Mining Tool for Surveillance Video

Nagia M. Ghanem, David Doermann, Larry Davis and Daniel DeMenthon

Institute for Advanced Computer Studies (UMIACS)
University of Maryland, College Park, MD 20742
{ghanem,doermann,lsd,daniel}@umiacs.umd.edu

ABSTRACT

This paper describes a system for the mining of surveillance video. Our main contributions are: providing a high level query language for submitting queries about spatial and temporal relations of background regions and moving entities, and about human activities; providing a compiler to map high level queries into a set of novel Petri net filters that utilize computer vision algorithms to answer components of the queries; and providing a powerful graphical interface where users have the ability to formulate the query visually.

Keywords: Petri nets, spatial temporal relations, video query, video surveillance, human activities

1. INTRODUCTION

Automatic analysis and interpretation of human activities have received a great level of attention from the computer vision community in recent years. This is motivated by many real world applications including video surveillance that require continuous observation by human operators. This paper describes a system that is able to answer user's queries about human activities. In response to user queries, the system returns video clips that satisfy the user query, removing any other clips that are not relevant to the query. This can reduce the tediousness of viewing large amounts of video data and improve productivity.

The system is able to answer queries about video data gathered from different surveillance cameras. These cameras are assumed to be fixed, so that the static environment of each scene can be described and used. This description is provided by selecting areas of interest (building entrance, garage entrance, prohibited-parking area, etc.). Queries usually describe scenarios. A scenario is built up using a set of spatial relations, a set of temporal relations and logical operators. Spatial relations can be classified into topological, directional or distance relations.[1] The topology is one of the following: Disjoint, meets, overlaps, equals, contains, inside, covers or covered by. The direction is one of eight directions (N, S, E, W, NE, etc.) A distance relation is a qualitative measure of the distance between two objects. A primitive spatial relation is a combination of a topology and a direction. A temporal relation between two events is one of Allen's thirteen primitive relations[2]: Before, meets, overlaps, during, starts, finishes, their inverses, and equals.

The system should be able to answer queries like the following:

- Identify all cars that park in a given (non-parking) area for more than 10 minutes
- Identify all cars that stop and let out a passenger, who then enters the building.
- Identify all persons that get out of a car and then get into another car within 5 minutes.

The main challenges for the system are as follows. First, we have to cross the semantic gap between the user-provided query and the low-level video features extracted by computer vision techniques such as color, shape, and motion. To overcome this challenge, we propose deriving a Petri net from user query automatically. This Petri net represents the query graphically and can be used in mapping video features detected by lower-level vision algorithms into higher-level semantics. Using Petri nets as an inference mechanism has many advantages, including:

- Petri nets can be used for both deterministic and stochastic inference of event occurrences.
- Petri nets have a nice graphical representation that uses 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.
- 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.
- Petri nets can be used to represent sequentiality, concurrency and synchronization of events.
- 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.
- Compared to classical rule-based expert systems, in terms of efficiency, Petri nets are shown to be more efficient. The RETE algorithm used in most expert systems implementations to improve their speed,[3] is applicable to Petri nets.[4] The main idea is to exploit temporal data redundancies (coming from the markings that are not changed during transition firing). It was shown that Petri nets can improve the use of working memory by splitting it into partitions corresponding to places. Petri nets also reduce the tree sizes used in testing.[5]
- 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 reevaluate past events.

More details about the proposed Petri net models will be given in Section 4. The second challenge is how to improve the productivity of an analyst dealing with large amount of video data. To overcome this challenge, the user can be given the option of selecting levels of recall, which the system will provide with high probability. User's feedback can be used to provide results with higher precision in subsequent stages. The third challenge is how to evaluate the performance of the system under different existing computer vision algorithms. This is done by building the system incrementally, adding modules providing additional levels of knowledge. The system has a plug-in architecture, so that integrating existing code for computer vision algorithms can be done easily. The system performance is reevaluated every time a knowledge level is added.

The rest of this paper is structured as follows. Section 2 summarizes previous work in the area of human activity recognition systems. Section 3 presents the system architecture. In Section 4, Petri nets are introduced and the mapping between different relations and their representations is explained. In Section 5, we provide information about the implemented system and experimental results. Section 6 concludes the paper and presents some ideas for future work.

2. RELATED WORK

Recognition of events from video data can be seen an inference problem, where some inference mechanism is applied to available knowledge to infer the occurrence of these events in the video data. In this section, a survey of these methods is given. Then we discuss how Petri nets are used as an inference mechanism in rule-based expert systems.

There have been many methods that apply deterministic inference to detect events in video data. Most of these methods assume that events can be decomposed into subevents, some of which can be directly detected by perceptual methods, accounting for a variety of temporal constraints. Then constraint propagation algorithms can be used.

In Past-Now-Future networks (PNF-networks),[6] Allen's temporal relationships[2] are used to express parallelism and mutual exclusion between different subevents. Then, Allen's interval algebra network is mapped into

a simpler three-valued PNF-network, to allow fast detection of actions and subactions. The arc consistency algorithm AC-2 is used to propagate the temporal constraints. This algorithm is linear in the number of constraints. But the computation of PNF restriction is NP-hard.

Declarative models[7] are used to describe all the activities (states of the scene, events and scenarios). The activities are described by the conditions between the objects of the scene. Then a classical constraint satisfaction algorithm called Arc Consistency-4 or AC4, is used to reduce the processing time for the process of recognizing activities in video sequences.

To increase the efficiency of processing temporal constraints, Vu et al.[8] suggest that in a preprocessing step, scenario models can be decomposed into simpler scenario models containing at most two sub-scenarios. Then, the recognition of these simpler scenarios just tries to link two scenario instances instead of trying to link together a whole set of combinations of scenario models. However, this method cannot be applied to partially ordered events, where there is no single order of events.

Petri nets have been suggested in by Castel et al.[9] as an inference mechanism to represent the dynamic 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 on 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 activities prototypes and state conditions. Transitions are associated with logical conditions and constraints.

Stochastic inference methods have also been applied successfully to event recognition from video data. Examples include Hidden Markov models, stochastic grammars and Bayesian networks.

Hidden Markov models (HMMs) were chosen to recognize American sign language.[10] HMMs are suitable for recognizing sequential events with different temporal durations but not for activities involving more than one actor. Coupled HMMs (CHMMs) were suggested to alleviate this problem by coupling the states of two HMMs to model interaction between persons.[11] For activities involving more than two persons, the model is complex and the number of parameters is large and difficult to learn from training data.

Stochastic context free grammars (SCFG) are used by Ivanov and Bobick[12] to recognize high-level activities. The limitations of this approach is that representing temporal and spatial relations between events is difficult. Also, inferring the grammar rules and their probabilities for each new domain is difficult.

Bayesian networks have also been used by many researchers. Buxton et al. used Bayesian Belief Network (BBN) for video interpretation in a traffic surveillance application.[13] The system described by Remagnino[14] supplies textual descriptions for dynamic activities occurring in a dynamic scene that include vehicles and pedestrians. But this system does not provide ways to handle situations with more than two objects. Intille and Bobick[15] used Bayesian networks to recognize several activities in a football match. However, no discussion about the learning of the network parameters is provided. Hongeng and Nevatia[16] suggested that for events with logical and temporal relationships between them, Allen's interval-to-interval relations can be used to describe temporal relations between subevents. The recognition is done by propagating temporal constraints and the likelihood degrees of subevents along the event graph. The advantage of this approach is that it can verify and propagate temporal constraints when events are uncertain, while other techniques for constraint satisfaction and propagation techniques usually assume that events and their durations are deterministic. A particular form of dynamic Bayesian networks, Recurrent Bayesian Networks (RBNs), have been used for the recognition of human behaviors through the temporal evolution of their visual features.[17] Although RBNs have the advantage of independence from the time scale of events, the learning problem is tedious and how to represent different temporal and spatial relations is not clear.

Petri nets have been used as an inference mechanism for rule-based expert systems. The rest of this section will survey the use of Petri nets in rule-based expert systems.

In 1987, Sahaoui et al. 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 Petri net representation increases the efficiency of rule-based expert system by providing parallelism and pipelining. Since then, many expert systems were developed that use Petri nets as a knowledge representation that guides the inference process. Examples include work done by Murata and Zhang,[18] Hura,[19] Li[20] and by Murata and Yim.[21]

Researchers have dealt with uncertainty in Petri nets for different purposes. Stochastic Petri nets,[22,23] Fuzzy Petri nets[24–27] and Possibilistic Petri nets[28] are examples of different types of Petri nets dealing with uncertainty.

The issue of efficient implementation of Petri nets has also been addressed by many researchers. The RETE algorithm has been applied to reduce the complexity of Petri nets and to achieve a linear performance in the number of knowledge base rules.[3] It has also been shown that Petri nets can improve the use of working memory by splitting it into partitions corresponding to places. Petri nets also reduce the tree sizes used in testing.[5]

3. SYSTEM ARCHITECTURE

Figure 1 shows the proposed system architecture. The system is designed to answer queries about surveillance video data gathered from different cameras. We formulate the system into an expert system framework. Therefore, a mapping between the system components and expert system components can be defined as follows:

- **The Abstract Knowledge Base**, or rule base, refers to statements of general validity that describe object types, states, primitive events and the structure of composite events. This information can be edited by the **Knowledge Editor Module**, to add, modify or delete certain pieces of information.
- **The Concrete Knowledge Base**, or fact base, refers to information about a particular scene, its zones of interest, objects in the scene, their trajectories, their speed, etc. The data gathered from different cameras are processed by an **Intermediate Vision Layer** that provides a Geometric Scene Description (GSD). A GSD is a quantitative object-level scene interpretation in terms of recognized objects and their (possibly varying) locations in the scene.
- **The User Interface Module** provides a query formation interface, a result display interface, and a knowledge-editor manager. In the query formation interface, for each surveillance camera, a static image representing its view is provided. The user can then select a view and mark areas of interest by drawing rectangles around these areas and giving them identifiers. Then, using the different menus, the user can design new queries by describing the events to be detected. Object types, object states, primitive events and previously defined composite events can be used in describing the new query. Additionally, query results can be explored through this interface where user feedback can be provided, and the query can then be reformulated and applied to the results of a prior query.
- **The Inference Engine** draws conclusions by applying the rules to the facts. This can be done by mapping the user query into a set of Petri nets that can use the GSD to infer high-level semantics and provide the query results in the form of video clips that satisfy the user query.

4. PETRI NETS REPRESENTATIONS

4.1. Petri Nets Basics

An ordinary Petri net is a bipartite graph with two types of nodes, places and transitions. In an ordinary Petri net, places hold tokens that can be used for firing transitions immediately as they are available. Transitions can fire only if every input place has a token. Firing a transition removes tokens from input places and insert tokens in output places. Figure 2 shows a Petri net with one transition. The transition has two input places and two output places. It is shown before and after the firing.

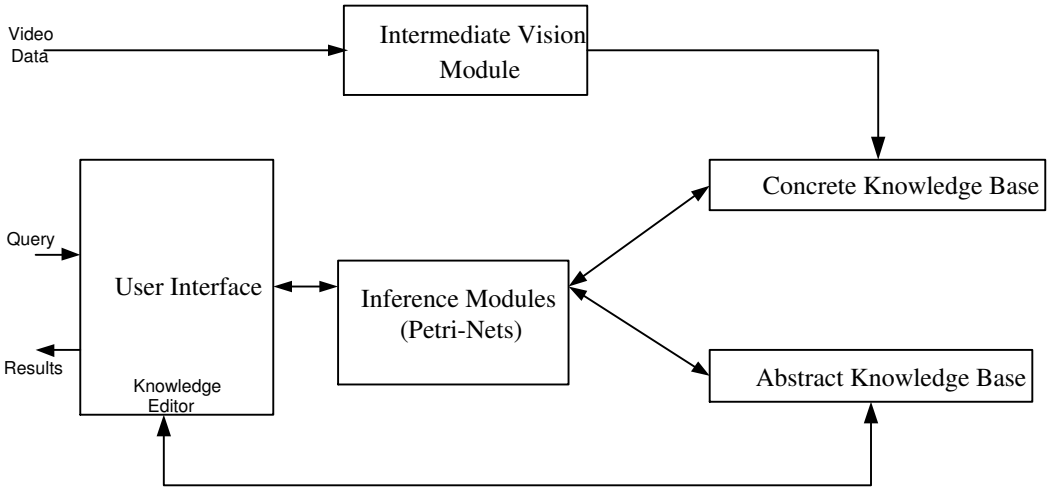

Figure 1. System Architecture

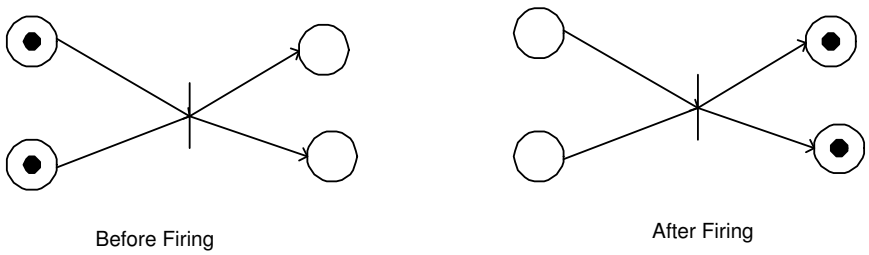

Figure 2. Simple Petri Net Before and After Firing

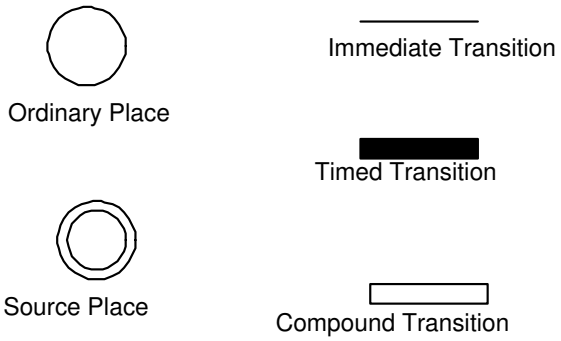

Figure 3. Graphical Notations for Petri Net Elements

4.2. Petri Nets Extensions

To be able to use Petri nets in our system, some enhancements to ordinary Petri nets are required. Some of these enhancements have been proposed in the literature but here we are trying to select the ones that are useful to our needs and add new enhancements. We propose the following extensions:

- **Places:**
  - **Ordinary Places:** The same as places in ordinary Petri nets. An ordinary place is represented by a single-line circle.
  - **Source Places:** Tokens are inserted by external modules, either at constant time intervals or at a variable rate. These places can act an input places for one or more transition but not as an output place. Source places are represented by double-line circles.
- **Transitions:**
  - **Immediate Transitions:** The same as transitions in ordinary Petri nets. This means that the transition fires immediately when every input place has the required tokens for firing.The transition may also have additional firing conditions that should be satisfied for the transition to fire. For example, tokens from different input places should relate to each other in some way (e.g. having same identity). An immediate transition is represented by a thin bar.
  - **Timed Transitions:** A timed transition is enabled if its input places have the required tokens and the firing rules are satisfied, but it may fire only after a given amount of time has elapsed. This time may be a constant or variable depending on the situation. In this case, the transition is said to be timed. Timed transition is represented by a filled-rectangle.
  - **Compound Transitions:** A previously defined Petri net can be reused in constructing new nets. Instead of redrawing it everytime it is reused, it can be represented by a compound transition. This also simplifies the graphical representation. A compound transition is represented by a unfilled rectangle.
- **Tokens:** Tokens can have identities (or colors) that represent different entities and can also hold information about these entities.

4.3. Relation Representations

4.3.1. Spatial relations

The Petri net for a spatial relation consists of only one spatial place and has no transitions. Only the external module that detects the existence of this spatial relation can place tokens in this place. A token is inserted in this place for the first video frame where this spatial relation is detected. The token is updated for every following frame where this spatial relation is still holding. For the first frame, where this spatial relation is no more satisfied, the lifetime of this token is computed. If it is in the range [minDur, maxDur] then it is unlocked and can be used for firing transitions, else it is either locked or dead and cannot be used for firing transitions. [minDur, maxDur] is the range into which the duration of the spatial relation should fall.

4.3.2. Temporal relations

Assuming the input arguments for a given temporal relation are intervals represented by places, then the temporal relation is represented by a transition whose input places are the arguments places and whose output is an ordinary place. The transition is augmented by conditions that should be satisfied for this temporal relation to hold. For example, to represent event A should occur "before" event B, then the condition on the transition is $EA < SB$, where a letter "S" before event name means the start time of the event and a letter "E" before the event name means the end time of the event. Other quantitative measures can be added to the temporal relations representations. For example, if we want to say event A should be before event B with the difference between them is no more than 5 minutes, the conditions on the transition representing the temporal relation are modified to reflect these new constraints. In this case the modified condition is" $EA < SB$ AND $SB - EA < 5$ minutes".

Figure 4.4 shows the representation for the temporal relation " A before B". Other relations have the same representation with matching conditions on the transition.

4.3.3. Logical relations

Figure 4.4 shows Petri nets representing the logical relation AND, OR and NOT. Here, we assume the operands are represented by places. The AND is represented by a transition, whose input places are the operands. The transition will fire only if every input place has one token. The OR is represented by a place. Every operand place will also have a transition that fires if this place has a token. Any of these transitions can place a token in the place representing the result. The NOT is represented by a transition, whose input place is the operand. But the arc connecting them is inhibitor, which means that the transition will fire if the input place does not have tokens.

4.4. Event Representations

Spatial, temporal and logical relations are the main constituents of events. A primitive event is defined as the occurrence of a spatial relation. A composite event may involve primitive events connected by spatial, temporal or logical relations. Many actors may be involved in the same composite event. Examples are given in the next section.

5. IMPLEMENTATIONS AND EXPERIMENTAL RESULTS

The system was implemented in Java for multi-platform portability.

Figures 5 5 show the query formulation interface where users have the ability to formulate the query visually. For each surveillance camera, a static image representing its view is displayed to the user. The user can then select a view and mark areas of interest by drawing rectangles around these areas and giving them IDs. For describing different spatial relations, we assume an area-based approach - the user selects an actor (car or person)

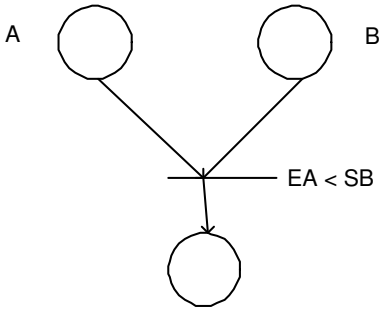

Figure 4. Temporal Relations Representation

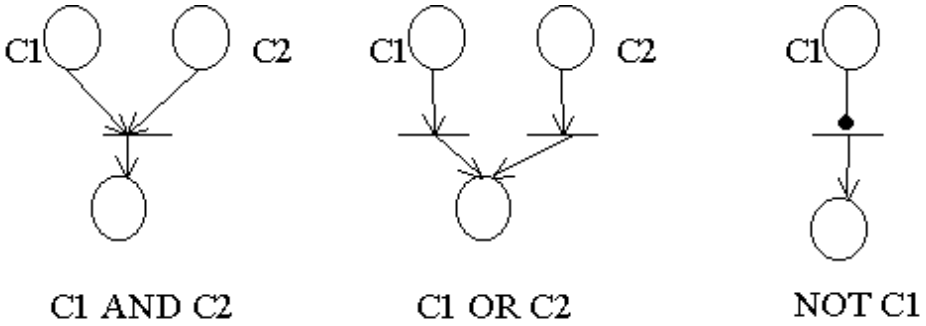

Figure 5. Logical Relations Representation

and a (previously marked) area of interest and then selects the topology and the direction for the spatial relation from a menu. The menu items depend on the position of the mouse click with respect to the area of interest (inside, on border or outside). Temporal relations can then be defined. The arguments for a temporal relation are either spatial relations or (previously defined) temporal relations. The operand for a temporal relation can be one of the thirteen Allen's temporal operands defined earlier. The final query can then be formulated by combining spatial and temporal relations using logical operators like AND, OR and NOT.

We are utilizing existing source code pieces to provide the system with Geometric Scene Description(GSD). One example that has been implemented is using the source code for the tracking algorithm developed by Elgammal[29] to provide object tracks over time.

Users can display the results for a given query so that he may reformulate the query or perform a refined query on the preliminary results. For this module, we are leveraging VideoMiner, a Java-based tool developed at our lab for displaying metadata about video data. We are extending its functionality to allow for user feedback. A query can be reformulated and applied on the results of a prior query.

In the following, a query example is given and the corresponding Petri net constructed by the program is presented.

Query Example

Identify all cars that stop and let out a passenger.

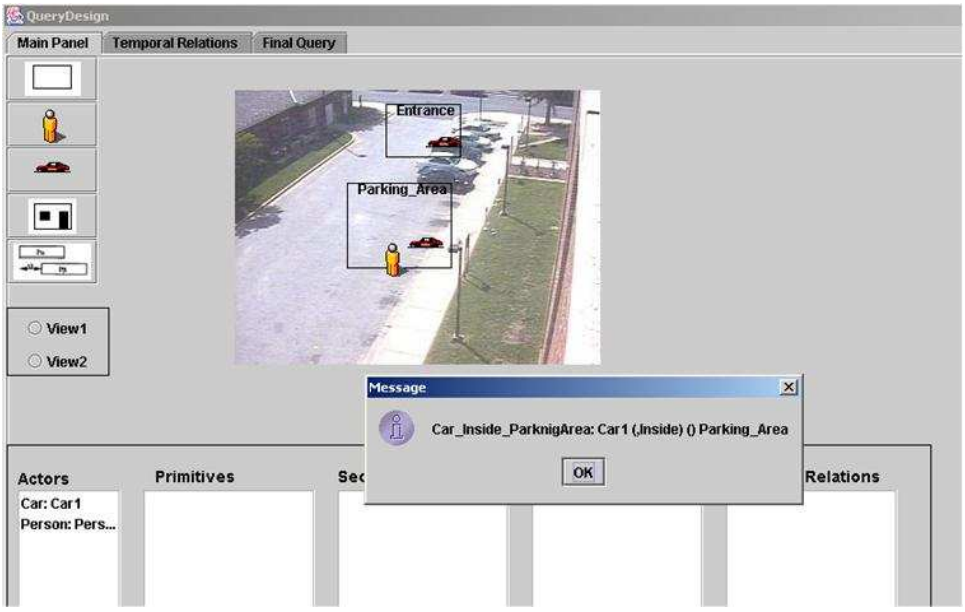

Figure 6. Program Interface, Main Panel

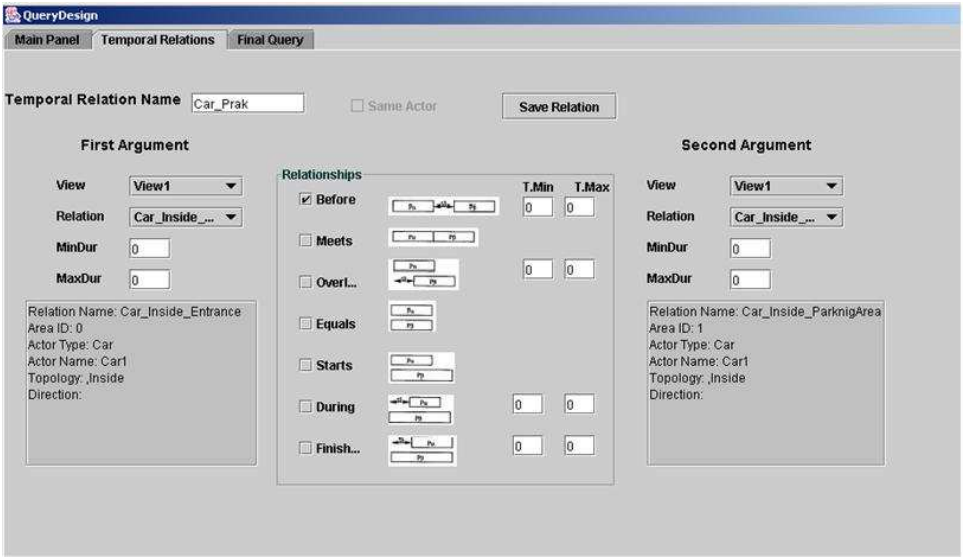

Figure 7. Program Interface, Temporal Relations

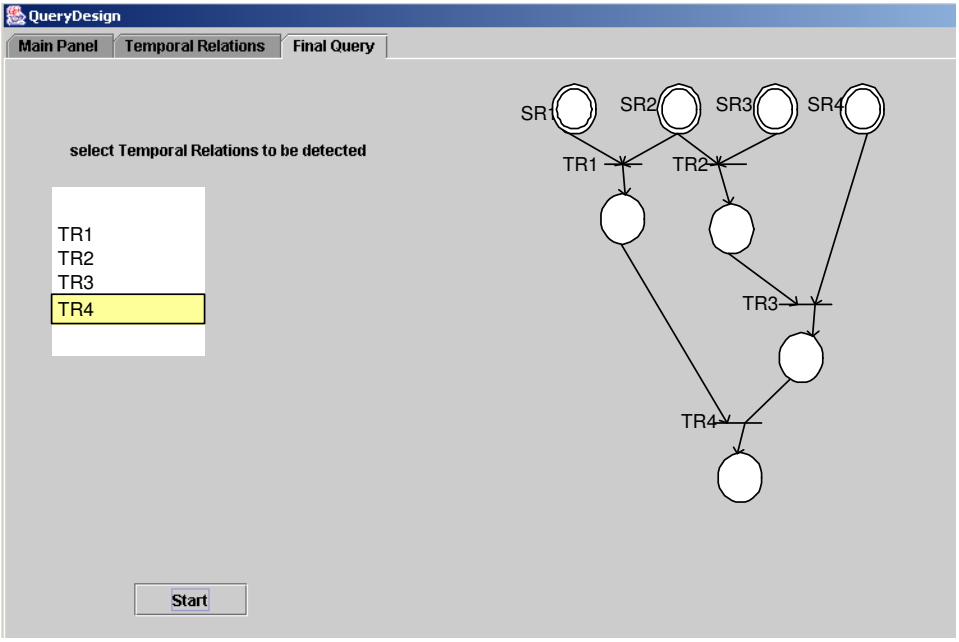

Figure 8. Generated Petri Net for the Given Example

First, we define two areas: Entrance, Parking-Area.
Second, the spatial relations involved are: SR1- Car-Inside-Entrance, SR2- Car-Inside-ParkingArea, SR3- Person-inside-ParkingArea and SR4- Person-Outside-ParkingArea.
Third, the temporal relations involved are: TR1- SR1 "before" SR2, TR2- SR3 "during" SR2 and TR3- TR2 "before" SR4.
Figure 5 shows the corresponding Petri net generated by the program. Each spatial relation is represented by a source place and each temporal relation is represented by a transition. A token in place $P_T R3$ means that the scenario has been identified.

6. CONCLUSIONS AND FUTURE WORK

We have described a system for the mining of surveillance video. The system provides a powerful graphical interface for submitting queries about spatial and temporal relations of background regions and moving entities, and about human activities. A mapping from different relation types into a set of Petri net filters is given. System implementation and a query example are discussed. However, the method described here does not take handling uncertainty inherent in video data into account. We are currently investigating how to incorporate probabilistic inference in Petri nets. Also, a complete ontology for event recognition should be defined to provide an even more powerful query language.

REFERENCES

1. S. Kaushik, E. Rundensteiner, "Direct Manipulation Spatial Exploration Using SVIQUEL," *Visual Database Systems Conference* , pp. 179–185, 1998.
2. J.F. Allen, "Maintaining Knowledge about Temporal Intervals," *Communications of the ACM* **26**, pp. 832–843, November 1983.

3. C. L. Forgy, "RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem," *Artificial Intelligence* **19**, pp. 17–37, 1982.
4. D.D. Burdescu and M. Brezovan, "High Level Petri Nets and Rule Based Systems for Discrete Event System Modelling," *International Journal of Smart Engineering System Design* **3**, pp. 81–97, 2001.
5. P.R. Muro-Medrano, J.A. Banares and J.L. Villarroel, "Knowledge Representation-Oriented Nets for Discrete Event System Applications,"
6. C.S. Pinhanez and A.F. Bobick, "Human Action Detection Using PNF Propagation of Temporal Constraints," *CVPR* , January 1998.
7. N. Rota and M. Thonnat, "Activity Recognition from Video Sequences using Declarative Models," *ECAI 2000* , pp. 673–680, 2000.
8. V. Vu, F. Bremond and Monique Thonnat, "Automatic Video Interpretation: A Recognition Algorithm for Temporal Scenarios Based on Pre-compiled Scenario Models," *ICVS* , pp. 523–533, 2003.
9. 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, Workshop on conceptual descriptions from images, Cambridge UK* , 1996.
10. T. Starner and A.Pentland, "Real-time American Sign Language Recognition from Video Using Hidden Markov Models," *Proceedings of International Symposium on Computer Vision* , pp. 265 –270, November 1995.
11. N. Oliver, B. Rosario and A. Pentland, "A Bayesian Computer Vision System for Modeling Human Interactions," *Proceedings of Intl. Conference on Vision Systems ICVS99. Gran Canaria. Spain.* , January 1999.
12. Y.A. Ivanov and A.F. Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**, pp. 852–872, 2000.
13. H. Buxton and Shaogang Gong, "Visual Surveillance in a Dynamic and Uncertain World," *Artificial Intelligence* **78**(1-2), pp. 431–459, 1995.
14. P. Remagnino, T. Tan and K. Baker, "Agent Oriented Annotation in Model Based Visual Surveillance," *ICCV, 4-7 January 1998,Bombay, India* , pp. 857–862, January 1998.
15. S.S. Intille and A.F. Bobick, "A Framework for Recognizing Multi-Agent Action from Visual Evidence," *AAAI/IAAI* , pp. 518–525, 1999.
16. S. Hongeng and R. Nevatia, "Multi-Agent Event Recognition," *ICCV* , pp. 84–93, 2001.
17. N. Moenne-Loccoz, F. Bremond and M. Thonnat, "Recurrent Bayesian Network for the Recognition of Human Behaviors from Video," *ICVS* , pp. 68–77, 2003.
18. T. Murata, D. Zhang, "A Predicate-Transition Net Model for Parallel Interpretation of Logic Programs," *IEEE Transactions on Software Engineering* **14**(4), pp. 481–497, 1988.
19. G.S.Hura, "Representation and Processing of Rule-Based Expert System Using Petri Nets: A Viable Framework ," *Proceedings of the 36th Midwest Symposium on Circuits and Systems* **2**, pp. 934 –937, 1993.
20. L. Li, "High-level Petri Net Model of Logic Program with Negation ," *IEEE Transactions on Knowledge and Data Engineering* **6**, pp. 382 –395, 1994.
21. T. Murata,J. Yim, "Petri net Methods for Reasoning in Real-Time Control Systems," *1995 IEEE International Symposium on Circuits and Systems* **1**, pp. 517 –520, 1995.
22. P. Kemper, "Transient Analysis of Superposed GSPNs," *IEEE Transactions on Software Engineering* **25**, pp. 182 –193, March-April 1999.
23. C. Lin, D.C. Marinescu, "Stochastic High-Level Petri Nets and Applications," *IEEE Transactions on Computers* **37**, pp. 815 –825, July 1988.
24. C. G. Looney, "Fuzzy Petri Nets for Rule-Based Decisionmaking," *IEEE Transactions on Systems, Man and Cybernetics* **18**, pp. 178 –183, Jan.-Feb. 1988.

25. S.M. Chen, J.Sh. Ke, J.F. Chang, "Knowledge Representation Using Fuzzy Petri Nets ," *IEEE Transactions on Knowledge and Data Engineering* **2**, pp. 311 –319, September 1990.
26. A. Konar,A.K. Mandal, "Uncertainty Management in Expert Systems Using Fuzzy Petri Nets," *IEEE Transactions on Knowledge and Data Engineering* **8**, pp. 96 –105, Febraury 1996.
27. H. Scarpelli, F. Gomide and R.R. Yager, "Reasoning Algorithm for High-Level Fuzzy Petri Nets," *IEEE Transactions on Fuzzy Systems* **4**, pp. 282 –294, August 1996.
28. J. Cardoso, R. Valette and D. Dubois, "Possibilistic Petri Nets ," *IEEE Transactions on Systems, Man and Cybernetics B* , pp. 573 –582, October 1999.
29. A. Elgammal, R. Duraiswami and L. S. Davis, "Efficient Computation of Kernel Density Estimation using Fast Gauss Transform with Applications for Segmentation and Tracking," *Second International Workshop on Statistical and Computational Theories of Vision, Vancouver, Canada* , 2001.