LECTURE NOTES IN CONTROL AND INFORMATION SCIENCES
Series Editor: M. Thoma
Series advisory board: A. Bensoussan · M.J. Grimble · P. Kokotovic · H. Kwakernaak · J.L. Massey · Ya.Z. Tsypkin

This series aims to report new developments in the fields of control and information sciences – quickly, informally and at a high level. The type of material considered for publication includes:

1. Preliminary drafts of monographs and advanced textbooks
2. Lectures on a new field, or presenting a new angle on a classical field
3. Research reports
4. Reports of meetings, provided they are
   a) of exceptional interest and
   b) devoted to a specific topic

The timeliness of subject material is very important.

The publication of Lecture Notes is intended as a service to the international scientific and engineering community, in that a commercial publisher, Springer-Verlag, can offer a wider distribution of documents which would otherwise have a restricted readership. Once published and copyrighted, they can be documented in the scientific literature.

TYPESCRIPTS

Typescripts should be written in English and be no less than 100, preferably no more than 500, typewritten pages in length, including illustrations. Lecture Notes are printed by photo-offset from printed copy delivered in camera-ready form by the authors/editors. This process does not include a copy-editing stage; therefore it is the responsibility of authors/editors to ensure the typescripts are as error-free as possible and printed on a laser or laser-quality printer with at least 300 dpi. Careful preparation will help keep production time to the minimum and ensure satisfactory appearance of the finished book.

For best results, text and illustrations in the Camera-Ready Copy should fill, but not extend beyond, an area of 120 × 195mm. Use a 10-point font size with line spacing of 2 points. A traditional serifed typeface (e.g. Times) is preferred. The text should be justified on both left and right margins. Headings should be capitalized and should, with the exception of the title, be aligned to the left. The font sizes preferred are: Title – 14pt bold; 2nd level – 12pt bold; 3rd level – 10pt bold. Line drawings should be submitted as clear, well-drawn originals, not a sketch or photocopy. They must be positioned on the Camera-Ready Copy typescript centred on the type width. Labelling should not be set smaller than 8pt unless previously agreed with the Springer-Verlag editor. The use of halftone illustrations is, in general, discouraged. However, if such illustrations are considered vital to the work, advance approval from the Springer-Verlag editor is required.

Authors of published monographs receive 30 free copies; editors of published multi-authored works and proceedings receive 20 free copies. Springer-Verlag retains copyright. However, all authors are free to use the material in other publications, with acknowledgment to Springer-Verlag.

Typescripts should be sent to the series editor, Professor Dr.-Ing. M. Thoma, Institut für Regelungstechnik, Technische Universität, Appelstrasse 11, D-30167 Hannover, Germany, or directly to the Engineering Editor, Springer-Verlag London Limited, for review.

Springer-Verlag, Heidelberger Platz 3, D-14197 Berlin, Germany
Springer-Verlag, Tiergartenstrasse 17, D-69121 Heidelberg, Germany
Springer-Verlag London Ltd, Sweetapple House, Catteshall Road, Godalming, Surrey GU7 3DJ, UK
Springer-Verlag, 175 Fifth Avenue, New York, NY 10010, USA
Springer-Verlag, 37-3, Hongo 3-chome, Bunkyo-ku, Tokyo 113, Japan

ISBN 3-540-76133-0

LNCIS 223 O. Khatib and J.K. Salisbury (Eds) Experimental Robotics IV

Lecture Notes in Control and Information Sciences 22

O. Khatib and J.K. Salisbury (Eds)

Experimental Robotics IV

The 4th International Symposium, Stanford, California, June 30 – July 2, 1995

Springer

Cooperative Autonomous Low-cost Robots for exploring Unknown Environments

Josep Amat
Automatic Control Department, UPC
Barcelona, Catalonia (Spain)
amat@esaii.upc.es

Ramon López de Màntaras, Carles Sierra
IIIA - Artificial Intelligence Research Institute, CSIC
Bellaterra, Catalonia (Spain)
{mantaras, sierra}@iiia.csic.es

Abstract

In this paper we present the results obtained with a troup of small autonomous vehicles designed to cooperatively explore unknown environments. In order to improve the covering of the explored zone the vehicles show different behaviours. The host that controls the troup of vehicles generates the most plausible map of the environment from the information obtained by the different components of the troup, which at the end of their mission return back. To perform the map generation a two-step algorithm, fusion and completion, based on fuzzy techniques, is presented.

1. Introduction

With the aim to explore an environment that is unknown but easily passable, a system constituted by a set of low cost, small autonomous vehicles has been developed. These vehicles follow the already classical line of insect robots [1] [2] [3] [4] [5] [6]. The goal of these autonomous vehicles is to obtain partial information about the environment during their exploration runs and afterwards to supply this information to a Master Autonomous Robot that in turn will be able to compute the most plausible map. With this information, the Master should be able to perform a given mission within structured environments. Using this Master-Multislave strategy to generate a model of the environment, we expect to achieve a better efficiency and a safer procedure than that which would be obtained based only on the Master perception capabilities.

The behaviour of these small autonomous vehicles has been programmed in an individual basis for obtaining a behaviour similar -to some degree- to that of ants in two aspects. First, in order to increase the coverage of the environment, the vehicles have a partially random moving behaviour; and second, the vehicles cooperate in the process of environment information gathering by transfering each other the perceived

environment when they meet. Sharing data in this way, allows the Master to get the information not only from the vehicles that successfully return after an exploratory run, but also from those that cannot return, provided that they have encountered vehicles that have safely returned.

The sensing capability of each small vehicle, to build its own partial map, comes from two kind of sensors: IR proximity sensors for environment data acquisition, and a relatively accurate odometric system for the estimation of the vehicle position during its run.

The next section of this paper describes the structure and the behaviour of the vehicles. Section 3 describes the fuzzy logic-based algorithms that we have developped in order to compute the most plausible map based on the partial maps perceived by the succesfully returning vehicles. Section 4 describes the results obtained to date and in section 5 we briefly point to some future work.

2. Structure of each mobile vehicle

Each vehicle has been designed with the aim of being small and cheap. It must have a high autonomy and be endowed with a computer, with which they can get and memorize, with the highest resolution, the observed portion of the environment. All these requirements have lead to a compromise solution consisting on small vehicles with three wheels. Two of them are steering wheels having independent motors.

The vehicles environment perception system and the communications with the host or with the other vehicles are based on IR impulse modulated sensors. Since some of the monirobots may not return to deliver their map to the Master robot, the following communication process is established: when two of them meet along their exploration run, they back-up from one to the other all the information they have acquired so far from the environment. In this way, when a vehicle reaches the host, it delivers both, the information acquired during its own run as well as the information obtained from other vehicles that it has encountered. This communication process allows to get all the information of a non-returning minirobot that had been transferred to a returning one.

2.1. Mechanical characteristics

The vehicle is 21 cm. long and 15 cm. wide (see Fig. 1). The 5cm. driving wheels allow the vehicle to save some small obstacles such as carpets or electrical wires. With the motors utilised, the vehicles can reach a speed up to 0.6 m/sec., and since the battery has a one hour autonomy at full regime, each vehicle can do a run of about 2000 m. long.

2.2. Sensing Capability

The vehicle is equipped with the following sensing elements:

- Impulse generators at each wheel for odometry.
- Five I.R. proximity sensors for obstacles detection.
- A proximity sensor to detect terrain horizontal discontinuities.
- Safety microswitches to detect collisions.
- One omnidirectional IR Emitter/Receiver sensor to detect other vehicles and to transmit data.

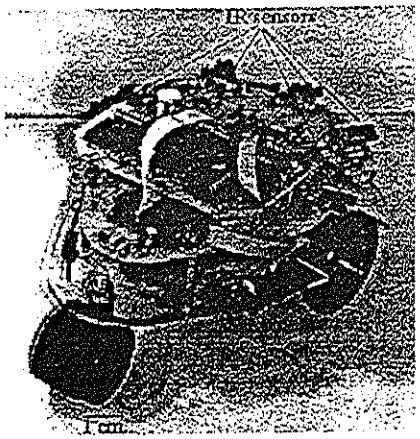


Figure 1. Autonomous Mini-Robot structure

- One IR Emitter with a sector scope of 90° to generate a priority signal (right hand preference).

The obtained odometric precision produces a 2x20 cm. uncertainty ellipse after a 10 meters run without direction changes. In order to use the data of this particular odometer, to get at each instant the vehicle position and also the environment information, a quality factor has been experimentally elaborated. This quality factor depends on the distance traveled L and on the number of turns N done. This quality factor is:

$$FC \quad \frac{1}{(1-\frac{L}{K_l})(1-\frac{N}{K_n})}$$

K_l and K_n are constant parameters that control the rate of decay of the quality factor.

When the vehicle detects and follows a wall, the corresponding information is incorporated into memory together with its associated quality factor. The host will generate a map of the environment by integration of the data supplied by the different vehicles in an efficient way (see Section 3).

2.3. Navigation Strategy

The navigation system incorporated to each vehicle has a random behaviour: The vehicle does a $\pm 45°$ or $\pm 90°$ turn, either randomly or when it detects an obstacle.

The random turns are done according to three significantly different probabilities: $P_1 > P_2 > P_3$, modelling three different behaviours:

- P_1 vehicle with an "anxious" behaviour.
- P_2 vehicle with "normal" behaviour.
- P_3 vehicle with "routine" behaviour.

When the vehicle finds a frontal obstacle, the turn can be done to the right or to the left based also on a probability value P_4. The vehicles having a probability $P_4 < 0.5$ will show a tendency to turn right more often than to turn left, whilst the vehicles having a probability $P_4 > 0.5$ will behave inversely.

Consequently, the different vehicles of the exploration troup will not show an identical behaviour. They can behave in six different ways corresponding to the different combinations of behaviours and turning tendencies. By using a troup of vehivles showing different behaviours, we expect to obtain a better covering of an environment than with vehicles showing the same behaviour.

Once a vehicle has run a length L_{P_i}, depending on the probability P_1, P_2 or P_3, where $L_{P_1} < L_{P_2} < L_{P_3}$, it starts its way back towards the host.

The algorithm used to guide the vehicle back towards the starting point generates a trajectory following an improvement of the travelled path contained in memory (in order to assure a path possibly free of obstacles). This improvement consists in eliminating loops.

2.4. Control System

The control unit in each vehicle has been designed having in mind that the hardware had to be as simple as possible but, on the other hand, it had to allow achieving a behaviour sufficiently smart in order to navigate efficiently. Furthermore the vehicle had to be based on a hardware flexible enough to allow for experimentation of navigation and control strategies. These requirements have resulted in a design which contains three different functional modules: The *navigation module* generates the trajectory to be followed; the *steering module* controls the motors in order to follow the generated trajectory; and the *perception module* acquires information of the environment by means of IR sensors. However, it is possible to replace this module by other modules adapted to different types of sensors.

The computer used to implement the navigation control unit is a 80C186 with a 1MB RAM to store the data of the perceived portion of the environment. The environment is discretized with a resolution of 4 cm which means that each vehicle can store maps of up to 40x40 m.

The steering control module operates with a much higher resolution since each encoder corresponds to a displacement of only 2 mm. and it is implemented on a 80C552

3. Environment map generation

The goal of map generation is to obtain the most plausible position of walls and obstacles. The information about their position conveyed by the different vehicles is imprecise. Furthermore, vehicles can detect portions of walls or obstacles with different degrees of imprecision. The main problem is to decide whether several detected portions, represented by imprecise segments, belong to the same wall or obstacle or not. This decision depends on the relative position of the segments as well as on their imprecision. The relative position of the segments can be represented by their euclidean distance. The distance is compared to a threshold to decide whether or not the segments represent different portions of the same wall or obstable. If two segments represent the same wall or obstacle, a segment fusion procedure is applied to produce a single segment. This process of segment fusion is followed by a completion process in which hypothesis are made with respect to non observed regions. The completion process is performed by means of hypothetical reasoning based on declarative heuristic knowledge about the structured environments in which vehicles evolve.

The map generation algorithm consists of two steps. The first one is the fusion of the map perceived by each vehicle, taking into account that the same vehicle can observe more than one portion of the same wall. The second step consists in a global fusion and completion of the maps perceived by several troup members. The fusion function is the same in both steps. However, since not all troup members may return home to give the information to the master and since, on the other hand, we want to have at any time the most plausible map based on the information obtained so far, it is necessary to develop an incremental approach to the map generation algorithm. Any time a vehicle returns home the algorithm is executed to update the map. The overall algorithm is as follows:

```
Function Map_generation(NewAntsMap, CurrentMap) =
  Begin
    NewCurrentMap = Fusion(CurrentMap ∪ Fusion(NewAntsMap, α, β), α, β);
    Return Completion(NewCurrentMap)
  End
```

where α and β are decision threshold parameters that will be explained later. In section 3.1 we give details of the segment fusion procedure and in 3.2 we outline the completion process.

3.1. Segment fusion

Let us start by defining the basic object of our approach which is the imprecise segment.

An *imprecise segment* S is a function that gives for any cartesian coordinates (x, y) the degree of possibility of the coordinates as being part of a wall. That is $S : \Re \times \Re \rightarrow [0, 1]$. This function can be seen as a possibility distribution [7] in the sense of fuzzy logic [8] and is determined by a segment that, for simplicity, we assume has a precise length, plus the imprecision measure of its position with respect to both axis. Furthermore, for each segment we keep the list of coordinates of the singular points that have been detected (i.e. corners and gaps in the real world).

Given the orthogonality of the structured environment, we have only two possible orientations for a segment: vertical or horizontal. That is $S = ((x, y_1), (x, y_2), e, p_1, ..., p_n)$ or $S = ((x_1, y), (x_2, y), e, p_1, ..., p_n)$. Where e is the imprecision and the p_i are the singular points. For simplicity, in the sequel we drop the list of singular points from the notation of segments.

For example, an imprecise horizontal segment $S = ((x_1, y), (x_2, y), e)$ means that there is a portion of a wall with coordinates $((a, b)(c, b))$ such that $y - e < b < y + e$ and $x_1 + e > a > x_1 - e$ and $x_2 - e < c < x_2 + e$. Similarly for the orthogonal case. Figure 2 shows an example of an imprecise horizontal segment.

As we have said, the main problem is to decide whether several imprecise segments represent portions of the same wall or obstacle or not. This decision depends on the relative position of the imprecise segments, like for example the case shown in figure 3.

$$s = ((x, y_1), (x, y_2), e), s' = ((x', y_1'), (x', y_2'), e')$$

This relative position is represented by two parameters Δ_x and Δ_y that represent the minimum distance with respect to the two axis:

$$\Delta_x(s, s') = |x - x'|$$

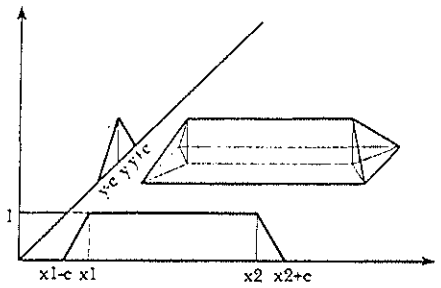
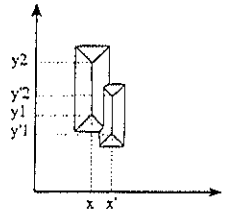


Figure 2. Horizontal Imprecise segment



A

Figure 3. Example of two overlapping imprecise segments

$$\Delta_y(s, s') = \begin{cases} 0 & \text{if } (y_1 - y_2') * (y_2 - y_1') < 0 \\ min(|y_1 - y_2'|, |y_2 - y_1'|) & \text{otherwise} \end{cases}$$

In the Δ_y expression, the first part stands for the case in which both segments overlap. In that case both terms in the product have a different sign.

Given a map M, represented as a list of segments ordered by imprecision, the "Fusion" function sketched below computes a new map M' whose difference with M is that some segments have been fused.

To do that, for each segment s, starting with the most precise, we compute its distance to the segments in M' with respect to both axis. Then, with those segments in M such that both distances are below a threshold we build the list of candidates for merging with s (the $MayMerge$ list).

Next, the function $Select\_Min\_Dist$ selects for fusion the segment s' in M' with the lowest average distance with respect to both axis. Finally, we merge s and s' with the Merging fuction and we replace s' in M' and s in M by the resulting merged segment s''. Furthermore, the singular points in M and M' are updated in order to keep the singular points of the merged segments properly connected with other segments meeting at these singular points. Obviously, in the case where the $MayMerge$ list for s is empty we simply add s to M'. This process is iterated while $M' \neq M$.

```
Function Fusion(M, α, β) =
  {M is a Map represented as a list of segments ordered by
  imprecision, α and β are decision thresholds}
  M' : Map
  Begin
    M' = M;
```

```
M = nil;
While M' ≠ M do
   M = M';
   M' = nil;
   Foreach s in M
      MayMerge = nil;
      Foreach s' in M'
         If Δx(s, s') < α  and  Δy(s, s') < β Then MayMerge = MayMerge + s'
         endif
      Endforeach
      If MayMerge = nil Then M' = M' + s
      Else  s" = Select_Min_Dist(MayMerge, s);
            M' = M' - s" + Merging(s", s)
            If singular(s") or singular(s) Then
               M = update_sing_points(M,s",s);
               M'= update_sing_points(M',s",s)
            Endif
      Endif
   Endforeach
Endwhile;
Return M'
End
```

The function $Select\_Min\_Dist(MayMerge, s)$ selects the segment s', in the list $MayMerge$, that has the minimum average distance to s with respect to both axes, that is, $\Delta_x(s,s') + \Delta_y(s,s')/2$.

The function $Merging(s", s)$ checks whether the orientation of the imprecise segments to merge is vertical or horizontal and uses the appropriate function: $Vert\_Merging$ or $Hor\_Merging$.

The vertical merging function is sketched next (the horizontal one is similar):

```
Function Vert_Merging (s, s') =
   {s = ((x, y1), (x, y2), e, p1...pn), s' = ((x', y'1), (x', y'2), e', p'1...p'm)}
   Begin
      y"1 = min(y1, y'1);
      y"2 = max(y2, y'2);
      e" = e * e' /(e + e');
      dist = abs(x − x');
      x" = e * Dist/(e + e');
      For i = 1 to n do p"i = (x", projy(pi));
      For i = 1 to m do p"i+n = (x", projy(p'i));
      Return ((x", y"1), (x", y"2), e", p"1...p"n+m)
   End
```

The coordinates $y"_1$ and $y"_2$ are obvious. We can easily see that $x"$ is the center of gravity of the masses $\frac{1}{e}$ and $\frac{1}{e'}$ of s and s', the x coordinate of $p"_i$ is $x"$ and the y coordinate $p"_i$ is the y coordinate of p_i, and $e"$ is the degree of imprecision of the merged segments computed as a function of e and e' in such a way that the mass of the merged segments $\frac{1}{e"}$ is the sum of the masses of the components of the merge $\frac{1}{e} + \frac{1}{e'}$, however alternative functions [9], like for example $\frac{1}{e"} = max(\frac{1}{e}, \frac{1}{e'})$, are also possible. Which function to use can only be decided experimentally. Figure 4 shows an example of vertical segments fusion.

Figure 5 graphically shows the merging result (B), along the x axis, for two vertical segments (A) with no singular points

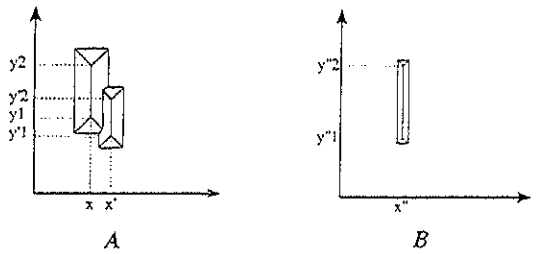


Figure 4. Vertical segments in A are fused into the segment in B

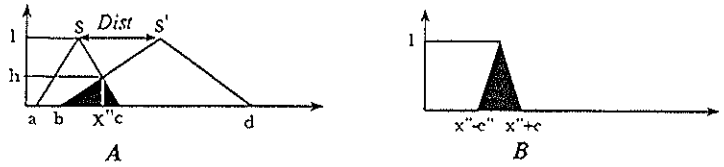


Figure 5. Merging of two vertical segment projections along the x axis

3.2. Global environment map completion

The fusion procedure is an initial step that only merges segments on a local basis using the geometrical properties of the segments. To improve the map, we perform a completion process based on a global view of the environment, including the map that each vehicle has from the other vehicles that it has encountered during its exploration, taking into account knowledge about the characteristics of the environment and knowing the paths followed by the vehicle. We adopt an heuristic knowledge-based approach. One example of such knowledge is expressed by the following heuristic rule:

```
If there exists a segment s1 and
      there exists a segment s2 and
      s1 and s2 are parallel and
      there are no singular points in the closest extremes of s1 and s2 and
      Δx(s1, s2) < α and
      k × β > Δy(s1, s2) > β and
      there is no vehicle path crossing the gap
Then Assume(Vert_Merging(s1, s2))
```

A set of fifteen rules covering many different situations has been identified.

4. Results

A first prototype of an autonomous vehicle has been physically built and tested. Comparing the environment map with obstacles obtained through IR sensing in real operating conditions with the real map of the environment allowed us to evaluate the accumulation of error due to the odometer and then to determine the values of the parameters K_l and K_n (see 2.2). This allowed us to compute the quality factor that is memorized for all segments.

Figure 6 shows different coverings of the environment obtained by computer simulation of two different behaviours (anxious and routine). We can observe that

the percentage of covered environment is very high with a relatively short running time. This is true for a major part of the experimented environments.

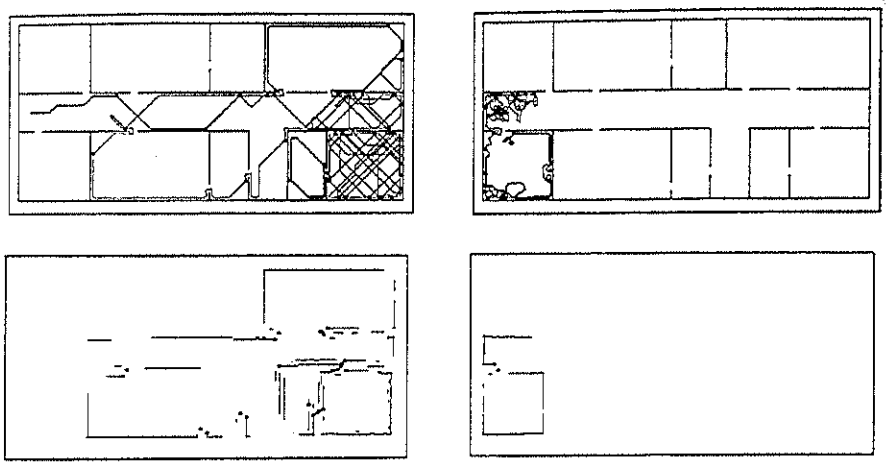


Figure 6. Examples of paths followed and maps generated with routine and anxious behaviours

Figure 7 shows the results obtained after applying the fusion process to a set of maps, including those of figure 6.

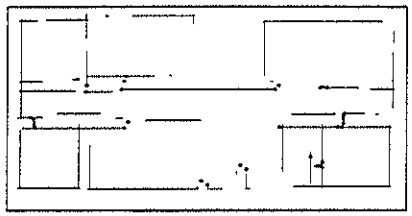


Figure 7. Fusion process result

5. Future work

The results obtained with this first prototype are encouraging enough to undertake the construction of a whole troup consisting of 15 vehicles. This troup will allow us to perform exhaustive experimentation with different number of vehicles and with different behaviour combinations. Concerning map generation, future work will be focused on refining fusion and implementing the completion procedure. Another interesting future work will be to set up the conditions under which it may be advisable to send further ants to explore those portions of the environment that remain incomplete.

6. Acknowledgements

We acknowledge the contributions of Francesc Esteva in the discussions concerning the fuzzy logic approach to map generation as well as the contributions of Gil Arbós, Maite López, Albert Sánchez and Angel Toribio who have worked hard in the implementation and experimentation of the prototype.

References

[1] R. Alami, R. Chatila, B. Espiau, Designing an Intelligent Control Architecture for Autonomous Robots, ICAR'93, Tokyo, 1993.

[2] R. A. Brooks, A Robust Layered Control System for a Mobile Robot, IEEE Journal of Robotics and Automation, RA-2, pp. 14-23, 1986.

[3] R. A. Brooks, Intelligence Without Reason, Proc. of IJCAI'91, pp. 569-595, 1991.

[4] R. A. Brooks, L. A. Stein, Building Brains for Bodies, Memo 1439, MIT, AI Lab., Cambridge, Massachussets, 1993.

[5] B. R. Donald, J. Jennings, D. Rus, Information invariants for cooperating Autonomous Mobile Robots, in Robotics Research, the Sixth International Symposium, pp. 29-48, Pittsburgh, USA, 1993.

[6] B. Jouvencel, J.E. Symphor, The variable Modelling of Mobile Robot Environments, IEEE IROS'91, Osaka, Japan, 1991.

[7] R. López de Màntaras, Aproximate Reasoning Models, Ellis Horwood Series in Artificial Intelligence, GB, 1990.

[8] L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems, 1, 3-28, 1978.

[9] D. Dubois, J-L. Koning, Social choice axioms for fuzzy set aggregation, Fuzzy Sets and Systems, 43, 257-274, 1991.