiotemporal and logical relations and to recognize the interactions of multiple objects in various video scenes using GSPN and marking analysis capabilities.

1. INTRODUCTION

The development of video surveillance systems presents many challenges in creating a robust applications that will effectively combine the methods for object detection, feature extraction, tracking, behaviour modelling and event interpretation. Any behaviour analysis process begins from choosing a powerful event representation method that supports the recognition of complex activities. *R.Nevatia et al.* presented the Event Recognition Language (ERL) [1] which can describe hierarchical representation of complex spatiotemporal and logical events. The proposed hierarchical event structure was constructed of primitive, single-thread and multi-thread events that were recognized using Bayesian network and semi-HMM methods. Lately, *R.Nevatia et al.* developed Video Event Representation Language (VERL) for describing ontology of events and Video Event Markup Language (VEML) to annotate instances of the events described in VERL [2]. Another representation technique base on hierarchical CASE representation was proposed by *M. Shah et al.* in [3] and then enhanced by [4].

The dynamic nature of video clips always requires robust modelling technique that can efficiently treat the uncertainty of the video scenes. Therefore, the Bayesian Networks [5-7] and various HMMs [8-10] have been widely used in the area of video event recognition. *K.Murphy* introduced the Dynamic Bayesian Networks (DBN) which generalizes HMMs by improving the state space representation [11].

Y.Ivanov and A.Bobick proposed the stochastic parsing approach [12] that combines the Coupled HMM (CHMM) for low level temporal event detecting and gesture classification with stochastic context-free parser (SCFG) for structural activity recognition using the external knowledge about the video domain.

Another modelling technique based on Petri nets was presented by *C.Castel et al.* [13]. This work proposed a symbolic language to capture the logical and algebraic conditions. Activity prototypes and state conditions were suggested and then interpreted by the Petri net graph. *N.Ghanem et al.* proposed using Petri nets for mining of surveillance video in [14]. A high level query language that allows the user to submit spatiotemporal queries about human activities was developed. The recognized events were hierarchically combined using primitive and composite events. Spatial, temporal and logical relations using the Petri nets were defined and illustrated on some real video episodes.

Our work extends the work of *N.Ghanem et al.* [14] and proposes to integrate Generalized Stochastic Petri Nets (GSPN) for video event representation and recognition. We introduce Petri net marking analysis for better video scene understanding of current state and for predicting of the next state upon the available training information. Finally, we present a surveillance system which is based on the proposed concepts and demonstrate experimental results on some real and synthetic video clips.

2. EVENT REPRESENTATION USING GSPN

The Petri net model is represented as a bipartite graph that consists of two node types: *places* and *transitions*. The nodes

can be connected by *regular* or *inhibit* arcs. Place nodes may contain a number of another graph component called *tokens*. Petri net components are illustrated on Figure 1.

The dynamic behaviour of the Petri net is obtained by *firing* tokens from one place node to another according to the parameters associated with transition nodes. The parameters that define transition node behaviour can be represented by the *enabling rules* and *firing rules*. Both the enabling and the firing rules are specified through arcs. An enabling rule defines the preconditions that should be satisfied before the transition node may fire, while the firing rules defines the marking modification that will take place in case of transition firing. A comprehensive presentation of the Petri nets structure, dynamic behaviour and modelling techniques are given in [15].

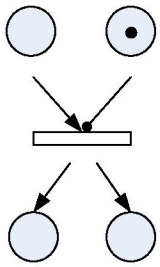

Figure 1: Petri net components.

In our representation model, each token represents an object that exists in the video scene. Places represent the object states and the transitions represent video events that provide dynamics of the behaviour model. Transition node firing is a result can be equivalent to the object state change in the real scene or can be result of a satisfied relation constraint.

The flexibility of an enabling rule is predefined by the tracking and classification information that is supplied by the intermediate video processing layer.

An enabling rule triggers transition firing if and only if:

1. Each input place contains token(s) that satisfy enabling rule constraint and the number of such tokens is greater or equal than a given arc multiplicity
2. Each inhibitor place contains a number of tokens smaller than a given threshold. This rule can be modified to applied the sane enabling rules on the tokens contained in the inhibit nodes.

Our representation model does not allow firing rule modifications, however it can be customized for implementation of complex firing behaviour. Any firing transition deletes from each input place as many tokens as the input arc multiplicity and then adds to each output place as many tokens as the output arc multiplicity. Occasionally, more then one enabling rule can be satisfied at the same time and depending on the graph structure, it can lead to a conflict because firing of one transition may immediately disable another transition. The conflicts can be resolved in a controllable way by adding *priority* parameter to each transition node. In case of conflict high priority transition will always fire before the low one.

The state change in the Petri net model is typically a result of some activity initiation/completion or a result of condition verification. Considering the first state, the *Timed Petri Net* introduces temporal specifications which are associated with *timed transitions*. The timer that is associated with a timed transition measures the duration of the activity in the modelled system. The activity starts when the conditions of the associated enabling rule are satisfied and stops when the enabling rule is violated. If the duration of the activity is greater then a predefined threshold, the timed transition is allowed to fire.

In some cases, the PN model with strictly defined timed transition delays does not accurately describe the behaviour of the underlying system. An alternative PN model that adopts timed transitions with exponential distributions for transition delays are known as *Stochastic Petri Net (SPN)*. The most popular method for random delay modelling is based on the negative exponential Probability Density Function (PDF). Obviously, the same model may enjoy of a significant advantage of using immediate transitions in PN models together with timed transitions. In our model immediate transitions coexist with timed transitions, transition priorities and random firing delays with negative exponential PDF and this creates *Generalized SPN (GSPN)* model [15].

The PN model state can be characterized by the number of tokens in the place nodes. This defines the current *marking* of the PN model. Prior to involving any PN model dynamics, it is possible to compute the set of all markings reachable from the initial marking and all the paths that the system may follow to move from state to state. The set of all reachable markings define *reachability set* of the graph. The *reachability graph* consists of nodes representing reachable marking and arcs representing possible transition from one marking to another. An example of reachability graph is demonstrated on Figure 2.

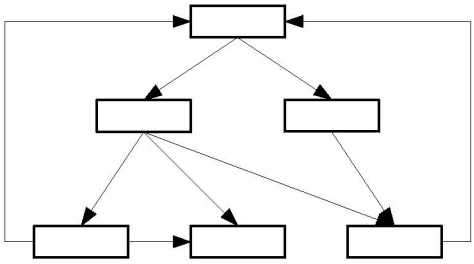

Figure 2: Reachability graph example.

In principle, the firing times are assumed to be independent of the marking of the GSPN model, however the transition parameters can be marking dependent. If one or more transitions are enabled in the same marking then it is possible to calculate the distribution of the sojourn time in that marking. Moreover, the probability of the transition node to fire first

can be computed using the PDFs of the transitions which also may be dependent on current marking. It is important to note that the specification of marking dependent parameters requires special attention to possible global states of the modelled system. The reachability graph of the GSPN system defines the possible transitions between the reachable markings; however the probability of each transition can be computed as:

$$\lambda_{n,k} = N_{n,k} / N_n \qquad (1)$$

Where: $\lambda_{n,k}$ is probability to move to marking M_k from marking M_n, $N_{n,k}$ is number of detected transitions from marking M_n to marking M_k, and N_n is number of M_n marking occurrences.

The reachability graph with the probabilities of marking transitions defines a *Discrete Time Markov Chain* (DTMC) that will describe the stochastic process associated with the system. The DTMC can significantly extend the understanding of the current system state upon the statistical parameters collected during the system training process. A brief presentation of the DTMC capabilities in GSPN can be found in [15].

3. SYSTEM MODELING WITH PETRI NETS

3.1. Logical Relations

A typical representation of logical relations in a Petri Net assumes that that the places and the tokens contained in these places are operands. The satisfied relation is represented by a fire event of the involved transition node. The implementation of the basic logical relations (AND, OR and NOT) can be found in [14].

3.2. Temporal Relations

Similar to the logical relation representation, the place nodes represent the input arguments while the combination of transition node represents the required temporal relation. The transitions are augmented by conditions that should be satisfied for this temporal relation to hold. The complete set of Allen's algebra relations and their implementation in Petri nets can be found in [14].

3.3. Spatial Relations

A spatial relation can be one of the following categories: topological, directional or distance relation. Currently our model supports only directional and distance relations which can be defined using the enabling rules of transition nodes. These rules define the distance between objects D[max,min] and difference in their orientation O[max,min].

3.4 GSPN Model Training

The probabilistic behaviour of the GSPN model is based on the random delays of timed transitions. The probability density of the timed transition delay is a PDF function of the form:

$$D_n = 1 - e^{-t_n / \mu_n} \qquad (2)$$

Where t_n is an enabling period of timed transition n and μ is an average delay of timed transition n. The average delay of the transition is automatically calculated using a training sequence of input videos. During the training, no timed transition can fire, so the average enabling period can be calculated. The training sequence must cover all typical activities in the particular domain. This allows the system to build the *reachability set* of the explored domain and build a Discrete Time Markov Chain.

Each marking node represents a legal state of the system that occurred during the training process and the links between nodes represent legal transitions between the states. These transition probabilities are calculated and used for construction a DTMC. In our surveillance system we use this model for prediction of the most probable next state in the scene and for calculation of conditional probabilities for certain events. This information is presented for the user in real-time and can be stored in textual file.

4. SURVEILLANCE SYSTEM OVERVIEW

The proposed video surveillance system is built of intermediate video processing module, behaviour configuration module and video event interpreter (Figure 3).

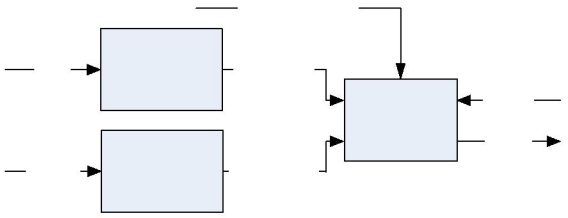

Figure 3: System architecture diagram.

The intermediate processing module performs motion detection, object detection and tracking functions. The output of this sub-system supports the format of the ground truth labeling files as defined in the CAVIAR project [16]. This sub module can be replaced with any format compliant datasets. For instance, we may use a set of synthetic video clips that symbolically draws object locations, but provides ground truth labeling files as in a real scene.

The behaviour configuration module implements a graphical interface for creating behaviour models using the GSPN graphical language. Currently, the correctness and efficiency of the constructed model relies on the user's expertise in Petri net modelling methods and the knowledge of the scene domain. In order to reduce the complexity of the modelling task and to enable model reuse, our system supports various templates. These templates can be created or extended by the user to adopt them for another model.

The video event interpreter analyzes the input video scene using the predefined behaviour model and generates textual expressions for the detected events. The interpretation process can run in two execution modes: training mode and normal mode. During the training all stochastic parameters of the model are calculated and stored. Once the model is trained, the detected states and transition parameters construct the DTMC which will be used for testing new scenes in the same domain.

The result of the interpretation is presented in the log window and then can be stored in a text file. The interpretation module can be configured to run automatically on video databases and store the interpretation results in textual files. This enables our system to analyze existing databases and to perform various user defined queries for specific events.

Our system consists of two separate applications that implement the behaviour modelling and the video event interpretation modules. All graphical interfaces are implemented in C# while the GSPN model is implemented in C++ to give better performance characteristics. The interpretation process can run in two execution modes: training mode and normal mode. During the training all stochastic parameters of the model are calculated and stored in the model.

5. EXPERIMENTAL RESULTS

The following experiment demonstrates GSPN behaviour modelling capabilities and the video event interpretation results. The input for the surveillance system can be obtained from two sources: a pre-processed video clip or a synthesised clip that can be generated using an animation tool that was developed. This tool allows creating a series of random symbolic clips that have common behaviour pattern as well as add abnormal activities that have to be detected. In the synthetic clips, moving objects are drawn as filled color circles and static objects are drawn by their contours. The background area can be schematically divided to interaction zones which are represented by different colors. The annotation file that is created with the generated clip is fully reliable and eliminates the need to handle erroneous or inaccurate tracking results.

The format of the annotation file that the interpretation module expects to receive from an external tracker is presented on Figure 4. The full definition of this format can be found in [16].

```
<frame number="1">
  <objectlist>
    <object id="0">
      <orientation>0</orientation>
      <box xc="10" yc="10" w="10" h="10" />
      <appearance>appear</appearance>
      <hypothesislist>
        <hypothesis id="0" prev="0.0" evaluation="1.0">
          <movement evaluation="1.0">walking</movement>
          <role evaluation="1.0">unknown</role>
          <context evaluation="1.0">unknown</context>
          <situation evaluation="1.0">unknown></situation
        </hypothesis>
      </hypothesislist>
    </object>
  </objectlist>
</frame>
```

Figure 4: Annotation file format.

5.1. Example 1: Security Check

This example assumes a public place where every visitor must pass a security check. Figure 5 shows the clip of abnormal behaviour where one visitor stops (2) for a security check while another one (3) passes near the guard (1) and evades the check. Similar scenes were generated using the same scenario. Figure 6 shows one of them where each circle represents an object in the real clip.

According to the proposed model, the surveillance system should raise a security alert on one of the two events:

- a visitor enters the hall without being checked,
- the security check is abnormally long.

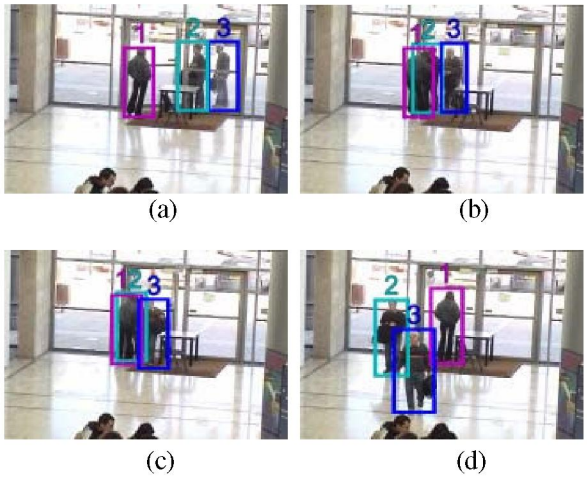

Figure 5: Real security check scene.

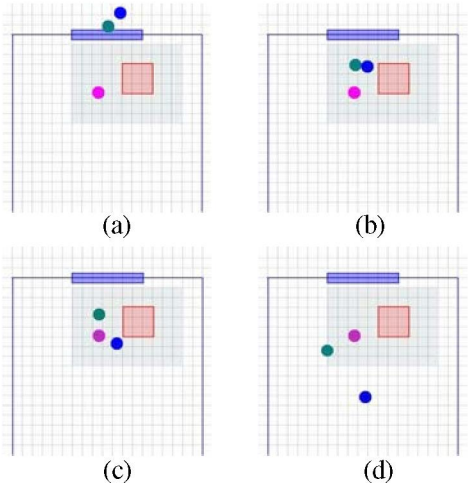

Figure 6: Synthetic security check scene.

The GSPN behaviour model for this example consists of several events as illustrated on Figure 7.

The interpretation starts from a special place node called 'Root Node'. In every new frame, this place contains tokens for all detected objects in the scene. Any new object is reported by the tracker as recently appeared one, then the transition 'Object_Appeared' fires and transfers the corresponding token to the place called 'Object_In_Scene'. There are 3 possible values of object movement characteristic: active, walking or running. The interpretation of the movement property will move the token to one of the appropriate states: 'Object_Is_Moving' or 'Object_Is_Active'. The transition 'Visitor_Entered_the_Hall' that has spatial constraint fires when an object passes through the door and enters the building. The transition 'Visitor_Was_Not_Checked' fires if the visitor leaves the area around the guard before the check begins. If the visitor approaches the guard and both stop moving, we consider that the security check is in progress and then the transition 'Guard_Met_Visitor' fires. The complete list of the detected events is presented in Table 1 and the interface of the interpretation module is presented on Figure 8.

The training sequence constructs the Petri net reachability set that we use for marking analysis. The created marking graph defines the DTMC that we use to calculate the most probable next marking state of the scene. This information improves our understanding of the current state as well. For instance, assume the visitor has entered the hall (Visitor_Walking_Towards_Guard). According to the model and the marking graph presented on Figure 9 the most probability next state is One_Visitor_Stopped_Near_Guard.

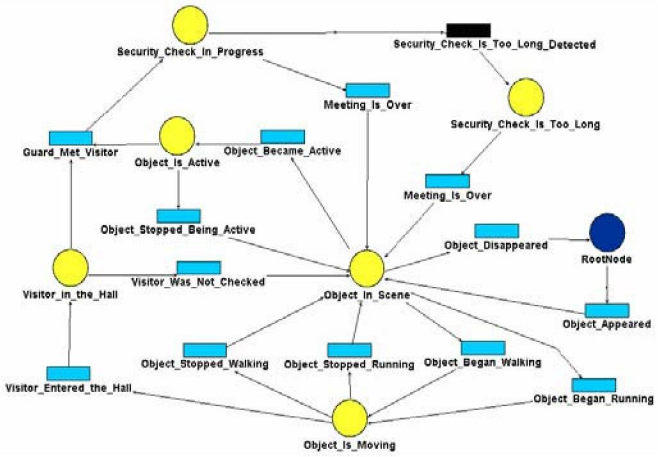

Figure 7: GSPN model for security check example.

Table 1: Interpretation results for security check example.

Frame	Message
1	'Object_Appeared' fired on objects: 0
20	'Object_Appeared' fired on objects :2
25	'Object_Appeared' fired on objects :6
56	'Visitor_Entered_the_Hall' fired on objects :2
61	'Guard_Met_Visitor' fired on objects :0, 2
61	'Visitor_Entered_the_Hall' fired on objects :6
68	'Visitor_Was_Not_Checked' fired on objects :6
71	'Security_Check_Is_Too_Long_Detected' fired on objects :0, 2
86	'Meeting_Is_Over' fired on objects :0, 2

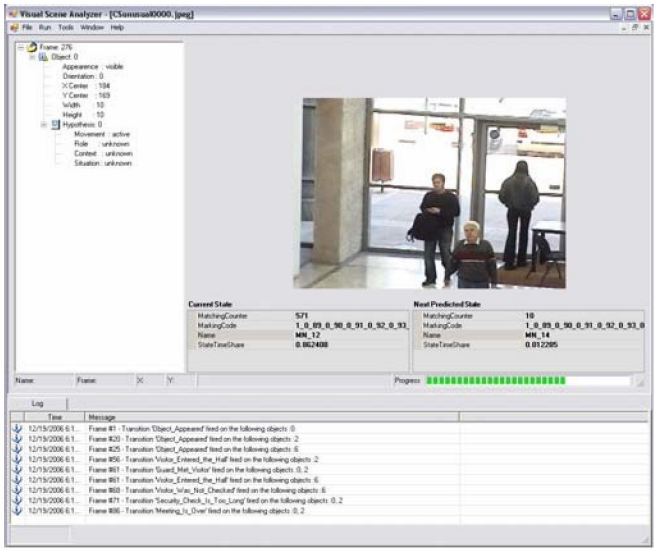

Figure 8: Interpretation module interface.

We can also calculate the probability that this visitor will be properly checked. All possible paths to reach the Guard_Checked_one_Visitor state form the state Visitor_Walking_Towards_Guard are highlighted by the red arcs on Figure 9. The total probability to check one visitor is the sum of probabilities to go over the two highlighted paths:

$$P = 0.72 \times 1 \times 0.63 + 0.72 \times 1 \times 0.37 \times 0.78 = 0.66 \quad (3)$$

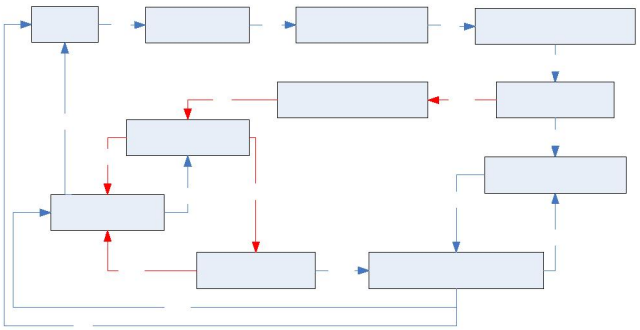

Figure 9: Marking graph for security check example.

5.2. Example 2: Traffic Junction Control

We assume that each car that enters the junction may cross the junction unless there is a car on its right side. Any violation of this rule is considered as abnormal situation and must be reported by the surveillance system. We generated a series of pseudo random synthetic clips where each vehicle is symboli-

cally marked as a colored circle. The bright grey region defines the internal area of the junction while the dark grey region defines the active zone where the vehicles begin their interaction. One of the synthetic clips is illustrated on Figure 10.

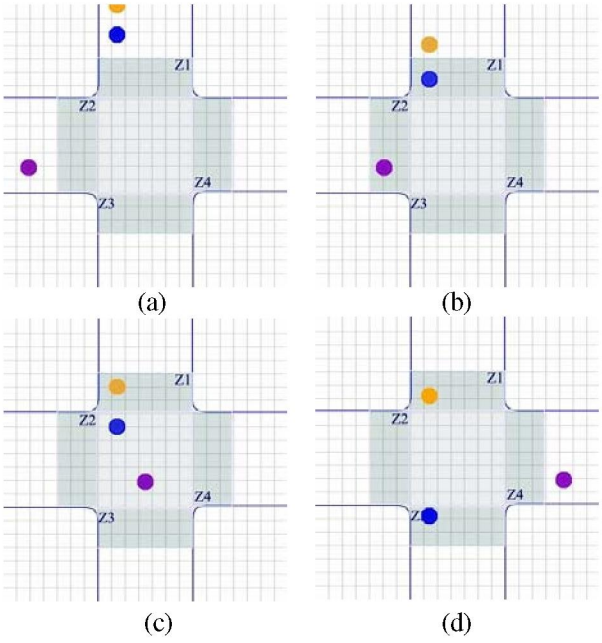

Figure 10: Synthetic road junction scene.

A similar situation was described and interpreted by R.Higgins in [17]. In this work the events are described using the VERL [2] and then recognized by a Bayesian network model. Each instance of similar objects is represented by duplicated networks that are processed simultaneously. This approach significantly increases the computation complexity of a multi object scene while the Petri net approach proposed in this work does not changes the network structure but adds new tokens for the detected objects. For instance, the junction control model does not limit the number of participating objects and does not change its structure during the recognition process. The completed behaviour model for this example is illustrated on Figure 11 and the results of the interpretation process are given on Table 2.

Table 2: Interpretation results for junction control example.

Frame	Message
#0	'Car_Appeared' fired on the objects :2
#1	'Car_Appeared' fired on the objects: 0
#10	'Car_Appeared' fired on the objects :1
#18	'Car_Entered_Z1' fired on the objects: 0
#23	'Car_Entered_Z2' fired on the objects :2
#34	'Car_Entered_Z1' fired on the objects :1
#52	'Car_In_Z1_Brakes_the_Low' fired on the objects: 0
#77	'Car_Entered_Z3' fired on the objects: 0
#80	'Car_Appeared' fired on the objects :3

In this example the surveillance system succeeded to detect a car that crosses the junction despite the fact that there is another car on its right side (Frame #52).

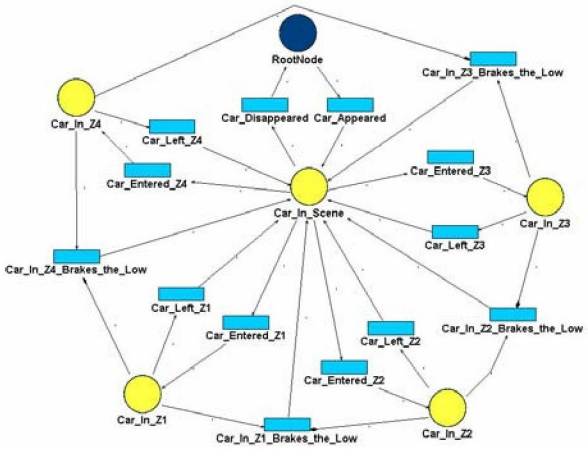

Figure 11: GSPN model for junction control example.

6. CONCLUSIONS AND FUTURE WORK

This paper introduces a new video event interpretation approach using GSPN. This approach integrates advanced GSPN features (timed transitions with random delays, conflict resolution using priorities, etc.) and proposes to add marking analysis into a GSPN model for better scene under standing and for next marking state prediction using historic data. Our work demonstrates the advantages of using random delays based on PDF and proposed a scheme for training timed transitions in a behaviour model. The reachability set is transformed to a DTMC and then used for predicting future states using the transition probabilities.

We present a new surveillance system that adopts GSPN modelling approach for video event representation and recognition. This system provides a powerful user interface for creating various behaviour models that are interpreted by the video event interpretation module.

Our future works will focus on extensions for the marking analysis approach which will allow controlling the spatial and the temporal scope of the analysis and proposing marking analysis models based on HMM or Bayesian networks. A few enhancements will be considered to deal with inaccurate or erroneous extraction of scene or object features.

7. REFERENCES

[1] R. Nevatia, T. Zhao, and S. Hongeng. "Hierarchical language-based representation of events in video streams", *In Proc. IEEE Workshop on Event Mining*, (2003).

[2] R. Nevatia, J. Hobbs, and B. Bolles, "An ontology for video event representation", *In Proc. Int. Workshop on Detection and Recognition of Events in Video*, (2004).

[3] A.Hakeem, Y.Sheikh and M.Shah, "CASEE: A Hierarchical Event Representation for the Analysis of Videos", *Proc. of AAAI*, (2004), pp.263-268.

[4] A. Hakeem and M. Shah, "Multiple agent event detection and representation in videos", *Proc. of AAAI*, (2005).

[5] K. Murphy, "Dynamic Bayesian Network Representation, Inference and Learning", *PhD Dissertation, University of California, Berkeley*, (2002).

[6] S. Intille and A. Bobick, "Representation and visual recognition of complex, multi-agent actions using belief networks" *In IEEE Workshop on the Interpretation of Visual Motion*, (1998).

[7] R. Nevatia, S. Hongeng, and F. Bremond, "Video-based Event Recognition: Activity Representation and Probabilistic Recognition Methods", *CVIU*, (2004), vol. 96(2), pp129-162.

[8] S. Hongeng and R.Nevatia, "Large-Scale Event Detection Using Semi-Hidden Markov Models", *IEEE ICCVs*, (2004), vol.2, pp1455–1462.

[9] N.Oliver, B. Rosario, and A. P. Pentland, "A Bayesian Computer Vision System for Modeling Human Interactions", *IEEE PAMI*, (2000), pp831–843.

[10] K. Murphy and M. Paskin, "Linear time inference in hierarchical HMMs", *Neural Information Processing Systems*, (2001).

[11] K. Murphy, "Dynamic Bayesian Network Representation, Inference and Learning", *PhD Dissertation*, (2002).

[12] Y.Ivanov and A.Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing", *CVPR*,(1998), vol.22, pp.852-872.

[13] C. Castel, L. Chaudron, and C. Tessier, "What Is Going On? A High Level Interpretation of Sequences of Images", *4th European Conference on Computer Vision*, (1995).

[14] N. M. Ghanem, D. Doermann, L. Davis and D. DeMenthon, "Mining Tool for Surveillance Video", *SPIE*, vol. 5307, (2003), pp. 259-270.

[15] M. Marsan, G.Balbo, G.Conte, S. Donatelli and G. Franceschinis, "Modeling With Generalized Stochastic Petri Nets", *John Wiley and Sons*, (1995).

[16] CAVIAR Project: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/

[17] R. Higgins, "Automatic event recognition for enhanced situational awareness in UAV video", *Military Communications Conference MILCOM2005*, pp1706-1711, Vol. 3, (2005).