

MORIA - A Robot with Fuzzy Controlled Behaviour

Hartmut Surmann and Liliane Peters

German National Research Center for Information Technology,
Institute for System Design Technology
53754 Sankt Augustin, Germany
E-mail: surmann@gmd.de.

Abstract. ¹ MORIA is a low cost fuzzy controlled autonomous service robot. The major goal of our experiment was to develop a human-like decision strategy that supports autonomous navigation of the system in office buildings. This paper presents the various steps during the design cycle, from specification to its implementation. Experimental results show that the developed prototype reacts autonomously in an appropriate manner to the various dynamic changes encountered in the test environment.

1 Introduction

A new breed of robots is entering our daily life. They clean the floors at airports [1], carry suit-cases in hotels [2] and guide you within a museum [3]. These robots, known as service robots, have four common features. (1) *They work within structured environments that are a-priori known.* Office environments can be considered structured as they have fixed building elements like: corridors, halls, elevators, rooms, etc. (2) *The complexity of the environment is high.* Not only do the structured building elements differ from building to building, but also the number of elements within a building is quite high especially if we consider that office building can have from two to thirty floors. (3) *The environment is highly dynamic.* The service robots have to operate together with humans in the same environment. That means sharing the same path and access points from one closed environmental section - corridor - to the next one. The complexity is increased even more if the service robots operate in groups. But not only mobile objects reflect the dynamic changes in the environment, static objects like, desks, chair, boxes, etc. can block or partially occlude parts of known paths. Their appearance and disappearance in the environment can not be foreseen (planned) a-priori. (4) *The users are technically unskilled personnel.* Therefore, not only does the human interface to the machine have to be simple, but an additional source of uncertainty has to be taken into consideration. These four features require a high navigation autonomy of the robot in the office environment, and a strategy for efficient task solving - from collision avoidance to rescheduling of missions. Although the first steps have been made, there is still

¹ in Fuzzy Logic Techniques for Autonomous Vehicle Navigation (D. Driankov & A. Saffiotti (Eds.)), Series: Studies in Fuzziness and Soft Computing. Vol. 61, Springer Verlag Berlin, 2001, pp. 343 - 366 ISBN 3-7908-1341-9, http://www.springer.de/cgi-bin/search_book.pl?isbn=3-7908-1341-9

a very long way to go until populations of robots “live” within our buildings or even apartments, and share all heavy, unhealthy, or time-consuming chores, so improving the quality of our daily life. Until now, cooperation has been mainly studied between one robot and a human or a few robots in a room like environment, e.g. [4] [5], [6]. What is still missing are the experiments with large numbers of robots (over 50) driving through busy buildings. The first step toward this vision is to develop a navigation and planning strategy for a single robot that can cope with a service-like environment. The increase in complexity through the introduction of a group is the next step.

Fuzzy logic theory offers a natural way to integrate these requirements in one system. It has already been shown that fuzzy rule based systems (FRBS) are important tools for modelling complex systems where, due to the complexity or the imprecision of the process to be controlled the knowledge base is acquired from human expert or from a referential data set with neural or genetic algorithms [7]. The advantage of fuzzy control is obvious when the control process is non-linear, has a time dependent behaviour and the available measurements have a poor quality or a high uncertainty. The fuzzy logic theory replaces crisp values like “true” and “false” with a range of truth values, crisp measurements like 0.5m or 1.2m, and linguistic terms like “near” or “default”. This linguistic description loosens the required accuracy of the sensorial measurement. Thus the output of the sensors have to guarantee a range of values corresponding to pre-defined linguistic terms e.g. “near” instead of absolute error deviation. The fuzzy terms in turn are described by a continuous membership function defined over the same range of values. This description mirrors natural language concepts and makes it possible to process linguistic concepts like the “distance is short” in a precise control system. For complete description of the fuzzy logic control method see [8].

In this chapter we propose a recurrent fuzzy system to implement the autonomous behaviour of the robot. It has to be mentioned that it is not the only approach using fuzzy logic for robot navigation. For comparison see [9–14], each one applying various navigation strategies. The originality of our approach is the introduction of an inferential reasoning process that solves inherent ambiguities within the environment. Thus the behaviour of the robot is not only dependent on the sensorial input but also on its past behaviour. This approach gives a high degree of autonomy to the robot. From the point of view of the robot the approach can be described to support an hierarchical architecture based on a high level planner and a low level executer [15]. Both levels are based on a fuzzy logic concept and cope successfully with the given service task.²

The high level planner needs a map of the environment to schedule and distribute the tasks. Although the environment is known a-priori, its structure can also introduce some hazards in addition to the ones coming from the sensors due to similarities in structure, steady changes in the environment and machine unfriendly paths, e.g. winding corridors, too many doors in a small area, etc. These aspects are supported by the map of the environment that underlays the planner. The designed navigation scheme has to be robust enough to recover from erroneous path planning on its own without rescheduling the complete path e.g. by manoeuvring autonomously out of a dead-end. Last, but not least some of the missing information given through incomplete commands to the

² A great number of different planners and testbeds can be found under <http://eksl-www.cs.umass.edu/planning-resources.html>.

system can be compensated through apriori knowledge about the environment by the system itself.

The presented experiment is the first step in a larger project dealing with service robots. In this experiment we have mainly concentrated on the first three features: autonomy, environment, and human interface. Robot cooperation has been taken into consideration while designing the planner and the map of the environment, thus preparing the ground work for the extension to a group of service robots.

The chapter is structured as follows. Section two presents the architecture of the implemented system. We start by giving the reason for a fuzzy approach, by introducing the problem to be solved and by giving the application constraints. Next a general overview of the architecture is introduced. In the following two sections we describe details of the architecture. In section three we describe the planner and the use of the environment map. In section four we present the navigation block and the related local perception of the environment. Section five presents the experimental results concentrating on the monitoring and communication capabilities of the system. We conclude our presentation with some final remarks and an outlook on future activities.

2 The Architecture of MORIA

2.1 Why A Fuzzy Approach?

As mentioned in the introduction, service robots operating in structured real-world environments require the ability to cope with uncertain, incomplete and approximate environmental information in real time. In addition, as the robots have to execute humans commands issued by personnel with different levels of qualification, uncertainties due to incomplete or missing information have to be tackled. The proposed recurrent fuzzy system (RFS) that implements an autonomous intelligent system takes into consideration all these uncertainties. Before discussing the proposed approach we would like to describe in more detail the uncertainties existing in the environment and their impact on the robot.

The high uncertainty existing in the environment is the main challenge in the control and behaviour of autonomous navigation. The types of uncertainty can be classified into sensory dependent and information dependent. Sensory dependent uncertainty caused by the low reliability of the sensors can be reduced through sensory fusion and by using additional, a-priori knowledge existing within the system. Information dependent uncertainty occurs when the map of the environment - a-priori known structured environment - doesn't fit reality due to some dynamic changes. The map can be considered incomplete. The missing information is due to:

- an incomplete plan of the environment (missing dynamic objects).
- steady changes in the environment (appearance and disappearance of dynamic objects); e.g. an object blocks certain path for a given period of time.
- transient perturbation in the environment (moving object - humans or robots) that distort environmental measures by occluding fixed objects like walls.

These various uncertainties can cause the following types of errors:

- wrong description of the local environment; the error can propagate into the navigation strategy and start a wrong manoeuvre
- wrong localisation of the robot in its own internal map; such an error could trigger another path estimation causing the robot to turn and start looking for its destination using another, generally longer path.

The first type of error has a local effect. The local navigation of the robot is changed but the path followed to reach the given target remains the same. The second type of error has a larger impact on the system. It introduces a global error jeopardising the success of the mission (task).

If we summarize the uncertainties described above it can be stated that knowledge of the environment is inherently partial and approximate. This has an impact on both the navigation and the task planning of the autonomous system. Sensing is noisy, the dynamics of the environment can only be partially predicted and the hard- and software tasks executed by the system (robot) are not completely reliable [16]. Classical path planning approaches have been criticised for not being able to cope adequately with this situation³. Traditional approaches to mobile robot navigation have dealt with these requirements by using computationally intensive planning algorithms and explicit pre-determined world models [18]. This is not necessarily a drawback for fixed-base manipulators but it is a problem for mobile robots for which computational resources must be carried on board.

In our opinion the first part of the answer is not to compensate for the uncertainty at a given processing level through higher precision in the following process step, but to make decisions based on qualitative measurements that give a certain range of values with a good precision. By using qualitative decision making the precision of the needed information decreases and thus better matches the input information requested by the control system. This qualitative decision making can be supported through a fuzzy approach where the decision for the behaviour is not decided through a crisp structure but through an evaluation of the best possibility within a given range. As will be shown in section four our navigation block utilises this decision making approach by introducing fuzzy state variables (FSV).

The second part of the answer is great autonomy of the system related to its ability to plan a path between two given points under the given conditions. A classical planning strategy needs an accurate map of the environment. While qualitative environmental information is enough for navigation (collision avoidance), this type of information is of course not enough for a precise topological map of the environment. Let's analyse in more detail if a precise geometric map of the environment is needed. Many path planning approaches describe the path as a list of coordinates. As long as the environment is restricted to a room, the amount of memory needed is manageable. But if we think of the service robots driving through a building the memory needed for such an accurate map increases very quickly and the processing time linearly with it. As already shown by Saffiotti [11] a fuzzy based controller has the advantage that the intuitive nature of collision-free navigation can be easily modelled using linguistic terminology (e.g. next right instead of 89.25° on precise radius) [14]. This implies that the low-level behaviour

³ Saffiotti [17] gives an interesting glance at the "planning vs. reactivity debate"

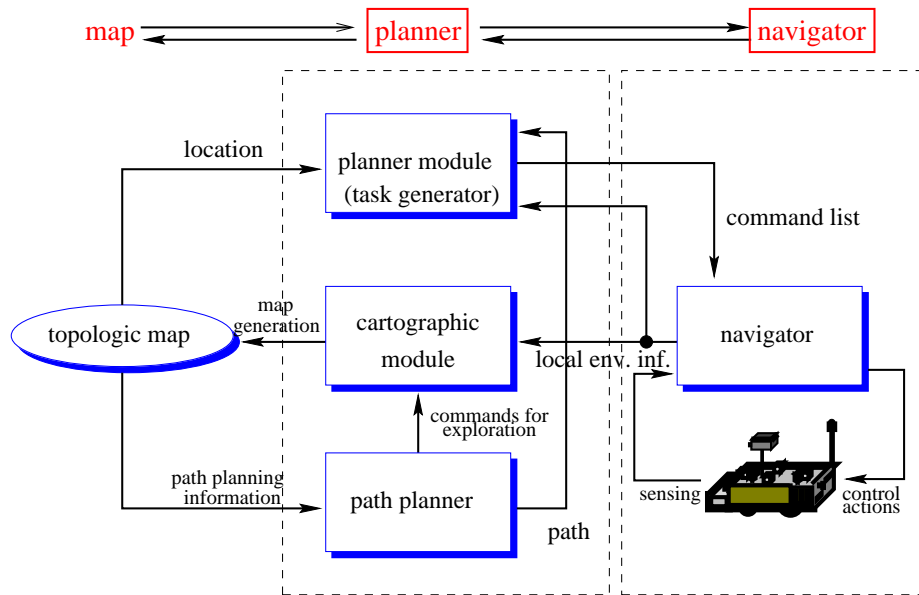


Fig. 1. System Architecture of MORIA

approach allows the storing of information at a higher level of abstraction. If this is so, a fusion between planning and navigation can be done by combining a high level goal with an autonomous navigation within this goal. This combination between high level goal and low level topologic map-based navigation can also be supported through the fuzzy state variables introduced. Thus the planner is also based on a topological map that doesn't keep the high accuracy of other conventional planners, but the goal of the mission is still reached. Using this approach we have extended the fuzzy concepts to the planning level too. The robot will always reach its global goal but nobody can predict the exact coordinates of its path except when they are given as an intermediate goal.

By using a fuzzy approach both for planning and for navigation our autonomous system behaves at the same time in a reactive and a goal oriented manner. The two different behaviours are blended together dependent on the environmental information and the internal fuzzy state variables. The obtained system behaviour although built up using only a few blocks is always changing and adapting in an adequate manner to the environment. The proposed block architecture of the system is presented in Fig 1.

2.2 Problem specification

MORIA (Fig 2) is designed as a transport system for ware houses and offices. The major hard constraint in our design specification is the cost of the control architecture. To be competitive in the market, our industrial partner requested a solution for autonomous navigation that would cost less than USD 6.000. This of course had an impact on the chosen sensors. The main task of the system is to drive within an building from A to B

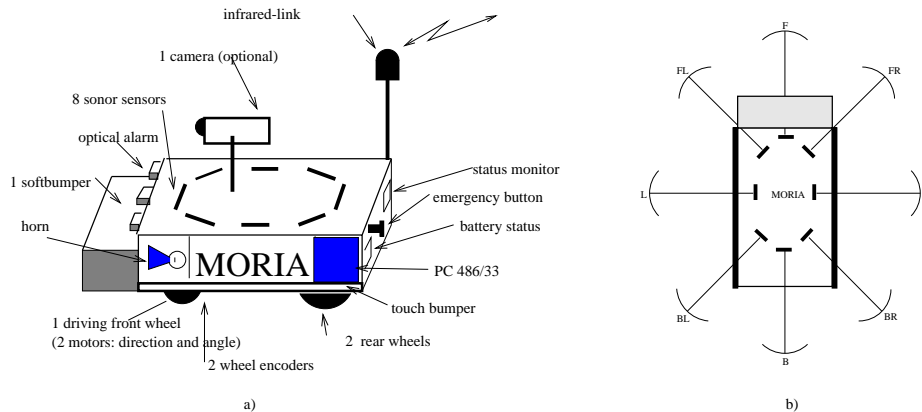


Fig. 2. The equipment of Moria

with the help of a map. The system should find the best path depending on the type of load. In case of an unforeseen obstacle or object temporarily blocking the path, the system should be able to recalculate the path without needing the support of a supervisor.

Navigation is based on 8 sonar sensors and two wheel-encoders. The distribution of the sonar sensors on the system is given in Fig. 2b. The alarm system is triggered by touch sensors integrated in the frontal, lateral and rear ledge bumpers. The warning system consists of a horn and three flashing lights. The lights are used to give the status of the vehicle using coded signals: X-X-X for forward, X-X-0 for left turn, 0-X-X for right turn, etc. The high-level planner can be located either on the robot or on an external computer. In the second case an infrared link is used for the communication between planner and navigator. This communication channel is important if a pool of robots are working in the same environments. Each robot has a navigator and one planner distributes the task and calculates the paths. The infrared link has a range of 40m, so for a building several access links are needed.

The interface for the user is a keyboard, a monitor and a joystick. The joystick is used for manual driving of the robot. The keyboard can be located either on the robot or on another computer. The same infrared link is used for remote control. The monitor, a 2D map of the environment, is on a remote system and receives the actual data from the robot via the infrared link. This approach enables us to use one interface between the low-level navigator and the high-level commanders (planner or human). The navigator doesn't see the difference between the two modes. The commands given by humans are of type "next left", etc. The system can be enhanced with an optional camera for additional tasks, e.g. visualisation in a 3D virtual reality (VR).

2.3 Proposed approach

The above mentioned specification requires an autonomous reaction (action) in the following situations:

- given the map of the environment the system is able to find a path between a given source A and a given destination B under special constraints. For example “the number of turns should be as low as possible” or “the corridor should be always larger than a given distance”, etc.
- the system should avoid collision while driving to its destination without losing the predefined path
- the system should identify dead-ends, find the best way out and if necessary recalculate the best path from the given position.

If we analyse these behaviour patterns and take into consideration their impact range we can divide them into global and local. Actions with global impact are related to path finding, localisation of the robot in the map and updating the map of the environment. The planner supports all these tasks. For these tasks a soft real time processing constraint is necessary. The planner performing these tasks is far-sighted and goal-oriented.

The local actions of the system are related to collision avoidance and to adapt to unforeseen situations appearing in the near neighbourhood. These navigation tasks require hard real-time processing. The navigator performing these tasks is short-sighted and reactive. Unexpected situations are mainly caused by moving objects. These objects (events) are local and can appear and disappear within an environment. Their impact could create a mismatch between the internal map of the environment and reality, thus have also a global impact. The system will first react via the navigator and adapt locally in real-time to the situation. The extracted information will be passed to the planner. If a global impact is detected this change will be processed by the planner. By choosing a reactive navigator and a goal-oriented planner the system is able to cope with known and unknown situation while keeping the imposed time constraints.

The block architecture of the system is presented in Fig. 1. If a task is given to the system then the planner starts the following process:

- the start and end point are detected in the internal map; additional information is requested from the internal knowledge base
- the path planner estimates the best path from A to B
- the task generator transforms the path into a list of linguistic commands of type “straight ahead”, “next left”, etc.

The navigator has only “one command in mind” at a time and tries to fulfil it while avoiding obstacles on its way. This means that the following processing steps are initiated by the navigator:

- the sensorial information is fused and evaluated
- the executed command is acknowledged so the planner can trigger the next command
- a topological description of the local environment is generated and sent to the planner to up-date the map

At the first glance the system behaves as expected. The planner takes care of global issues and has a longer estimation time for this while the navigator reacts quickly only to sensorial information and keeps the imposed real-time constraint.

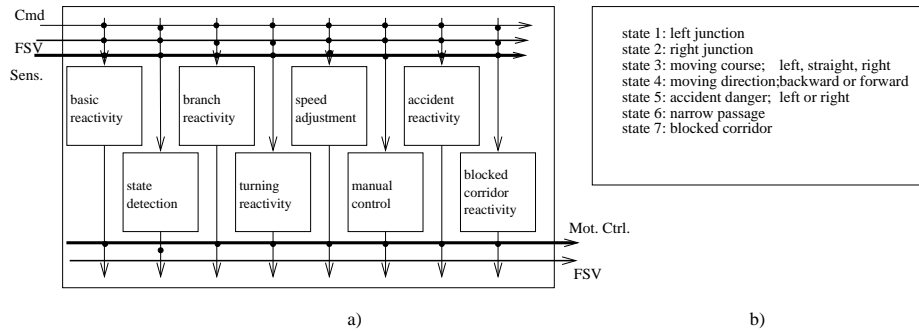


Fig. 3. Detail of the recurrent fuzzy system; a) fuzzy behaviour blocks; b) fuzzy state variables. Left junction, Right junction and moving direction have crisp, non-overlapping membership functions and the other states Gaussian like, overlapping MFs.

But this approach is not sufficient for solving conflicting problems between navigator and planner or reacting adequately to unexpected situations. A simple example illustrates possible conflicts. Suppose somebody puts a table in the corridor (large enough to block it for the robot). This information is not in the map, therefore the path planner gives a command to go straight through the corridor ignoring the change. The local perceptual block of the navigator identifies the blocked corridor. For a while the command coming from the planner has to be post-poned and the behaviour “basic reactivity” has to be changed into “blocked corridor reactivity” of the navigator (see Fig. 3a). The planner has to be informed about the occurrence of the special situation. From the point of view of task distribution, the navigator is now the master until the unexpected situation is solved. To handle such situations we introduced fuzzy state variables (FSV). At all times these variables identify the local situation based on the sensorial input, the current global command and the previous “state-of-mind” of the robot (previous fuzzy state) (see Fig. 3b). If we blend these inputs together (command, sensorial input, and the fuzzy state variables) into one fuzzy inference engine, we have a goal oriented reactive autonomous system (see Fig 7). The modular design approach has led to a recurrent fuzzy system implementation[19] of the autonomous behaviour.

2.4 A Recurrent Fuzzy System

A (first order) recurrent fuzzy system (RFS) is characterised by rules in which one or more variables appear both in the premise and consequent parts, like x in: IF $x(t-1)$ is A_j^k AND $u(t-1)$ is B_j^k THEN $x(t)$ is C_j^k , where A_j^k represents the membership functions coding the linguistic term k associated to the variable j [19]. A recurrent fuzzy system means a fuzzy system with inferential changing, an approach very typical of fuzzy reasoning or fuzzy expert systems, but very infrequently in fuzzy control where one shot input/output structures are commonplace. An n -th-order RFS may also contain *internal variables* from several (n) different blocks of rules. The fuzzy internal variables also have their own temporal dynamics. Fig. 5 shows two examples of fuzzy state variables for Moria.

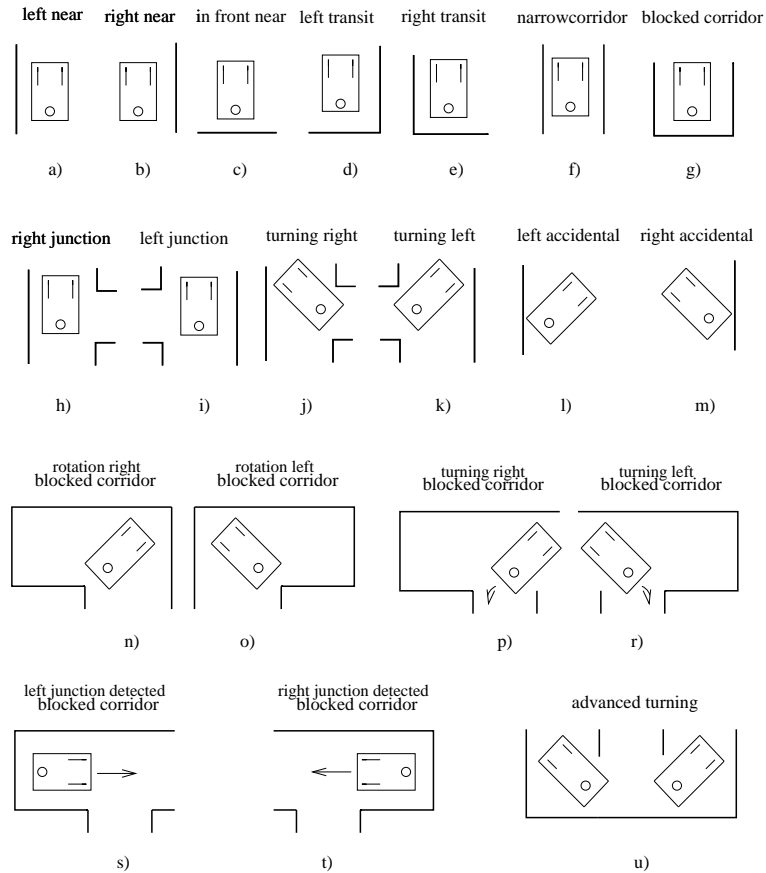


Fig. 4. Local Perceptual State and the names of the related fuzzy state variables

In contrast to RFS, most autonomous robots have standard FS without memory. They act or react only based on the actual sensorial input. Ambiguous states such as “narrow passage” or “accident danger” (Fig. 4 f,p-r) are difficult to recognise immediately when the robot is in locomotion. By introducing fuzzy state variables (FSV) we added a local memory to the navigation process and gave the system the capability to react in a temporally dynamic way to complicated situations (Fig. 4n-u) based on the near past. Another advantage is the smooth passage from one behaviour to the next through the FSV. The different strategies are smoothly blended together (in a fuzzy manner) depending on the degree of the belonging of the environmental conditions to a defined state. Not all states are fuzzy, some are crisp like “left junction”. An example of the used fuzzy controller algorithm is given in Fig. 6. The calculation of the center of gravity for the result function *speed* is the major bottleneck of the fuzzy algorithm. Therefore, a modified result function calculation [20] is suggested, in which the center of area M_i and the area A_i of a membership function are calculated before run time (m

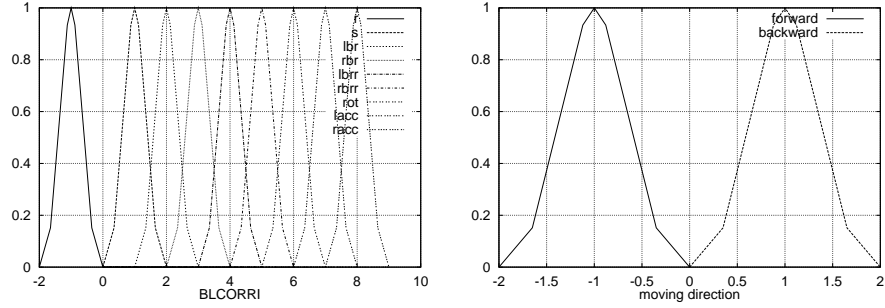


Fig. 5. Example of overlapping and non-overlapping membership functions for the fuzzy state variable blocked corridor and moving direction

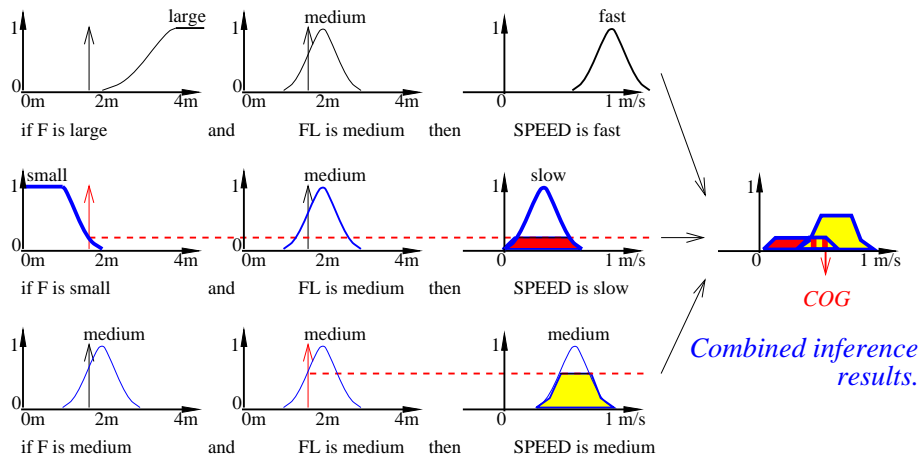


Fig. 6. Example of the fuzzy controller algorithm. F: front sensor. FL: front left

= number of rules):

$$A_i = \int speed(e) de, \quad M_i = \int e \times speed(e) de$$

$$COG_I = \frac{\sum_{i=1}^m \omega_i \times M_i}{\sum_{i=1}^m \omega_i \times A_i} \quad (1)$$

The set of fuzzy state variables represents a local perceptual and action memory of the fuzzy controller (FC) (Fig. 3). Since standard fuzzy membership functions may implemented both as crisp or fuzzy values, states could also be crisp or fuzzy.

Furthermore a fuzzy state variable represents a landmark for the topological map. Let's look at a more detailed representation of the system architecture (Fig. 1).

The planner generates a list of commands (see section 3). Only one command at a time is sent to the FRBS. The next one is triggered by the sensorial information coming

from the Local Perceptual Space and the FSV. For example if the command is “next left” and the FSV =1 (Fig. 3b) this will trigger the next command from the planner list.

The navigator has several behaviour modules (Fig. 3a) each being a fuzzy rule-base. If the situation i occurs, the corresponding fuzzy rules are selected through an additional fuzzy rule (context rule), and selects the condition variable i . In contrast to the approach presented by [13] our vehicle doesn't need the definition of special rule weights to change from one navigation state to the next. This is done by the firing value of the context rules and which are not set a-priori. Furthermore the new fuzzy state variables at the next time step $t+1$ is calculated from the sensorial input and the actual fuzzy state variable at time step t . As an example, if the premise “IF left sensor = very small AND front sensor = default AND right sensor = very small” is activated then a narrow corridor is detected and the robot has to drive slowly. Under these conditions FSV 5 and 6 are activated (Fig 3b). The fuzzy rules have the following form for the motor control:

IF COMMAND is C_i
AND SENSOR(1) is I_{h1} ... AND SENSOR(m) is I_{hm}
THEN SPEED is S_j AND ANGLE is A_j
or
IF COMMAND is C_i
AND STATE(1) is $ST1_{l1}$... AND STATE(n) is STn_{ln}
AND SENSOR(1) is I_{h1} ... AND SENSOR(m) is I_{hm}
THEN SPEED is S_j AND ANGLE is A_j

For the estimation of the actual FSV the form is:

IF COMMAND is C_i
AND STATE(1) is $ST1_{l1}$... AND STATE(n) is STn_{ln}
AND SENSOR(1) is I_{h1} ... AND SENSOR(m) is I_{hm}
THEN STATE(1) is $ST1_{l1}$... AND STATE(n) is STn_{ln}

The collision avoidance strategy based on these rules is explained in more detail in section 4. The implemented recurrent fuzzy system has 29 inputs: 7 FSV's, 8 sensorial inputs, 1 command input, 8 sensorial derivatives and 5 FSV derivatives. The number of outputs is 14: 2 for motor control and 12 FSV's. The implemented rule-base has 352 rules. These rules can be divided into blocks corresponding to the various behaviours presented in Fig. 3. The advantages of the approach can be summed up:

- keep the modular design of high-level planner and low-level navigator
- have a smooth passage from goal-oriented to reactive behaviour without conflicts
- keep a modular structure of the autonomous behaviour by implementing several behaviour blocks separately
- extend to new behaviour types easily by adding new fuzzy state variables and new behaviour blocks.

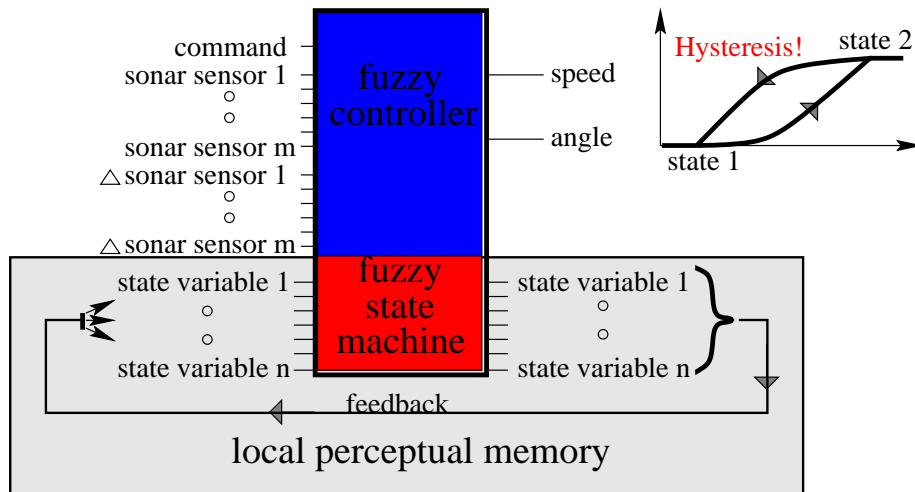


Fig. 7. A Recurrent Fuzzy System. With the help of the internal fuzzy state variables hysteresis could be realized. The concept of fuzzy state variable may contain both: crisp and fuzzy membership functions

3 Goal-oriented Planner

3.1 Path Planner

Most path planning approaches are based on high precision, metric approaches and demand very reliable and accurate sensor devices as well as great computational power. In addition, these methods have difficulties during navigation in complex and dynamically changing environments where unknown obstacles appear on an a-priori planned path [17, 14]. The conventional path planning approaches for mobile robots may be divided into two categories. One is 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.

In our approach the navigator is analogous to a driver and a planner to a co-driver of the system. The planner is in charge of the map and of finding new routes (paths) when unexpected obstacle appear. It utilises a topological map of the network of corridors and has the path-finding sophistication to plan a route for the robot. The planner uses only computed state variables of the navigator combined with fuzzy metric information. The fuzzy state variables contain information about actions of the robot, relative fuzzy distances since the last fuzzy state interaction and information according to the current situation. The task generator (see Fig.1) transforms the found path into a list of commands (Table 1). Fig. 8 shows an example.

In cases where obstacles suddenly appear, fast reaction of the robot is needed. Since MORIA drives with a maximum speed of $1.0 \frac{m}{s}$ the robot may have to make a hard break and/or turn. To avoid a jerky driving style we implemented some additional commands (4-7 in table 1) where the navigator executes immediately without exploring the best

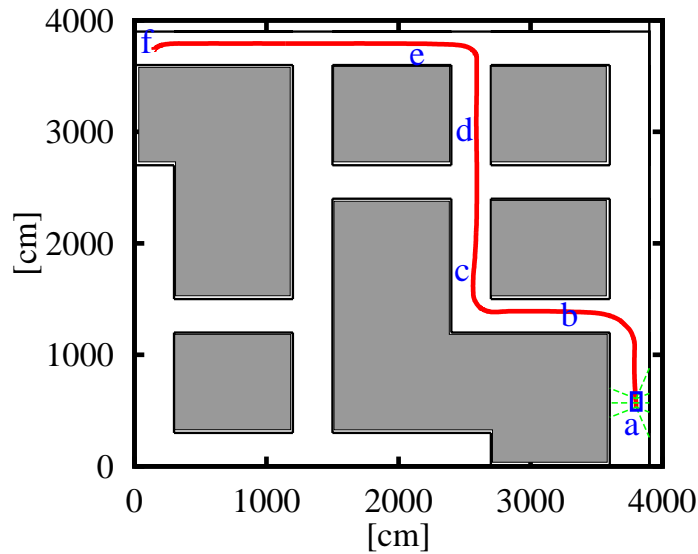


Fig. 8. Path-planning and the corresponding command list: next left (a); next right (b); straight ahead (c); next left (d); straight ahead (e); stop(f).

next environmental possibility.

Obviously, metric information about precise distances are not necessary. The maze of corridors can be represented as a graph. The path planning algorithm acts like a vectored graph search. To find the best route is an NP complete problem. We restricted the search by reducing the maximum number of corridors to one hundred. This is a realistic figure even if the building is very complex with a large number of floors. The search can be done recurrently for each floor. A detailed presentation of the search algorithm can be found in [15].

3.2 Map of the environment

Geometric approaches use distance information from several sensors to build geometric maps mostly based on probability theory e.g. [21]. These approaches need a lot of precise information and have difficulties in distinguishing 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, e.g. [22], requires from the robot to drop distinct markers. Our approach explores different levels of map representation simultaneously without dropping distinct markers. Each map type has limitations and is therefore used for part of the planner tasks.

The service robots act in a structured a-priori known environment. The static map of the environment is usually available in a computer readable format. In our case it is read automatically into the system. Part of it is shown in Fig. 9. This map is then converted into a topological graph that is used to search paths. On the graph level, we try to extract

Table 1. List of commands understood by the navigator. They are implemented as Gaussian like membership functions with $y(x) > 0$ in the given interval and $y(x) = 1$ in the middle of the interval

Commands	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)	$10 \in [9.5, 10.5]$

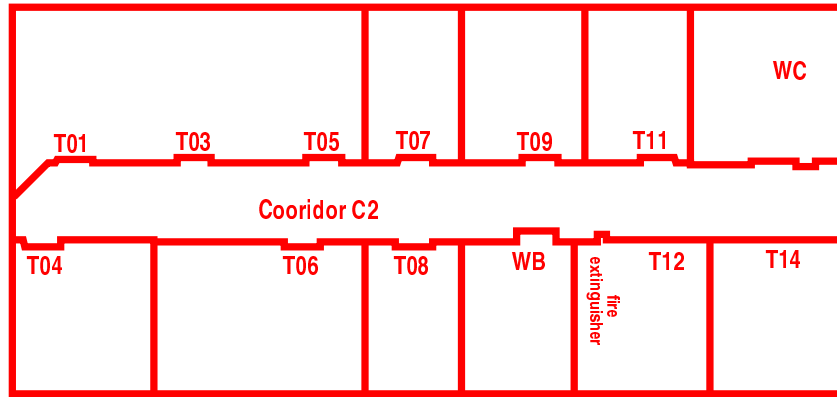


Fig. 9. Robot environment: corridor of a building

places, paths and their fuzzy metric relations based on the fuzzy state variables [23] of the recurrent fuzzy controller.

Some words have to be said about the map update procedure. At the navigation level we have a perception module where the local sensory information is kept and a local representation of the environment is available. Fig. 11 shows a topological map and Fig. 10 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. The navigator uses this local map for collision avoidance. This information is also sent to the planner together with the recognised structure through the identified fuzzy state variables. In this way dynamic objects can be handled and map updates made.

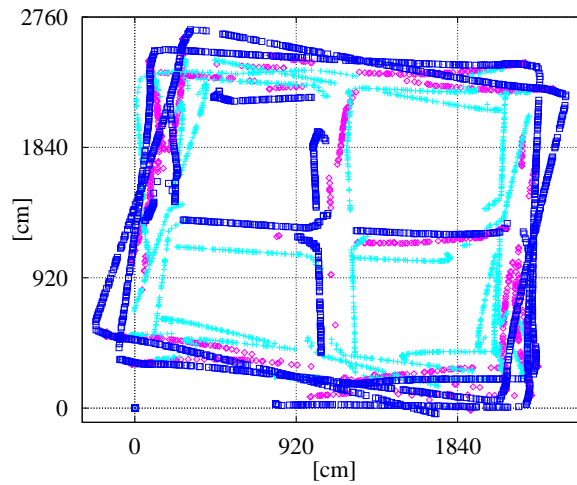


Fig. 10. Map generated by the sensorial information

4 Reactive Navigator

The recurrent fuzzy inference engine of the system operates based on the fuzzy input variables: command, sensorial information, the FSV, the speed and direction (angle) of the motor, the angle of the front wheel, and the new fuzzy states. The avoidance procedure is based on a reactive control loop between sensors and motors by slightly changing the driving angle and reducing the motor speed.

The “basic reactive behaviour” of the fuzzy controller uses only three sensors. For driving forward the sensors FL, F, FR as shown in Fig. 12. For driving backward the corresponding symmetric ones BL, B, BR, are used. The lateral sonar sensors left L and right R are used for docking or driving parallel to a wall by evaluating BL, L, FL for the left wall and BR, R, FR for the right one. The range of values of each sensor is divided into three linguistic terms: “very near”, “near”, “standard”. For each combination of the three membership functions a fuzzy rule is given (see Table 2). The rule base of the basic behaviour is completely defined (27 rules).

The system output values for speed and rotation angle (direction) are estimated through superposition of the various forces acting in the decision mechanism: sensorial forces, command forces, FSV forces. Let’s consider first the forces introduced by the sensorial input as shown in Fig. 12. The firing rule will be:

IF FRONT-SENSOR is very-near AND FRONT-LEFT-SENSOR is near AND FRONT-RIGHT-SENSOR is very-near
 THEN SPEED is (positive)-small, ANGLE is negative-small

The vehicle will start turning to the left as the FL sensor is only near compared with the other two values. If we add the command coming from the planner to the inference process the behaviour could be changed:

case 1:

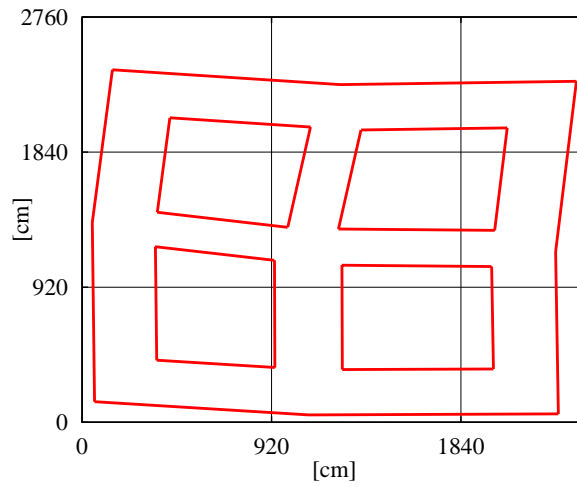


Fig. 11. Topological map built on sensorial information

Table 2. Basic reactivity rule base

left	center	right	angle	speed
normal	normal	normal	zero	high
normal	normal	small	s. left	med.
normal	normal	v. small	left	small
normal	small	normal	s. left	med.
normal	v. small	normal	left	small
small	normal	normal	s. right	med.
v. small	normal	normal	right	small
.
.
v. small	v. small	v. small	zero	zero

IF COMMAND is straight-ahead
 THEN SPEED is default, ANGLE is zero.

case 2:

IF COMMAND is left-rotation
 THEN SPEED is very-small, ANGLE is negative-very-large.

If case 1 is active the command forces don't change the motor control requested by the sensors. If case 2 is active the final output will be:

...

THEN SPEED is almost-very-small, ANGLE is negative-medium

If we integrate (superimpose) the command rules and the sensorial rules then the rule

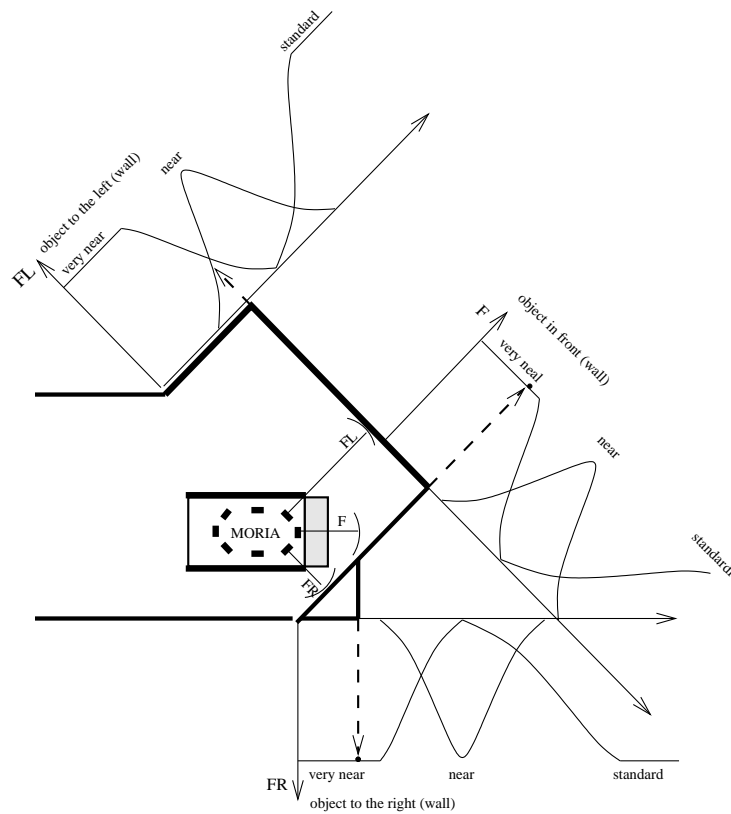


Fig. 12. Fuzzy direction forces for a blocked corridor and the input membership functions

for case 1 is:

IF COMMAND is straight-ahead
 AND IF FRONT-SENSOR is very-near AND FRONT-LEFT-SENSOR is near AND
 FRONT-RIGHT-SENSOR is very-near
 THEN SPEED is positive-small, ANGLE is negative-small

Now let's introduce the force of the FSV. Their impact on the behaviour can be best explained when a blocked corridor appears. The basic reactive rule base will decrease the speed and slowly increase the turning angle to the left (negative large). The fuzzy state detection will fire the following rule:

IF FRONT-SENSOR is very-near AND FRONT-LEFT-SENSOR is very-near AND
 FRONT-RIGHT-SENSOR is very-near
 THEN FSV_{env} is blocked-corridor.

This fuzzy state will change the behaviour of the system from basic to "block-reactivity". The firing rule is:

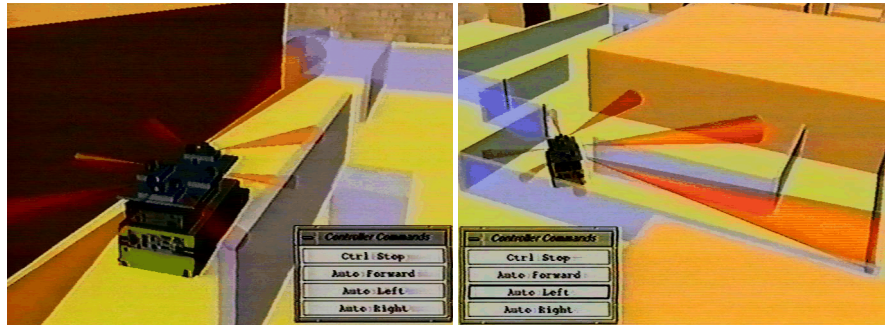


Fig. 13. 3D graphical user interface which shows a plan view of the robot's progress through the test environment.

IF FRONT-SENSOR is very-near AND FRONT-LEFT-SENSOR is near AND FRONT-RIGHT-SENSOR is very-near
 AND FSV_{env} is blocked-corridor
 THEN $FSV_{direction}$ is backwards.

The system starts the recovering manoeuvre from a blocked corridor and the FSV informs the planner about the identified state. The planner can either keep the last command if the robot was on exploration tour or recalculate a new path. As presented in section 2.3 the behavioural blocks are fired in parallel and all inputs are estimated at once thus producing the described recurrent fuzzy system.

5 Implementation

5.1 Design and Simulation Environment

This project used the design and simulation environment that was initially developed for the digitally implemented fuzzy control system reported in [23]. The modelling system FunnyLab [24] was used to create and edit the membership functions and the rule base. On a PC 486 / 33 MHz the generated fuzzy engine processes at a speed of more than 2 MFIPS (million fuzzy inference per second) including defuzzification [25]. 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 1. The motion simulator has been demonstrated to produce an accurate representation of the dynamics of MORIA during 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 additional feature which allows single-stepping. At each step it is possible to watch the activated fuzzy rules, the sensorial output, and the

values of the 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 the actual firing rule, the tuning of the behaviour blocks, or the fuzzy state variables.

5.2 The Robot MORIA

The vehicle MORIA is a mobile device with dimensions of 157cm x 90cm x 73cm (L,W,H). The weight of the vehicle is 400kg and it has a payload capacity of 150kg (Fig. 14). The vehicle is driven by two motors acting on a single wheel situated in the front of the vehicle, one motor that is reversible provides the driving torque, and the other one steers the vehicle by changing the orientation of the driving wheel. Due to this construction, the forward and backward driving strategies are different. It is equipped with 8 ultrasonic sensors that have a range between 50 - 400 cm.



Fig. 14. MORIA at the Hannover Fair'96.

Computational capabilities of MORIA consist of 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 computers is available via an on-board infrared sensor.

5.3 Experimental results

The vehicle was tested in the corridors of our research institution and several public presentations were made during the Hannover Fair'96 and GMD Open House days in the last 3 years e.g. during the Hannover Fair the robot drives one week without an accident and only two deadlocks. The first stage of the implementation, the low level reactivity and high-level planner were presented at the IEEE FUZZ/IFES'95 where it took the "Intelligence Award".

The experimental results proved not only the robustness (Σ of user helps / (number of operating hours)) of the approach but also showed the simple programmability of the system through the separately designed behaviour blocks. Depending on the test environment (width of the corridor) the same rule base with different definition of the linguistic terms of the sensorial information was successfully. Experiments with different behaviour types were also tested and an easy download mechanism supported this adaptivity. The complexity of the environment was increased every year. At the end of our experiments with MORIA the system was able to drive through the various floors using the behaviour blocks and the number of rules presented in this paper.

The major gain from these experiments was the experience we acquired in specifying the features of a service robot for buildings. Based on this experience we have defined a new type of robot (a group of three) which we are currently studding The behaviour developed for MORIA was ported and the new mechanical construction of the system was designed to better fit our local environment. The new family of robots have a greater range of operation, e.g. they can use the elevator. The monitoring capabilities were enhanced to permit goal-oriented navigation for all three robots to be performed on a remote computer.

6 Conclusion and future work

This paper described a low-level navigator and a high-level planner that were implemented as a fuzzy recurrent system. By introducing the fuzzy state variables we were able to drive and control the robot through a complex environment and the system reacted autonomously to unexpected situations such as moving or dynamic objects. Our experiment has also shown that autonomous systems can be controlled with human like behavioural commands thus supporting a human friendly interface. A new generation of robots is currently being implemented in the ARIADNE project based on this experience. The main goal of our research will be the cooperation of the robots under test. This feature will be based on an extension of the global planner housed in a remote system and the reactive navigator present on the autonomous platform.

Acknowledgement J. Huser, Shuwei Guo, Stefan Virsik, Markus Eder and J. Wehking contributed to the development of MORIA by implementing several modules. TZN Unterlüß(Germany) supported part of this work. Julian Kolodko and the reviewer make suggestion for improving this paper. Our thanks to all of them.

References

1. R. D. Schraft and H. Volz, *Serviceroboter, Innovative Technik in Dienstleistung und Versorgung*. Berlin, Heidelberg, New York: Springer Verlag, 1996.
2. E. Prassler, R. Dillmann, and H.-B. K. (Hrsg.), *Robotik in Deutschland: Lehre, Forschung und Entwicklung*. Aachen: Shaker Verlag, 1998.
3. S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, pp. 21–71, 1998.
4. R. Bajcsy and J. Kosecka, "The problem of signal and symbol integration: A study of cooperative mobile autonomous agents behaviors," in *Proceedings from DAGM Symposium Bielefeld, Germany, September, 1995*.
5. K. Sekiyama and T. Fukuda, "Modeling and controlling of group behavior based on self-organizing principle," in *Proc. of the 1996 IEEE International Conference on Robotics and Automation (ICRA'96)*, vol. 2, pp. 1407–1412, 1996.
6. A. Birk, "Behavior-based robotics, its scope and its prospects," in *The 24th Annual Conference of the IEEE Industrial Electronics Society, IECON'98*, 1998.
7. H. Surmann, A. Kanstein, and K. Goser, "Self-Organizing and Genetic Algorithms for an Automatic Design of Fuzzy Control and Decision Systems," in *Proceedings of the First European Congress on Fuzzy and Intelligent Technologies, EUFIT'93, Aachen*, pp. 1097 – 1104, 7. - 10.09.1993.
8. C. C. Lee, "Fuzzy Logic in control systems: fuzzy logic controller," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 20, pp. 404–435, 1990.
9. E. H. Ruspini, "Fuzzy Logic in the FLAKEY Robot," in *IIZUKA'90, Proceedings of the International Conference on Fuzzy Logic and Neural Networks, Iizuka/Japan*, pp. 202 – 206, 20.7. - 23.7.1990.
10. Y. Maeda, M. Tanabe, M. Zuta, and T. Takagi, "Hierarchical Control for Autonomous Mobile Robots with Behavior-Decision Fuzzy Algorithm," in *Proceedings of the IEEE International Conference on Robotics and Automation, Nice/France*, pp. 117 – 122, 12.5. - 14.5.1992.
11. A. Saffiotti, E. H. Ruspini, and K. Konolige, "Blending Reactivity and Goal-Directedness in a Fuzzy Controller," in *Second IEEE International Conference on Fuzzy Systems, San Francisco*, pp. 134 – 139, 28.3. - 1.4.1993.
12. Y. Watanabe and F. G. Pin, "Sensor-Based Navigation of a Mobile Robot Using Automatically Constructed Fuzzy Rules," in *International Conference on Advanced Robotics, Tokyo/Japan*, pp. 81 – 87, 1.11. - 2. 11.1993.
13. P. Reignier, "Fuzzy logic techniques for mobile robot obstacle avoidance," *Robotics and Autonomous Systems*, vol. 12, pp. 143 – 153, 1994.
14. E. Tunstel and M. Jamshidi, "Fuzzy Logic and Behavior Control Strategy for Autonomous Mobile Robot Mapping," in *FUZZ-IEEE World Congress on Computational Intelligence, Orlando*, pp. 514 – 517, 26.6. - 2.7.1994.
15. H. Surmann, J. Huser, and J. Wehking, "Path planning for a fuzzy controlled autonomous mobile robot," in *Fifth International Conference on Fuzzy Systems, FUZZ-IEEE 96, New Orleans, USA*, pp. 1660 – 1665, Sept. 1996.
16. A. Saffiotti, E. H. Ruspini, and K. Konolige, *A Multivalued Logic Approach to Integrating Planning and Control*. Menlo Park, CA 94025, USA: Artificial Intelligence Center, SRI International, Technical Note No. 533, 4/1994.
17. A. 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), pp. 122 – 126, 1993.
18. E. von Puttkamer, "Sensorintegration zur Geometrischen Weltmodellierung," *it+ti*, vol. 1/94, pp. 34 – 38, 1994.

19. V. Gorrini and H. Bersini, "Recurrent Fuzzy Systems," in *FUZZ-IEEE World Congress on Computational Intelligence, Orlando*, pp. 193 – 198, 26.6. - 2.7.1994.
20. T. InfraLogic, *Fuzzy-C Development System User's Manual*. Togai InfraLogic, 1990.
21. S. Thrun, "A Livelong Perspective for Mobile Robot Control," *Intelligent Robots and Systems, Munich, September 12-16*, vol. 1, pp. 23–30, 1994.
22. G. Dudek, "Robotic exploration as graph construction," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 6, pp. 859–865, 1991.
23. H. Surmann, J. Huser, and L. Peters, "A fuzzy system for indoor mobile robot navigation," in *Fourth IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 95, Yokohama, Japan*, pp. 83–88, 20–24 Mar. 1995. Distinguished with *Robot Intelligence Award*.
24. H. Surmann, K. Heesche, M. Hoh, K. Goser, and R. Rudolf, "Entwicklungsumgebung für Fuzzy-Controller mit neuronaler Komponente," *VDE-Fachtagung: "Technische Anwendungen von Fuzzy-Systemen"*, Dortmund, pp. 288–297, 12–13 Nov. 1992.
25. H. Surmann, A. P. Ungerling, T. Kettner, and K. Goser, "What kind of hardware is necessary for a fuzzy rule based system," in *FUZZ-IEEE World Congress on Computational Intelligence, Orlando*, pp. 274 – 278, 26.6. - 2.7.1994.