A Petri Net-based Approach for Event Detection in Pedestrian Crossing Sequence

P. M. Ashok Kumar[1] and Arun Kumar Sangaiah[2*]

[1]Department of CSE, VelTech Technical University, Avadi, Chennai, Tamil Nadu, India
[2]School of Computing Science and Engineering, VIT University, Vellore, Tamil Nadu, India; sarunkumar@vit.ac.in

Abstract

In this paper we present an approach to automatically detect anomalous traffic events like pedestrians crossing the junction, based on traffic video of low-level features such as size of the blob, spatial location, and velocity. The construction of Petri-Nets was used for both semantic description and event detection within traffic videos. The major novelties of this paper are extensions to both the modeling and the recognition capacities of Object Petri-Nets (PN). The detection of object level features are done with the help of state of art techniques like Gaussian Mixture of Models (GMM), and a series of Petri-Nets composed of various objects is proposed to describe the video content. The expected outcome of the proposed framework is that we can easily build semantic detectors based on PNs to search within traffic videos and identify interesting events. Experimental results based on recorded traffic video data sets and synthetic data sets are used to illustrate the potential of this framework.

Keywords: Block Motion Estimation, Gaussian Mixture of Models, Petri Nets

1. Introduction

In Computer Vision Community, the process of monitoring the abnormal behaviour of people and objects automatically, within public places, especially airports, metro stations and traffic intersections, by means of visual aids (cameras), has increased substantially, for safety and security purposes. As a result the amount of video data gathered daily by surveillance cameras increases. Manual detection and recognition of abnormal events by manual operator becomes difficult and report in a timely manner. This process is always subject to human errors and fatigue. Thus, automating the process of video event detection and recognition is one important task of computer vision research. In most of these systems, the goal is to reduce the gap between high level semantics representing human's activity and lower level data given by vision modules. These lower level modules include Background modelling, Object detection and tracking, information representation. Higher level event detection/recognition

module is based on both stochastic inference according to earlier researches (1–6) and deterministic inference (7–10) method. According to the literature review, in this paper we have used deterministic inference using Petri nets.

This paper describes an approach for online detection of Anomalous events in video junctions or intersections based on Coloured Petri nets. With the help of domain expert, Petri nets are developed by forming temporal, spatial and logical relations between primitive events for the events of interest[19]. Our approach consists of Lower level vision modules to extract primitive events. These primitive events are processed by Petri nets filters to detect composite events of interest. This approach is general enough to be applied to any surveillance domain (car parks, airports, indoor scenes, etc.). Thus, a brief introduction about Petri nets and its advantages are given in this section. A Petri net is an abstract model of the flow of information in a system[1]. Using Petri nets as a representation and as a filtering mechanism has the following advantages:

Author for correspondence

a) Petri nets can be used for both deterministic and stochastic inference of event occurrences.

b) Petri nets have a nice graphical representation that uses just a few types of elements. This representation has a well-defined semantics so that it is easy to understand the model and to learn the language.

c) Petri nets have a precise mathematical model that can be used for analysis. For example, there are well-defined algorithms for detecting deadlock and inconsistency in the data.

d) Petri nets can be used to represent sequentiality, concurrency and synchronization of events.

e) Petri nets can be used to represent events in a top-down fashion at various levels of abstraction, i.e. they can be used to model a composite event hierarchically from simpler event models.

f) Compared to classical rule-based expert systems, in terms of efficiency, Petri nets are known to be as efficient as expert systems.

The RETE algorithm, used in most expert systems implementations to improve speed[2], is applicable to Petri nets[3]. The main idea is to exploit temporal data redundancies (coming from the markings that are not changed during transition firing). At any time during the interpretation process, the positions of tokens in the Petri net summarize what happened in the past (keep history) and predict what will happen in the future. In this way, composite events are recognized incrementally and there is no need to re-evaluate past events.

The rest of this paper is structured as follows: Section 2 summarizes previous work done in the area of event recognition. In Section 3, the basic concepts of Petri nets are given. Section 4 presents the system architecture. In Section 5, ontology for event representation is described. In Section 6 the mapping between ontology entities and their Petri net representations is explained. In Section 7, we provide information about the implemented system and experimental results. Section 8 conclusion of the paper and presents ideas for future work.

2. Related Works

Detection of Anomalous events from video data can be seen as an inference problem, where some inference mechanism is applied to available knowledge with the help of domain expert to infer the occurrence of these events in the video data. In this section, we discuss a number of deterministic methods available for event detection and then how Petri nets can be used as an inference mechanism in rule-based expert systems. C. S. Pinhanez et al.[4] proposed for fast detection of actions and sub actions by incorporating Allen's temporal relationships into a simpler three-valued PNF-network. The arc consistency algorithm AC-2 is used to propagate the temporal constraints. Rota et al.[5] proposed for describing all activities (states of the scene, events and scenarios), represented by conditions between the objects of the scene and Arc consistency-4 or AC-4, is used to reduce the processing time for recognizing activities in video sequences. For increasing the efficiency of processing temporal constraints, Vu et al.[6] suggested to decompose the scenario models into simpler scenario models, in a preprocessing step. The recognition of simple scenarios just tries to link two scenario instances. For loosely coupled events, the method leads to several decompositions which is more expensive for recognition.

Castel et al.[7] used Petri nets as an inference mechanism to represent the evolution of a car parking scene with human and vehicles. A symbolic language is defined to capture the logical and algebraic conditions that are handled in a set of prototypes. An Activity prototype is a set of logical and algebraic relations holding for a finite set of objects and scene elements. A Plan prototype is a set of relations between some Activity prototypes and some state conditions. The Plan prototype is interpreted as a Petri net. Places are associated with Activity prototypes and state conditions. Transitions are associated with logical conditions and constraints. Stochastic inference methods like Hidden Markov models[8,9], stochastic context free grammars[1] and Bayesian networks[4,10–13] have been applied successfully to event recognition from video data.

Petri nets have been used as an inference mechanism for rule-based expert systems[20]. In 1993, Srinivasan and Jafari showed the similarities between a rule-based expert system and a Petri net: transitions can represent rules, markings can represent facts, and the token player can represent the inference engine. They also showed that using a Petri net representation increases the efficiency of rule-based expert systems by providing parallelism and pipelining. Since then, many expert systems were developed using Petri nets as a knowledge representation that guides the inference process. Examples include work done by Murata and Zhang[14], Hura[15], Li[3] and by Murata and Yim[13]. Earlier studies proposed Object Petri-Nets[5,6]

(or colored Petri-Net) for class of event model. In this, each token represents the detected object in the scene and places represents particular states of the objects. Enabling rules in transitions in this model class are conditions on the properties of the object tokens. The events of interest are the transitions themselves and a complex event of interest may lie at the end of a chain of a number of such events.

3. Proposed Work

The proposed Anomalous event detection method is used to detect the abnormal events such as unnecessary crossing of Pedestrians, which normally occurs at traffic junction. First, Petri net model is designed for a specific event in traffic junction scenario, with the help of domain expert. The next thing is to apply the test video containing abnormal events, to the Petri net models. Figure 1 shows the proposed architecture for abnormal event detection. Initially, the moving foreground objects in the video are detected using Gaussian Mixture Model (GMM) and tracked using Block Motion Estimation using Three Step Search Algorithm.

Whenever new objects are created, new coloured tokens are generated inside the Object's places and moved when ever primitive event condition is met. When the tokens reach the final state, we can say that anomalous event has occurred and the token with distinct colour represents the association with that anomalous event i.e., pedestrian crossing/traffic rule violation.

3.1 Object Detection and Tracking

In this Paper, our emphasis is mainly on event modeling task. So, we used state of art techniques like Gaussian mixture models[7] for object detection, Morphological operations are applied for noise elimination to get resultant blob. Based on the sizes of the blob are analyzed and classified in to moving vehicles, person, or pedestrian. Object Tracking is done in two steps (i.e., tracking by detection technique)[21–24]:

1. Detecting moving objects in each frame.
2. Associating the detections corresponding to the same object over time.

The association of detections to the same object is based solely on motion and distance proximity. In this application, we used a simple tracking method to track multiple objects, under the assumption of object's movement in minimal between the frames and all objects are occlusion free.

Our simple tracking method works as follows:

1. **Input:** Our input video is converted in to sequence of frames ($f_1, f_2, f_3, \ldots, f_n$).
2. **Initialization:** Let f_0 be the first frame for which detection is available with GMM[6]. Create a new track set T_J^0 for each blob or block detection j.
3. **Iteration:** loop over frames i from f_1 to f_n.
4. **Object detection and Track Assignment:** Perform Block Motion Estimation using Three Step Search
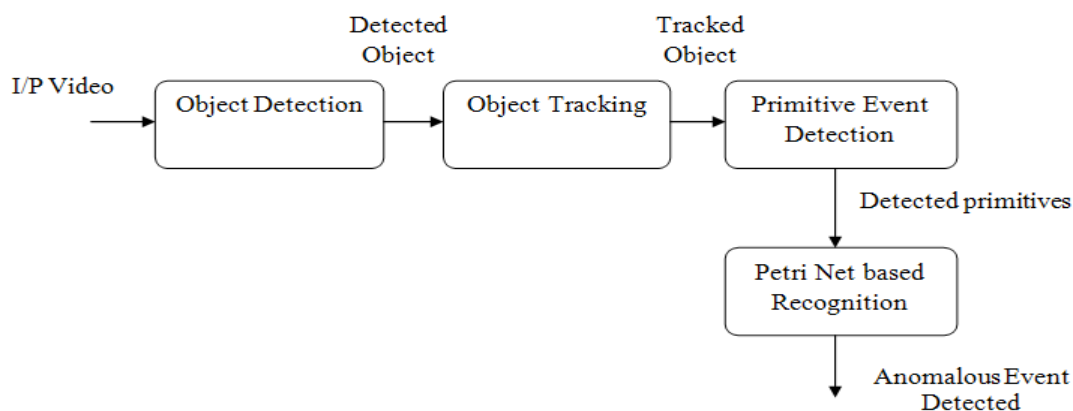

Figure 1. Proposed Petri Net framework for Abnormal event detection.

Algorithm[18] based on Bock matching criteria i.e., Mean Square Error(MSE)

$$MSE\left(d_1, d_2\right) = \frac{1}{N_1 \times N_2} \sum_{(n_1, n_2)} \sum_{\in B} \left[s(n_1, n_2, k) - (n_1 + d_1, n_2 + d_2, k+1)\right]^2 \quad (1)$$

where B is $N_1 \times N_2$ block of pixels, and s(x, y, k) denotes a pixel location at (x, y) in frame k, (d_1, d_2) is the displacement of the center pixel of the block.

5. **Delete Lost Tracks:** Each track keeps count of the number of consecutive frames, where it remained unassigned. If the count exceeds a specified threshold, (i.e., object might have left out of that frame) and we delete those tracks.
6. **Create New Tracks:** Introduce new Blobs by applying GMM[6] at every 10 frames. We eliminate noise object detection with the help of size and appearance.
7. **Output:** In our case, object's id, spatial location and name are extracted to form a formatted output, for further processing in our proposed framework.
8. Track new and old blobs by using steps 4–7.

3.2 Primitive Event Detection

A primitive event is defined as the simplest type of events inferred directly from the observables in the video data (e.g. position, trajectory, speed, etc.). The main function of this module is to detect types of primitive events occurred by interpreting the data provided by the object detection and tracking module. It will keep track information about different object states, with respect to object's position, speed and acceleration. Any change in these parameters signals the detection of an event. In our work, we defined all primitive events with respect to pedestrian and moving vehicles and which are as follows: *"car appeared, car disappeared, person appeared, person disappeared, car entered in zone, car not in zone, person is moving, person is not moving, car accelerates, car decelerates"*.

We classified the object based on the size of the blob to determine whether it is pedestrian or moving vehicle and marked all the pedestrian crossing regions as critical zones manually. If the instance of the primitive event is car appeared/person appeared is said to be detected of an object appears for the first time and remains as a foreground object for at least *k* frames. On the other hand, if an existing object car/person, which cannot be tracked for at least k frames is said to disappear, an instance of the primitive event car/person disappears is said to be detected. The object detection and tracking module computes trajectories motion of a tracked object, which is described in terms of its position, speed, and acceleration. When the moving object is is moving, it starts decelerating and when the change in object position during a specified number of frames is under a given threshold, an instance of the primitive event car decelerates is detected. On the other hand, if the object is not moving and then it starts accelerating, an instance of the primitive event car accelerates is detected. For each marked critical zone, if the object pedestrian is moving (based on motion parameters), then primitive instance person is moving is detected and if the object pedestrian is not moving then primitive instance person is not moving is detected, in that region. Car entered in zone primitive event is detected based on the object name, objects location and its motion pattern.

To test if an object is inside or outside a given ROI, we test whether there is overlap between the bounding box around the object and the ROI polygon. But this can result in a large number of false positives as while the bounding box of an object may overlap the ROI polygon, the object itself may be completely outside the ROI. To solve this problem, we use the bounding box information only. For example, for a pedestrian to be inside a ROI polygon, the bottom line of its bounding box should be inside the ROI polygon. In other words, the pedestrian's feet are inside the ROI polygon, since the bottom line of the bounding box usually touches the pedestrian's feet. For a vehicle to be inside a ROI polygon, we need to ensure that its wheels are inside the polygon. Since we only have the 2D information about the objects, it is not practical to find the wheels. Instead, we estimate the bounding box side that touches the front side of the vehicle. To do this, we first measure the direction in which the vehicle is moving and then identify the bounding box side in this direction. Assuming that this side represents the front of the vehicle, to detect a vehicle Enters ROI, we test whether this line lies inside or at least intersects the ROI polygon. We also wait until the overlap area is above a pre-defined percentage of the ROI area. This approach reduces the false positives rate.

3.3 Petri-net based Framework for Anomalous Detection

In this section, we describe about Token based Petri net models for a traffic intersection scenario as shown in Figure 2 and each tokens represents distinguished foreground object which can be vehicles or pedestrian.
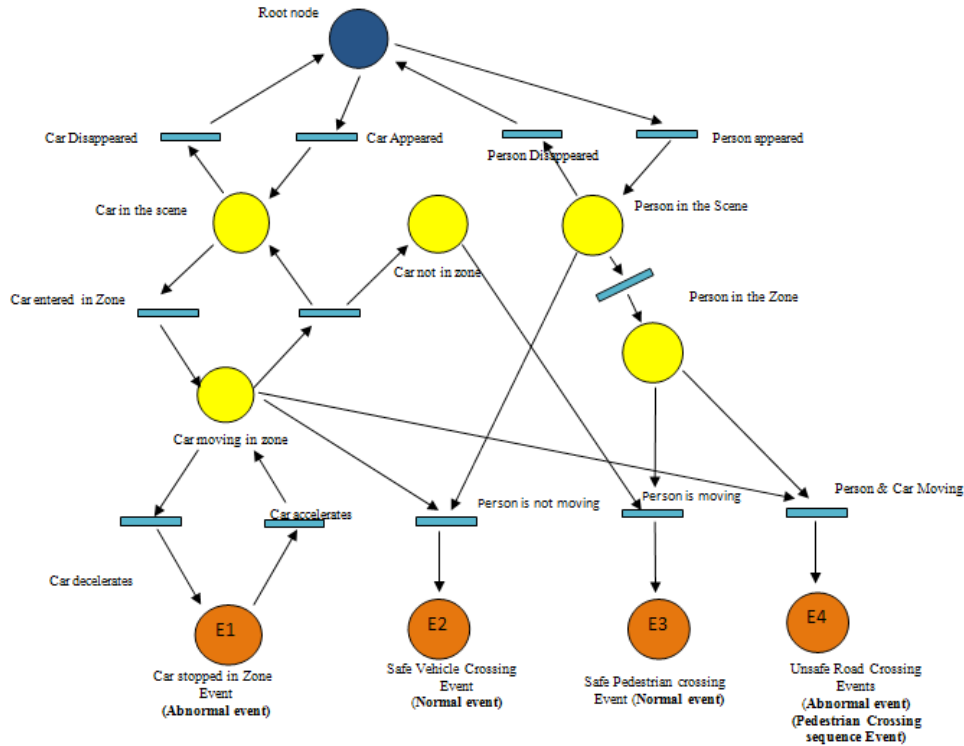
Figure 2. Proposed petri net based framework for event detection.

The state of each token is based on the position present in places and movement of these tokens from one place to another place is determined by the condition placed on each transition. A transition is enabled whenever there is at least one token in each of its normal input places and no tokens in any of its inhibitor input places; otherwise, it is disabled. An enabled transition fires by removing one token per place from a random subset of its normal input places and depositing one token per place in a random subset of its output places. We used immediate transition which fires in the instant, as it becomes enabled.

This model can be used for any traffic scenario, provided we should mark the critical & non critical regions. These critical zones are marked uniquely. Our main emphasis is on to detect and record both Normal and Abnormal events related to pedestrians and vehicles behaviors as they occur; This also can be used as a tool to analyze video archives to study pedestrians and vehicle behaviors in the intersection.

Here, we define and detect four types of Events and are listed as following:

a) **Car stopped in Zone Event, E1**: Vehicles that stop at the Critical regions like cross walk regions, No

parking region etc., and denoted as E1 – Abnormal Event.

b) **Safe Vehicle Crossing Event, E2**: Vehicles crossing the zone region while pedestrians are waiting and denoted as E2 – Normal Event.

c) **Safe Pedestrian Crossing Event, E3**: Pedestrians crossing the zone region while Vehicles are not in zone and denoted as E2 – Normal Event.

d) **Unsafe Road Crossing Events, E4**: Pedestrians who cross the road using marked zone, while car is also moving towards/in the zone – Abnormal Event.

We discuss the representation of the traffic intersection events in our Petri nets framework. The primitive library used has six primitive events, which are: *"car appeared, car disappeared, person appeared, person disappeared, car entered in zone, car not in zone, person is moving, person is not moving, car accelerates, and car decelerates"*. With the help of domain knowledge available, four events are identified with their associated triggering object and their states.

The Normal Event *"Vehicle Crossing Event, E2"* is identified with tokens at object state E2. This occurs when vehicle is crossing the critical zone while the person is present

static in non critical region depending on primitive events *"car entered in zone"*, *"person appeared"* and the normal event *"Safe Pedestrian Crossing Event, E3"* occurs when pedestrian crossing zone while vehicle is static in the scene. The above events activate depending on their transition conditions placed on their object transitions and these are considered as normal events as per the domain knowledge.

The Abnormal Events *"Car stopped in Zone Event, E1"* is detected when the vehicle is stopped at zone irrespective of the state of the pedestrian. The event *"vehicle stopped"* depends on condition placed on its position, velocity gradient at the transition. The Abnormal Events *"Unsafe Road crossing Event, E4"* is detected when both vehicles and pedestrians moving in the same zone. The main advantage of using Petri net models is that we can model the complex temporal relation that exists between primitive events in Normal and Abnormal Events. The above things are represented in the Petri net representation.

4. Results and Discussion

We applied our framework on a 14-minute video stream captured from a static camera monitoring a traffic intersection. The system is used to model and detect all types of normal events and abnormal events described in Section 3. Once a violation is detected, an alert is made by the system displaying the vehicle or the pedestrian committing the violation and the time when it happens.

The Figure 3 shows recognized instances of different event by the system. The marking of critical zones is shown in Figure 3a. The Figure 3b shows the Normal event "**Safe Vehicle Crossing Event, E2**" i.e., Pedestrian present in different zone and waits for all the vehicles to cross over. This is realized with the static velocity condition and movement of vehicles, with conditions placed at the transitions. This normal behaviour is shown in Figure 3b. in the figure we have represented only a single frame for showing normal events.

Figure 3c represents "**Unsafe Road Crossing Events, E4**" and it represents the scenario of pedestrians using the cross walk region while the vehicle is moving towards it and the primitive events detected are Car moving in zone and pedestrian moving in the same zone. It is abnormal events detection. Figure 3d shows the "**Safe Pedestrian Crossing Event, E3**" and represents the scenario of pedestrians cross the zone while vehicles are moving in non critical region. It is Normal Event as we have defined in previous section. We also applied the same system to a set of synthetically generated data, with some nodes (pedestrians,
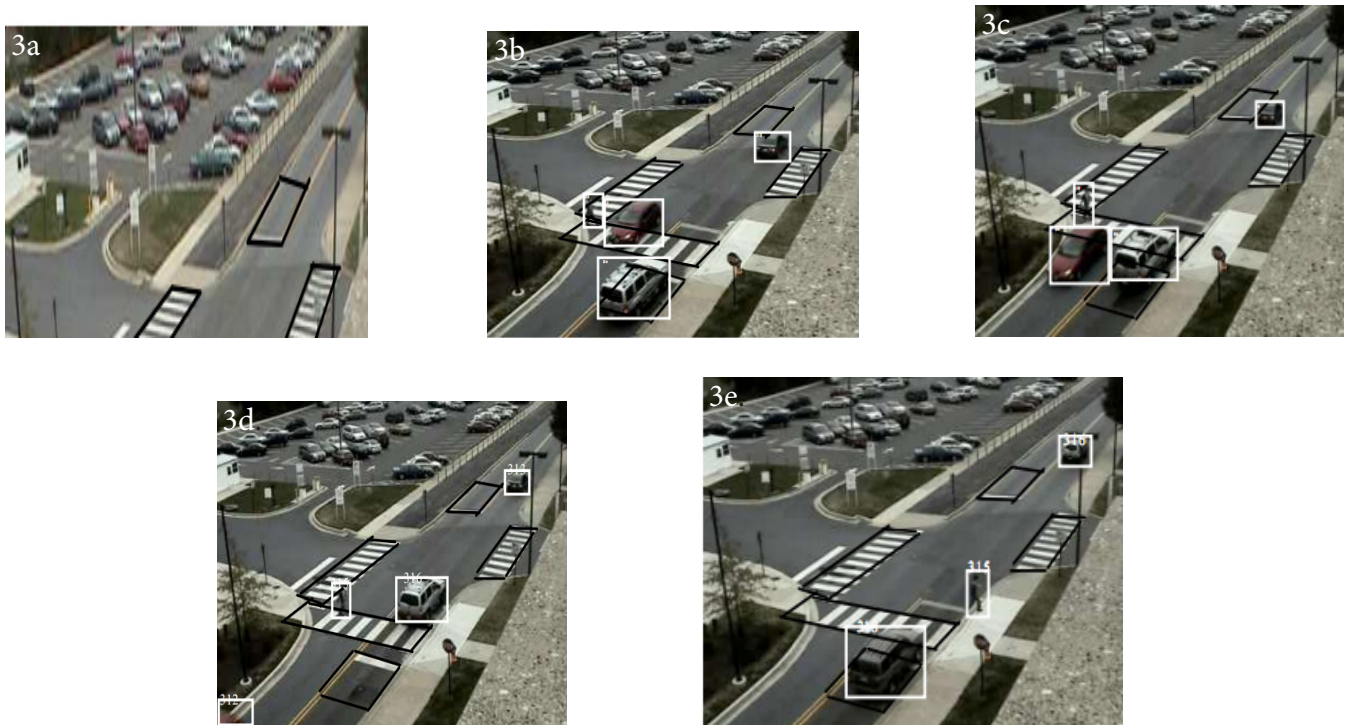

Figure 3. Events detected.

Table 1. Traffic Intersection Event Detection Results

S.No	Event Name	Ground Truth		True Positives		False Positives		Detection rate (%)		False alarm rate (%)	
		I	II	I	II	I	II	I	II	I	II
1	Car stopped in Zone Event, E1	5	15	4	13	1	2	88%	89%	12%	11%
2	Safe Vehicle Crossing Event, E2	29	20	27	18	2	2	94%	90%	6%	10%
3	Safe Pedestrian Crossing Event, E3	24	25	21	22	3	3	88%	89%	12%	11%
4	Unsafe Road Crossing Events, E4	14	25	13	23	2	2	93%	92%	7%	8%
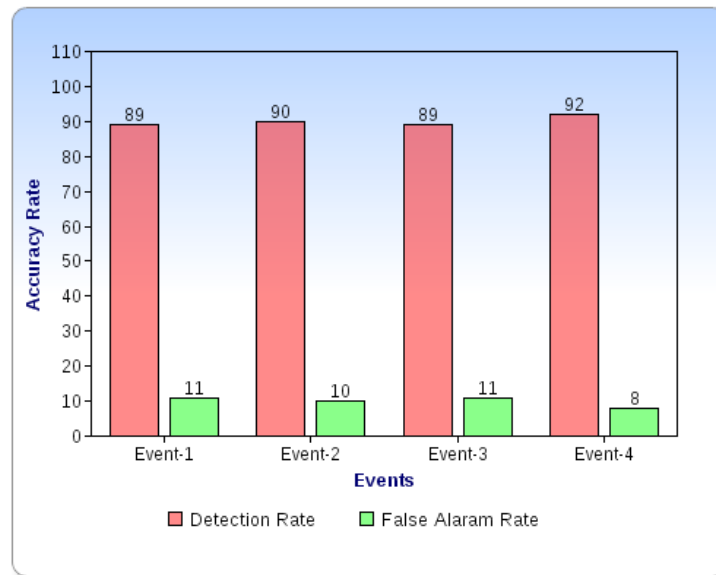
Figure 4. Prediction Accuracy Rate of the events.

vehicles) randomly moving from one region to another region. The results are also tabulated from Table 1. We compared these results with results from original video and are tabulated.

5. Conclusion

Most of the Events are complex in surveillance videos are difficult to model because of the complex logical and temporal relations that exists between different objects. So, we designed Petri net models mainly for four types of events namely "Car stopped in Zone Event, E1", "Safe Vehicle Crossing Event, E2", "Safe Pedestrian Crossing Event, E3", "Unsafe Road Crossing Events, E4". Our model comprises of different states corresponding to different objects in the scene, with conditions placed on the immediate transition and successfully detected. In the real traffic video taken,

we got an average detection rate shown in Figure 4 for all events greater than 88%. The remaining 12% false alarm rate is mainly due to approximate location data, misclassification rate. In future we can improve this aspect of lower level modules, thereby reducing the false alarm rate.

6. References

1. Ivanov YA, Bobick AF. Recognition of visual activities and interactions by stochastic parsing. IEEE Trans Pattern Anal Mach Intell. 2000; 22(8):852–72.
2. Forgy CL. RETE: A fast algorithm for the many pattern/ many object pattern match problem. Artif Intell. 1982; 19(1):17–37.
3. Li L. High-level petri net model of logic program with negation. IEEE Trans Knowl Data Eng. 1994; 6(3):382–95.
4. Pinhanez CS, Bobick AF. Human action detection using PNF propagation of temporal constraints. 1998 IEEE

Computer Society Conference on Computer Vision and Pattern Recognition; 1998 Jun 23–25. IEEE. p. 898–904.

5. Rota N, Thonnat M. Activity recognition from video sequences using Declarative models. Proceedings of the 14th European Conference on Artificial Intelligence; 2000 Aug 20–25; Berlin, Germany. p. 673–680.

6. Stauffer C, Grimson W. Adaptive background mixture models for real-time tracking. IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 1999 Jun 23–25; Fort Collins, CO. IEEE. p. 246–52.

7. Castel C, Chaudron L, Tessier C. What is going on? A high level interpretation of sequences of images. 4th European conference on computer vision, Workshop on conceptual descriptions from images. United Kingdom: Cambridge Publication. 1996.

8. Oliver N, Rosario B, Pentland A. A Bayesian computer vision system for modeling human interactions. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2000; 22(8):831–43.

9. Starner T, Pentland A. Real-time American sign language recognition from video using hidden markovmodels. Proceedings of International Symposium on Computer Vision; 1995 Nov; Coral Gables, FL. p. 265–70.

10. Buxton H, Gong S. Visual surveillance in a dynamic and uncertain world. Artificial Intelligence. 1995; 78(1–2):431–59.

11. Hongeng S, Nevatia R. Multi-agent event recognition. Eighth IEEE International Conference on Computer Vision, ICCV 2001. 2001; Vancouver, BC. p. 84–93.

12. Intille SS, Bobick AF. A framework for recognizing multi-agent action from visual evidence. AAAI '99/IAAI '99 Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence. 1999. p. 518–25.

13. Murata T, Yim J. Petri net methods for reasoning in real-time control systems. IEEE International Symposium on Circuits and Systems. 1995 Apr 30–May 3. p. 517–20.

14. Murata T, Zhang D. A predicate-transition net model for parallel interpretation of logic programs. IEEE Trans Software Eng. 1988; 14(4):481–97.

15. Hura GS. Representation and processing of rule based expert system using petri nets: a viable framework. Proceedings of the 36th Midwest Symposium on Circuits and Systems. 1993 Aug 16–18; Detroit, MI. p. 934–37.

16. Borzin A, Rivlin E, Rudzsky M. Surveillance event interpretation using generalized stochastic petri nets. Eighth International Workshop on Image Analysis for Multimedia Interactive Services, 2007. WIAMIS '07. 2007 Jun 6–8. Santorini.

17. Ghanem N, DeMenthon D, Doermann D, Davis L. Representation and recognition of events in surveillance video using petri nets. 2004 Jun 27.

18. Li R, Zeng B, Liou ML. A new three- step search algorithm for block motion estimation. IEEE Trans Circ Syst Video Tech. 1994; 4(4):438–42.

19. Allen JF. Maintaining knowledge about temporal intervals. Comm ACM. 1983; 26(11):832–43.

20. Burdescu DD, Brezovan M. High level petri nets and rule based systems for discrete event system modelling. International Journal of Smart Engineering System Design. 2001; 3:81–97.

21. Hongeng S, Nevatia R. Multi-agent event recognition. Eighth IEEE International Conference on Computer Vision. ICCV. 2001; 2:84–91.

22. Moenne-Loccoz N, Bremond F, Thonnat M. Recurrent bayesian network for the recognition of human behaviors from video. Lect Notes Comput Sci. 2003; 2626: 68–77.

23. Pinhanez CS, Bobick AF. Human action detection using pnf propagation of temporal constraints. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 1998 Jan.

24. Vu V-T, Bremond F, Thonnat M. Automatic video interpretation: a recognition algorithm for temporal scenarios based on pre-compiled scenario models. Lect Notes Comput Sci. 2003; 2626:523–33.

25. Srinivasan VS, Jafari MA. Fault detection/monitoring using time Petri nets. IEEE Transactions on Systems, Man and Cybernetics. 1993; 23(4):1155–62.