Fuzzy Logic Rules for Mapping Sensor Data to Robot Control

Jianwei Zhang, Frank Wille and Alois Knoll
Faculty of Technology, University of Bielefeld, 33501 Bielefeld, Germany
Email: zhang|fwille|knoll@techfak.uni-bielefeld.de

Abstract

We use fuzzy logic rules to directly map sensor data to robot control outputs by classifying a set of typical subtasks, such as "path tracking", "local collision avoidance", "contour tracking", "situation evaluation", etc. With the help of existing heuristics, the decision-making process for each subtask can be modelled and represented with "IF-THEN" rules. The underlying concepts of mapping with fuzzy logic rules are briefly explained by considering the proximity sensors, the control of speed and steering angle of a mobile robot. The development of these fuzzy rules is explained, typical rules for dealing with various motion situations are listed. The modularly developed fuzzy rule bases can be integrated to realise task-level programming and the exploration task. Experiments with the mobile robot validate this concept.

1 Introduction

The conventional robot control architecture employs the so called SMPA (*Sensing-Modelling-Planning-Action*) strategy, which is based on the classical symbolism of AI. Recently, problems are found out with such a control architecture: 1). Algorithms for modelling and planning can be highly complex; 2). The time delay from perception to action is usually long due to the computational distance between them; 3). A system based on such an architecture is not fault tolerant. Therefore, a lot of recent work on robot control aims at finding efficient sensor-based solutions to shorten the distance between perception and action. The behaviour-based approaches [Se95] use the parallel instead of hierarchical control structure and try to implement the so-called *embodied, situated* behaviours without building complete world models and planning any actions and motions. Obviously, both control architectures have advantages and disadvantages, so it becomes an important topic how to integrate them. Our previous work [ZWK96] presents a fuzzy control solution towards such an integration for robot motion control.

Beyond the classical control algorithms, like PID control and potential field [BLL92], "intelligent computing methods", like neural networks and fuzzy logic, are increasingly applied in sensory systems and robot control. Fuzzy control approach is gradually becoming an important approach for sensor-based control of robots. Applications range from the purely reactive fuzzy controller, e.g. [PW93], to the mixture of "behaviours" like single-goal directness and reactive collision-avoidance, e.g. [Ish95] and [Rus95].

In this paper, we present the development of modular fuzzy rule bases for realising several typical subtasks, such as "*path tracking*", "*local collision avoidance*", "*contour tracking*", "*situation evaluation*", etc. The concepts of mapping the sensor space to the control output space with fuzzy logic rules are illustrated in section 2. The formulation and implementation of the fuzzy rule bases are presented in details in section 3. Section 4 demonstrates briefly the realisation of collision-free movement from start to goal as well as an exploration task. This section also discusses the problem of controller optimisation. The last section summarises the advantages of using fuzzy logic rules for robot control.

2 Robot, Sensors and their Connection

2.1 A Mobile Robot System for Experiments

The concept presented in this paper has been implemented for the real mobile gripper system *Khepera*. The mobile platform of *Khepera* is of circular shape with a diameter of 52mm. Additional modules can be mounted on the top of *Khepera*, e.g. a gripper module, see Fig. 1. The environment is currently observed by eight infra-red (IR) sensors (six at the front and two at the rear), while a vision module is now being tested. The sensibility of the IR sensors varies for different objects and is limited to 5cm. The directly controlled values are the velocities of the robot's left and right wheel, which are denoted as v_l and v_r respectively. In order to test robot independent control programs, we have derived a computation table with which the robot's forward speed (*Speed*) and steering angle (*Steer*) can be translated to v_l and v_r.

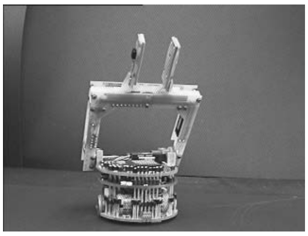


Figure 1. A mobile robot gripper system for experiments

Since the proximity sensors as well as the controller outputs *Speed* and *Steer* are imprecise, it does not make sense to develop complicated, exact algorithms to use the sensor data for world modelling and to control the robot motion with a high resolution. If the control of a mobile robot is compared with the driving behaviour of a human, it can be well understood that fuzzy logic rules emulating the human decision-making process with "IF-THEN" rules can be applied in the design of such a robot controller.

2.2 Sensor Data

In order to develop a robust on-line robot controller, external and internal sensor data should be applied directly in each control cycle instead of building and updating the world model. If sensor data is coupled with motion control in a simple form, the robot can decide its reaction in time. The word "*situatedness*" used by Brooks ([Se95]) contains the similar idea. Simon [Sim69] summarised with "*bounded rationality*" the principle that human-beings often use only incomplete or imprecise knowledge for problem-solving.

Sensor data needed for direct integration in motion control possess the following features:

- They are relative. These data are mainly derived from the external sensor measurements and their derivatives or the differences between the sensor values and the internal model. Such a variable value is not related to the robot or sensor alone, but to the interaction between the robot and its environment.
- They are local. Normally, only part of the environment, which is directly involved in the current robot motion, is perceived by the sensor system. Each sensor measurement represents one aspect of the object's features. No time-costly sensor fusion is performed (sensor data fusion is then transformed to task fusion).
- They are task-oriented. Modelling and interpretation of the sensor data depend on the control tasks. Only the control-relevant data are selected, pre-processed and represented.

2.3 Fuzzy Logic Rules Connecting Sensor Data and Motion Control

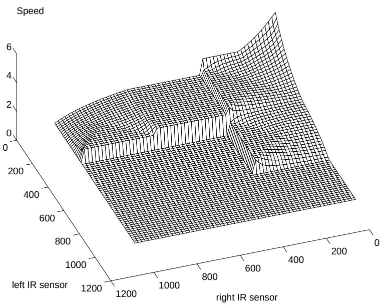
The variables of the logic sensors as well as the control variables can be viewed as *linguistic variables*, such as *Sensor_Left, Sensor_Right, Speed, Steer*. Each linguistic variable can be covered with overlapping *linguistic terms* specified by fuzzy sets, like NB (*negative big*), NM (*negative middle*), NS (*negative small*), Z (*zero*), P (*positive*), PM (*positive middle*) and PB (*positive big*).

Fuzzy rules for control have the following form:

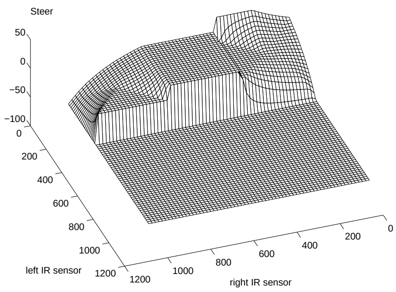
"**IF** (a set of conditions is fulfilled) **THEN** (a set of consequences can be determined)"

To illustrate the procedure of mapping sensor space to control space, a simplified example for tracking the contour of an object is shown in Fig. 2, in which the mappings of the input variables *Sensor_Left* and *Sensor_Right* to the output variables *Speed* and *Steer* are depicted. The upper figure (a) shows that the *Speed* output will get a high value, when both IR sensors supply a low input (no obstacle in vicinity), and a low value otherwise. The lower figure (b) shows the dependency between the IR sensors and the steering angle. *Steer* will be negative if the the right IR sensor has a high value and positive if the sensor has a low value, but only if the left IR sensor detects no obstacle on the left side at the same time. So the robot is able to follow the contour of an object in clockwise direction. The linguistic terms of the input and output variables can be specified with fuzzy sets using triangles, trapezes, B-spline basis functions[1], or they can be just selected as a crisp value (fuzzy-singleton). The linguistic terms of variables *Speed* and *Steer* in Fig. 3 are defined by triangular fuzzy sets.

[1] We introduced an approach to model fuzzy sets with B-spline basis functions, see [ZK96].



(a) the forward speed



(b) the steering angle

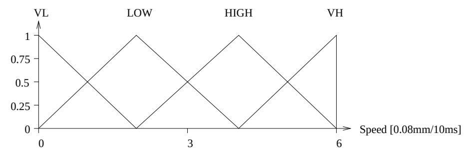
Figure 2. Mapping sensor data to the control output

3 Development of the Rule Bases

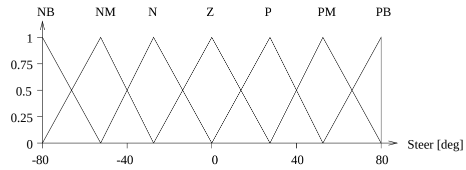
3.1 Path Tracking (PT)

First, we introduce the rule base "path tracking" (Fig. 4), which generates the appropriate speed and steering angle to be able to follow the current path segment to the next subgoal. Subgoals can be planned under the given representation of the environment, see [ZWK96]. It is required to pre-calculate the two input variables *shortest_distance_to_path* (denoted by a) and *angle_of_divergence* (denoted by d) first.

- d: The shortest distance between the robot and the path segment, connecting the previous subgoal and the next one. This linguistic variable is covered with the following linguistic terms, each of which is defined by the fuzzy sets shown in Fig. 5(a):
  - NB: far from the path to the left
  - NM: not too far from the path to the left



(a) the forward speed



(b) the steering angle

Figure 3. Linguistic terms of the two output variables

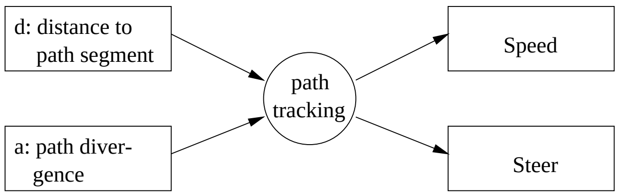
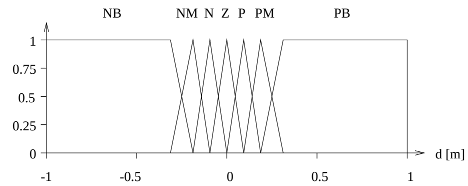


Figure 4. Connecting the input and output variables with the rule base "path tracking"

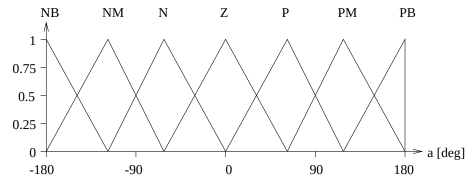
  - N: slightly from the path to the left
  - Z: almost on the path
  - P: slightly from the path to the right
  - PM: not too far from the path to the right
  - MB: far from the path to the right

- a: The angular divergence between orientation of the path and the robot with the following linguistic terms (Fig. 5(b)):
  - NB: driving into the opposite direction, slightly to the left
  - NM: direction is totally off the path to the left
  - N: direction is slightly off the path to the left
  - Z: almost on the path segment
  - P: direction is slightly off the path to the right
  - PM: direction is totally off the path to the right

- MB: driving into the opposite direction, slightly to the right



(a) the shortest distance from path



(b) the angular divergence

Figure 5. Linguistic terms of two inputs for "path tracking"

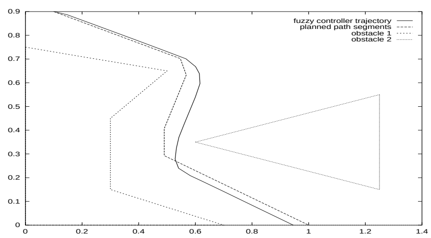


Figure 6. Trajectory of the controller using the rule base "path tracking"

By classifying "situations" of the robot's current position to the path segment to be tracked, rules for path tracking to the next subgoal can be developed. In appendix A, 49 rules for "path tracking" are listed. It is the task of the main control program to verify the current robot position and to switch to the next path segment.

A typical fuzzy rule of this module looks like this:

IF (d IS N) AND (a IS Z) **THEN** (*Speed* IS HIGH) AND (*Steer* IS P),

which is the fuzzy logic representation of the following heuristic rule: "If the robot is located slightly to the left of the path, but its orientation is almost on the path, then it will steer slightly to the right by applying a high speed."

Fig. 6 shows an example of the trajectory, realised by the fuzzy controller, to track a sequence of pre-planned path segments.

3.2 Local Collision Avoidance (LCA)

The LCA rule base (Fig. 7) is assigned to the task to avoid collisions with unknown or moving obstacles. By observing the current values of the proximity sensors, LCA calculates the speed and steering angle, which is required to avoid obstacles.

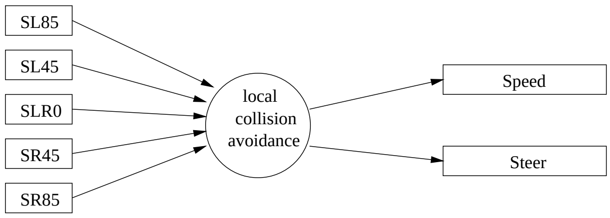


Figure 7. Connecting variables with the rule base "local collision avoidance"

- *SL85*, *SL45*, *SLR0*, *SR45*, *SR85*[2]: current value of the proximity sensors.

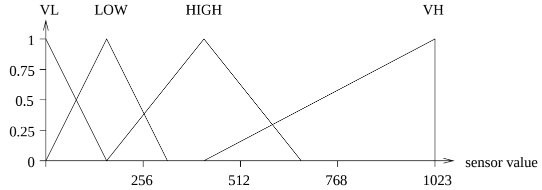
  The four linguistic terms (Fig. 8 (a)) are based on triangular membership functions, which have different distances from each other, because of the non-linearity of these sensors.

  - VL: no obstacle in sight
  - LOW: obstacle is far away
  - HIGH: obstacle is near
  - VH: almost in touch with obstacle

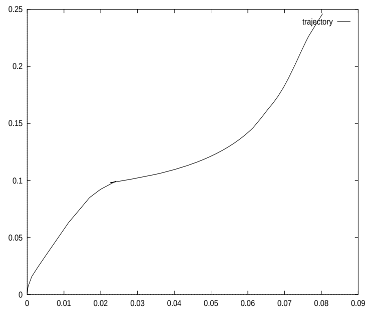
The fuzzy rules can be extracted by modelling the human experiences coping with the following situations: "dead end", "obstacle from right", "obstacle from left", "obstacle ahead", "obstacle from half-left/right", "no obstacle nearby". The whole rule set is listed in appendix C.

Fig. 8 (b) shows the trajectory of the robot avoiding an unknown obstacle in front of its motion direction on the left.

[2]They are referring to the IR sensors arranged at different angles, e.g. *SL85*: "sensor on the left at angle 85°.



(a) Linguistic terms for proximity sensors



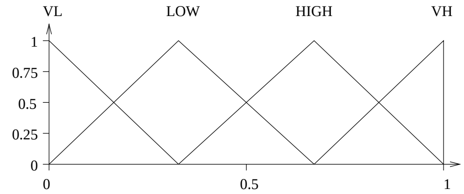
(b) Trajectory of avoiding an unknown obstacle, scales in m

Figure 8. Avoiding local collisions based on proximity sensors

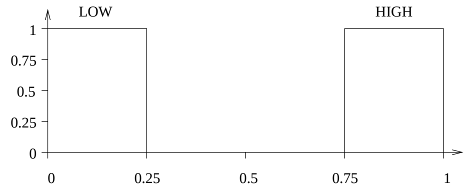
3.3 Situation Evaluation (SE)

The rule base "situation evaluation" uses the proximity sensors as input (Fig. 8 (a)) and generates two output variables: the importance priority K and the replanning selector *Replan*. The rule base calculates the importance priority of each module for all possible situations.

- K: importance priority for the LCA rule base. Each specific situation gets its importance priority assigned (Fig. 9 (a)).
  - VL: no obstacle avoidance, subgoal approach only
  - LOW: slightly doing obstacle avoidance, mainly subgoal approach
  - HIGH: mainly obstacle avoidance, slightly trying to approach subgoal
  - VH: obstacle avoidance has priority, subgoal approach is irrelevant
- *Replan*: deciding if a situation, which requires the path planning procedure to be invoked once again, is reached. That will be indicated by a high value in *Replan* (Fig. 9 (b)).
  - LOW: no replanning required
  - HIGH: replanning required



(a) the *importance priority*



(b) *replanning*

Figure 9. Linguistic terms for two state variables

The coalesced rule base for local collision avoidance and situation evaluation is listed in appendix C.

A typical fuzzy rule of this module looks like this:

IF (*SL85* IS HIGH) AND (*SL45* IS VL) AND (*SLR0* IS VL) AND (*SR45* IS VL) AND (*SR85* IS VL) **THEN** (*speed* IS LOW) AND (*Steer* IS PM) AND (K IS HIGH) AND (*Replan* IS LOW)

"If the leftmost proximity sensor detects an obstacle which is near, and the other sensors detect no obstacle at all, then steer halfway to the right at low speed. Mainly perform obstacle avoidance. No replanning required."

3.4 Contour Tracking (CT)

By evaluating the current values of the five proximity sensors (see LCA), this rule base generates *Speed* and *Steer* for tracking the contour of an object, Fig. 10. If the object was found, the robot always tries to follow its outline clockwise, which is reflected by a "HIGH" state in the third output variable, called *InContact*.

The rule base is listed in appendix B. A typical fuzzy rule of this module looks like this:

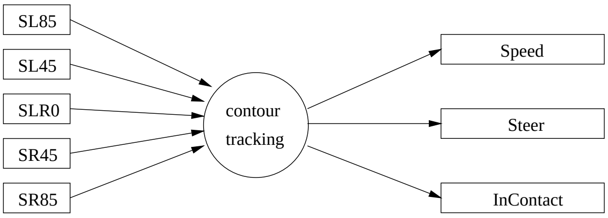


Figure 10. Connecting variables with the rule base “contour tracking”

IF ($SL85$ IS VL) AND ($SL45$ VL) AND ($SR85$ IS HIGH) **THEN** ($Speed$ IS LOW) AND ($Steer$ IS Z) AND ($InContact$ IS $HIGH$)

“If the proximity sensors on the left side detect nothing and the rightmost sensor detects that the obstacle is nearby, then continue to follow the obstacle’s outline straight ahead at low speed. The robot is in contact with the obstacle.”

4 Experiments

4.1 Integrating the Deliberative and the Reactive Strategies

The fuzzy rule bases introduced above are applied to integrate the deliberative and the reactive strategies. A test environment (Fig. 11) is built which includes not only the known static obstacles but also unknown moving objects. The first step of this integrated concept is to search a sequence of critical points. A coarse, rather than a detailed geometric fine path is to be found through which collisions with the known obstacles can be avoided. In the second step of motion execution, on-line sensor data of different granularity provide the information needed for avoiding uncertainties which cannot be pre-modelled.

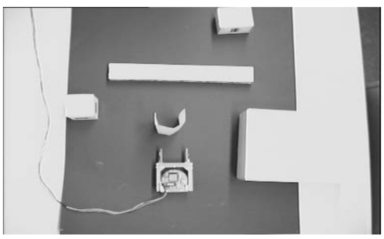


Figure 11. A test environment

A fuzzy controller is used for executing subgoal-guided motions. Fuzzy rule bases, like “local collision avoidance”, can work together with the rule base “path tracking”. The flow chart of the robot control program is shown in Fig. 12. In this way, during motion between subgoals, the robot does not move along a statically planned trajectory, but under the control of a subgoal-guided, sensor-based controller. On-line sensor data can be evaluated to detect local collisions and the motion control is adapted to the dynamic environment.

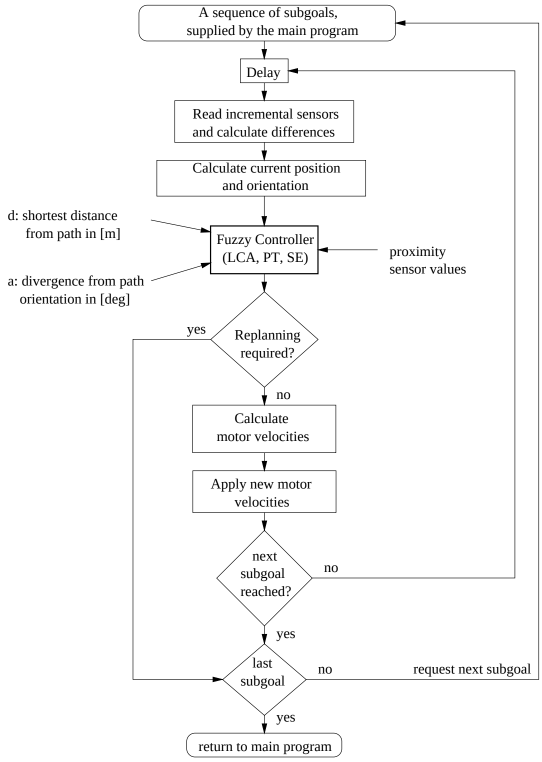


Figure 12. Flow chart for integrating “path tracking” and “local collision avoidance”

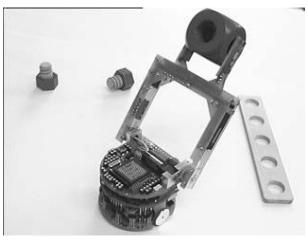


Figure 13. Grasping an object after the exploration

4.2 An Exploration Task

As long as the robot is in contact with the obstacle the main program can measure the length of an obstacle's side by reading the incremental sensor of the outer wheel. So it can decide whether the object is small enough for grasping it. After a definite time limit has passed, without discovering a side which was too long, the grasping process is initiated. When the object was grasped (Fig. 13), the robot continues driving by doing *normal* LCA and waits for the user to give a command, so that the object can be dropped. After that, the whole cycle is repeated, see Fig. 14.

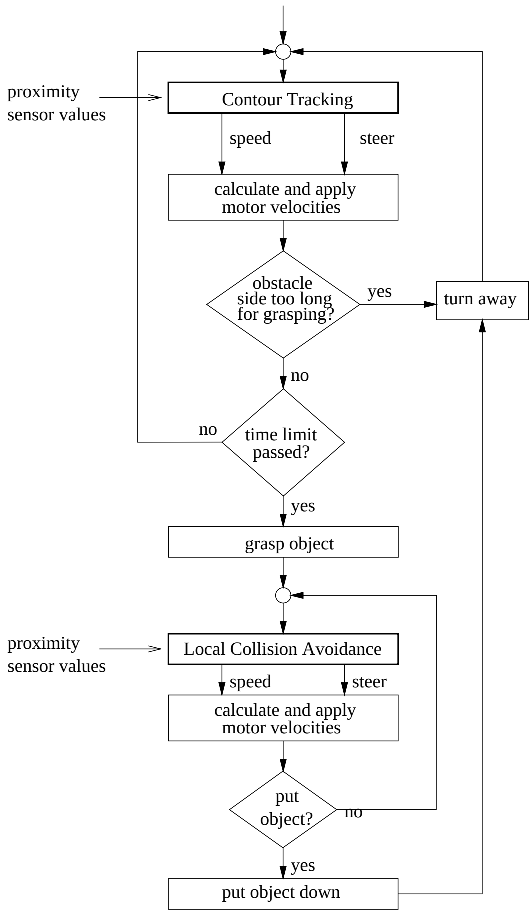
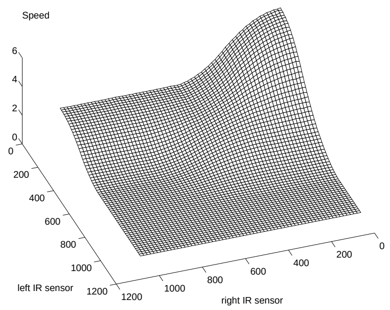


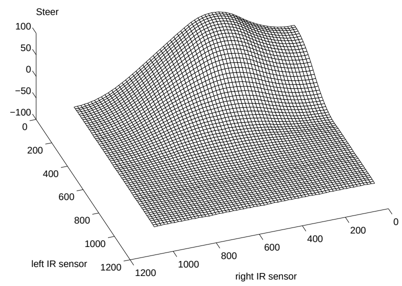
Figure 14. Flow chart for contour tracking with object detection

4.3 Improvement and Self-Optimisation of the Fuzzy Controller

The classical fuzzy controller employs the "Max-Min" (or "Max-Product") inference and centroid defuzzification method. If only triangles and trapezes are used for specifying the fuzzy sets, it can be found that the output of the fuzzy controller is not smooth or sometimes even discontinuous, see Fig. 2. By using B-spline fuzzy controllers [ZK96], the controller output can be significantly smoothed. If B-spline basis functions of order 3 with piecewise polynomials of degree 2 are used, the outputs of the same example of "contour tracking" are C^1-continuous, i.e. both the output and its derivative are continuous, see Fig. 15.



(a) the front speed



(b) the steering angle

Figure 15. 2-Input-2-output mapping for subtask "contour tracking" with B-spline basis functions as membership functions for inputs

We are currently working on the adaptation and self-learning of the B-spline fuzzy controller for mobile robots. Since the values for specifying linguistic terms of the controller outputs actually determine the shape of the input-output space, they are selected as the main parameters for optimisation. We have successfully used such a fuzzy controller to approximate any nonlinear functions, which can be applied for the supervised learning. Furthermore, the structure of the B-spline fuzzy controller is also suitable for unsupervised learning, if a performance function of robot motion is chosen. Our future work will be on developing an adaptive sensor-based (distance sensor and vision based) control system which can work in totally unknown as well as partly-known environments.

5 Summary

The main advantages of using fuzzy control for mobile robots can be summarised as the following:

modularity: Actually, the principle of fuzzy control is intrinsically modular: a rule base is generated by increasingly developing each single rule, which has the linguistic interpretation and its own control function. The order of these rules does not make any difference, both during the controller design and the rule evaluation. If we regard a rule base for fulfilling a certain subtask as a separate module, it is easy to understand that different rule bases can be developed independently and then integrated for realising a high-level task.

efficiency: The modular design features enable a significant reduction of developing time, which is achieved by simple design of a single rule base, fast prototyping and efficient debugging.

transparency: Since the sensor-based robot control strategy takes advantages of the heuristics of human experiences, the control procedure is then well interpretable, which is one important property of the intelligent control philosophy.

real-time features: Thanks to the simple computation and the possibility of parallel processing of fuzzy rules, fuzzy controllers can run in real-time. If the dedicated hardware is applied, the execution time can be reduced to less than the processing time of many sensor data.

adaptability: By selecting a type of fuzzy controller and designing appropriate learning algorithms, the adaptability of the robot controller in an unknown environment can be realised.

References

[BLL92] J. Barraquand, B. Lanlois, and J.-C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on System, Man and Cybernetics*, 22(2):224–241, 1992.

[Ish95] S. Ishikawa. A method of autonomous mobile robot navigation by using fuzzy control. *Advanced Robotics*, 9(1):29–52, 1995.

[PW93] F. G. Pin and Y. Watanabe. Driving a car using reflexive fuzzy behavior. *IEEE International Conference on Fuzzy Systems*, pages 1425–1430, 1993.

[Rus95] E. H. Ruspini. Fuzzy logic-based planning and reactive control of autonomous mobile robots. In *IEEE International Conference on Fuzzy Systems*, pages 1071–1076, 1995.

[Se95] L. Steels and R. Brooks (editors). *The Artificial Life Route to Artificial Intelligence: building Embodied, Situated Agents.* Lawrance Erlbaum Associates Publishers, 1995.

[Sim69] H. A. Simon. *The Sciences of the Artificial.* MIT Press, Cambridge, MA, 1969.

[ZK96] J. Zhang and A. Knoll. Constructing fuzzy controllers with B-spline models. In *IEEE International Conference on Fuzzy Systems*, 1996.

[ZWK96] J. Zhang, F. Wille, and A. Knoll. Modular design of fuzzy controller integrating deliberative and reactive strategies. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1996.

A Appendix A - Rule Base PT

Input		Output	
d	*a*	*Steer*	*speed*
Completely off the path on the left side			
NB	NB	PB	LOW
NB	NM	PB	LOW
NB	N	PM	LOW
NB	Z	PM	HIGH
NB	P	P	HIGH
NB	PM	Z	VH
NB	PB	N	HIGH
Far away on the left side			
NM	NB	PB	LOW
NM	NM	PB	LOW
NM	N	PM	LOW
NM	Z	PM	HIGH
NM	P	P	HIGH
NM	PM	Z	HIGH
NM	PB	N	HIGH
Slightly left of the path			
N	NB	PB	LOW
N	NM	PB	LOW
N	N	PM	HIGH
N	Z	P	HIGH
N	P	Z	VH
N	PM	N	HIGH
N	PB	NB	LOW
Almost on the path			
Z	NB	PB	LOW
Z	NM	PM	LOW
Z	N	P	HIGH
Z	Z	Z	VH
Z	P	N	HIGH
Z	PM	NM	LOW
Z	PB	NB	LOW
Slightly right of the path			
P	NB	PB	LOW
P	NM	P	HIGH
P	N	Z	VH
P	Z	N	HIGH
P	P	NM	HIGH
P	PM	NB	LOW
P	PB	NB	LOW
Far away on the right side			
PM	NB	P	HIGH
PM	NM	Z	HIGH
PM	N	N	HIGH
PM	Z	NM	HIGH
PM	P	NM	LOW
PM	PM	NB	LOW
PM	PB	NB	LOW
Completely off the path on the right side			
PB	NB	P	HIGH
PB	NM	Z	VH
PB	N	N	HIGH
PB	Z	NM	HIGH
PB	P	NM	LOW
PB	PM	NB	LOW
PB	PB	NB	LOW

B Appendix B - Rule Base CT

Input					Output		
SL85	SL45	SLR0	SR45	SR85	*Speed*	*Steer*	InContact
VL	VL	VL	VL	VL	VH	Z	LOW
LOW	-	-	VL	VL	LOW	NM	LOW
HIGH	-	-	VL	VL	VL	NM	LOW
VH	-	-	VL	VL	VL	NM	LOW
VL	VL	LOW	VL	VL	LOW	N	LOW
-	-	HIGH	-	-	VL	NM	LOW
-	-	VH	-	-	VL	NM	LOW
VL	VL	VL	VL	LOW	LOW	PM	HIGH
VL	VL	-	-	HIGH	LOW	Z	HIGH
VL	VL	-	-	VH	VL	NM	HIGH

C Appendix C - Rule Base LCA und SE

Input					LCA output		SE output	
SL85	SL45	SLR0	SR45	SR85	Sp.	St.	K	Repl.
Dead end situation. Requires replanning.								
VH	VH	-	VH	VH	VL	Z	VH	HIGH
HIGH	VH	VH	VH	VH	VL	Z	VH	HIGH
VH	HIGH	VH	VH	VH	VL	Z	VH	HIGH
VH	VH	VH	HIGH	VH	VL	Z	VH	HIGH
VH	VH	VH	VH	HIGH	VL	Z	VH	HIGH
HIGH	HIGH	VH	VH	VH	VL	Z	VH	HIGH
VH	HIGH	HIGH	VH	VH	VL	Z	VH	HIGH
VH	VH	HIGH	HIGH	VH	VL	Z	VH	HIGH
VH	VH	VH	HIGH	HIGH	VL	Z	VH	HIGH
Collision avoidance in free space - Obstacle from right								
VL	VL	VL	VL	LOW	HIGH	N	LOW	LOW
VL	VL	VL	LOW	LOW	LOW	NM	LOW	LOW
VL	VL	LOW	LOW	LOW	LOW	NB	HIGH	LOW
VL	LOW	LOW	LOW	LOW	LOW	NB	HIGH	LOW
VL	VL	VL	VL	HIGH	LOW	NM	HIGH	LOW
VL	VL	VL	LOW	HIGH	VL	NB	HIGH	LOW
VL	VL	LOW	LOW	HIGH	VL	NB	VH	LOW
VL	VL	VL	HIGH	HIGH	VL	NB	VH	LOW
VL	VL	HIGH	HIGH	HIGH	VL	NB	VH	LOW
VL	VL	VL	VL	VH	VL	NB	VH	LOW
VL	VL	VL	LOW	VH	VL	NB	VH	LOW
VL	VL	VL	HIGH	VH	VL	NB	VH	LOW
VL	VL	LOW	HIGH	VH	VL	NB	VH	LOW
VL	LOW	HIGH	HIGH	VH	VL	NB	VH	LOW
VL	VL	VL	VH	VH	VL	NB	VH	LOW
VL	VL	LOW	VH	VH	VL	NB	VH	LOW
VL	VL	VH	VH	VH	VL	NB	VH	LOW
VL	LOW	VH	VH	VH	VL	NB	VH	LOW
LOW	HIGH	VH	VH	VH	VL	NB	VH	LOW
Collision avoidance in free space - Obstacle from left								
LOW	VL	VL	VL	VL	HIGH	P	LOW	LOW
LOW	LOW	VL	VL	VL	LOW	PM	LOW	LOW
LOW	LOW	LOW	VL	VL	LOW	PB	HIGH	LOW
LOW	LOW	LOW	LOW	VL	LOW	PB	HIGH	LOW
HIGH	VL	VL	VL	VL	LOW	PM	HIGH	LOW
HIGH	LOW	VL	VL	VL	VL	PB	HIGH	LOW
HIGH	LOW	LOW	VL	VL	VL	PB	VH	LOW
HIGH	HIGH	VL	VL	VL	VL	PB	VH	LOW
HIGH	HIGH	HIGH	VL	VL	VL	PB	VH	LOW
VH	VL	VL	VL	VL	VL	PB	VH	LOW
VH	LOW	VL	VL	VL	VL	PB	VH	LOW
VH	HIGH	VL	VL	VL	VL	PB	VH	LOW
VH	HIGH	LOW	VL	VL	VL	PB	VH	LOW
VH	HIGH	HIGH	LOW	VL	VL	PB	VH	LOW
VH	VH	VL	VL	VL	VL	PB	VH	LOW
VH	VH	LOW	VL	VL	VL	PB	VH	LOW
VH	VH	VH	VL	VL	VL	PB	VH	LOW
VH	VH	VH	LOW	VL	VL	PB	VH	LOW
VH	VH	VH	HIGH	LOW	VL	PB	VH	LOW
Avoiding direct collision with obstacle ahead								
VL	VL	LOW	VL	VL	LOW	Z	HIGH	LOW
VL	VL	HIGH	VL	VL	VL	Z	VH	LOW
VL	VL	VH	VL	VL	VL	PB	VH	LOW
VL	LOW	HIGH	LOW	VL	LOW	Z	VH	LOW
VL	HIGH	HIGH	HIGH	VL	VL	Z	VH	LOW
VL	HIGH	VH	HIGH	VL	VL	PB	VH	LOW
VL	VH	VH	VH	VL	VL	PB	VH	LOW
Avoiding direct collision with obstacle from half-left/right								
VL	LOW	HIGH	VL	VL	LOW	PM	VH	LOW
VL	LOW	VH	VL	VL	VL	PB	VH	LOW
VL	LOW	LOW	VL	VL	LOW	PM	HIGH	LOW
VL	HIGH	VH	VL	VL	VL	PB	VH	LOW
VL	VH	HIGH	VL	VL	VL	PB	VH	LOW
VL	VH	VH	VL	VL	VL	PB	VH	LOW
LOW	HIGH	HIGH	LOW	VL	VL	PB	VH	LOW
HIGH	VH	HIGH	VL	VL	VL	PB	VH	LOW
HIGH	VH	VH	LOW	VL	VL	PB	VH	LOW
HIGH	VH	VH	HIGH	VL	VL	PB	VH	LOW
HIGH	VH	VH	HIGH	LOW	VL	PB	VH	LOW
VL	VL	HIGH	LOW	VL	LOW	NM	VH	LOW
VL	VL	VH	LOW	VL	VL	NB	VH	LOW
VL	VL	LOW	LOW	VL	LOW	NM	HIGH	LOW
VL	VL	VH	HIGH	VL	VL	NB	VH	LOW
VL	VL	HIGH	VH	VL	VL	NB	VH	LOW
VL	VL	VH	VH	VL	VL	NB	VH	LOW
VL	LOW	HIGH	HIGH	LOW	VL	NB	VH	LOW
VL	VL	HIGH	VH	HIGH	VL	NB	VH	LOW
VL	LOW	VH	VH	HIGH	VL	NB	VH	LOW
VL	HIGH	VH	VH	HIGH	VL	NB	VH	LOW
LOW	HIGH	VH	VH	HIGH	VL	NB	VH	LOW
No obstacle in vicinity								
VL	VL	VL	VL	VL	VH	Z	VL	LOW