Modeling and recognition of video events with Fuzzy Semantic Petri Nets *

Piotr Szwed

AGH University of Science and Technology
pszwed@agh.edu.pl

Abstract. This paper addresses the problem of modeling and automated recognition of complex behavior patterns in video sequences. We introduce a new concept of Fuzzy Semantic Petri Nets (FSPN) and discuss their application to recognition of video events. FSPN are Petri nets coupled with an underlying fuzzy ontology. The ontology stores assertions (facts) concerning classification of objects and detected relations. Fuzzy predicates querying the ontology content are used as guards of transitions in FSPN. Tokens carry information on objects participating in a scenario and are equipped with weights indicating likelihood of their assignment to places. In turn, the places correspond to scenario steps. The Petri net structure is obtained by translating a Linear Temporal Logic formula specifying a scenario in a human-readable form. We describe a prototype detection system consisting of an FSPN interpreter, the fuzzy ontology and a set of predicate evaluators. Initial tests yielding promising results are reported.

Keywords: Petri nets, fuzzy ontology, event recognition, temporal logic

1 Introduction

Automatic recognition of complex behavior patterns in analyzed video sequences is a challenging area in computer vision. Such patterns, commonly referred as scenarios or events, can be perceived as combinations of simpler events describing interactions between objects that are either detected and tracked, or predefined as components of a scene configuration. Practical implementations of event recognition systems face two problems [13]. The first is related to methods used for extraction of features, which are further used to recognize and discriminate events. The methods are often delivering uncertain or noisy data. The second problem is an approach to scenario modeling. Definitions of scenarios, to be meaningful and manageable, should preferably be decoupled from a software implementation and use semantic description of objects and their relations. In

* This is the draft version of the paper presented at the **KICSS 2013**, **8th International Conference on Knowledge, Information and Creativity Support Systems**, November 7–9, 2013 Kraków, Poland.

particular, such semantic information should be communicated to system operators in case of intelligent video surveillance systems or used as a video metadata when applied in automated video indexing engines.

In this paper we introduce a new concept of Fuzzy Semantic Petri Nets (FSPN) and discuss their application to an automated analysis of video surveillance scenarios. FSPN are Petri nets coupled with an underlying fuzzy ontology. The ontology, apart from defining a terminology, i.e. names of classes and relations, stores assertions (facts) concerning classification of objects and detected relations between them. These assertions can be queried with unary or binary predicates returning fuzzy truth values from $[0, 1]$. Predicates are used in guards of transitions in FSPN controlling in that way flows of tokens. Tokens carry the information on objects participating in a scenario and are equipped with fuzzy weights indicating likelihood of their assignment to places. In turn, places correspond to scenario steps.

The conceptual layout of a proposed video surveillance system is depicted in Fig. 1. We focus on components marked in gray: *Fuzzy Ontology* constituting an intermediate abstraction layer between description of tracked objects, *Scenario Specification* expressed in Linear Temporal Logic (LTL) and *Fuzzy Semantic Petri Nets*, which are obtained by translating LTL formulas.

The proposed approach stems from an observation that in the domain of formal software verification LTL has been successfully applied to specify and check temporal requirements pertaining to behavior of concurrent programs. Hence, an idea of applying LTL to the detection of video events occurrences. However, in opposition to models of programs, whose behavior can be observed with the 100% accuracy, results of video content analysis are inherently uncertain, as they are calculated in several steps, including background subtraction, object recognition, classification and tracking. Each of them may produce small cumulating errors. Moreover, semantic scenario specifications may introduce vagueness, by referring to such terms as *near*, *walking*, *running*, which can be differently understood and defined. Fuzziness is the proposed mean to manage uncertainty and vagueness.

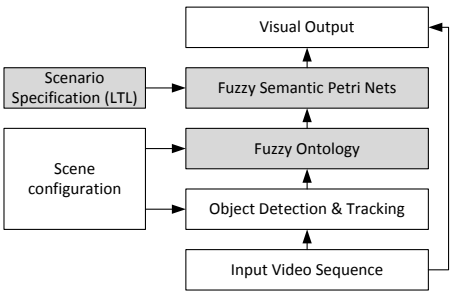


Fig. 1: Conceptual model of a system for semantic video events analysis

The paper is organized as follows: the next Section 2 reports known approaches to the specification and analysis of events. Section 3 discusses application of Linear Temporal Logic in this field. It is followed by Section 4, which describes the fuzzy ontology. FSPN are defined in Section 5. A system implementing the proposed approach is presented in Section 6 and finally Section 7 gives concluding remarks.

2 Related works

Recognition of video events has been intensively researched over last fifteen years. A large number of methods is reported in recent surveys: [13] and [2]. Systems for video recognition usually have a layered architecture, e.g. [8, 17], in which lower level layers provide an *abstraction* of meaningful aspects of video sequences, whereas higher level layers are related to formalisms used for *event modeling* and algorithms that detect events or scenarios based on formal specifications. The greatest diversity can be observed in approaches to event modeling .

Probabilistic state-based methods use models comprised of states and transitions, in which transitions are attributed with probability factors learned from annotated video. During the analysis of an input video sequence a likelihood of a situation is computed. This group include methods based on neural networks [5], Hidden Markov Models, Dynamic Bayesian Networks [1] and stochastic Petri nets [14].

In grammar based methods complex activities are represented by production rules that generate strings of atomic actions. Hence, complex events can be recognized by language parsing techniques [11]. In the review [2] a limitation of these methods as regards concurrent activities was indicated. The criticism seems to be founded in a case, where sequences of single actions are analyzed. However, in a more general setting, e.g. this provided by the Kripke structure [12], each string element is a set of low level events occurring parallelly and, in consequence, the concurrent events can be tracked (see Section 3).

Description based approaches specify events and scenarios using high level languages, either textual [19], or graphical as Situation Graph Trees [18, 17] and Petri nets [10, 4, 14]. The methods falling into this category are considered *semantic*, as specifications are prepared by experts, who give meaningful names to events, engaged objects, actions and conditions. Descriptions are often hierarchical: complex events can be expressed as graphs of subevents. Models also may include constraints and knowledge about scene objects, e.g. in [17] they are expressed as formulas of a Fuzzy Metric-Temporal Horn Logic. In some approaches events and their ingredients: types of participating objects and relations are defined as ontologies [7, 3].

Petri Nets (PNs) are applied in the field of event detection in two modes [13]. In the first mode of *object PNs* tokens represent objects, places object states and transitions events of interest. Such approach was applied to the surveillance of traffic [10] and people [6]. In the second mode of *plan PNs* places correspond to subevents building up a plan. A presence of a token in a place indicates that

a particular event assigned to the place is occurring. The latter approach was applied in [4] to people surveillance.

3 Scenario specification with temporal logic

Temporal logic [16] is a symbolic language allowing to express temporal (unquantified) relationships between events or conditions. Formulas in temporal logic are constructed from propositions linked by classical logic operators and *temporal operators*. For Linear Temporal Logic (*LTL*), most often used operators are: $\Box q$ (the formula q is *always* true starting from a certain moment in time) and $\Diamond p$ (the formula q will *eventually* became true in the future).

To give a simple example, the temporal formula $a \Rightarrow \Diamond b \Rightarrow \Box c$ specifies, that at the beginning a is satisfied, then b happens, and finally c becomes always true.

Semantics of LTL is defined with a model called *Kripke structure* [12], which can be defined formally as a tuple $K = (Props, S, T, s_0, L)$, where $Props$ is a set of atomic propositions, S is a set of states, $T \subseteq S \times S$ is a flow relation, s_0 is an initial state and $L\colon S \to 2^{Props}$ is a function, that assigns *sets* of true propositions to states. For LTL the flow relation T together with the initial state s_0 defines a linear sequence of states (worlds): $s_0, s_1, s_2, \ldots, s_n, \ldots$.

In case of a formal verification conducted by model checking, the states represent snapshots of program memory in consecutive time moments. For the intended application of LTL to specify surveillance scenarios, the sequence of states corresponds to a video sequence or, more precisely, to the sets of objects recognized in particular frames, their properties and relations. Hence, in formal specifications of events we replace propositions by unary and binary predicates in Fuzzy Description Logics [15] querying those relations.

We will discuss the scenario specification on an example of graffiti painting event shown in Fig. 2. The event develops in the following steps (subevents): (i) *init*: a person appears on a scene, (ii) *move*: the person moves towards the wall, (iii) *front*: the person is in front of the wall, (iv) *appear*: a graffiti appears on the wall, (v) *remain*: the graffiti remains on the wall.

init move front appear remain

Fig. 2: Graffiti event steps

The scenario of the event can be formalized as LTL formula comprised of propositions corresponding to subevents (1). It should be noted that to some

extend these lower-level events can overlap in time, e.g. when a graffiti appears the person is probably still in front of the wall.

$$init \Rightarrow \Diamond move \Rightarrow \Diamond front \Rightarrow \Diamond appear \Rightarrow \Box remain \tag{1}$$

In the next stage, the initial scenario specification is refined by replacing propositions with conjunctions of unary and binary predicates. During the recognition, their arguments are bound with detected objects or elements of the scene configuration. The target scenario specification for the discussed event is given by the formula (2).

$$\begin{aligned} &Person(p), atBorder(p), Wall(w) \\ &\Rightarrow \Diamond(movesTowards(p,w))_{\{3,\infty\}} \\ &\Rightarrow \Diamond(inFrontOf(p,w))_{\{10,\infty\}} \\ &\Rightarrow \Diamond(newObject(g), inside(g,w), notInsideSomeWindow(g))_{\{3,\infty\}} \\ &\Rightarrow \Box(inside(g,w), isStill(g), notInsideSomeWindow(g))_{\{3,\infty\}} \end{aligned} \tag{2}$$

The scenario references three objects: p of type $Person$, w of type $Wall$ and g, an object that is introduced i 4 th step. It is required that g appears inside the wall w, but not inside a window. The specification relies on classification of objects: p is a $Person$, their relations, e.g.: graffiti g is inside a wall w, and a scene configuration that should provide information on shapes of walls and windows. The later for the discussed example is given in Fig. 3. Integer numbers in curly brackets define time intervals $[t_l, t_h]$. A subevent should last at least t_l time units to be accepted as a scenario step occurrence. Then it can be reported, but also enable the transition to a next subevent. Compound events older then t_h are ignored.

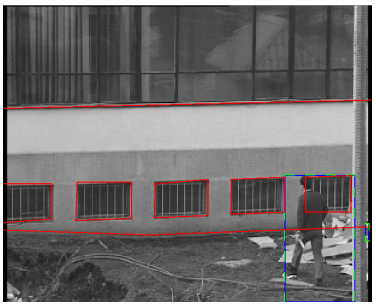


Fig. 3: Scene configuration: the wall and the windows boundaries are marked

4 Fuzzy ontology

The fuzzy ontology constitute an intermediate layer between information on tracked objects and fuzzy Petri nets. Whereas objects within the tracking model are described with numeric values, like size, distance or speed, the ontology provide a kind of linguistic abstractions, e.g. *a person*, *an object is inside other object* or *a person is in front of other object*.

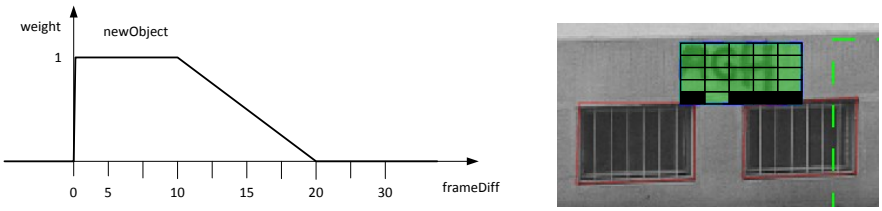
Ontologies are often described as unions of two layers: terminological: *TBox*, which comprises concepts and relation types (including taxonomic relations between concepts) and assertional: *ABox* gathering facts about individuals and their relations. For *fuzzy ontologies* and corresponding Fuzzy Description Logics these relations are extended by adding weights being real numbers from $[0, 1]$. They can be used to express uncertainty, e.g. with respect to class membership or relation occurrence. The formalization of fuzzy ontology language including fuzzy classes, roles (object properties) and datatypes can be found in [15].

In the presented solution the TBox is limited to fuzzy concepts, like *Person*, *Wall*, taxonomic relations and object properties: *inside*, *notInsideSomeWindow*. The ABox is comprised of individuals including tracked objects and predefined scene objects, fuzzy class membership relations that are represented by unary predicates returning values from $[0, 1]$, e.g. $Person(x)$ and asserted fuzzy relations between individuals: $inFrontOf(x, y)$. It should be noted that a unary predicate describing property of an object, e.g. $isStill(x)$ can be considered equivalent to the class membership axiom: $x \in isStill$.

The ontology supports queries for objects present in ABox and their relations. Assertions on relations in ABox are made with *evaluators*, functions (or more precisely function object in an object-oriented implementation) that examine object model and calculate fuzzy weights of predicates. In opposition to approach proposed in [15] evaluators are external entities beyond the ontology. This allows greater flexibility in their construction. In many cases they have a form of fuzzy membership functions described by line segments, similar to these discussed in [18], but they can be also based on other features, as Jaccard metrics applied to object areas.

The predicate $newObject(g)$ references temporal information stored in the underlying frame sequence model. Its evaluator shown in Fig. 4a. uses a membership function that takes as the argument the difference between current frame number and the frame, in which the object g appeared.

The predicate $inside(x, s)$ (Fig. 4b.) divides the object x into a grid of cells and calculates how many of them overlaps with a scene object s. It is used internally by the predicate $notInsideSomeWindow(x)$, which denotes a class of objects satisfying the axiom $\neg\exists w \cdot Window(w) \sqcap inside(x, w)$. The condition was introduced to prevent from classifying as objects left on the wall visual changes occurring inside the windows, e.g. reflections on the glass, window opening, people moving behind. In this case the predicate value is calculated by the implemented reasoner according to the formula: $1 - \max\{w \in Window : inside(x, w)$. For the example in the Fig. 4b. the evaluator $notInsideSomeWindow(g)$ yields the value $22/25 = 0.88$.



a. b.

Fig. 4: Examples of evaluators: (a) newObject(x) (b) inside(x,s)

5 Fuzzy Semantic Petri Nets

Classical method of verification, whether a sequence of worlds satisfy an LTL formula, consists in converting it into more manageable representation, namely a Büchi automaton [9]. Such automaton accepts infinite sequence of symbols, which can be considered as subsets of logical propositions having true value in the subsequent worlds forming a semantic model.

In the presented approach we use Fuzzy Semantic Petri Nets as a tool for scenario analysis. The nets have a typical structure of Büchi automata, however, they can process multiple tokens. This feature is particularly important, because it allows to reason about overlapping scenario occurrences, in which participate various combinations of objects.

To manage uncertainty of an input data and vagueness of specifications, fuzzy predicates returning values from $[0, 1]$ are used as transition guards. These values are then combined with the weights of tokens flowing through a net. Tokens, in turn, represent scenario occurrences. This enables monitoring the scenario steps and reasoning about their likelihood. Moreover, sequences of accepted states strictly defined with LTL formulas can be to some extent interleaved with states not satisfying the specified conditions. In such case, the weight of a token expressing the scenario likelihood gradually decrease and, after passing a certain threshold, the token can be removed.

Formal definition of FSPN comprises three concepts: Petri net structure, a binding and fuzzy marking. We start with some auxiliary definitions. Unary predicate is defined as a pair (n, v_s) where, n is a predicate name and v_s is a variable name referring to a *subject* of the predicate. Binary predicate is a triple (n, v_s, v_o); the variable v_o is a predicate *object*. Set of all unary and binary predicates is denoted by $Preds$. By $Vars(p)$ we denote a set of variables appearing in the predicate p. Analogously, for a set $C \subseteq Preds$ we define $Vars(C)$, as $\bigcup_{p \in C} Vars(p)$.

Definition 1 (Petri net structure).
Petri net PN is a tuple $(P, T, F, Preds, G, L, H)$, where P is a set of places, T is a set of transitions, P and T are satisfying $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$. $F \subseteq P \times T \cup T \times P$ is a set of arcs (flow relation), and $Preds$ is a set of unary

and binary predicates. $G\colon T \to 2^{Preds}$ *is a guard function that assigns sets of predicates to transitions.* $L\colon P \to \mathbb{N} \cup \{0\}$ *is a function assigning lower bound to a place; this value defines how long a token should stay in a place to be allowed to leave it.* $H\colon P \to \mathbb{N} \cup \{\omega\}$ *assigns upper bound to a place. The symbol* ω *represents infinity.*

The set of input places for a transition $t \in T$ is denoted as $\bullet t = \{p \in P\colon (p,t) \in F\}$ and the set of output places as $t\bullet = \{p \in P\colon (t,p) \in F\}$

Definition 2 (Binding). *Let* V *be set of variables and* I *a set of objects.* Binding b *is defined as a partial function from* V *to* I*. A variable* v is bound *for a binding* b*, iff* $v \in \mathbf{dom}\, b$*. A set of all bindings is denoted by* B*.*

Let $p \in Preds$ a predicate and $b \in B$ be a binding. Predicate value for a binding $val\colon Preds \times B \to [0,1]$ is a function that assigns value from the interval $[0,1]$ to a pair (p,b). If $Vars(p) \setminus \mathbf{dom}\, b \neq \emptyset$, then $val(p,b) = 0$.

Definition 3 (Fuzzy marking). *A set of fuzzy tokens* FT *is defined as* $FT = B \times \mathbb{R} \times (\mathbb{N} \cup \{0\}) \times (\mathbb{N} \cup \{0\})$*. Components of a token tuple* $(b, w, c, \tau) \in FT$ *are the following:* $b \in B$ *denotes a binding,* $w \in [0,1]$ *is a fuzzy weight,* $c \geq 0$ *is a counter storing information, how long the token rests in a place and* τ *is a time stamp. For a Petri net* $PN = (P, T, F, Preds, G)$ fuzzy marking $FM : P \to 2^{FT}$ *is defined as a function that assigns sets of fuzzy tokens to places of the net.*

The definition of FSPN is too general with respect to the structure that is required to represent a scenario expressed with an LTL formula. In fact, we use state machines, i.e. Petri nets satisfying $|\bullet t| \leq 1$ and $|t\bullet| = 1$ for each $t \in T$.

Fig. 5 gives an example of a net representing the formula (2). Logical conditions appearing in subformulas become guards for two transitions: the first leading to a place and a self-loop (a transition, for which $\bullet t = t\bullet$ holds). The net in Fig. 5 represents a particular simple case. Translation of more complex formulas, e.g. containing disjunctions, may result in multiple transitions linking places or forks.

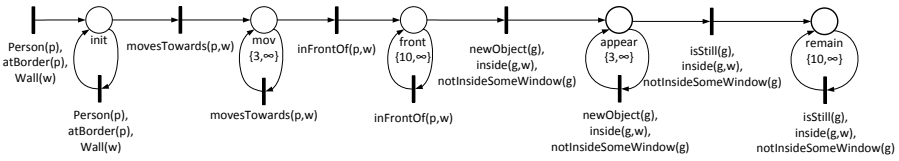


Fig. 5: Fuzzy Semantic Petri Net representing the graffiti painting event

The behavior of FSPNs defined in previous section differs from the standard semantics for Petri nets, as they are not intended analyze such issues as concurrency or conflicts, but to perform a kind of fuzzy reasoning and classification of sequences of events. A single step of FSPN execution is comprised of three basic stages:

1. *Firing enabled non-initial transitions and generating new tokens.* During this stage for each pair consisting of a transition t an a token ft in its input place, the transition guard is evaluated, then aggregated with the token weight and assigned to a new token ft' introduced to the transition's output place. The new token obtains a timestamp equal to the iteration number. There are, however, some variations to the above procedure: new tokens must have weight above a certain threshold (we used 0.25 in experiments); in the case, where the transition guard contains a free variable, it must be bound to an object in the ontology.
2. *Removing old tokens.* It is assumed, that creation of a new token ft' from ft consumes a portion of its weight. If this value falls below a certain threshold, the token ft is removed. Also in this step, multiple tokens sharing the same binding and assigned to the same place are aggregated.
3. *Firing initial transitions.* Finally, new tokens are introduced into the net, by firing initial transitions (i.e. satisfying $\bullet t = \emptyset$). For each initial transition variables appearing in its guard are bound to objects, then the guard value is calculated and used as a weight of new tokens. A threshold (0.2) preventing from creation of tokens with a small weight is used, as well as there is implemented a mechanism, which does not allow introducing tokens with a binding already present in the net.

The semantics of Petri nets proposed in this paper is close to referenced in Section 2 *plan PNs*, as tokens represent combination of objects participating in scenarios. There are, however, some salient differences. 1) In probabilistic PNs discussed in [4] in case of a conflict (e.g. two enabled transitions sharing input place with a single token) only one transition with a higher learned probability would fire, whereas in our model they both can be executed and produce two tokens.This allows to reason concurrently about scenario alternatives. Moreover, a weak initial likelihood of a scenario branch can be amplified by future events. 2) In our approach all enabled transitions are executed in a single parallel step. 3) Petri nets modeling scenarios are actually state machines. Their structure is sufficient to construct the Büchi automaton [9] representing an LTL formula.

6 Event detection system and initial experiments

In this section we describe a prototype system allowing to test defined events. The system takes at input an annotated video sequence defining tracking information. For each frame a list of segments and identified objects is provided. The data does not represent ground truth, but real output from experimental tracking algorithms developed within a project SIMPOZ [1] aiming at implementation of an automated video surveillance system.

The architecture of the system is presented in Fig. 6. Main components are: the *Fuzzy ontology*, a set of *Evaluators*, i.e. functions calculating fuzzy values of predicates from low-level features of tracked objects, and the *Fuzzy Semantic*

[1] `http://www.simpoz.pl`

Petri Net. The system is also equipped with a GUI providing visual output shown in Fig. 7.

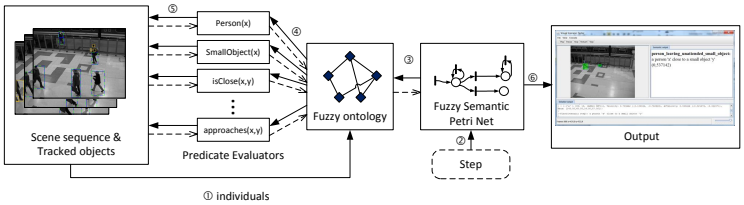


Fig. 6: Architecture of the scenario recognition system

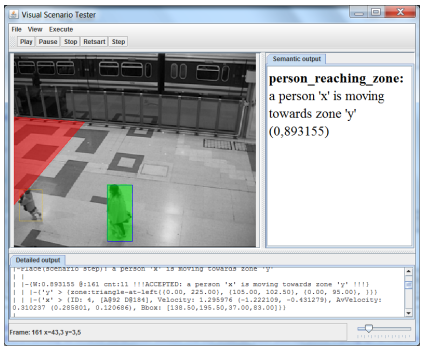


Fig. 7: Visual scenario tester

The system is entirely written in Java. Its performance is quite good: for three concurrently analyzed scenarios and a scene with a few tracked objects, a single reasoning iteration, during which the ontology is updated, evaluators are called and multiple transitions in Petri nets are fired, is executed within 0.1ms to 1.6ms (average 0.45 ms).

To facilitate the evaluation of a FSPN at the *design* time, the framework collects analytic information related to weights of tokens and their flows. This information helps to evaluate the recognition capabilities of defined FSPN and implemented evaluators. We will discuss this an example given by the formula (2) and corresponding Petri net in Fig. 5.

Fig. 8 presents in form of a Gannt chart weights of tokens assigned to net places at consecutive frames. For the purpose of presentation their values were shifted by adding 2, 4, 6 and 8 for tokens in *move*, *front*, *appear* and *remain*. Hence, each elevation above a baseline represents a subevent occurrence. The expected and successfully recognized event occurrence is accomplished within

the frames 286–316. It can be observed that compound subevents partly overlap, moreover, multiple transitions (marked with gray arrows) occur. Subevents *init*, *front*, *appear* and *remain* are stable, both as regards duration and amplitude. The presented time series exhibit interesting property related to reusing of ontology and predicate *evaluators*. The *movesTowards* evaluator that was used in specification of *move* event (see formula 2) was actually prepared for other experiment, during which people violating an artificial zone on a floor were detected (c.f. Fig. 7). As it can be noticed, the evaluator is inappropriate in the case of interactions with vertical objects, because it can not produce stable events. In the further development, it was replaced by a new corrected implementation: *movesTowardsVerticalObject*.

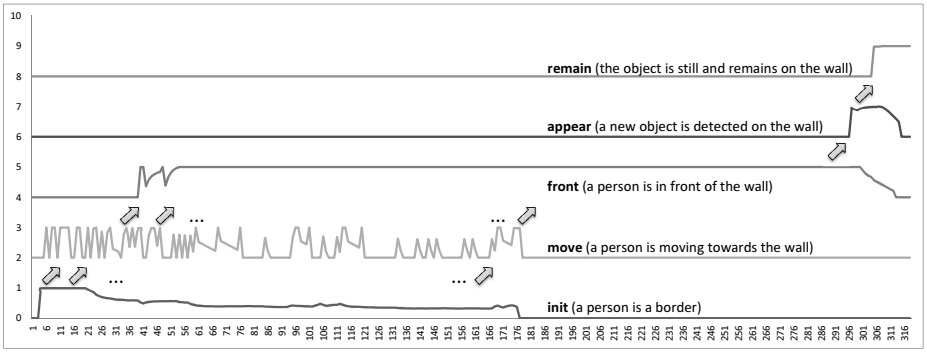


Fig. 8: Weights of tokens assigned to places at consecutive frames

Analogous experiments were conducted for several event recognition tasks including abandoned luggage and detecting violation of a surveillance zone. Tests for the abandoned luggage and graffiti painting events yielded 100% correct results (true positives). For a zone violation the recognition ratio was about 76%. Detailed analysis revealed that in this case the lower performance was caused by tracking problems (lost of identity in case of occlusion and in some cases invalid segmentation).

7 Conclusions

In this paper we address the problem of modeling and recognition of video events. To summarize our contribution: firstly, we propose to apply a temporal logic formalism to specify event scenarios and further to translate them to Petri net structures; secondly, we introduce Fuzzy Semantic Petri Nets; finally, we describe a proof of concept prototype system that interprets a data resulting from a tracking algorithm, represents it as a content of a fuzzy ontology and detects event occurrences with a FSPN interpreter. An advantage of FSPN is their ca-

pability of detecting concurrently occurring events, in which participate various combinations of objects, analyze scenario alternatives and their likelihoods.

References

1. Aggarwal, J., Park, S.: Human motion: modeling and recognition of actions and interactions. In: 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on. pp. 640–647 (Sept 2004)
2. Aggarwal, J., Ryoo, M.S.: Human activity analysis: A review. ACM Computing Surveys (CSUR) 43(3), 16 (2011)
3. Akdemir, U., Turaga, P., Chellappa, R.: An ontology based approach for activity recognition from video. In: Proceedings of the 16th ACM international conference on Multimedia. pp. 709–712. ACM (2008)
4. Albanese, M., Chellappa, R., Moscato, V., Picariello, A., Subrahmanian, V.S., Turaga, P., Udrea, O.: A constrained probabilistic Petri net framework for human activity detection in video. Multimedia, IEEE Transactions on 10(8), 1429–1443 (Dec 2008)
5. Barnard, M., Odobez, J.M., Bengio, S.: Multi-modal audio-visual event recognition for football analysis. In: Neural Networks for Signal Processing, 2003. NNSP'03. 2003 IEEE 13th Workshop on. pp. 469–478 (Sept 2003)
6. Borzin, A., Rivlin, E., Rudzsky, M.: Surveillance event interpretation using generalized stochastic Petri nets. In: Image Analysis for Multimedia Interactive Services, 2007. WIAMIS'07. Eighth International Workshop on. pp. 4–4. IEEE (2007)
7. Bremond, F., Maillot, N., Thonnat, M., Vu, V.T., et al.: Ontologies for video events. Research report number 51895. Tech. rep., INRIA Sophia-Antipolis (2004)
8. Brémond, F., Thonnat, M., Zúniga, M.: Video-understanding framework for automatic behavior recognition. Behavior Research Methods 38(3), 416–426 (2006)
9. Büchi, J.R.: On a Decision Method in Restricted Second-Order Arithmetic. In: International Congress on Logic, Methodology, and Philosophy of Science. pp. 1–11. Stanford University Press (1962)
10. Ghanem, N., DeMenthon, D., Doermann, D., Davis, L.: Representation and recognition of events in surveillance video using Petri nets. In: Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on. pp. 112–112 (June 2004)
11. Joo, S.W., Chellappa, R.: Attribute grammar-based event recognition and anomaly detection. In: Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on. pp. 107–107 (June 2006)
12. Kripke, S.: Semantical considerations on modal logic. Acta philosophica fennica 16(1963), 83–94 (1963)
13. Lavee, G., Rivlin, E., Rudzsky, M.: Understanding video events: A survey of methods for automatic interpretation of semantic occurrences in video. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 39(5), 489–504 (Sept 2009)
14. Lavee, G., Rudzsky, M., Rivlin, E., Borzin, A.: Video event modeling and recognition in generalized stochastic Petri nets. Circuits and Systems for Video Technology, IEEE Transactions on 20(1), 102–118 (Jan 2010)
15. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the Semantic Web. Web Semantics Science Services and Agents on the World Wide Web 6(4), 291–308 (2008)

16. Manna, Z., Pnueli, A.: Temporal logic. In: The Temporal Logic of Reactive and Concurrent Systems, pp. 179–273. Springer New York (1992)
17. Munch, D., Jsselmuiden, J., Arens, M., Stiefelhagen, R.: High-level situation recognition using fuzzy metric temporal logic, case studies in surveillance and smart environments. In: Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on. pp. 882–889 (Nov 2011)
18. Nagel, H.H.: Steps toward a cognitive vision system. AI Magazine 25(2), 31 (2004)
19. Vu, V.T., Bremond, F., Thonnat, M.: Automatic video interpretation: A novel algorithm for temporal scenario recognition. In: International Joint Conference on Artificial Intelligence. vol. 18, pp. 1295–1302. Lawrence Erlbaum Associates Ltd (2003)