

# An Agent-Like Architecture for Autonomous Robot Motion in Unknown Environment

G. B. Stamou, S. N. Raptis, and S. G. Tzafestas

Intelligent Robotics and Control Unit  
National Technical University of Athens  
Zografou 15773, Athens, Greece

## Abstract

*This paper proposes a general methodology for solving specific categories of robotic problems. This methodology decomposes the problem to simpler ones, every one of which is faced by an appropriate agent-like subsystem. The overall output is produced by merging the outputs of these agents taking into account its "degree of responsibility". This way, a complex function's universe of discourse is broken down to "domain of responsibility" of the respective agents.*

*This general methodology is then applied to solve the problem of path planning in unknown environment. To this end, every part of the system is specified and analyzed, and appropriate agents are defined.*

*Some experimental results obtained through simulation are finally given.*

## 1. Introduction

Autonomous mobile robots constitutes one of the most important fields of robotics and a very active research area for the last years. One of the key aspects in the mobile robots area is the path planning problem. The research on this problem has, for many years, been divided in two major categories, namely global path planning and local path planning.

*Global path planning* makes use of some available a priori knowledge relative to the environment and the objects that consist it, in order to move the robot towards a target position. To this end, many methods have been proposed in the technical literature, which differ in the philosophy of the solving algorithm, the knowledge representation scheme, etc. Some of the most important methods are:

- the configuration space method [1], developed by Lozano-Perez [2] and other researchers [3],
- the generalized Voronoi diagrams [4]
- the methods of artificial intelligence [5], and
- lately a very interesting and promising approach: the artificial magnetic field methodology [6].

The common problem in all the above global path planning methods is the need of possessing full knowledge of the environment and the obstacles. In many cases this demand may not be satisfied. That is the reason why local path planning techniques, capable to deal with generally unknown environments, have been developed.

In *local path planning* the robot makes use of information obtained by various sensors in order to successfully move to the target position. Dividing the research work in the field of local path planning into categories is not a straightforward task. Considering the kind of sensors used, one can find publications that make use of cameras [7], simple distance measuring sensors [8], etc.

To deal with the uncertainty introduced by the sensor measurements, fuzzy logic methods have been proposed [9].

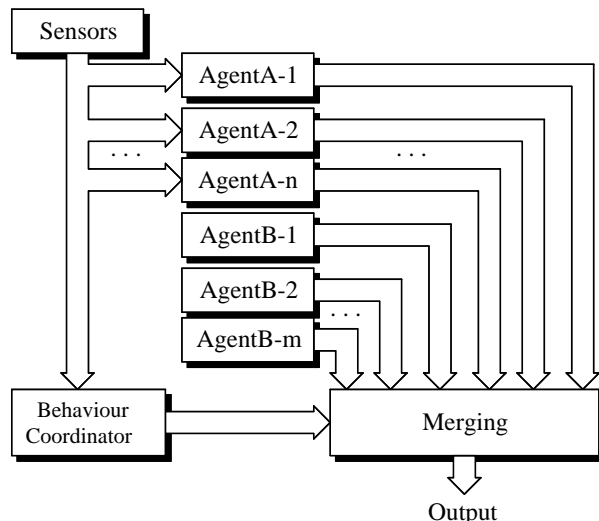
Quite popular in the field of obstacle avoidance are the hierarchical model [10], and lately Saridis' intelligent control scheme [11], often making use of fuzzy control methodology [12]. Although the hierarchical model aims at reducing the large complexity of path planning, problems arise due to the strict hierarchy and sequential nature of execution. The reason for this is that the complexity introduced by the identification tasks and the tasks that require intelligence, is not faced but only transferred to higher levels. To the end of solving this problem, Brooks [13] combines asynchronous units together, to each one of which a different role is assigned. However, these units are not independent since they communicate to each other. In a recent work of Boem and Cho [14], a combination of two independent units is presented, the one of which has an obstacle-avoidance behaviour and the other having a goal-seeking behaviour. Combination of these two units (which do not communicate to each other), is achieved through a 'behaviour-selector' which makes use of a bistable switching function to activate each unit.

The method proposed in this paper was inspired by Minsky's theory [15] and extends the above logic. The complex behaviour required to lead a robot

towards a target position can be reproduced by a combination of simpler independent ‘behaviouristic elements’, e.g. heuristics of the form ‘move towards the obstacles’, ‘move along the goal direction’, ‘avoid the obstacles that move to your direction’, etc. Many such antagonistic behaviouristic elements which are appropriate for different circumstances may be taken into account and may be implemented and operate independently. Some of them make use of the sensor measurements while others do not. An appropriate combination of such elements may lead to a system that exhibits the desired overall behaviour.

## 2. Structure of the Proposed Model

The proposed model is first presented in its general form and will then be specialized to the path planning problem by appropriately configuring each part. The system’s general form is given in Fig. 1. It consists of  $n$  agents connected to the sensors (i.e. their behaviour depends on the specific circumstances) and  $m$  agents that do not depend on the inputs. Every agent produces an output independently from all the other agents. All these partial outputs are appropriately merged by the behaviour coordinator. The sensor data is input to the behaviour coordinator which may also have some kind of memory in order to recognize more efficiently the present situation.



**Fig. 1:** General structure of the proposed model

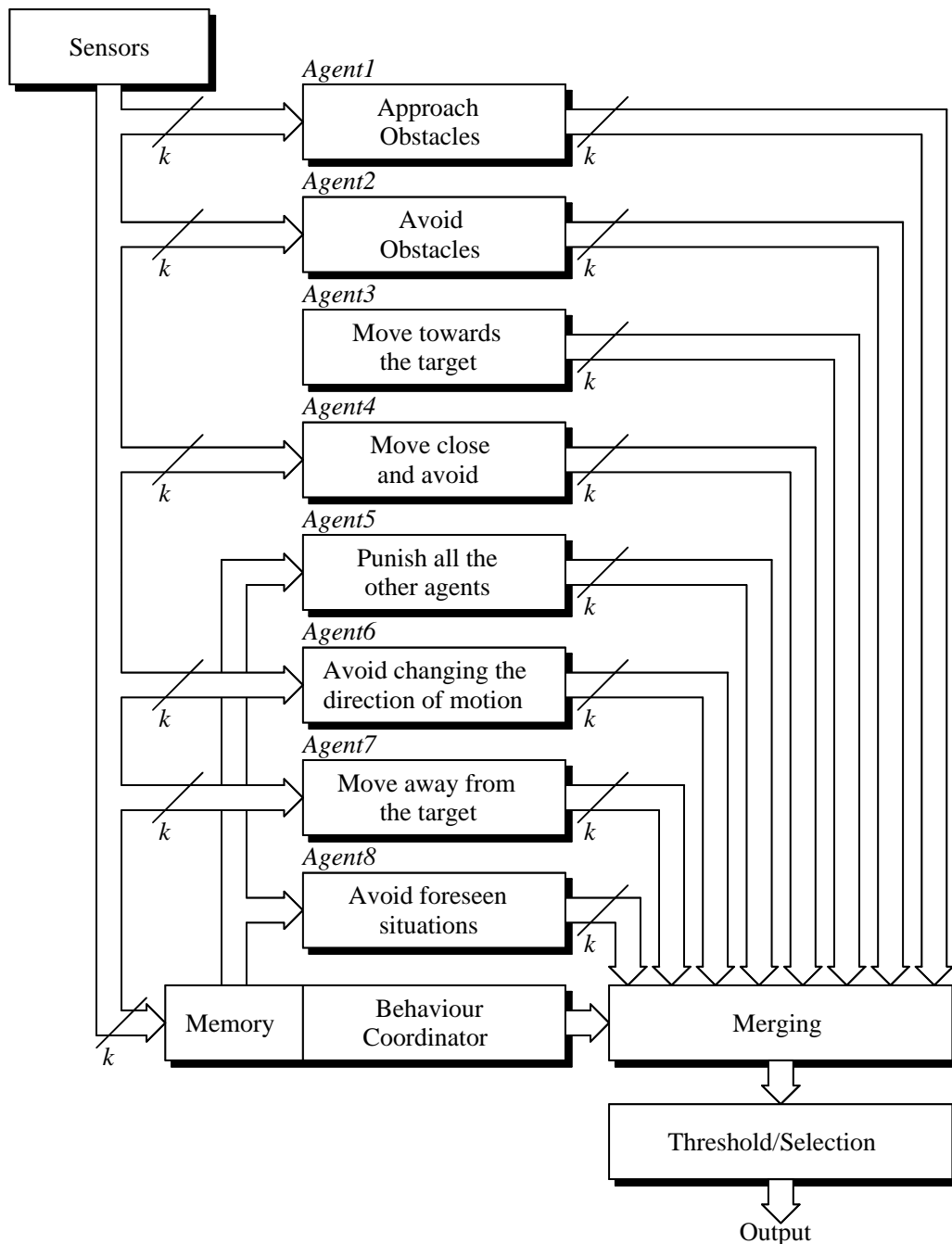
Each agent is implemented in such a way that a specific behaviour, appropriate for some specific situations, is exhibited. This may be easily achieved by using conventional algorithmic methods, or fuzzy system methods, or neural methods (when adaptation is desirable). The complexity and intelligence of the system lies on the way the partial

outputs of the agents are merged, i.e. on the way the overall system output is produced. The approach adopted here is the assignment of a weight to each agent's output that will determine its contribution to the overall output. This way, the restriction of activating a single agent at each time is overcome and we are led to a combined decision depending on the degree that the advice of every agent is taken into account.

Assuming that the path planning system has  $k$  inputs (coming from the sensors) and  $k$  outputs (only one of which is non-zero each time (indicating the direction to be followed)), we can easily understand that this system has to implement a (very complex) function of  $k$  inputs and  $k$  outputs. Of course, if we additionally desire to control the velocity and/or the acceleration of the robot, more outputs would be required. With the proposed method we attempt to divide the universe of discourse into subsets, and to implement the subsystems that approximate this function in every one of these subsets. Some difficulties arise during the determination of the subsets in which every subsystem is supposed to operate. This partially results from the fact that these subsets may be overlapping. As it will be shown in the next section, we use heuristic rules to determine the optimal domains of discourse of every agent. To this end, we will implement the behaviour coordinator system based on fuzzy logic methods.

## 3. Development of a Path Planning System

In this section a system is developed that deals with the path planning problem in a completely unknown environment in the presence of obstacles, where some restrictions (that will be discussed later) apply. Let us assume that we must drive a robot in a two-dimensional space from a starting position to a target position. The only thing known about the target is the direction connecting the robot to it (not the target's actual position). To accomplish this task we make use of  $k$  distance sensors which are uniformly positioned on the robot, i.e. each sensor is directed  $360^\circ/k$  from the previous and the next one. Let's assume that a body-attached Cartesian coordinate system has its Ox axis along the direction of the target position. This is the direction of the first sensor. We force  $k$  to be multiple of 4 in order to achieve a smooth placement of the sensors along the primary axes. This technique was successfully used in [9] for solving the same problem. The system that will be presented here, is based on the method developed in the previous section and is pictorially represented in Fig. 2.



**Fig. 2.** The proposed path-planning system

The output of the  $k$  sensors is fed to the appropriate agents (the input to the fifth and the eighth agent is the previous states from the memory), and every agent assigns  $k$  priorities, one for every direction. The output of every agent is then multiplied by a weight that characterizes its degree of "responsibility" and is provided by the behaviour coordinator. The result is finally brought to the merging subsystem, where the priorities supplied by all the agents for a specific direction are multiplied. The threshold unit ensures that a direction leading closer to an obstacle than a prespecified value will certainly be rejected. This threshold value depends on the dimensions of the

robot and the nature of the specific problem at hand. This unit works in a binary way: either a direction is safe or not, since collision is not a fuzzy concept!

To conclude the algorithm, a step along the direction of maximum priority takes place, and the process is continuously repeated until the robot reaches the target position.

It is important to highlight that all the agents are functioning independently from each other and the only duty of the behaviour selector is to assign a degree of "responsibility" to each one of them. Thus, problem decomposition is taking place, since the original problem is divided into eight much

simpler subproblems, with the avoided complexity not being transferred to the task of recomposing the overall output from the partial outputs.

In the following we will analyse the operation of every agent along with the heuristic rules used to determine its domain of responsibility.

The first agent is a system that forces the robot to move close to the obstacles. This is necessary for two reasons. The first reason is that by getting close to the obstacles, the robot increases its resolution since it is easier to recognize small corridors or passes among the obstacles which may sometimes be the only way to reach the target position. The second reason is that this agent enables the robot to actually enter inside an identified corridor. This would have been avoided if no such rule existed since other directions would look more appealing. This agent is appropriate ("responsible") for handling situations where the distances measured by the sensors are similar (i.e. small differences between them appear).

The second agent is a system that forces the robot to move away from the obstacles, i.e. it favours the directions along which long distances are measured. This is useful firstly because the further the robot moves from the obstacles, the safer the produced trajectory is. Moreover, this provides a way to optimize our trajectories. This agent is responsible for situations where the obstacles are moving fast or when the distances measured by the sensors differ a lot.

The third agent is a system that forces the robot to move to the direction of the target. Its usefulness is obvious since this direction, along with its neighbouring ones, should be the most favourable. However, this does not always lead to desirable results, since when we are close to obstacles it is more important to avoid them than to move towards the target. So this agent is responsible for handling cases where there is enough space to move along the target direction and/or its neighbour directions.

The role of the fourth agent is to improve the trajectories with regard to the distance covered. Its logic is to avoid the obstacles while moving as close as possible to them. These directions can be identified by large differences between the distances measured along two successive directions. This agent is almost always "responsible" except of some situations that will be considered later.

The fifth agent is a subsystem that is activated periodically and checks if any progress has been made, i.e. if the current state relatively to the one evaluated during its last activation is "improved". If no significant progress is observed, a "reaction"

process is initiated, the role of which is to suppress the action of all the other agents for a specific number of steps, and to lead the robot to the most unknown directions (i.e. directions leading to positions it has the least visited before). This agent is very important and has a global responsibility since moving around with no progress is always undesirable and the agents that lead to this situation should be "punished". Punishment has the meaning of ignoring, for some time, what these agents suggest. The unknown positions are identifiable by the help of an associative memory implemented using neural network concepts.

The sixth agent is a subsystem that helps the robot avoid to continuously change its direction of motion. If a direction is chosen, the robot shouldn't easily change it except, of course, if another direction appears to be much more promising. This agent is responsible for handling situations where the direction leading to the target (and its close neighbours,) are forbidden due to the presence of obstacles at small distances.

The seventh agent is implemented in such a way that it drives the robot away from the target! Its presence in the system is necessary since this may be the best thing to do under certain circumstances. This is the system's defence against getting trapped into a "local minimum". Without the presence of such an agent, it would be difficult for the robot to achieve this. The circumstances under which this agent is to be responsible, are the ones where the third agent is the least responsible, i.e. the complement of the responsibility domain of the third agent.

The eighth agent is responsible to avoid getting involved into foreseen situations. Obviously, these should be avoided since they are examined in the past and a path, if existed, would have already been found. This agent also helps the robot to avoid deadends caused by continuously looping around the same positions. To this end, the neural associative memory mentioned earlier is used. By adopting neural methods the system inherits the very important feature of generalization, i.e. not only the already visited situations but also their neighbours will be avoided (to a lesser, of course, degree).

This set of agents is not, of course, the only possible set one can think of. According to the specific problem at hand and the specific demands, we may add or remove agents from the system. The system is flexible enough to allow us to directly insert or delete agents in a straightforward manner. The way each agent is implemented, can be determined according to the role this agent is supposed to play in the system. Fuzzy logic

provides a good solution when problems related to the accuracy of the sensor measurements occur. Neural networks, on the other hand, may enrich the system with learning/adaptation capabilities. Finally, simple algorithmic procedures may prove to be efficient enough under certain circumstances. The behaviour coordinator consists of eight different subparts, one for each agent, and provides to each one of them the heuristics needed to determine the domain of responsibility of that agent. The method used here is based on fuzzy systems theory for three major reasons:

- fuzzy logic provides the simplest way to translate heuristic rules to a computational algorithm,
- the system needs to deal with the uncertainty introduced by the sensor measurements, and
- the domain of responsibility of each agent is, by its nature, fuzzy.

Presently, we are working on the completion of this system via neurofuzzy techniques so as to improve the fuzzy part of the system. These methods have become quite popular recently and are continuously being improved. In [16] and [17] a quite representative sample of the work in this research area is presented.

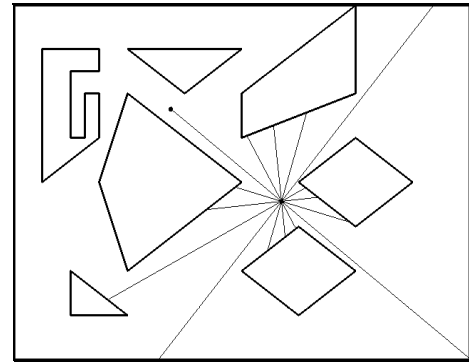
A first version of the system (without any neurofuzzy techniques included) has already been implemented and the results are being shown in the Figures 3 and 4.

#### 4. Results

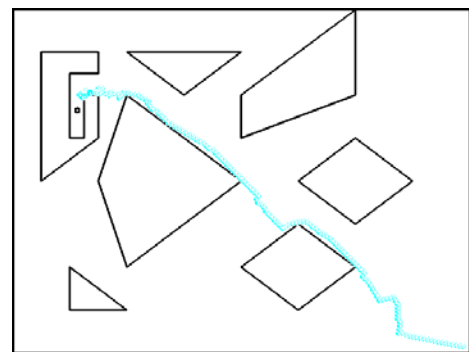
These figures illustrate the division of the space into  $k$  directions (here  $k=16$ ), and some demo paths obtained using a simulation program written in C++ language. In each case, various parameters were altered.

#### 5. Conclusions

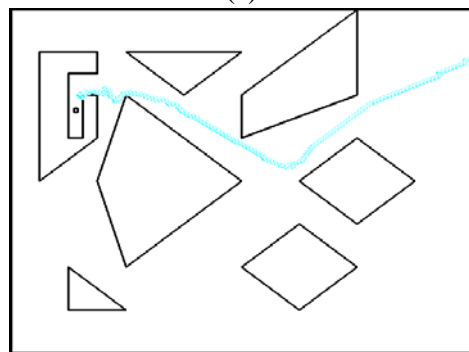
The method proposed in this paper is very general and may be applied in many robotic problems. One of the advantages of this method is its intrinsic flexibility, which extensively assists the implementation of the system. One of the disadvantages that we are trying to avoid through the use of neurofuzzy techniques, is the difficulty of determining the domain of responsibility of each agent, and the need of explicitly stating it by the appropriate heuristic rules.



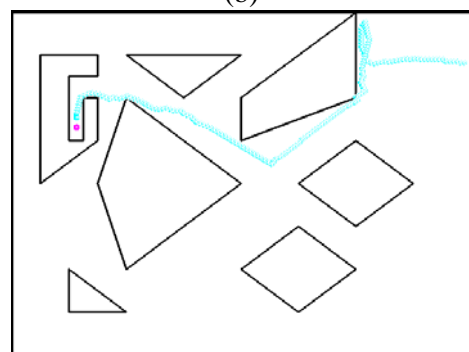
**Fig. 3:** The division to directions



(a)



(b)



(c)

**Fig. 4:** Some representative paths

#### 6. References

- [1] Lozano-Perez, T. and Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles", *Comm. ACM*, Vol. 22, No. 10, 1979.
- [2] Lozano-Perez, T., "Spatial Planning: A Spatial Configuration Space Approach", *IEEE Trans. Computers*, Vol. 32, No. 2, 1983.

- [3] Brooks, R. A. and Lozano-Perez, T., "A Subdivision Algorithm in Configuration Space for Findpath with Rotation", *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 15, No. 2, 1985.
- [4] Takahashi, O. and Schilling, R. J., "Motion Planning in a Plane Using Generalized Voronoi Diagrams", *IEEE Trans. Robotics and Automation*, Vol. 5, No. 2, 1989.
- [5] Kambhampati, S. and Davis, L. S., "Multiresolution Path Planning for Mobile Robots", *IEEE Journal of Robotics and Automation*, Vol. 2, No. 3, 1986.
- [6] Lin, C. T. and Lee, C. S. G., "A Multi-Valued Boltzmann Machine", *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 25, No. 4, 1995.
- [7] Tsai, W. H. and Chen, Y. C., "Adaptive Navigation of Automated Vehicles by Image Analysis Techniques"
- [8] Jarvis, R., "Distance Transform Based Path Planning for Robot Navigation", in *Recent Trends in Mobile Robots* (Zheng, Y. F., ed.), World Scientific.
- [9] Tzafestas, S. and Stamou, G., "A Fuzzy Model for Autonomous Robots Path Planning", *Proc. EURISCON'94*, Malaga, Spain, 1994.
- [10] Fujimura, K. and Samet, H., "A Hierarchical Strategy for Path Planning Among Moving Obstacles", *IEEE Trans. Robotics*, Vol. 5, No. 1, 1989.
- [11] Saridis, G. N., "Intelligent Robotic Control", *IEEE Trans. Automatic Control*, Vol. 28, No. 5, 1983.
- [12] Sawaragi, T., Itoh, K., Katai, O., and Iwai, S., "Integration of Symbolic Path-Planning and Fuzzy Control for Intelligent Mobile Robot", in *Fuzzy Logic* (Lowen, R. and Roubens, M., eds.), Kluwer, 1993.
- [13] Brooks, R. A., "A Robust Layered Control System For A Mobile Robot", *IEEE Journal of Robotics and Automation*, Vol. 2, No. 1, 1986.
- [14] Beom, H. R. and Cho, H. S., "A Sensor-Based Navigation for a Mobile Robot Using Fuzzy Logic and Reinforcement Learning", *IEEE Trans. System, Man, and Cybernetics*, Vol. 25, No. 3, 1995.
- [15] Minsky, M. L., "Society of Mind", Simon & Schuster, New York, 1986.
- [16] Berenji, H. R. and Khedhar, P., "Learning and Tuning Fuzzy Logic Controllers Through Reinforcements", *IEEE Trans. Neural Networks*, Vol. 3, No. 5, 1992.
- [17] Lin, C. T. and Lee, C. S., "Reinforcement Structure/Parameter Learning for Neural-Network-Based Fuzzy Logic Control Systems", *IEEE Trans. Fuzzy Systems*, Vol. 2, No. 1, 1994.