Path Planning for a Fuzzy Controlled Autonomous Mobile Robot

Hartmut Surmann, Jörg Huser and Jens Wehking

German National Research Center for Information Technology,
53754 Sankt Augustin, Germany, Tel:++49/2241/14-2518, Fax: -2342,
E-mail: surmann@gmd.de.

Abstract

Path planning is one a crucial task in navigation of autonomous mobile robots. This paper proposes a high-level path planning approach based on a fuzzy navigator in uncertain environments which is indepenend of crisp metric information. The navigator acquires information about the environment with eight ultrasonic sensors. The information are evaluated by the navigator utilizing fuzzy rules. In order to acquire the information about the environment around the mobile robot MORIA, the planner only uses computed state variables of the navigator combined with fuzzy metric information. The fuzzy state variables contain information about the actions of the robot, relative fuzzy distances since the last fuzzy state interaction and information according to the current situation. The roles of the navigator and planner are comparable with the behaviour of a car driver and his co-driver reading a map. The approach is developed in a series of simulations and tested on the robot MORIA.

I. Introduction

Incorporating inexact reasoning into usable form can present problems of imprecision or uncertainty. Fuzzy theory addresses such problems by attaching measures of credibility to propositions. Mamdani's work on fuzzy control [1], which was motivated by Zadeh's approach to inexact reasoning [2], led many researchers to work in this field. The basic idea of this approach was to incorporate the knowhow of a skilled human operator into fuzzy sets (FS) and fuzzy rules (FR), which would then be combined by the fuzzy implication and the compositional rule of inference.

An autonomous mobile robot (AMR) has to cope with uncertain, incomplete or approximate information when going about its tasks as a transportation or cleaning robot in administrative buildings, factories or hospitals. Moreover, it has to identify sudden changes that it perceives in its surroundings, and react and maneuver in real time [3], [4]. A control system utilizing fuzzy rules has recently been developed in this laboratory. It has been demonstrated to be able to guide a real AMR through a network of corridors of varying widths, containing occasional random obstructions such as fire extinguishers [5] (Robot Intelligence Award).

This work is supported by TZN Unterlüß, Germany.

Inaccuracy of different components *eg*, sensor measurements and motor drift makes the problem of planning and controlling autonomous mobile robots very complicated. Most path planning approaches base on high precision, metric approaches and demand very reliable and accurate sensor devices as well as large computational power, *eg* [6]. In addition, these methods have difficulties during navigation in complex and dynamically changing environments where unknown obstacles appear on an a-priori planned path [7], [8]. The conventional path planning approaches for mobile robots may be divided into two categories. One is the global path planning based on a-priori complete information about the environment. The other is the local path planning based on sensor information in uncertain environment where size, shape and location of obstacles are unknown.

Fig. 1. **The autonomous vehicle MORIA.**

Here, we outline our proposal for a two-level integrated approach that imitates a driver (navigator) and co-driver (planner) in an unknown simple net-

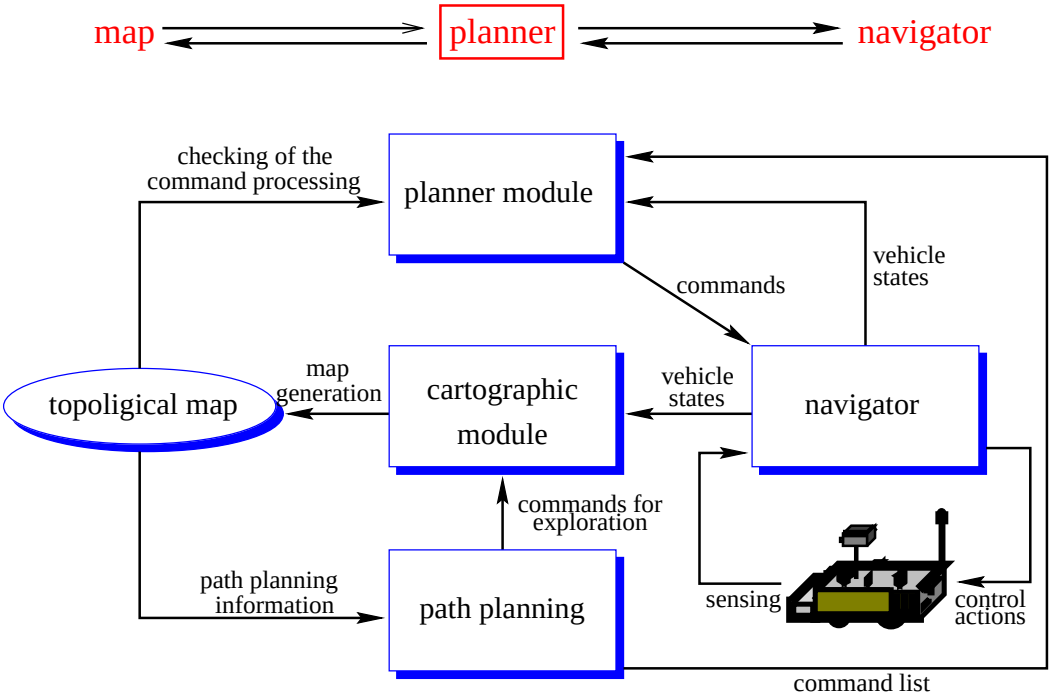


Fig. 2. **MORIA's embedded modules.**

work of corridors. The navigator can successfully drive the robot through the network of corridors, avoiding collisions with walls or other obstacles, responding to simple commands from the planner such as "go straight ahead" or "take next left" but has no the path-finding sophistication. The co-driver disposes of a topological map of the network of corridors and it has the path-finding sophistication to plan a route for the robot. The plan given to the navigator is a list of commands *eg*, "straight", "left", "straight" etc. The planner guides the navigator by these simple commands. The navigator itself has the capabilities to identify the nature of the surroundings, to recognise the current command and to combine this knowledge to select the next behaviour pattern.

Before we discuss the planning approach in section III, a brief overview of MORIA's architecture and the fuzzy navigator is given. We do not introduce fuzzy rule-based systems (FRBS). An extended introduction to FRBS can be found in many good textbooks, *eg* [9], [10].

II. The Robot MORIA

The vehicle MORIA used in our experiments is 157cm x 90cm x 75cm in size (including the bumper 24 cm), and has a weight of 400kg with a payload of 150kg. It is equipped with 8 ultrasonic sensors which supply the distance information required for navigation in a range of 50 - 400 cm. The vehicle is driven by two motors acting on a single wheel, one motor (reversible) to provide the driving torque, and the other motor to steer the vehicle by changing the orientation of the driving wheel. The external appearance of the robot is shown in figure 1. Computational capabilities of MORIA are based on an industrial PC (486/66 MHz, 16 MBytes) with extended I/O possibilities. The PC board collects the output of the sonar sensors and controls the two motors. A communication link to other mobile platforms or remote mainframes are possible through the on-board installed infrared sensor. The basic architecture of the autonomous platform is shown in figure 2.

The Fuzzy Navigator

Traditional navigators available for autonomous indoor vehicles are based on strategies that require a precise geometric map of the environment [11], [12]. Moreover, additional landmarks are often necessary to compensate uncertainties arising from control errors, sensor noise or inaccurate models of the environment [13]. Our strategy uses a reactive and goal-oriented navigation system based on fuzzy set theory [2], [3]. Through this approach, environmental uncertainties are compensated. The planner specifies the global driving direction by means of linguistic instructions, *eg*, "turn left at the next junction," while the navigator explores the current surroundings and selects the appropriate local path [14]. Unlike conventional systems, the navigator reacts on environmental

changes while maintaining the given global direction. Actual perceptions, linguistic instructions and the local history of the robot are combined and stored in fuzzy state variables. These state variables build the memory of the fuzzy controller and will recur in the following control loop. As different behaviours are predefined within the recurrent fuzzy system [15], the occurrence of deadlock situations like blocked corridors are easily solved. Thus, the computational rule of inference in the fuzzy algorithm allows the processing of apparently contradictory information.

To allow these effects, the fuzzy inference engine in the digital fuzzy controller requires more inputs than just the sensor inputs; it also needs fuzzy variables representing the current state of the controller and the robot, such as whether a branch has been detected, whether the robot is in the process of making a turn and so on. By the same token, the fuzzy controller requires more outputs than just the identification of the set of rules; it needs to output variables to signal other outcomes of its inference process (*eg* has a turn been completed, has a branch been passed without being taken *etc.*) for use by the higher level route planner, or for use by itself as inputs in its next inference round (table I). This strategy has previously been shown to be effective [5].

TABLE I

The high-level, or goal-oriented commands supplied by the navigator routine to the planner.

Planner/Command	Fuzzy number
Stop (s)	$0 \in [-0.5, 0.5]$
Straight Ahead (a)	$1 \in [0.5, 1.5]$
Take Next Left (l)	$2 \in [1.5, 2.5]$
Take Next Right (r)	$3 \in [2.5, 3.5]$
Go Backward (b)	$4 \in [3.5, 4.5]$
Turn Left Immediately (li)	$5 \in [4.5, 5.5]$
Turn Right Immediately (ri)	$6 \in [5.5, 6.5]$
Go Forward (f)	$7 \in [6.5, 7.5]$
Change Direction (cd)	$8 \in [7.5, 8.5]$
new: Last Left (ll)	$9 \in [8.5, 9.5]$
new: Last Right (lr)	$10 \in [9.5, 10.5]$

Commands 4-7 are "manual" commands, where the response contains no automatic collision avoidance strategy.

To obtain the degree of sophistication of the navigation system some level of memory is necessary. This can be provided by a digitally based controller which affects the behaviour pattern of the fuzzy controller. The purpose of the navigator is then to

1.) identify the nature of the surroundings
2.) remember the current behaviour pattern
3.) recognize the current goal-oriented command
4.) combine the above knowledge to select the next behaviour pattern.

III. Path Planning

The more a system goes ahead according to a precise plan, the more effective it is surprised by hazard. Human beings explain paths in an unknown environment (*eg*, buildings) fuzzy like "go straight ahead" and then "take second corridor left" etc. We have the capabilities to reduce the problem complexity to the necessary information. Obviously, metric information about precise distances are not necessary. Since our navigator "understands" linguistic high-level commands, path planning can concentrate on the generation of high-level description lists with the commands shown in table I. On this first level, a map of an indoor environment is represented in a graph which is a collection of related corridors. Figure 3 shows an example of the command list and the corresponding path run by the navigator. The path planning algorithm acts like a vec-

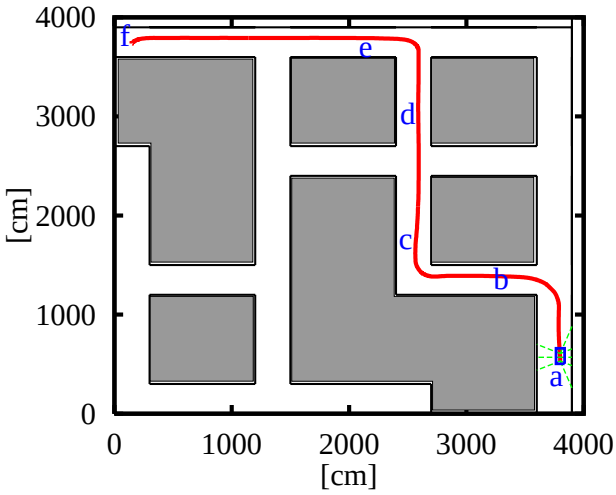


Fig. 3. **Calculated commands of the planner and executed path of the navigator (a: take next left; b: take next right; c: straight ahead; d: take next left; e: straight ahead; f: stop).**

tored graph search and is presented in the appendix. Finding the best route is an NP complete problem, so that the search algorithm has to be restricted. The restriction is given in real world buildings because the number of corridors is small (< 100). For about 100 corridors is the calculation time less than one second. In addition to corridors, the planner can also determines a relative coordinate for the navigator to conserve the graph representation in non corridor enviroments *eg* empty areas.

IV. Explorations

Robotic exploration and map-based navigation is difficult to accomplish over large spatial scales. Geo-

metric approaches use the distance information from several sensors to build geometric maps mostly based on probability theory *eg* [16], [17]. These approaches need a lot precise information and they have difficulties to distinguish between static and dynamic objects. An off-line generation of a topological map out of the geometric information can lead to errors. Exploration as a graph construction without distance information, *eg* [18], needs to drop distinct markers by the robot.

Our approach explores different levels of maps simultaneously without dropping distinct markers. Each map type has limitations that are relevant to the exploration tasks. Robust performance in both maping and navigation means not only that performance should be excellent when all resources are plentiful but that performance should degrade gracefully when resources are limited. On the graph level, we try to extract places, paths and their fuzzy metric relations based on the fuzzy state variables [5] of the recurrent fuzzy controller. On the lowest level we store the robot sensory information. The navigator uses the corresponding map according to the required information. Figure 4 shows a topological map and figure 5 a sensor reading of the same simple test environment. The position error of the robot after one loop is around 2.0m in the sensor plot. Important for a further path planning is that the topological map shows the relationships between the corridors, even if the metric information is only fuzzy.

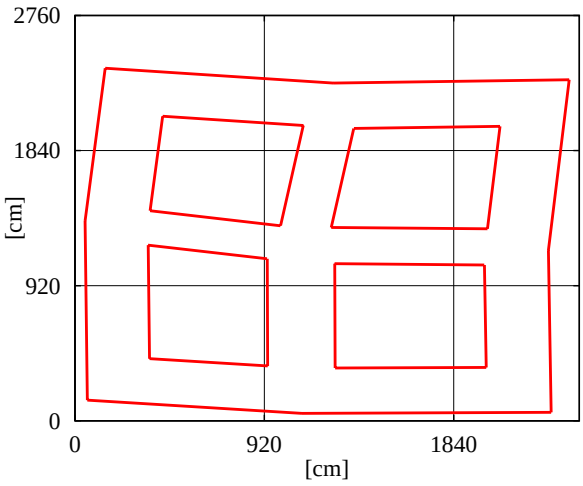


Fig. 4. **Explorated topological map of the planner.**

New Commands

One problem during exploration is the fast reaction-time. The robot immediately has to turn in a corridor after the navigator detects a left or right branch. Since MORIA drives with a maximum speed of $1.0\frac{m}{s}$ the robot has to make a hard break and turn.

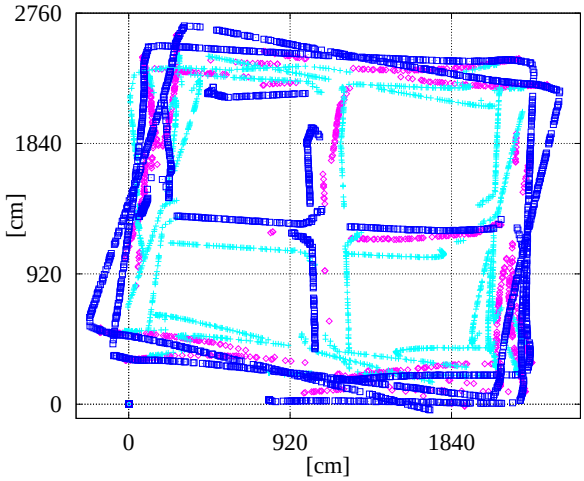


Fig. 5. **Sensor plot.**

To avoid such terrible driving style we implemented two new commands called “take last left” and “take last right” (table I). Figure 6 shows the behaviour of the robot during exploration given the new command “take last left”. When the robot detects the left branch it rolls to a stop (a), drives backwards (b) and turns left.

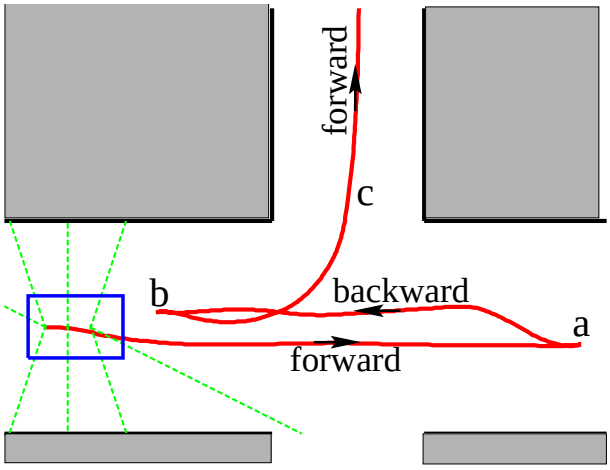


Fig. 6. **Robot behaviour for the new high-level command “last left”.**

V. Development and Simulation Environment

The development and simulation environment used for this project was basically developed for the digitally implemented fuzzy control system reported in [5]. The modeling system FunnyLab [19] was used for the creation and editing of the membership functions and the rule base. The knowledge base file produced by FunnyLab was read into a simulation environment

developed in-house at GMD, which includes not only a fuzzy inference engine, but also a motion simulator which incorporates a model of the real, measured dynamics of the robot MORIA. A graphical user interface which shows a plan view of the robot's progress through the test environment, displays the values of certain key parameters. It allows the instantaneous entry of the planner commands given in table I.

The motion simulator has been demonstrated to produce an accurate representation of the dynamics of MORIA during the earlier development work, and allows a realistic assessment of the effects of the fuzzy controller, given that it includes the effects of the inertia and finite response time of the robot.

The user interface includes an additonal feature which allows single-stepping. At each step it is possible to ascertain the activated fuzzy rules, what distance values are being returned by the sensors, and the values of some of the important fuzzy variables. This feature is a powerful tool for examining the operation of the rule base over critical stretches of travel. This facilitates the rapid identification of which rules are serving the purpose for which they were intended, which are firing in unexpected circumstances, and which need tuning to produce smoother, more reliable behaviour.

VI. Appendix

Definition 1 (Map:) A map is a 2-tupel $P = (K, G)$ consisting of a non- empty set K and a set $G \subseteq K \times K$. The elements of K signed as crossings and the elements of G as **paths**. For the crossings $x, y \in K$ are the path $g \in G$ from x to y labelled with $\vec{xy}$ or (x, y) for the crossings $x, y \in K$.

Definition 2 (Adjacent crossings, adjacent paths:) Let $a, b, c, d \in K$ be crossings then is

1.) the crossing b adjacent to the crossing a if a path $\vec{ab} \in G$ exists.

2.) the path $g_i = \vec{ab}$ adjacent to $g_j = \vec{c,d}$ if $b = c$.

If $\vec{ab} \in G$ but $\vec{ba} \notin G$ then $\vec{ab} \in G$ is a on- way path. In a one way path only b is adjacent to a (achievable) but not vice versa.

Algorithm 1 (Path planning:) Let $P = (K, G)$ be a finite map and $g_0 = \vec{x_0 x_1}, g_m = \vec{x_{m-1} x_m} \in G$ two corridors with the starting junction x_0 and the end junction x_m. Let SL be a plan from the starting corridor and ZL a plan from the target corridor. Furthermore, let $MA_m \in \{-2, -1, 0, 1, 2\}$ be a set of markers for each corridor $g \in G$.

Initialization: At the beginning, there are the plans from the starting corridor $SL = g_0, i = 0$ and from the target corridor $ZL = g_m, j = m$, i.e $g_i \in SL$ is the current corridor in the plan form start and $g_j \in ZL$ the current corridor in the plan from the target. $MA_k = 0, k = 1 \ldots m-1$ is marked with 0. Furthermore, the set of markers $MA_0 = 1$ and $MA_m = -1$ are marked. [1]

1.) If $g_i = g_j$, i.e. the current corridor g_i in the plan from the starting corridor is equal to the current corridor in the plan from the target corridor then $L = SL \cup ZL$ and L is the searched plan.

2.) If $MA_i = -1$ for the current corridor i then $L = SL \cup ZL_i$, where ZL_i is the plan from corridor g_i to the target corridor.

3.) If $MA_j = 1$ for the current corridor j then $L = SL_j \cup ZL$, where SL_j is the plan from the corridor g_i to the starting corridor.

4.) If $MA_i = -2$ for the current corridor i, i.e. the corridor g_i was selected in the plan from the target corridor, then $L = SL \cup L_{new}$ with L_{new} = Find a plan from g_i to g_m.

5.) if $MA_j = 2$ for the current corridor j, i.e. the corridor g_j was selected in the plan from the starting corridor, then $L = L_{new} \cup ZL$ with L_{new} = Find a plan from g_j to g_0.

6.) Let $g_{i,j} = \vec{x_a x_b}$. Calculate the output degree $k = d_+(x_a) = | NB_{g_{i,j}} |$ with $NB_{g_{i,j}} := \{\vec{x_a x_c} | g_{i,j} = \vec{x_a x_b} \wedge g_{i+1,j-1} = \vec{x_a x_c} \wedge x_c$ is adjacent to x_a, $x_a, x_b, x_c \in G\}$.

(a) If $k = 0$ then mark $MA_{i,j} = 2, (-2)$, remove $g_{i,j}$ in the plan from the starting (target) corridor and determine the current sub procedure call. For $k > 0$ sort the set $NB_{g_{i,j}}$ of possible adjacent paths.

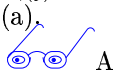
(b) Select from $NB_{g_{i,j}}$ the next corridor g_{i+1} (g_{j-1}).

(c) If $MA_{i+1,j-1} = 1, (-1)$ then ignore the corridor, i.e. $NBg_{i,j} = NBg_{i,j} \setminus g_{i+1,j-1}, k = k-1$ and continue with (a).

(d) Add $g_{i+1}, (g_{j-1})$ to the plan from the starting (target) corridor, i.e. $SL = SL \cup g_{i+1}$ $(ZL = g_{j-1} \cup ZL)$ and mark $MA_{i+1} = 1$ $(MA_{j-1} = -1)$.

(e) Find a plan from g_{i+1} to g_j (from g_i to g_{j-1}).

(f) If L is a plan then determine the call. Otherwise let $NBg_{i,(j)} = NBg_{i,(j)} \setminus g_{i+1,(j-1)}, k = k-1$ and continue with (a).



Algorithm 1

Remark: The path has no loops if 6.(c) is exchanged through: If $NB_{g_{i,j}} \exists MA_{i+1,j-1} = 1, -1$, $i, j = 1 \ldots k$ then $NBg_{i,j} = \emptyset, k = 0$ and continue with 6.(a).

Theorem 1 (Path planning) Let $P = (K, G)$ be a finite map and $g_0, g_n \in G$ two corridors. The algo-

[1] $MA_i, 0 = 1 \ldots m$ with $g_i = 1$ $(g_i = -1)$ is marked, if g_i is in the plan form the starting corridor (target corridor). The corridor is marked $g_i = 2$ $(g_i = -2)$ if the corridor is already visited but not included in the current plan from the start- or target. This improves the run time performance.

rithm 1 finds a plan between $g_0, g_n :\Leftrightarrow$ a noose free path exists between g_0, g_n.

Proof: "$\Rightarrow$" This direction is trivial. "$\Leftarrow$" The proof is given in [20] and goes inductive over n.

References

[1] E. H. Mamdani, "Application of fuzzy algorithms for control of a simple dynamic plant", *Proc. IEE*, vol. 121, Nr. 12, pp. 1585 – 1588, 1974.

[2] Lotfi A. Zadeh, "Fuzzy sets", *Information and Control*, vol. 8, pp. 338 – 353, 1965.

[3] Alessandro Saffiotti, Enrique H. Ruspini, and Kurt Konolige, *A Multivalued Logic Approach to Integrating Planning and Control*, Artificial Intelligence Center, SRI International, Technical Note No. 533, Menlo Park, CA 94025, USA, 4/1994.

[4] Patrick Reignier, "Fuzzy logic techniques for mobile robot obstacle avoidance", *Robotics and Autonomous Systems*, vol. 12, pp. 143 – 153, 1994.

[5] Hartmut Surmann, Jörg Huser, and Liliane Peters, "A fuzzy system for indoor mobile robot navigation", in *Fourth IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 95, Yokohama, Japan*, 20–24 Mar. 1995, pp. 83–88, Distinguished with *Robot Intelligence Award*.

[6] Peter Hoppen, Thomas Knieriemen, and Ewald von Puttkamer, "Laser-radar based mapping and navigation for an autonomous mobile robot", in *IEEE int. conf. on Robotics and Automation 1990, Cicinnati, Ohio*, 13. - 18.05.1990, pp. 948–953.

[7] Alessandro Saffiotti, "Some Notes on the Integration of Planning and Reactivity in Autonomous Mobile Robots", in *Procs. of the AAAI Spring Symposium on Foundations of Automatic Planning*, Stanford, CA, 1993, pp. 122 – 126.

[8] Edward Tunstel and Mo Jamshidi, "Fuzzy Logic and Behavior Control Strategy for Autonomous Mobile Robot Mapping", in *FUZZ-IEEE World Congress on Computional Intelligence, Orlando*, 26.6. - 2.7.1994, pp. 514 – 517.

[9] Bart Kosko, *Neural Networks and Fuzzy Systems*, Prentice Hall, Englewood Cliffs NJ, 1991.

[10] D. Driankov, H. Hellendorn, and M. Reinfrank, *An Introduction to Fuzzy Control*, Springer Verlag, Berlin, Heidelberg, New York, 1993.

[11] R. Dillmann and U. Rembold, "Schwerpunktthema: Robotik", *Informationstechnik und Technische Informatik 1/94*, 1994.

[12] P. Levi and Th. Bräunl, *10. Fachgespräch Autonome Mobile Systeme*, Institut für Parallele und Verteilte Höchstleistungsrechner, Universität Stuttgart, 1994.

[13] Ewald von Puttkamer, "Sensorintegration zur Geometrischen Weltmodellierung", *it+ti*, vol. 1/94, pp. 34 – 38, 1994.

[14] Jörg Huser, Hartmut Surmann, and Liliane Peters, "A fuzzy system for realtime navigation of mobile robots", in *Information Processing for Intelligent Service Robots, Workshop 8, KI-95, Bielefeld*, 11–12 Sept. 1995, pp. 170–172.

[15] Vittorio Gorrini and Hugues Bersini, "Recurrent Fuzzy Systems", in *FUZZ-IEEE World Congress on Computional Intelligence, Orlando*, 26.6. - 2.7.1994, pp. 193 – 198.

[16] Alberto Elfes, "Sonar-based real-world mapping and navigation", *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3, pp. 249 – 265, June 1987.

[17] S. Thrun and T. M. Mitchell, "Lifelong robot learning", *Robotics and Autonomous Systems*, vol. 15, pp. 24–46, 1995, Also appeared as Technical Report IAI-TR-93-7, University of Bonn, Dept. of Computer Science III, 1993.

[18] Greogory Dudek, Micael Jenkin, Evangelos Milos, and David Wilkes, "Robotic Exploration as Graph Construction", *IEEE Transactions on Robotics and Automation*, vol. 7, no. 6, pp. 859 – 865, Dec. 1991.

[19] Hartmut Surmann, Kai Heesche, Martin Hoh, Karl Goser, and Rainer Rudolf, "Entwicklungsumgebung für Fuzzy-Controller mit neuronaler Komponente", *VDE-Fachtagung: "Technische Anwendungen von Fuzzy-Systemen", Dortmund*, pp. 288–297, 12–13 Nov. 1992.

[20] Hartmut Surmann, Jörg Huser, and Jens Wehking, *Topolgische Karten für fuzzy-gesteuerte mobile Roboter I*, GMD - Forschungszentrum Informationstechnik GmbH, St. Augustin, June 1996.