Topological Modeling with Fuzzy Petri Nets for Autonomous Mobile Robots

Javier de Lope[1], Darío Maravall[2], and José G. Zato[1]

[1] Dept. Applied Intelligent Systems, Technical University of Madrid, Spain
{jdlope,jzato}@eui.upm.es
[2] Dept. Artificial Intelligence, Technical University of Madrid, Spain
dmaravall@fi.upm.es

Abstract. In this paper a novel method of reference places' detection to build topological models is described, as well as an algorithm for route planning based on Fuzzy Petri Nets. The proposed method for the detection of reference places does not employ sensory information but the information provided by the robot's control subsystem. The reference places and the navigation strategies between places are used for constructing the model's environment with a Fuzzy Petri Net. The route planning algorithm propagates over the net the certainty value of places and transitions. After finishing the propagation the transitions and the places store the information needed to make decisions about the navigation strategies of the robot route.

1 Introduction

There is an isomorphism between the environment's model of a mobile robot and the control subsystem responsible of its actions. Therefore we can consider the models of the world as the robot's tools for planning its actions to reach a concrete purpose. This planning is useful to predict some robot behaviors before the corresponding world's situations or states happen; i.e. to predict the environment's states to act accordingly.

Model building can be classified into two large groups: (a) metric models and (b) topological models. In a metric mode the information about the position and orientation of the robot is numerically specified in relation to a fixed reference system. In topological models the information about the world is based on a group of reference places that can be recognized by the robot and on a group of links between accessible reference places.

The choice of a particular model depends on the problem to be solved. Metric models are suitable for situations where high accuracy in the robot's navigation is required. Topological models are more suitable for situations in which a very

[0] *Proc. of 11th Int'l. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA-98-AIE*, A.P. del Pobil, J. Mira and M. Ali (eds.), Lecture Notes in Artificial Inteligence, 1416, Springer-Verlag, Berlin, 1998, pp. 290–299.

precise knowledge of the robot's position is not vital and therefore it is sufficient for the navigation of the robot to recognize the reference places rather than to compute its exact position. Examples of metric models are expounded in [9] and [3]. Topological models are proposed in [4], [5], [8], [6] and [13].

In this paper a novel method of reference places' detection to build topological models is described, as well as an algorithm for route planning based on Fuzzy Petri Nets.

2 Detection of Reference Places

Most of the mobile robots navigation systems based on topological models use the sensory information as the main source for the detection of reference places. For instance, Kuipers and Byun [4, 5] employ a detection method of what they call distinctive places (i.e. reference places) based on production rules about the sensory information coming from the environment. Matarić [8] defines a procedure to detect reference places (landmarks) using the continuos monitoring of the sensory measurements closely related to a navigation based on wall following. Kurz [6] proposes a detection approach very similar although more general. Serradilla [13] introduces a novel concept, the sensory gradient, to detect reference places so that when great changes in the sensory gradient operator occur the robot detects a *relevant sensory place* —i.e. a reference place—. Nehmzow and Smithers [11] propose a rather different approach and instead of using the sensory information to detect reference places, the robot's internal behavior produced by its control subsystem allows the detection of the reference places.

In this paper we propose a method for the detection of reference places that does not employ sensory information but the information provided by the robot's control subsystem. The changes in the behavior modes of the control subsystem generated by the presence of obstacles allow the creation of reference places. Unlike Nehmzow and Smithers' our method uses several navigation strategies, including wall following, that enriches the world model built by the robot. Apart from wall following other strategies are free movement or wandering and detour that permits the robot to make decisions about alternative trajectories.

In Fig. 1 three reference places are displayed. In all of them the robot abandons its current navigation strategy in order to avoid the collision with the obstacle in front of it. These situations are recognized as reference places because the robot must abruptly change its trajectory and they can be used for future planning of navigation missions.

In the above mentioned three situations the robot must change suddenly its current trajectory turning to the side without obstacle. In cases (a) and (b) there is only one alternative: to turn right and left, respectively. In case (c) the robot can turn in both directions.

Once the collision situation has been overcome the robot must continue with a navigation strategy. For the two first cases any of the aforesaid strategies may be used. However in case (c) the wall following strategy cannot be activated because it could lead the robot to an infinite loop state.

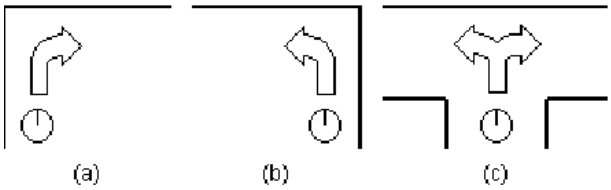

Fig. 1. Detection of reference places

When a reference place is detected some relevant information for later use must be retained. Basically this information concerns the state of the robot and some features of the environment of that reference place. From the standpoint of the topological model building the information of each reference place is divided into two groups: (a) information about the physical place and (b) information about the navigation strategies activated since the last reference place. In Table 1 these two different pieces of information of a reference place are shown.

Table 1. Information about places and transitions

Reference Places	Transitions
Robot Cartesian coordinates	Origin and destination places
Robot angular coordinates	Duration
Type of reference place	Exit turn direction
Concavity for calibration	Activated strategy

This information and the number of times that a reference place has been visited are important for making decisions about the direction and the navigation strategy for the next visits to a particular reference place. The reason for making a place with concavities for calibration is to use some places for the dynamical calibration of the robot's odometric information as suggested by Kurz [6].

3 Fuzzy Petri Nets for Topological Modelling

Unlike other topological modelling methods our approach admits two possible interpretations. On the one hand the model can be viewed as a set of places accessible by the robot by means of navigation strategies. On the other hand the inverse interpretation is also correct: the world model is formed by a set of robot states determined by the navigation strategies related to each of the reference places detected in the environment.

This double interpretation and the fact that the links between reference places are unidirectional make possible the use of Petri nets [10] for its rep-

resentation. More specifically we are going to introduce fuzzy Petri nets [7, 2, 14] because the application of inference tools is very useful for route planning.

The reference places correspond to the nodes or places of the net and the robot states —generated by the navigation strategies— to the transitions of the net. A Petri net can be interpreted either as a sequence of places or as a sequence of transitions. Regarding our navigation system the second interpretation is more interesting because the quantitative information about the exact position of the reference places is not necessary and at the same time the robot can reach these places using navigation strategies based on world features rather than on physical positions. In this way the routes can be interpreted as suggestions like in the work of Agre and Chapman [1] and like in the potential fields approach of Payton [12], so that the route towards a goal place can be established through navigation strategies activated in each reference place. An example of route planning should be the following: "go ahead towards the wall, then turn to the right and take the first way on the left".

A Petri net can be defined as the tuple $PN = \{P, T, F, W, M_0\}$ where $P = \{p_0, \ldots, p_m\}$ is the finite set of places, $T = \{t_0, \ldots, t_n\}$ is the finite set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is the set of arcs, $W : F \to \{1, 2, 3, \ldots\}$ is the weighting function for arcs and $M_0 : P \to \{0, 1, 2, \ldots\}$ is the initial marking of the net that may or may not exist. As additional restriction we include $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$.

For our navigation system the set of places P is formed by positions and orientations of the robot and the set of transitions T consists of the navigation strategies such that a transition $t_q = t_{i-j}$ drives the robot from place p_i to place p_j. The weighting function W assigns value 1 to each arc linking places and transitions as each transition determines the movement of the robot between two places. The net marking depends on the place where the robot is situated. Therefore the firing of the transition t_{i-j} depends on the position of the robot.

Fuzzy Petri nets are derived from Petri nets and can be described as the tuple $FPN = \{P, T, D, I, O, f, \lambda, \alpha, \beta\}$ where P and T stands for the same than in a PN, $D = \{d_0, \ldots, d_m\}$ is a finite set of propositions such that $P \cap T \cap D = \emptyset$ and $|P| = |D|$, $I : T \to P^\infty$ is an input function that defines a correspondence between the transitions and theirs input places, $O : T \to P^\infty$ is an output function that stands for the correspondence between the transitions and theirs output places, $f : T \to [0, 1]$ is an association function that establishes a correspondence between transitions and the real-valued segment $[0, 1]$ and determines the certainty of each transition, $\lambda : T \to [0, 1]$ is an association function fixing a correspondence between the transitions and the interval $[0, 1]$ and settles the activation threshold for each transition, $\alpha : P \to [0, 1]$ is a function that establishes a correspondence between the places and the interval $[0, 1]$ and $\beta : P \to D$ is a biyective correspondence between places and propositions.

When fuzzy Petri nets are used for knowledge representation the set of propositions D defines the antecedent and the consequent of the rules: i.e. the transitions. In our navigation system the propositions di are statements like "the robot is at place pi". The truth-value of this proposition, as in a general-purpose

fuzzy Petri net, is given by $\alpha(p_i)$. The input function I applied to a transition t_{i-j} will provide the set of places pi from which can be fired. The output function O applied to the same transition t_{i-j} will give as a result the set of places p_j accessible from p_i that in our case —for restriction— is a single place. The value given by the function f over the transition t_{i-j} is interpreted as the cost associated to this transition and it is determined by factors like the execution time of the corresponding navigation movement and by the navigation strategy's complexity. The value given by the function λ applied to the transition indicates the activation threshold of this transition and its value is estimated through experience.

4 Route Planning Algorithm

The route planning algorithm based on fuzzy Petri nets that we propose in this paper propagates over the net the certainty value of places and transitions. The initial choice of the places' values guarantees that after their propagation the least valued transitions are those that lead faster to the goal.

It is relevant to emphasize that thanks to the concurrent nature of Petri nets the algorithm permits to directly plan the paths that lead to several goal places by just an appropriate initialization of the net's places. Furthermore with an updating of the transitions' values that we describe later the planned routes appear ordered in such a way that the first one leads to the nearest goal place. Obviously the path planning for a single goal place is a particular case.

The values of places and transitions propagate from the goal places towards the place where the robot is situated. In this way for each transition the corresponding value is stored and therefore it is possible to explicitly know the larger paths produced by either more expensive transitions or by a higher number of transitions. Therefore the right decision at each place is to choose the transition with the minimum value. Notice that if the propagation were the opposite to the above mentioned —i.e. from the starting place towards the goal places— then it would not be possible to evaluate the optimum transition at each place of the net.

The initialization of the transition values is not necessary as they are updated at each propagation step. On the contrary the goal places must have values less than the rest of the places in order to guarantee that the least valued places lead to the goal more rapidly.

Each propagation step consists of three phases. The first one corresponds to what in fuzzy reasoning is known as *composition*. In this phase the transition values from each node is updated by accumulating the estimated cost in crossing each transition. Each transition of the net leads to a single place by the very nature of the net that is a model of the environment. Therefore the updated value of each transition is obtained by adding the value of the place where the transition ends and the estimated cost of the transition. The transitions ending in a goal place have a value equal to its estimated cost. For the rest of the transitions its estimated cost is added to the previous path. Obviously the

transitions' values cannot be higher than unity. The updating of the transitions values $f(t_i)$ is made as follows:

$$f(t_i) = \min\left[1, C(t_i) + \alpha(p_j)\right] \tag{1}$$

where p_j is the destination place for transition t_i and $C(t_i)$ is the weighted cost associated with the firing of transition t_i. This cost is estimated as:

$$C(t_i) = \frac{c_i}{\sum_{j=0}^{i} c_j} \tag{2}$$

where c_i is the real cost for transition t_i measured as a combination of the time expended in physically performing this transition and the complexity of the behaviors emerging during the transition.

The second stage —*thresholding*— is very common in fuzzy reasoning. It assigns value 1 to the transition t_i such that its computed value does not exceed the activation threshold, so that the number of rejected transitions for the navigation planning can be easily controlled:

$$f(t_i) = 1;\ \text{iif}\ f(t_i) > \lambda(t_i) \tag{3}$$

Finally the third phase, usually known as *inference* in fuzzy reasoning, updates the values of the places depending on their transitions. The final value determines the optimum place. If the value of this path has been previously computed then the place's value does not change. However if the minimum value of all the transitions leaving the place is less than the value of the place itself then the last value must be updated with the value of the transition. The values $\alpha(p_i)$ of the net are updated as follows:

$$\alpha(p_i) = \min\left[\alpha(p_i), f(t_i)\right] \tag{4}$$

where the transitions t_j are all those leaving the place p_i —i.e. $p_i \in I(t_j)$—.

The propagation of the values of all places and transitions must continue until the initial place is no longer equal to 1. This means that a path linking the initial place with its nearest goal place has been obtained. Furthermore by the building itself of the paths linking the initial place to the goal places the found path is of minimum cost.

After finishing the propagation the transitions and the places store the information needed to make decisions about the control strategies for the navigation of the robot towards the goal. At each place the transition with the minimum value is taken. Furthermore the transition with value unity are rejected.

The pseudo-code of the above described propagation algorithm is the following:

1. Initialize the goal places p_g so that $a(p_g) = 0$ and the rest of the values p_i gives $a(p_i) = 1$.
2. While initial or starting place p_s holds $a(p_s) = 1$:
   (a) Update the transitions t_i by means of expression (1).
   (b) Update the transitions t_i as in expression (3).
   (c) Update the places p_i by applying expression (4).

5 Experimental Results

In the sequel several examples of model building developed using our own mobile robot Nomad-200's simulation environment are discussed. Special care has been taken to work with realistic simulated situations, both regarding the sensors of the robot —16 ultrasound ring and odometric information— and the environment. In these examples the simulated world embraces an office-like environment measuring about 50 square meters.

In the Fig. 2 Petri net built by the system is displayed over the simulated environment. The reference places detected by our system, that correspond to the net's places, are represented by circles with its reference number. Place p_0 is the initial place of the net from which the robot starts its exploration of the environment and it has been marked with number 0 —left topside of the Figure—. The lines linking the places stand for the transitions.

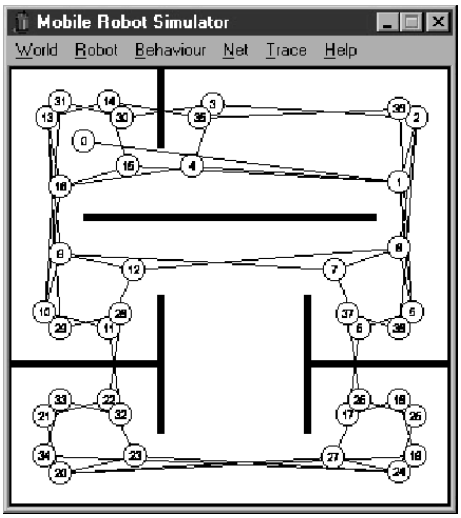

Fig. 2. Model of the environment obtained with a wall following strategy

In Fig. 3 two robot trajectories starting at place p_5 and ending in the left bottom corner are shown. In (a) the goal place is p_{20} and the places traversed by the robot are: p_6 with free movement strategy; p_{17} with a strategy based on following the wall through the left and p_{18}, p_{19} and p_{20} with free movement strategy.

The $f(t_{i-j})$ values of Table 2 explain the robot trajectory. Place p_5 has a single alternative of going to p_6. From this place it is possible either to go to place p_7 with $f(t_{6-7}) = 1.0$ and with a free movement strategy or to go to place p_7 with $f(t_{6-7}) = 0.106055$ and with a wall left-following strategy. As the second

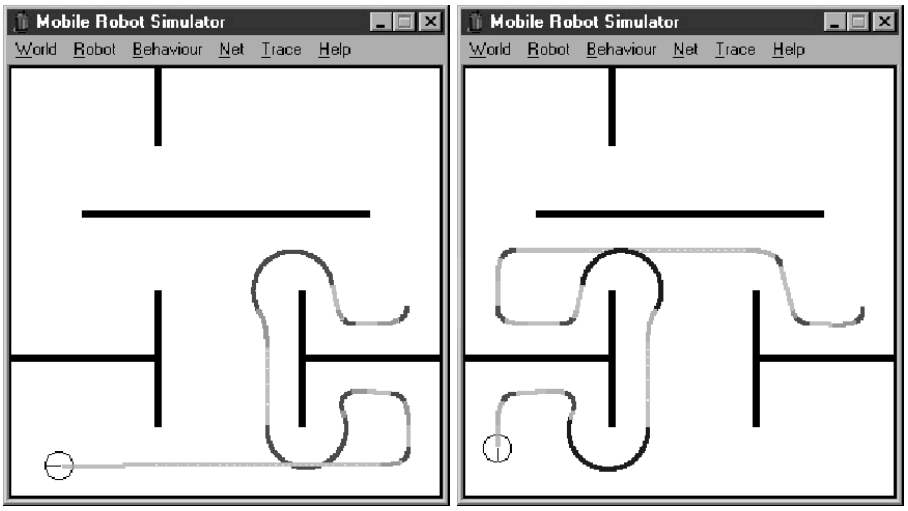

Fig. 3. Robot trajectories: (a) from p_5 to p_{20} and (b) from p_5 to p_{34}

alternative is of minimum cost the robot follows this transition. Afterwards at each place there is a single alternative, always by means of a free movement strategy.

In Fig. 3 (b) the goal place is p_{34} and the places crossed by the robot are: p_6, p_7, p_9, p_{10} and p_{11} with free movement strategy; p_{32} with wall right following strategy and p_{33} and p_{34} with free movement strategy.

In Table 3 the values of $f(t_{i-j})$ obtained by our algorithm are displayed. For this case place p_6 is chosen instead of p_7 because $f(t_{6-7}) = 0.122264$ and $f(t_{6-17}) = 1.0$. At place p_9 appears another important decision that in this case is to follow the way to p_{10} as $f(t_{9-10}) = 0.085341$ and $f(t_{9-13}) = 0.102383$. The last important decision occurs in $p_1 1$ and because $f(t_{11-32}) = 0.069075$ and $f(t_{11-12}) = 0.103076$ the wall on the right of the robot is followed to reach place p_{32}. Following the only alternatives for the subsequent places the robot finally arrives at the goal place p_{34}.

Table 2. The $f(t_{i-j})$ values for the trajectory from p_5 to p_{20}

Transition	$f(t_{i-j})$	Transition	$f(t_{i-j})$	Transition	$f(t_{i-j})$
$5 \rightarrow 6$	0.113172	$27 \rightarrow 24$	0.062213	$20 \rightarrow 21$	0.033987
$6 \rightarrow 17$	0.106055	$34 \rightarrow 24$	0.090440	$21 \rightarrow 22$	0.026306
$17 \rightarrow 18$	0.053048	$24 \rightarrow 25$	0.052837	$22 \rightarrow 23$	0.017729
$18 \rightarrow 19$	0.045602	$25 \rightarrow 26$	0.045269	$23 \rightarrow 20$	0.010053
$19 \rightarrow 20$	0.038038	$26 \rightarrow 27$	0.037715	Others	1.0
$23 \rightarrow 24$	0.082065	$27 \rightarrow 20$	0.029695		

Table 3. The $f(t_{i-j})$ values for the trajectory from p_5 to p_{34}

Transition	$f(t_{i-j})$	Transition	$f(t_{i-j})$	Transition	$f(t_{i-j})$
$5 \rightarrow 6$	0.129381	$12 \rightarrow 9$	0.095279	$30 \rightarrow 31$	0.109952
$6 \rightarrow 7$	0.122264	$9 \rightarrow 10$	0.085341	$31 \rightarrow 10$	0.101367
$7 \rightarrow 9$	0.114471	$2 \rightarrow 3$	0.139749	$10 \rightarrow 11$	0.077547
$9 \rightarrow 13$	0.102383	$3 \rightarrow 4$	0.117285	$11 \rightarrow 32$	0.069075
$13 \rightarrow 14$	0.119200	$4 \rightarrow 16$	0.109152	$32 \rightarrow 33$	0.016011
$14 \rightarrow 15$	0.111066	$16 \rightarrow 10$	0.093346	$33 \rightarrow 34$	0.007337
$11 \rightarrow 12$	0.103076	$3 \rightarrow 30$	0.135354	Others	1.0

6 Conclusions

A dynamical learning-based method for topological modeling of environments based on fuzzy Petri nets has been presented. This novel topological map building for autonomous mobile robots can be interpreted in a double way: (a) as a set of successive reference places of the environment and (b) as a sequence of robot's control strategies or behaviors.

The main advantage of the proposed method lies on the use of the aforementioned interpretation (b); i.e. by using the changes on the control strategies or internal behaviors —states— of the robot rather than using the sensory information. So that two improvements are obtained: (1) independence from the particular sensory equipment of the robot and (2) the control subsystem does not have to rely on problematic, external information.

Besides this approach to detect reference places for map building of the environment, the paper presents an algorithm based on fuzzy Petri nets for route or path planning. This algorithm is ver efficient from the computational point of view and guarantees an optimum path from the starting to the goal place. The paper discusses in particular the important issue of the propagation in the fuzzy Petri net of the values associated with the topological map of the environment.

Several examples of model building —i.e. topological maps— by applying the proposed method are also discussed. These and other similar cases, developed in a complex, realistic simulated environment was the first step before the implementation on a Nomad-200 mobile robot platform. Currently the proposed method has been fully implemented on the Nomad-200 and successfully tested on real navigation through the premises —an office-like environment— of the Computer Science Faculty at the Technical University of Madrid.

References

1. Agre, P.E., Chapman, D.: What are plans for? In: P. Maes (ed.): Designing Autonomous Robots. MIT Press, Cambridge, Massachusetts (1991) 17–34
2. Chen, S.M., Ke, J.S., Chang, J.F.: Knowledge representation using Fuzzy Petri Nets. IEEE Trans. on Knowledge and Data Engineering **2(3)** (1990) 311–319

3. Cox. I.J.: Blanche — An experiment in guidance and navigation of an autonomous robot vehicle. IEEE Trans. on Robotics and Automation, **7(2)** (1991) 193-204
4. Kuipers, B.J., Byun, Y.T.: A robust, qualitative approach to a spatial learning mobile robot. In SPIE Vol. 1003 Sensor Fusion: Spatial Reasoning and Science Interpretation (1988) 366–375
5. Kuipers, B.J., Byun, Y.T.: A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. In W. van de Velde, (ed.): Toward Learning Robots. MIT Press, Cambridge, Massachusetts (1993) 47–63
6. Kurz, A.: Constructing maps for mobile robot navigation based on ultrasonic range data. IEEE Trans. on Systems, Man and Cybernetics—Part B: Cybernetics, **26(2)** (1996) 233–242
7. Looney, C.L.: Fuzzy Petri Nets for rule-based decisionmaking. IEEE Trans. on Systems, Man, and Cybernetics, **18(1)** (1988) 178–183
8. Matarić, M.J.: Integration of representation into goal-driven behavior-based robots. IEEE Trans. on Robotics and Automation, **8(3)** (1992) 304–312
9. Moravec, H.P., Elfes, A.: High resolution maps from wide angle sonar. In Proc. of IEEE Int. Conf. on Robotics and Automation (1985) 116–121
10. Murata, T.: Petri Nets: Properties, analysis and applications. Proc. of the IEEE **77(4)** (1989) 541–580
11. Nehmzow, U., Smithers, T.: Using motor actions for location recognition. In Proc. of the First European Conf. on Artificial Life (1991) 96–104
12. Payton, D.W.: Internalized Plans: A representation for action resources. In: P. Maes (ed.): Designing Autonomous Robots. MIT Press, Cambridge, Massachusetts (1991) 89-103
13. Serradilla, F.: Arquitectura cognitiva basada en el gradiente sensorial y su aplicación a la Robótica Móvil. Ph.D. Dissertation, Dept. Artificial Intelligence, Technical University of Madrid (1997)
14. Yu, S.K.: Comments on "Knowledge representation using Fuzzy Petri Nets". IEEE Trans. on Knowledge and Data Engineering **7(1)** (1995) 190–191