An Extended G-Net Model for Knowledge Representation*

M. Gerkšič and N. Pavešić

Faculty of Electrical Engineering and Computer Science, University of Ljubljana, Slovenia

A new, generalized scheme for knowledge representation of fuzzy timed systems is proposed. The scheme is based on principles of a G-net model. We extended the existing G-net capabilities by the possibility of describing fuzzy defined and time dependent relationships among the system agents.

A fuzzy system is described by fuzzy relations. A fuzzy production rule is represented by a fuzzy term and valued by a certainty factor. Possibilities of modeling different types of knowledge are given.

Time dependencies in the systems and their representations vary according to the nature of the problems studied and a type of represented knowledge.

The represented scheme is capable of a static and a dynamic knowledge representation in a unique framework and enables interaction and coordination of both types of knowledge.

Finally, some illustrative examples are given.

1. Introduction

In the field of knowledge representation, many different approaches of modeling have been developed. They differ in the type of used structure, type of represented knowledge and in the set of used reasoning methods.

When modeling situations from different real (or simplified)-world vision system domains, we rarely come across problems where everything can be described in a small, one-purpose system. Moreover, in modeling a real world problem we always deal with different types of knowledge joined. Additionally, knowledge can be hierarchically organized in various levels or layers to represent the problem domain in the way that is easier to comprehend. Then, what is sought is a good knowledge organization enabling interactions among all levels and types of knowledge. Or, in terms of vision system domain, efficient robot action is a set of various agents such as sensors, situation analysis, general knowledge, domain knowledge, reasoning, conclusions..., compound together in a compact entity.

Among various approaches of knowledge modeling we have chosen the Petri-net formalism as a base of our robot vision system. By means of this formalism many different models have been built up to represent block world scenes, level-organization, time dependencies, fuzzy reasoning, etc. [7], [8], [9].

We have decided building on a very general knowledge representation scheme, with the possibility to combine static and dynamic knowledge and fuzzy described and time dependent parts. The combination of static and dynamic knowledge representation has been already done by Deng et al. in [2] and it is named G-net. In this paper, a fuzzy, time dependent knowledge base for computer vision systems, based on the G-net structure, is introduced. In addition, some general principles of reasoning in this structure are proposed. The detailed description of the reasoning algorithm is given in the article by Parkelj et al. [4] of this issue.

To illustrate the proposed scheme, two examples are given. The first one is representation of static knowledge with gradual inclusions of time and fuzzy dependencies. The other is a de-

* We gratefully acknowledge the support of the Ministry of Science and Technology of Slovenia and Alexander von Humboldt Foundation.

scription of a dynamic, time-dependent process with a fuzzy quality control element.

2. Knowledge Representation Using Petri-Nets

A structure of a Petri-net knowledge representation system can be defined as a 7–tuple [8]:

$$KRP = (P, T, I, O, \mu, \alpha, \beta),$$

where

- $P = \{p_1, p_2, \ldots p_n\}$ is a finite set of places;
- $T = \{t_1, t_2, \ldots t_m\}$ is a finite set of transitions;
- $I : T \rightarrow P^{\infty}$ is the input function, a mapping from transitions to bags of places;
- $O : T \rightarrow P^{\infty}$ is the output function, a mapping from transitions to bags of places;
- μ is a marking vector, denoting current state of the net by the distribution of tokens;
- $\alpha : P \rightarrow C$ is an associating bijective function, a mapping from places to set of knowledge concepts;
- $\beta : T \rightarrow \Sigma$ is an associating surjective function, a mapping from transitions to set of relations.

This is a very universal structure and allows various interpretations in potential fields of applications. An outline of the implemented model ,as well as the set of used reasoning mechanisms, will depend on the type of the represented knowledge.

To start modeling, knowledge should be classified, according to the problem given, into one of the two general types: static knowledge and dynamic knowledge.

2.1. Static Knowledge

Here, we refer to static knowledge when we have to represent some facts or general information, to describe a scene by its elements and relations among them. The basic Petri-net would be modified trough the elements of the sets of knowledge concepts C and relations Σ.

- C = { objects, classes of objects, object properties }
- Σ = { semantic, structural, positional relations among objects, property relations }

The other modification would be in the set of inference mechanisms. We are interested in information on the physical position of an object, its properties, its classification, thus we require inference mechanisms [6] in order to provide this information from given facts and knowledge.

- inference mechanisms = {inheritance, object recognition, intersection search}

2.2. Dynamic Knowledge

Dynamic knowledge is introduced when we have to represent some processes or algorithms. In this case the sets of knowledge concepts C and relations Σ can be interpreted as follows:

- C = {states of a process, control p-nodes}
- Σ = {functional and causal-effect relationship, actions in the process, environment actions, control t-nodes}

The inference mechanisms have to enable analysis of the net behaviour [3]. Therefore:

- inference mechanisms = {reachability, liveness, safety, *etc.*}

3. G-net as a Knowledge Representation Scheme

G-nets enable combining of static and dynamic knowledge into a unified structure for knowledge representation and inference. The G-net can be introduced as an extended Petri net, and represented as a 9–tuple [2]:

$$G = (k, P, T, m, I, O, \alpha, \beta, D).$$

The differences between G and Petri net are:

- k is a parameter used to determine the type of knowledge represented by a G-net. The value of k can be either "static" or "dynamic". According to this value the set of the inference mechanisms is chosen.
- m is a marking function which indicates tokens distribution. The tokens are "colored": *white* and *black* for dynamic, and *forward*

and *backward* for static knowledge. A *white* token is a normal enabling token as defined in a Petri-net. A *black* one, when generated, destroys all *white* tokens in the output place. By using *forward* and *backward* tokens each transition can be fired bi-directionally and can be used as well for recognition (*forward*) as for inheritance (*backward*).

- *D* is a triple denoting the properties of a semantic relation in the static part of the G-net structure. It is defining the set of semantic relations used in the G-net, the inheritance set of a given semantic relation and the set of reverse semantic relations.

The transformation of the G-net into a knowledge table, as proposed in [2], can be used to facilitate the reasoning process. Here, for the clarity reasons, we are pointing out just a switching place between a static and dynamic knowledge representation.

The switching place is a special type of a place. It contains information of an elementary G-net: name, type, marking, etc. Additionally, it is a place as any other in the G-net and can be used accordingly. It can be assigned transitions and thus connect in type or in level different "sub" G-nets.

New, resulting features of the unified structure are then:

- the efficiency of knowledge managing,
- modular and hierarchical construction and
- clarity and systemization of represented knowledge.

4. Time dependencies

When modeling time dependencies, we have found just a few approaches and many different interpretations of these approaches according to the problem domain. We have decided on the approach described by Ribarić in [10]. Types of the time dependencies can be separately considered for static and dynamic knowledge.

In a static knowledge model a time dependency means a dynamic, time dependent changing of the model. This can be achieved either by a predicate or by an interval associated to every net transition. In both cases the net structure is extended by an additional condition for transition firing.

Dynamic knowledge models are frequently describing processes (or process protocols). Therefore, the time dependency can be bounded to

- a certain state of the system,
- coordination among different processes,
- process or real time.

The first two items have rather control while the third one action meaning.

The first two can be modeled implicitly by either using *inhibitor arcs* or *black* tokens. *Inhibitor arcs* are transitions or part of transitions that can enable (disable) or control another transition. Similar mechanism is a *black* G-net token. The *black* token is a G-net mechanism and when generated, destroys all *white* (normal, enabling) tokens in the controlled place. Graphically it is represented by a transition arc ending with a back-arrow instead of an arrow. The modeling of this can be done by defining appropriate input and output set of a transition. Another possibility in the mechanism for Petri-net knowledge representation are functions for changing the marking of the net. With them we can change the marking of the net at the beginning, to denote the situation in the scene, or in the middle of the run, on a special request of some actions. Mechanisms of the *black* token and the time interval are not mutually exclusive and can also be used simultaneously, when necessary or requested. The only proper solution for the modeling of time in this case is again the use of a time interval.

The time interval shows to be very useful and can be applied in many ways. It is sufficient to represent all time dependencies at a static knowledge representation and even some at dynamic. Although we mentioned the *inhibitor arc* as a possible solution it can be substituted by the time interval in some cases. The only situation where it can not be accepted is when a *black* token mechanism is used.

A time interval can be defined either as a pair of two real-time moments denoting the start and the end time point, or the start and duration, or two real numbers showing just the time relationships, etc. In any case it is defining the value of

a predicate associated to transition. The predicate has a logical value and it is an additional condition to fire a transition.

Formally we define the time interval by the finite set Θ of ordered pairs associated to transitions $t_j \in T$:

- $\Theta = \{(\Theta_{1,j}, \Theta_{2,j}); \quad j = 1, 2, \ldots, m; \; \Theta_{1,j} < \Theta_{2,j}\}$, where
- $\Theta_{1,j}$ and $\Theta_{2,j}$ are the start and end point of the time interval when the transition t_j is enabled;
- the transition t_j is enabled by $w(t_j) = w_j$ where: $w_j = \begin{cases} 1; & \Theta_{1,j} < \tau < \Theta_{2,j}, \\ 0; & \text{otherwise}, \end{cases}$; and
- τ stands for current time.

To be able to "calculate" with the time intervals, what is very convenient for knowledge representation, the mapping function δ is introduced:

$$\delta : \Delta \rightarrow \mathbf{R},$$

where:

- Δ is the ordered set of start, end and inner points of time intervals and
- $\mathbf{R}$ the set of real numbers.

This function maps time into the set of real numbers:

- $\delta : \Theta_j \mapsto \Theta_j' \; ; \; \Theta_j' \in \mathbf{R}$.

5. Fuzzy Dependencies

There are as many different fuzzy set theory based knowledge representation as researchers dealing with them, but here we concentrate on one only. interpretations of a representation the fuzzy set theory and only one aspect will be shown here. We will take the mechanism [1] and then treat the static and the dynamic knowledge representation separately.

A static fuzzy system usually represents the problem of recognition or inheritance of an object/ concept, where input data are given descriptively or inexactly. A typical problem would be a representation of medical knowledge.

On the other hand, with dynamic fuzzy systems the main stress is on fuzzy system control. Input data can be quite exact but numerous and thus difficult to manage consistently. The control of the process is reduced to a few rules for the most typical situations, but defined loosely enough to be of a wide use.

Typical problems are arising from complex situations with unpredictable variable parameters as driving a car, balancing of a moving object, etc.

Concerning static problems, first, properties can be defined fuzzy by using linguistic qualifiers. Then, concepts represented by places become fuzzy sets. Values can be assigned to tokens. Thus we denote the value of a membership function for a fuzzy set represented by the place.

For dynamic problems, further, the rules (transitions) can be weighted by a certainty factor to denote possibility of a rule. Also, additional type or a color of a token (*enable*—*e*) is needed to enable process control and feedback loops.

To enable all these features, the net structure is extended by the following definitions:

- $\gamma : P \rightarrow [0, 1]$ is an association function, denoting value of a token in a place: $\gamma(p_i) \in [0, 1]$;
- $f : T \rightarrow [0, 1]$ is an association function, denoting certainty factor of the transition (rule) $f(t_i) \in [0, 1]$;
- $\lambda \in [0, 1]$ is a threshold value, an additional condition to fire a transition.

6. The Extended G-net Scheme

Let us recollect all the pieces of our jigsaw together:

Petri-net= $PN = (P, T, \mu, I, O)$

$KRP = PN + (\alpha, \beta, C, \Sigma)$

G-net = $G = KRP + (k, m, D); \qquad (\mu \mapsto m)$

time= (Θ, δ, w)

fuzzy= $(\gamma, f, \lambda, M); \qquad (m \mapsto M)$

Extended G-net= $EGN = G+$ time $+$ fuzzy=

$(P, T, I, O, \alpha, \beta, C, \Sigma, k, M, D, \Theta, \delta, w, \gamma, f, \lambda)$

This 17–tuple is the first iteration of Extended G-net description. To obtain the proper Extended G-net structure a few changes are made. First, we have sorted the given parts according to their function and divided them into two groups: parts of net structure and parts for knowledge representation. This division can be represented by the following expression:

$$EGN = \text{network} + \text{knowledge} =$$
$$= (P, T, I, O, M, w, \gamma, f, \lambda)$$
$$+(\alpha, \beta, C, \Sigma, \Theta, \delta, k, D,)$$

The next step was to find whether there are some common expressions in the different parts, that can be joined and thus the net structure simplified. No simplification can be done in the "knowledge" part. This part is a kind of an interface between the knowledge and its net representation and therefore this relation must be clear and exact. In the net part some modifications can be made.

The first one was indicated at the beginning of this section, along by the structure of the Extended G-net, with changing of the function of the token distribution. At the beginning, there was μ in the Petri-net, giving the number of tokens in places. Then it was changed to m in the G-net. Here the tokens had four colours, a pair for each type of knowledge, with their meanings unchanged. After introducing fuzzy reasoning, another modification happened and the function was renamed to M. In the fuzzy reasoning, as described above, tokens have values between 0 and 1 or e.

In the second modification we assembled the firing condition. A transition can fire, if it is enabled and if there are enough tokens in all preceding places. In assembly of the structures, we have gathered two enabling predicates. The first one is $w(t_i)$ or w_i, from the time dependencies and has a binary value. The second is defined by λ, the threshold value of the fuzzy rule. We propose the following assembly. Instead of taking $\lambda \in [0, 1]$, as defined by the fuzzy part, let $\lambda \in [0, 2]$. The condition of firing a fuzzy transition remains the same, but if we want to disable the transition we add 1 to the current value of the threshold $\lambda := \lambda + 1$. So the value of an enabled transition is between $[0, 1]$, and of a disabled in $[1, 2]$. With this mechanism we can enable and disable a transition with one predicate, without losing information of the threshold. At the end, we have to adjust the enabling predicate for the transition in the time dependent and non-fuzzy system. According to the former mechanism, the "predicate", now λ instead of w_i, must be in inverted logic: 0 for enabled and 1 for disabled.

What is gained by this approach? Surely, this method has some disadvantages: instead of logical predicate a real value is used to control the transition. Even where we could look at it as a binary value, it has an inverted value. On the other hand, what we achieved is the possibility of using the same structures for normal and fuzzy reasoning with a minor change in the time part of knowledge representation.

The final version of the Extended G-net is the following 16–tuple:

$$EGN = \text{network} + \text{knowledge} =$$
$$= (P, T, I, O, M, \gamma, f, \lambda)$$
$$+(\alpha, \beta, C, \Sigma, \Theta, \delta, k, D) =$$
$$= (P, T, I, O, M, \gamma, f,$$
$$\lambda, \alpha, \beta, C, \Sigma, \Theta, \delta, k, D)$$

In spite of greater complexity, derived structure is comprehensive and extensive so that can be used in a wide spectrum of problems. comprehensive and extensive wide spectrum of problems it is not so terrible. All the unnecessary parts can be disabled or just put aside by choosing the appropriate "indifferent" values. Indifferent values are constant values that "hide" unoccupied segments and enable simplified reasoning. In the example of the firing condition the indifferent value would be $\lambda(t_i) = 0; \quad \forall t_i \in T$ and all transitions are enabled.

7. Examples

7.1. A Static Knowledge Representation Example

The example of a static, time dependent knowledge represented with EGN is given in figure 1. In the figure we can see a normal static knowledge of a concept "a horse". From this scheme we can learn:

```
Mare is_a Lipicaner is_a Horse
```

`Stallion is_a Lipicaner is_a Horse`

`Stallion is_a_son_of_a Mare`

(For the ease of the comprehensiveness of the problem, the nets are marked by knowledge items from the sets $C, \Sigma, \ldots$ instead by the net items from $P, T \ldots$.)

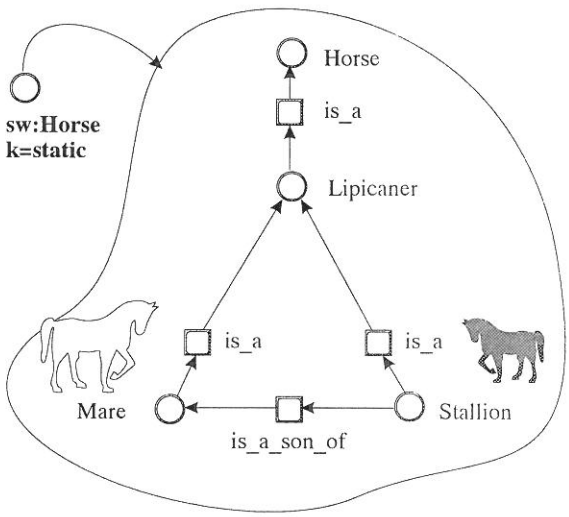

Fig. 1. Concept `a_horse`.

In figure 2 this concept is widened with some time dependent knowledge. The time, in this case, means the age of a horse.

The additional results of the inheritance reasoning in this scheme would be:

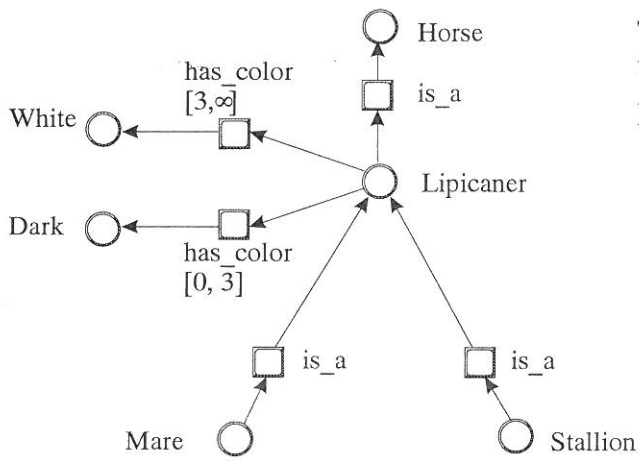

Fig. 2. Time Dependent Knowledge

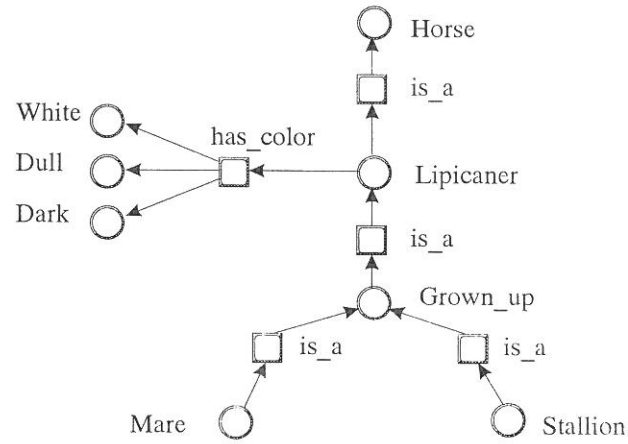

Fig. 3. Fuzzy defined static knowledge.

$\tau = 7$: `Mare is_a Lipicaner has_color White`

$\tau = 2$: `Stallion is_a Lipicaner has_color Dark`

However, the age is a typical fuzzy term. Another example is a color. We can not tell exactly, at which moment a horse, if it is a lipicaner, ceases to be dark and becomes white. We know that it is dark when born, and white when grown up. Regarding the time and colour, the change is gradual.

In figure 2, another place is added. It is representing the fuzzy set `Grown_up`. The first transition in this net is then a mapping from the real horse's age to the membership function of the fuzzy set `Grown_up`. The transitions towards Lipicaner and Horse have the value 1 because the memberships to these sets are not fuzzy-defined themselves. The value of the token is important when evaluating the color of the horse.

The evaluation for the stallion is shown in figures 4 and 5.

Finally, the results for this highly simplified ex-

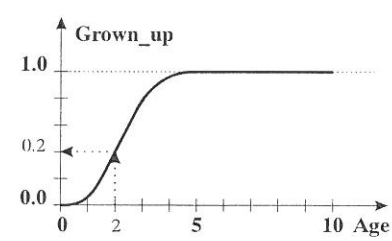

Fig. 4. Evaluation of the membership of the fuzzy set `Grown_up`.

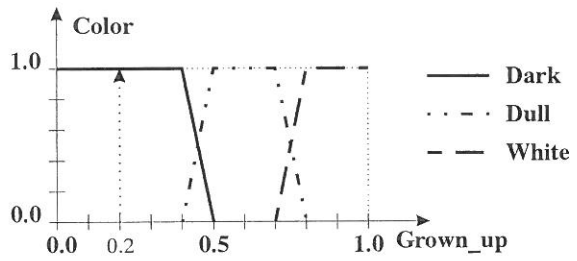

Fig. 5. Evaluation of the membership of the fuzzy set Dull.

ample are:

```
Stallion has_color Dark (1.0)

Mare has_color White (1.0)
```

For the study of the fuzzy logic and realization of this examples we also used [13], [11] and [12].

7.2. A Dynamic Knowledge Representation Example

Let us consider another horse, of a different color.

For dynamic knowledge representation we have chosen an example from quality control. We have a line for filling up diverse perfume bottles. We have a simple machine that fills all the bottles approximately equally, a sensor of height and a corrector. Since we do not want to have a jam in the line every time we have to correct a bottle, another, queuing place is added at the beginning. At the filling machine is a control mechanism, which is demanding a free line before the next bottle is proceeded. The system is shown in figure 6.

The net structure displayed in figure 6, is explained in detail in the tables 1 and 2.

8. Conclusion

The Extended G-net we have proposed, enables modeling combined knowledge types in a unified scheme. Although we have examined different kinds of representations separately, they are not mutually exclusive but can be combined. Even though the structure is comprehensive it can be easily simplified by assigning appropriate constant values to unnecessary parameters.

Since the fuzzy logic is very generalized it can be degraded by additional limitation rules to model a stochastic, multilevel or a normal logic system.

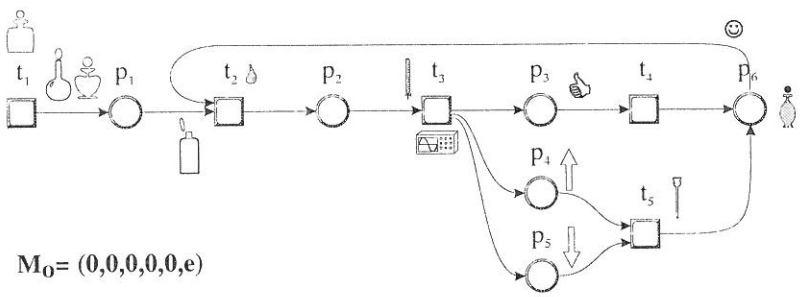

Fig. 6. A line for filling up the bottles.

place	concept	token value	meaning
p_1	EMPTY	N_0	number of waiting bottles
p_2	FULL	[0, 1]	membership f. of fuzzy set FULL
p_3	GOOD	[0, 1]	membership f. of fuzzy set GOOD
p_4	MUCH	[0, 1]	membership f. of fuzzy set MUCH
p_5	LITTLE	[0, 1]	membership f. of fuzzy set LITTLE
p_6	DONE	e	bottle finished, enable next bottle

Table 1. The Table of Places.

transition	action	token calculation	meaning
t_1	NEW	+1	new bottle in the queue
t_2	FILL	fuzzy	fill the bottle
t_3	MEASURE	norming	filled % of bottle volume
t_4	CORRECT	e	bottle corrected, fill next enabled
t_5	END	e	fill next enabled

Table 2. The Table of Transitions.

The examples have shown the use of defined structure in a complex problem domains.

References

[1] CHEN S. M., KE J. S., CHANG J. F., "Knowledge Representation Using Fuzzy Petri Nets". *IEEE Trans. on Knowledge and Data Eng.*, Vol. 2, No. 3, pp. 311–319, September 1990.

[2] Deng Y., CHANG S. K., "A G-Net Model for Knowledge Representation and Reasoning". *IEEE Trans. on Knowledge and Data Eng.*, Vol. 2, No. 3, pp. 295–310, September 1990.

[3] MURATA T., "Petri Nets Properties, Analysis and Applications." *Proc. of the IEEE*, Vol. 77, pp. 541–580, April 1989.

[4] PARKELJ M., PAVEŠIĆ N., "Fuzzy Expert System for Pattern Recognition". *The Second German–Slovenian Workshop: 3–D Scene Aquisition Modeling and Understanding*, in this issue, June 1994.

[5] RIBARIĆ S., PAVEŠIĆ N., "Declarative and Procedural Knowledge Interaction in Robot Vision System". *Proceedings of the Slovenian–German Workshop: Image Processing and Stereo Analysis*, pp. 53–71, December 1992.

[6] RIBARIĆ S., K. ŠPARAVEC, "Program Implementation of Inheritance in Knowledge Representation Scheme". (In Croatian), *14th Conference "MIPRO '92" Proceed. of the MIS — Microcomputers in Intelligent Information Systems*, pp. 33–88, Opatija, May 1992.

[7] RIBARIĆ S., PAVEŠIĆ N., "Knowledge Representation Scheme For Blocks World Scenes". *Automatika 31*, pp. 49–54, No. 1–2, 1990.

[8] RIBARIĆ S., "Knowledge Representation Scheme Based on Petri Net Theory". *Int. Journal of Pattern Recognition and Artificial Intelligence*, pp. 691–700, Vol. 2, No. 4, 1988.

[9] RIBARIĆ S., GYERGYEK L., PAVEŠIĆ N., "Knowledge Representation Schemes in Computer Vision and Pattern Recognition". *Automatika* , Vol. 29, No. 1/2, pp 5–8 pp. 1–7, 1988.

[10] RIBARIĆ S., "An Original Scheme For Representation of Time Varying Knowledge". (In Croatian), *Invited paper on 11th Conference "MIPRO '89", Opatija, Proceed. of the MIS*, pp.4–41 — 3–56, May 1989

[11] VIRANT J., "An Implementation of the Fuzzy Logic in the Modern Systems". (In Slovene), *Didakta*, Ljubljana, 1992.

[12] VODOPIVEC T., "Don't Worry, Be Fuzzy". (In Slovene), *Programer 24*, pp. 22–27, Novo Mesto, 1994.

[13] ZADEH L. A., "Fuzzy Logic". *Computer*, pp. 83–92, April 1988.

Contact address:

M. Gerkšič, N. Pavešić
Faculty of Electrical Engineering and Computer Science
Laboratory of Artificial Perception
Tržaška 25, Ljubljana
University of Ljubljana, Slovenia
e-mail: maja.gerksic@fer.uni-lj.si

MAJA GERKŠIČ (1967) is an assistant researcher in the Laboratory of Artificial Perception at the Faculty of Electrical and Computer Engineering at University of Ljubljana. She received her B. Sc. degree in Electrical Engineering from University of Ljubljana in 1991. Her area of research interest is in representation of knowledge for systems of artificial intelligence, using Petri nets and Fuzzy logic.

NIKOLA PAVEŠIĆ, please, see pp. 191.