Original paper

Soft Computing 1 (1997) 180–197 © Springer-Verlag 1997

The uses of fuzzy logic in autonomous robot navigation

A. Saffiotti

Abstract The development of techniques for autonomous navigation in real-world environments constitutes one of the major trends in the current research on robotics. An important problem in autonomous navigation is the need to cope with the large amount of uncertainty that is inherent of natural environments. Fuzzy logic has features that make it an adequate tool to address this problem. In this paper, we review some of the possible uses of fuzzy logic in the field of autonomous navigation. We focus on four issues: how to design robust behavior-producing modules; how to coordinate the activity of several such modules; how to use data from the sensors; and how to integrate high-level reasoning and low-level execution. For each issue, we review some of the proposals in the literature, and discuss the pros and cons of fuzzy logic solutions.

Keywords Fuzzy logic, Robotics, Navigation, Intelligent control, Autonomous behavior

1 Introduction

The goal of autonomous mobile robotics is to build physical systems that can move purposefully and without human intervention in unmodified environments – that is, in real-world environments that have not been specifically engineered for the robot. The development of techniques for autonomous robot navigation constitutes one of the major trends in the current research on robotics. This trend is motivated by the current gap between the available technology and the new application demands. On the one hand, current industrial robots lack flexibility and autonomy: typically, these robots perform pre-programmed sequences of operations in highly constrained environments, and are not able to operate in new environments or to face unexpected situations. On the other hand, there is a clear emerging market for truly autonomous robots. Possible applications include intelligent service robots for offices, hospitals, and factory floors; maintenance robots operating in hazardous or hardly accessible areas; domestic robots for cleaning or entertainment; semi-autonomous vehicles for help to disabled people; and so on.

Despite the impressive advances in the field of autonomous robotics in recent years, a number of problems remain. Most of the difficulties originate in the nature of real-world, unstructured environments, and in the large uncertainties that are inherent to these environments. First, prior knowledge about the environment is, in general, incomplete, uncertain, and approximate. For example, maps typically omit some details and temporary features, spatial relations between objects may have changed since the map was built, and the metric information may be imprecise and inaccurate. Second, perceptually acquired information is usually unreliable. The limited range, combined with the effect of environmental features (e.g., occlusion) and of adverse observation conditions (e.g., poor lighting), leads to noisy and imprecise data; and errors in the measurement interpretation process may lead to incorrect beliefs. Third, real-world environments typically have complex and unpredictable dynamics: objects can move, other agents can modify the environment, and relatively stable features may change with time (e.g., seasonal variations). Finally, the effect of control actions is not completely reliable: wheels may slip, and a gripper may lose its grasp on an object.

Traditional work in robotics has tried to overcome these difficulties by carefully designing the robot mechanics and sensors, or engineering the environment, or both. This approach is systematically adopted in industrial robots, but some amount of environment engineering has often been introduced in autonomous robotics as well, from the early days of Shakey [79] until today's service robots that follow a white or magnetic strip on the floor. Careful robot and environment engineering, however, increases costs, reduces robot's autonomy, and cannot be applied to all environments. If we want to build easily available robots that inhabit our homes, offices, or factory floors, we should accept the idea that the platform cannot be overly sophisticated, and that the environment should not be modified. The main challenge of today's autonomous robotics is to build robust control programs that reliably perform complex tasks in spite of the environmental uncertainties.

Received: *31 March 1997* / *Accepted*: *24 September 1997*

A. Saffiotti
IRIDIA, Université Libre de Bruxelles
50 av. F. Roosevelt, CP 194/6
B-1050 Brussels, Belgium
e-mail: asaffio@ulb.ac.be

This work was partially supported by the Belon project, founded by the *Communauté Française de Belgique*. I thank Enrique Ruspini and Kurt Konolige for illuminating discussions. A preliminary draft of this survey was fortunate to receive comments by many of the cited authors. I wish to thank them all, and in particular Isabelle Bloch, Marco Dorigo, Frank Hoffmann, François Michaud, François Pin, Claudio Sossai, Harmut Surmann, Eddie Tunstel, Giovanni Ulivi and Chris Voudouris.

It often claimed that the qualitative nature of fuzzy logic makes it a useful tool for dealing with problems characterized by the pervasive presence of uncertainty. The pages to follow are devoted to test the validity of this claim in the field of autonomous robotics. Before we begin our survey, however, we need to better clarify the intended meaning of the two key terms "fuzzy logic" and "uncertainty."[1]

According to Zadeh [132], we can talk of fuzzy logic in a *narrow* sense or in a *wide* sense. Taken in its narrow sense, fuzzy logic refers to the study of formal logical systems, equipped with a syntactic and a semantic apparatus, where propositions admit partial degrees of truth (e.g., [87, 52]). Very few systems of this kind have been proposed in the literature on autonomous robotics (but see [15]), and most occurrences of the expression "fuzzy logic" in this field take it in its wider sense; this sense encompasses any technique based on the theory of fuzzy sets. Loosely speaking, any formalism which is grounded in set theory can be "fuzzified" by replacing (crisp) subsets and individual elements by fuzzy sets, and set-theoretic operations by fuzzy operations. Examples include fuzzy topology, fuzzy mathematical morphology, fuzzy linear programming, fuzzy expert systems, and, of course, fuzzy rule-based control. In this survey, we use the expression "fuzzy logic" in this general sense to denote any technique based on fuzzy sets.

The second term we need to clarify is "uncertainty." Strictly speaking, uncertainty is not a property of information, but rather a property of an agent – or, more precisely, of the agent's mental state. An agent may be uncertain about the existence of an object, the value of a property, the truth of a proposition, or the action to perform. Yet, it is very common to talk of uncertain information. We maintain that what is actually meant by this is information which is "weak" in some respect, thus inducing a state of uncertainty in the agent. Consider a robot wishing to grasp a given block; this task requires that the position of the block be known with a high degree of precision – how high depends on the robot's grasping mechanism. The item of information (i) "The block is on the table" is too weak for this task, as it does not give us a unique position; we talk in this case of *imprecise* information. The item (ii) "The block is about the center of the table" is *vague*, as it does not give us a crisp position. And the item (iii) "The block was seen yesterday at coordinates (1, 3) " is *unreliable*, as the block may no longer be there. All these items may be regarded as uncertain, as they leave the agent in a state of uncertainty about the actual position of the block. Note that considering an item of information as uncertain may depend on the specific task, as in (i), or on the source of information, as in (iii).

In this note, we informally talk of *uncertain information* to mean information (knowledge or data) which has some weak property in the above sense. An essential requirement for an uncertainty representation formalism is thus the ability to capture these weak properties, that is, to represent the information at the level of detail which is available. As we shall show, fuzzy logic techniques have attractive features in this respect.

The rest of this paper is devoted to discuss how fuzzy logic can be used, and has been used, to address some of the problems posed by autonomous robot navigation. In the first part, we present in deeper detail the main challenges posed by autonomous navigation, and introduce an architectural framework that helps us to separate the different problems. In the second part, we focus on four of these problems: how to design robust behavior-producing modules; how to coordinate the activity of several such modules; how to use data from the sensors; and how to integrate high-level reasoning and low-level execution. For each problem, we discuss the ways in which fuzzy logic can help us, and review several proposals appeared in the literature. The exposition is informal, and we address the reader to the bibliography for the technical details. A few comments on the pros and cons of using fuzzy logic in this domain are appended to each section, and summarized in the conclusions. An up-to-date version of this note is maintained on-line at http://iridia.ulb.ac.be/FLAR/survey.html.

2
The challenges of autonomous navigation

Any approach to control a dynamic system needs to use some knowledge, or *model*, of the system to be controlled. In the case of a robot, this system consists of the robot itself *plus* the environment in which it operates. Unfortunately, while a model of the robot on its own can normally be obtained, the situation is different if we consider a robot embedded in the type of real-world, unstructured environments which we would like to consider. As noticed in the Introduction, these environments are characterized by the ubiquitous presence of

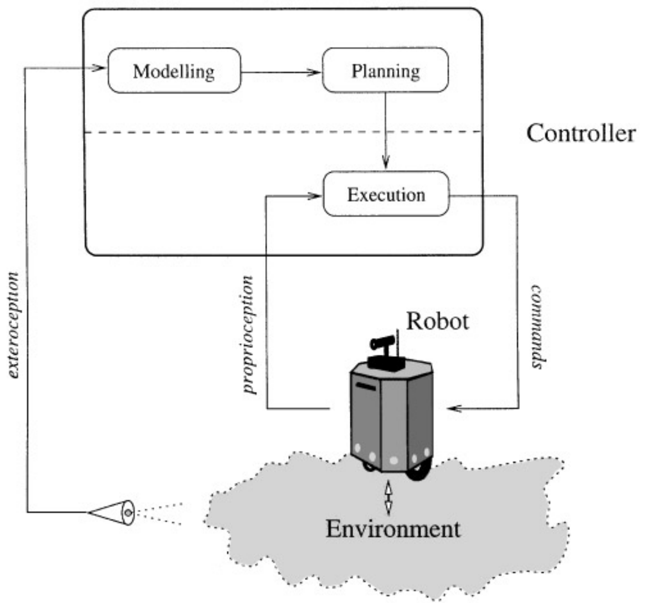

Fig. 1. Hierarchical architecture. The high-level layer builds a model of the environment and generates a plan for action. The low-level blindly executes this plan

[1] The discussion that follows is only meant to clarify the use of these terms in the context of this paper, and should not be taken as an attempt at defining them. Finding a precise definition of uncertainty is a very important task indeed, which is the object of a vivid debate in the field of artificial intelligence, but which is far beyond the scope of this paper. (See [46, 103, 60, 111] for a few samples of this debate.)

uncertainty; more annoyingly, the nature of the involved phenomena is such that we are often not able to precisely model or quantify this uncertainty. Consider, for example, the uncertainty induced by the presence of people. People move around, and they may change the position of objects and furniture; in most cases, however, we cannot hope to write, say, a meaningful probability distribution that characterizes these events. The interaction of the robot with the environment causes similar difficulties: the results of the robot's movement and sensing actions are influenced by a number of environmental conditions which are hard to be accounted for. For example, the error in the robot's motion may change as a result of a wet floor; the quality of visual recognition may be influenced by the lighting conditions; and the reliability of distance measured by a sonar sensor is influenced by the geometry and the reflectance properties of the objects in the environment.

A common strategy to cope with this large amount of uncertainty is to abandon the idea of completely modeling the environment at the design phase, and endow the robot with the capability of building this model by itself on-line. This strategy leads to the so-called *hierarchical* architectures, sketched in Figure 1.[2] The robot uses *exteroceptive* sensors, like a camera or a sonar sensor, to observe the state of the environment; it uses *proprioceptive* sensors, like a compass or shaft encoders on the wheels, to monitor the state of its own body.[3] By using the exteroceptive sensors, the robot acquires a model of the workspace as it is at the moment when the task must be performed. From this model, a planning program builds a plan that will perform the given task in the given environment. This plan is then passed to a lower-level control program for execution. Typically, execution proceeds "blindly" – the controller may use a model of the robot and monitor the state of the robot's effectors (proprioception), but it does not try to sense or model the environment anymore. In a sense, the hierarchical approach factors the environment out of the controlled system, thus making the control problem tractable. This approach has been extensively used in the robotics literature; in most cases, the plan consists of a path leading to the goal, and execution consists in tracking this path.

It is not difficult to see the limitations of the hierarchical approach when dealing with real-world environments. The model acquired by the robot is necessarily incomplete and inexact, due to the uncertainty in perception. Moreover, this model is likely to rapidly become out of date in a dynamic environment, and the plan built from this model will then turn out to be inadequate to the environment actually encountered during execution. The fact that the modeling and planning processes are usually computationally complex and time consuming exacerbates this problem: intuitively, the feedback loop with the environment must pass through all these processes – for this reason, this approach is also known as the "Sense-Model-Plan-Act", or SMPA approach. The complexity of the processes in the SMPA loop makes the response time of the robotic system of the order of seconds, far too much for dynamic environments.

By the mid-eighties technological improvements had caused the cost of mobile platforms and sensors to drop, and mobile robots began to appear in several AI research labs. Research on autonomous navigation was strongly pushed, and a number of new architectures were developed that tried to integrate perception and action more tightly. The general feeling was that planning should make as few assumptions as possible about the environment actually encountered during execution; and that execution should be sensitive to the environment, and adapt to the contingencies encountered. To achieve this, perceptual data has to be included into the executive layer, as shown in Figure 2. This apparently simple extension has two important consequences. First, it makes robot's interaction with the environment much tighter, as the environment is now included in a closed-loop with the (usually fast) execution layer. Second, the complexity of the execution layer has to be greatly increased, as it needs to consider multiple objectives: pursuing the tactical goals coming from the planner; and reacting to the environmental events detected by perception.

Following the seminal works by Brooks [21], Payton [88] and Arkin [2], most researchers have chosen to cope with this complexity by a divide and conquer strategy: the execution layer is decomposed into small independent decision-making processes, or *behaviors*. Figure 3 illustrates a general behavior-based organization of the execution layer. Each behavior fully implements a control policy for one specific sub-task, like following a path, avoiding sensed obstacles, or crossing a door.

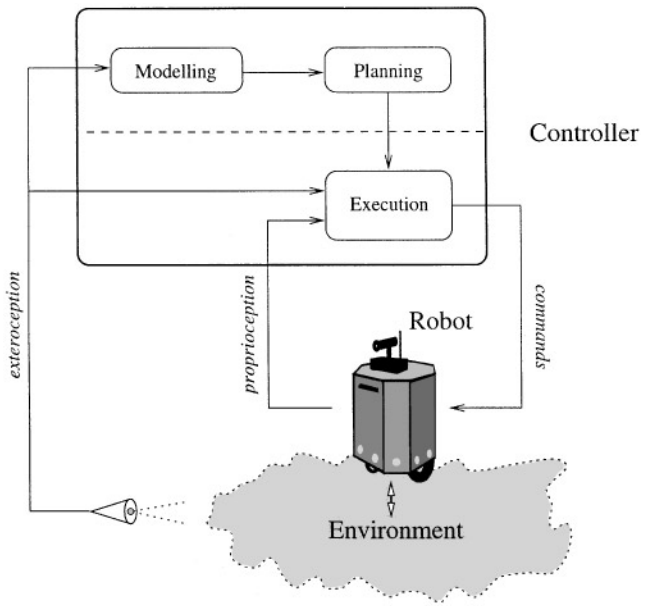

Fig. 2. Hybrid architecture. The lower layer uses perception to dynamically adapt plan execution to the environmental contingencies. The execution module is complex, because it must simultaneously consider demands coming from the plan and from the environment

[2] The term "hierarchical" is overloaded and inevitably ambiguous. It is used here to refer to a control hierarchy where the higher-level decides the set-point to be achieved by the lower-level. Some authors prefer the term "functional decomposition" to identify this type of architecture.
[3] In general, the exteroceptive sensors are physically mounted on the robot, but we prefer to draw them as a separate entity to emphasize the difference between exteroceptive and proprioceptive information. The picture should also include boxes for the sensor interpretation processes; we omit them, in order to better focus on the architectural aspects that are most relevant to our discussion.

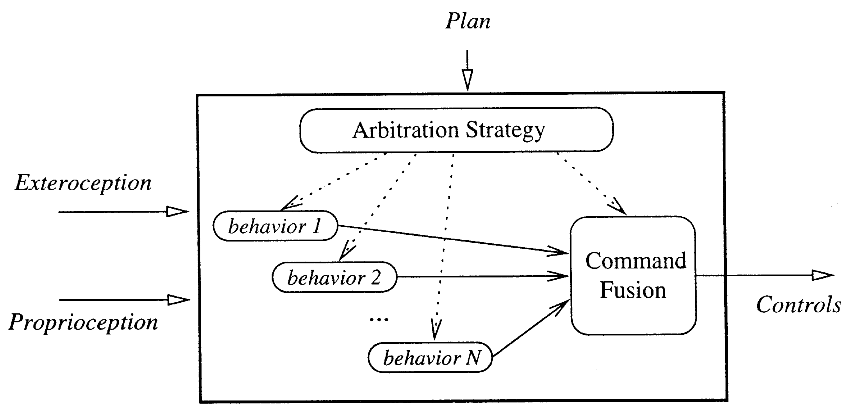

Fig. 3. Behavior-based organization of the execution module. Complexity is managed by a divide and conquer strategy.

The *arbitration strategy* decides which behaviors should be activated depending on the current goal and on the environmental contingencies. Several behaviors may be concurrently activated: in these cases, some form of *command fusion* is needed to combine the results from these behaviors into one effector command. Many proposals in the autonomous robotics literature adhere to this scheme, but differ in the emphasis put on each part. For instance, most proposals only allow one behavior at a time to run, and hence do not need a fusion policy. Brooks' subsumption architecture [21] is peculiar in that the arbitration policy is hard-wired at design time, and the modeling and planning layer is omitted altogether – indeed, Brooks claims that intelligent behavior can be achieved without using higher level processing at all [22].

Hybrid architectures do not solve the autonomous navigation problem, but they provide a convenient framework in which the different sub-problems can be dealt with and integrated. In the rest of this paper, we shall focus on some of these sub-problems which: (i) play an important role in autonomous navigation; and (ii) are particularly prone to solutions based on fuzzy logic. In particular, we shall consider the following issues:

- How to design individual robust behavior-producing modules;
- How to combine several such behaviors;
- How to use the information provided by the sensors in the modeling and in the execution modules; and
- How to integrate processes and representations that belong to different layers.

3 Behavior design

The first and most common application of fuzzy logic techniques in the domain of autonomous robotics is the use of fuzzy control to implement individual behavior units. Fuzzy logic controllers incorporate heuristic control knowledge in the form of if-then rules, and are a convenient choice when a precise linear model of the system to be controlled cannot be easily obtained. They have also shown a good degree of robustness in face of large variability and uncertainty in the parameters. (See, for instance, [61, 63, 30] for a reminder of the basic principles of fuzzy control.) These characteristics make fuzzy control particularly suited to the needs of autonomous robot navigation.

3.1 Proprioceptive behaviors

The "classical" type of control regime, or behavior, used in mobile robots is path tracking: the controller is given a path in the form of a sequence of coordinates in some reference frame, and it generates motor commands in order to follow this path as closely as possible. To do this, the controller needs to know the position of the robot with respect to the path, and the kinematic and dynamic characteristics of the robot. The position is usually inferred from measurements from the wheels encoders or other internal sensors: hence we talk of *proprioceptive* behavior.

Path tracking may be surprisingly difficult (see [68] for an overview). The kinematics and dynamics of the robot may be complex and non-linear, and the interaction between the vehicle and the terrain may be hard to model in general. As noted by Martínez *et al.* ([72], p. 26)

> Even when kinematics or dynamic models are considered, path tracking involves a substantial amount of heuristic knowledge.

These characteristics led several authors to use fuzzy control techniques for path tracking. An early proposal was presented by Isik [54], who considers tracking of a discrete path (i.e., a set of way-points to the goal), but does not report experimental results. More recently, Benreguieg *et al.* [9] and Zhang [133] have proposed fuzzy controllers for the pass-through-way-points problem. The latter work is based on the emulation of spline curves. Splines have nice properties from the point of view of motion control, but their computation is expensive; Zhang's fuzzy emulation can generate a good approximation in milliseconds or less.

Other researchers have applied fuzzy control to a simplified version of path tracking: to follow a straight line. Nishimori *et al.* [81] report a series of experiments where they tried several combinations of fuzzy operators and defuzzification techniques. Tanaka and Sano [119] derive fuzzy rules for line following from a fuzzy model of the controlled vehicle; this allows them to give a stability proof in the sense of Tanaka and Sugeno [120]. Both works have only been tested in computer simulations.

Finally, Ollero and colleagues have considered two ways to apply fuzzy control techniques to the tracking of arbitrary continuous paths. In the first one, the fuzzy controller directly generates the robot's controls [72]; in the second one, it generates the parameters used by a geometric method [83]. Both have been tested on a real robot with good results.

Generating a path is but one way to provide a behavior with a plan leading to a global goal. Another popular technique is to generate a potential field that has maxima around the obstacles, and a minimum at the goal location: the high-level process generates such a potential field based on a model of the environment, and the robot then navigates through this field by gradient descent [62]. A gradient descent policy can easily be implemented by fuzzy rules: for example, Makita *et al.* [71] propose a system based on this idea, which has been tested in simulation. (Note that other approaches dynamically generate the potential field during execution based on sensor data [56].)

3.2 Sensor-based behaviors

Tracking a precomputed path is an effective way to bring the robot to a target position when two conditions are verified: the assumptions used during the computation of the path are still valid at execution time (e.g., the environment was correctly modeled, and it has not changed afterwards); and the robot is able to reliably establish its position with respect to the path. As we have seen, these conditions are rarely met in real world environments, and many researchers have preferred to equip their robots with sensor-based behaviors. A sensor-based behavior implements a control policy based on external sensing; intuitively, the robot moves with respect to features in the environment, rather than with respect to an internally represented path. This type of control is also known as *compliant* control. Typical examples include moving along a wall or a contour, reaching a light source or a beacon, and avoiding obstacles.

The first reported uses of fuzzy control in mobile robotics belong to this type. For example, in 1985 Sugeno and Nishida developed a fuzzy controller able to drive a model car along a track delimited by two walls [115]. A single rotating sonar sensor was used to measure the distances from the walls. The fuzzy controller ran on-board on a 8080 microprocessor, and used heuristic rules derived from the observation of a human driver. The authors report encouraging results, but note that the controller is not robust with respect to sensor's errors, and that speed is limited to a very slow 1.7 cm/sec. Later experiments seem to have improved speed, but not tolerance to sensor's errors [114].

Shortly after the publication of Sugeno and Nishida's fuzzy car, Takeuchi *et al.* [118] developed a fuzzy controller for obstacle avoidance. The controller used a simple algorithm to obtain information about occupied and free areas in front of the robot from a video camera. The rules were experimentally derived with the help of a simulator. The authors were satisfied with the experimental performance, although perceptual errors in the vision system sometimes led to failures.

In the subsequent years, an increasing number of authors implemented fuzzy sensor-based behaviors in their robots. In most cases, attention was focused on the same two fundamental tasks as above: following environmental features (walls, road edges, white lines on the floor, or other), and avoiding obstacles. Examples can be found in [18, 89, 92, 127, 122, 5, 83] – all based on simple sensors (sonar or infrared) except [18] who uses visual data. A different type of behavior, called "tactical," is proposed by Murphy and Hawkins [78]. A tactical behavior acts as a supervisory controller, observing the recent execution in order to modify some navigation parameters; the authors report on a tactical fuzzy behavior for speed regulation based on the amount of recent turning, and on the density of obstacles. Pin and Bender [91] propose yet another type of behavior, called "memory processing," whose task is to manage an internal state used to detect and escape from limit cycles.

More extensively developed autonomous robots have been equipped with a wider repertoire of different behaviors, covering all the elementary sub-tasks that they need to perform. This is the case of the autonomous robots Flakey [106, 108], Marge [44], Moria [117], and Lobot [122]; these robots include fuzzy behaviors for going to a given position, for orienting towards a target, for docking to an object, for crossing a door, and so on. Other authors have developed several types of fuzzy behaviors, but have only reported experiments in simulation (e.g., [10, 39, 6, 75]).

3.3 Complex behaviors

All the behaviors discussed up to here can be classified as *basic*, in that they only care for one single objective, or performance criterion – for example, a path tracking behavior does not account for the possibility of an obstacle blocking the path. Some authors have used fuzzy control to implement *complex* behaviors, that take multiple objectives into account: for example, following a given path while avoiding unforeseen obstacles in real time. A complex behavior of this type can be regarded as a full implementation of the execution module in Figure 2.

Fuzzy controllers are typically designed to consider one single goal. If we want to consider two (or several) interacting goals, we have two options. We can write a set of complex rules whose antecedents consider both goals simultaneously; or we can write two sets of simple rules, one specific to each goal, and combine their outputs in some way. For example, if the two goals are tracking a path and avoiding obstacles, the first approach consists in using fuzzy rules of the general form

IF *path-condition*$_1$ AND *obstacle-condition*$_1$ THEN *command*$_1$,
IF *path-condition*$_2$ AND *obstacle-condition*$_2$ THEN *command*$_2$,

while the second approach would use fuzzy rules of the general form

IF *path-condition* THEN *command*$_1$,
IF *obstacle-condition* THEN *command*$_2$.

Examples of both approaches have been presented in the literature. The first approach has been used for navigating to a target while avoiding obstacles by Skubic *et al.* [110], who used a miniature infrared-based robot; and by Li [66], who used a simulated sonar-based robot. It was also used by Altrok *et al.* [125] on a model car for following a wall-bounded race track while compensating for the skidding and sliding due to the high speed.

The second approach has been used by Yen and Pfluger [131] and by Baxter and Bumby [7] for integrating path tracking and obstacle avoidance (only tested in simulation); and by Maeda *et al.* [70] for integrating vision-based wall following and obstacle avoidance on a Hero 2000 robot. The last work is peculiar in that the authors use predictive fuzzy control for the obstacle avoidance part. It should be noted that in each case the authors need to take extreme care of how to fuse the commands issued by the different rule-sets. We shall come back to this issue in section 4.

Whether a complex behavior is better implemented by a monolithic or a partitioned set of rules is a difficult question. The monolithic solution can take better care of the interactions between the goals, and should be preferred when these interactions are important. Unfortunately, the monolithic solution can easily become intractable, as the number of rules tends to grow exponentially in the size of the input space. When this space is large, a partitioned solution is likely to be easier to write and to debug; however, this solution leaves us with the difficult problem of how to re-combine the outputs of the individual rule-sets.

3.4 Discussion

Fuzzy control is credited with being an adequate methodology for designing robust controllers that are able to deliver a satisfactory performance in face of large amounts of noise in the input and of variability in the parameters. The key to this robustness is to be found in the interpolation mechanism implemented by fuzzy controllers, which embodies the idea that similar inputs should produce similar actions. In addition to this, the rule format and the use of linguistic variables make fuzzy control an adequate design tool for non-linear systems for which a precise mathematical model cannot be easily obtained, but for which heuristic control knowledge is available. Finally, fuzzy controllers lend themselves to efficient implementations, including hardware solutions.

These characteristics fit well the needs of autonomous robotics, where: (i) a mathematical model of the environment is usually not available; (ii) sensor data is uncertain and imprecise; and (iii) real-time operation is of essence. It is no surprise, then, if fuzzy control has been the first application of fuzzy logic techniques in this domain, and it is still the most common one. Early applications of fuzzy control in the robotics field include Sugeno and Nishida's model car [115], and the ping-pong playing robot developed by Hirota *et al.* [49].

In general, authors have noted three main advantages of the fuzzy control technology. Firstly, the fuzzy rule format makes it easy to write simple and effective behaviors for a variety of tasks, without having to use complex mathematical models. Secondly, thanks to their qualitative nature, fuzzy behaviors are prone to be transfered from one platform to another with few modifications [92, 78, 53]. Finally, the interpolative nature of fuzzy control results in smooth movement of the robot, and in graceful degradation in face of errors and fluctuations in sensor's data.

While the possibility to write an effective controller without using an explicit mathematical model constitutes a strength of fuzzy control, it is also the source of two important difficulties: first, having no mathematical model, we cannot use classical tools for formal design; second, once we have the controller, we cannot give any guarantee that it will produce the desired behavior other than empirical testing.

Concerning the first point, the design of fuzzy controllers is typically done by eliciting the fuzzy rules and the (input and output) membership functions from a human who knows how to control the system; debugging and tuning then proceeds by trials and errors. Whether or not this knowledge elicitation process is more effective than trying to build an analytical model of the system depends on the specific domain and task. A few proposals have appeared in the literature that use different techniques for obtaining the data needed to build a fuzzy controller; these include building the controller from a fuzzy model of the system [119], and extracting these data from the observation of the actions of a human operator [115, 69]. Several researches have also explored the use of learning techniques [10, 70, 23, 42, 51, 123, 19].

The second problem mentioned above, the formal analysis of a fuzzy behavior, is the object of intensive research in the field of fuzzy control. Some tools exist to prove stability *given that* a model of the system is available (see, e.g., [57] and bibliography therein; and [120] for an application to robotics). However, one may feel that what is really needed is a new set of *qualitative* performance criteria, and a set of formal tools that can tell when a given fuzzy controller will (approximately) satisfy these criteria [109].

4 Behavior coordination

Since the first appearance of behavior-based approaches in the mid-eighties, authors have noticed the importance of the problem of behavior coordination: how to coordinate the simultaneous activity of several independent behavior-producing units to obtain an overall coherent behavior that achieves the intended goal. The simplest example is the coordination of an obstacle avoidance behavior and a goal reaching behavior in order to safely reach the target despite the presence of unexpected obstacles.

Still today, the problem of behavior coordination is generally recognized as one of the major open issues in behavior-based approaches to robotics. In what follows, we show that fuzzy logic offers useful mechanisms to address this problem. As suggested in Figure 3, we split the behavior coordination problem into two conceptually different problems: (i) how to decide which behavior(s) should be activated at each moment – and, possibly, how much so; and (ii) how to combine the results from different behaviors into one command to be sent to the robot's effectors – possibly, taking weights into account. We call these the *behavior arbitration* and the *command fusion* problem, respectively.[4]

4.1 Arbitration

The first aspect of behavior combination is how to decide which behavior unit(s) should be activated in each situation.

[4] Strictly speaking, there is a third aspect to behavior coordination: the inter-behavior communication. This issue does not seem to have been given solutions specific to fuzzy logic, and we shall ignore it here.

The arbitration policy determines which behavior(s) should influence the operation of the robot at each moment, and thus ultimately determines the task actually performed by the robot. Early solutions, like the subsumption architecture proposed by Brooks [21], relied on a fixed arbitration policy, hard-wired into a network of suppression and inhibition links. This rigid organization contrasts with the requirement that an autonomous robot can be programmed to perform a variety of different tasks in a variety of different environments. (In fact, Brooks' robots were usually built to perform one single task.) Most architectures use dynamic arbitration schemas where the decision of which behavior to activate depends on both the current plan and the environmental contingencies; the plan is usually generated by higher-level reasoning modules [88, 36, 3, 73, 41, 80]. Note that many of these architectures do not allow for the concurrent execution of behaviors, and thus do not require a subsequent step of command fusion.

Both fixed and dynamic arbitration policies can be implemented using the mechanisms of fuzzy logic. The two main advantages in doing so are: (i) the ability to express partial and concurrent activations of behaviors; and (ii) the smooth transitions between behaviors.

The first appearance in the fuzzy literature of a fixed-priority scheme to arbitrate between rule-sets is probably due to Berenji *et al.* [11], who applied it to the classical cart-pole control problem. In their fuzzy controller, a higher priority rule-set takes care of balancing the pole on the cart; when the pole is approximately balanced, a lower priority rule-set is activated to bring the cart to the desired position. In the robotics domain, Pin and Watanabe [92] have proposed a fixed arbitration schema based on the suppression and inhibition mechanisms used in subsumption architectures, but generalized to operate on fuzzy behaviors. Interestingly, they also provide a method to compile a set of fuzzy behaviors, together with explicit priority relations, into VLSI hardware circuitry.

More flexible arbitration policies can be obtained using fuzzy meta-rules, or *context rules*, of the form

IF *context* THEN *behavior*,

meaning that *behavior* should be activated with a strength given by the truth value of *context*, a formula in fuzzy logic. When more then one behavior is activated, their outputs will have to be fused as discussed in the next subsection. Fuzzy context rules have been initially applied by Sugeno *et al.* [113] to switch between flight modes in a fuzzy-controlled unmanned helicopter; and by Saffiotti *et al.* [106] in the mobile robot FLAKEY.

It should be noted that using fuzzy meta-rules for expressing behavior arbitration policies is independent of the way in which individual behaviors are implemented – that is, these do not need to be fuzzy. For example, Ghanea-Hercock and Barnes [42] and Pan *et al.* [86] use fuzzy rules for arbitrating non-fuzzy behaviors. In most cases, however, fuzzy context rules have been used to arbitrate behaviors which are themselves implemented by fuzzy logic [44, 127, 117, 124, 75]. All these systems allow the concurrent execution of different behaviors, possibly with different degrees of activation, with the exception of [9] who use fuzzy rules to select a single behavior (control strategy) to be used depending on the situation.

4.2 Command fusion

The simplest way to fuse the commands from different behaviors is to use a *switching* scheme: the output from one behavior is selected for execution, and all the others are ignored; which output is selected depends on the arbitration strategy. The switching scheme is widely used in autonomous robotics [21, 88, 36, 80, 28]. Unfortunately, this simple scheme may be inadequate in situations where several criteria should be simultaneously taken into account. To see why, consider a robot that encounters an unexpected obstacle while following a path, and suppose that it has the option to go around the obstacle from the left or from the right. This choice may be indifferent to the obstacle avoidance behavior. However, from the point of view of the path-following behavior, one choice might be dramatically better than the other. In most implementations, the obstacle avoidance behavior alone could not know about this, and would take an arbitrary decision.

To overcome these limitations, several researchers allow the parallel execution of different behaviors, and use a weighted combination of the commands they issue. The most popular approaches of this type are based on a *vector summation* scheme: each command is represented by a force vector, and commands from different behaviors are combined by vector summation. The robot "responds" to the force resulting from the combination [56, 2, 62].

When the output of a behavior is represented by a fuzzy set, we can see the problem of command fusion as an instance of the problem of combining individual preferences. Following Ruspini [102], we can see each behavior-producing unit as an agent expressing preferences as to which command to apply; degrees of preference are represented by a possibility distribution (or fuzzy set) over the command space. We can then use fuzzy operators to combine the preferences of different behaviors into a collective preference, and finally choose a command from this collective preference. This view can be given a formal setting based on the interpretation of fuzzy logic as a logic of graded preferences [97, 8, 33].

According to this view, command fusion is decomposed into two steps: (i) preference combination, (ii) decision. Fuzzy logic offers many different operators to perform combination, and many defuzzification functions to perform decision. It is important to note that the decision taken from the collective preference can be different from the result of combining the decisions taken from the individual preferences. Intuitively, each individual decision issued by a behavior tells us which is *the* preferred command according to that behavior, but does not tell anything about the desirability of alternatives. Preferences contain more information, as they give a measure of desirability for each possible command: combining preferences thus uses more information than combining vectors, and can produce a different final decision. Figure 4 graphically illustrates this point in the case of two behaviors $B1$ and $B2$ both controlling the steering angle. This argument explains why fuzzy command fusion is fundamentally different from vector summation.

Several proposals that use fuzzy logic to perform command fusion have appeared in the literature. Curiously enough, the first such proposal was made, in a naive form, by two roboticists who were unaware of fuzzy logic but were frustrated

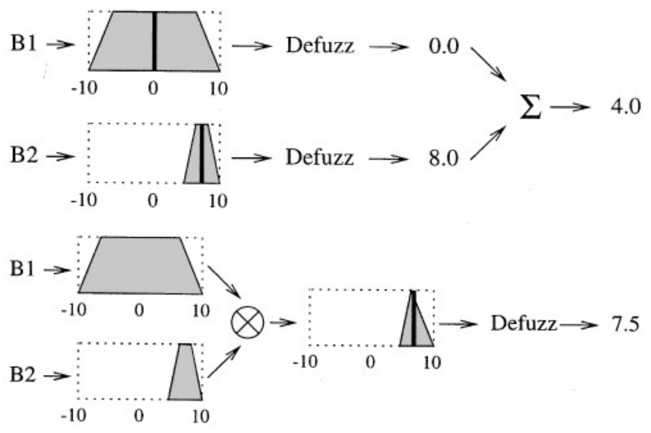

Fig. 4. Two approaches to command fusion. Top: combining individual decisions. Bottom: combining individual preferences. The final result may be different

by the pitfalls of the existing on-off arbitration schemas [98]. Their suggestion has been later restated in terms of fuzzy logic by Yen and Pfluger [131] and Baxter and Bumby [7]. Other authors have proposed trivialized forms of fuzzy command fusion. For instance, Goodridge *et al.* [44] use weighted singletons as fuzzy outputs and COG defuzzification; and Pin and Watanabe [92] use symmetric rectangles and COG. Both methods can be shown to be equivalent to a vector summation scheme. The majority of authors, however, have opted for a full-fledged form of fuzzy fusion: combination of arbitrary fuzzy sets followed by a defuzzification step [131, 7, 106, 127, 86, 117, 74, 124].

Most proposers of fuzzy command fusion have considered the problems that may arise from blindly applying defuzzification to the combined fuzzy set. In particular, when this set is not unimodal defuzzification may result in the selection of an "undesirable" control value, i.e., a value which lays in the gap between two peaks of the combined set, and has low membership in this set. In the case of robot control, this may mean that the robot, having the option of avoiding an obstacle from the right or from the left, decides to go straight. Yen and Pfluger [131, 90] and Baxter and Bumby [7] have addressed this problem by defining different defuzzification schemes (see [130] for a more elaborate approach). Saffiotti *et al.* [106] use ordinary COG, but insist that the fuzzy behavior designer should make sure that the output of the behavior is unimodal; Pin and Watanabe [93] require the designer to explicitly state "dominance" relations between all potentially conflicting behaviors. The rationale in the last two cases is that inconsistencies and ambiguities in a rule-set should be prevented by careful design rather than corrected by arbitrary mathematical manipulations.

Interpreting the generation of a non unimodal output as a sign of inconsistence in the arbitration rules allows a more analytical treatment of conflicts between behaviors. For example, Surmann [116] uses a fault detection approach; and Rausis *et al.* [96] analyze a rule-base from the perspective of validation of fuzzy data bases (see [34] for an extensive account of this type of validation problem). Goodridge *et al.* [43] delay detection and resolution of conflicts until execution time: the occurrence of an undesirable defuzzified value is permitted, but it is reported to the higher-level reasoning modules, which are responsible for analyzing the problem and breaking the tie.

4.3 Context-dependent blending

The most general form of behavior combination that we can realize using fuzzy logic is obtained by using both (i) fuzzy meta-rules to express an arbitration policy, and (ii) fuzzy combination to perform command fusion. This form of combination, which generalizes the approaches listed above, was initially suggested by Ruspini [100], and fully spelled out by Saffiotti *et al.* [106, 105] under the name of *context-dependent blending* of behaviors, or CDB.

CDB is a general mechanism that allows one to express different patterns of behavior combination. For instance, we can use a perceptual condition C to decide between two alternative behaviors; e.g., the following rules can be used to navigate to a target while reactively avoiding obstacles on the way:

IF obstacle-close THEN Avoid-Obstacle
IF $\neg$(obstacle-close) THEN Go-To-Target.

When the obstacle is only partially close, both behaviors are partially activated; thus, the commands issued by the Go-To-Target behavior can be taken into account during obstacle avoidance maneuvers. As we have seen above, this is an important feature for reactive navigation. We can also sequence two behaviors B_1 and B_2 aimed at two goals G_1 and G_2 by using context rules of the form

IF G_1 not achieved THEN B_1
IF G_1 achieved THEN B_2.

This is the way to prioritize fuzzy rule-sets used in Berenji's cart-pole system above. Note that blending is this example is goal-driven, while it was event-driven in the previous one. These two types of blending can be combined into an arbitrarily complex set of context rules to represent a full *plan* for action, telling which behavior(s) should be used in each situation. We shall come back on this issue in Section 6.

CDB can be implemented in a hierarchical fuzzy controller as shown in Figure 5. A few observations should be made on this architecture. First, as noted above, it is essential that the defuzzification step be performed after the combination. Second, although in Figure 5 all the context-rules are grouped in one module, the same effect can be obtained by including each context-rule inside the corresponding behavior; this solution would be more amenable to a distributed implementation. Third, CDB can be iterated: we can use the structure in Figure 5 to implement individual behaviors, and combine several such (complex) behaviors using a second layer of context-rules; and so on. (Defuzzification should still be the last step.) Hierarchical fuzzy controllers have already been proposed in the literature [95, 129, 99]; they have been applied to building complex robot behaviors by CDB in [24, 126, 124]. Importantly, the hierarchical organization answers the criticism of non-scalability often moved to fuzzy control techniques [12].

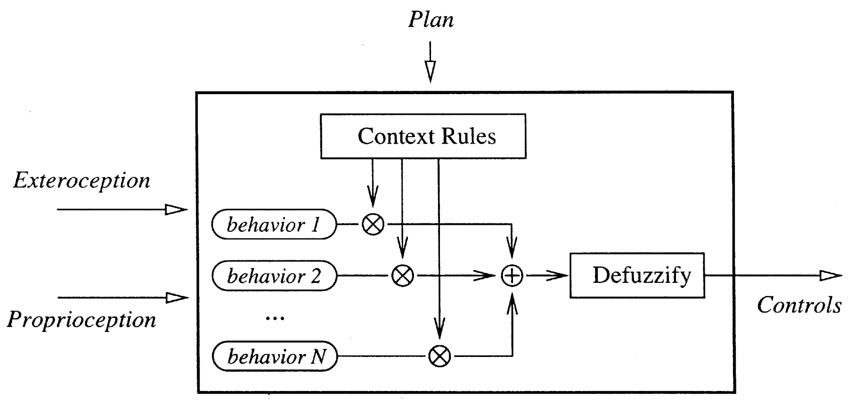

Fig. 5. Context-depending blending (CDB) of behaviors can be implemented in a hierarchical fuzzy controller. (Adapted from [106].)

Following its implementation on FLAKEY, CDB has been used by several researchers in autonomous robotics. Voudouris [126, 127] organizes fuzzy behaviors in a hierarchical structure based on fuzzy decision trees; behaviors in the leaves are blended by CDB. Surmann [116, 117] has implemented CDB on the robot MORIA using recurrent fuzzy systems. Bonarini and Basso [19] use an evolutionary algorithm to learn coordination policies expressed by fuzzy rules and executed by CDB (see also [123]). Hasemann [48] uses CDB to coordinate several behaviors to grasp and handle paper rolls. Arrúe *et al.* [4] use CDB to blend reactive and goal-directed behavior during outdoor navigation. Finally, Michaud [74] extends CDB by considering both a measure of desirability and a measure of undesirability for each behavior. Distinguishing between the desirability and the undesirability of control actions may be extremely useful for handling potentially conflicting behaviors (see [90] for an early use of this concept).

Similar forms of CDB have also been independently developed by other researchers. For example, Goodridge *et al.* [44,43] have implemented a simplified form of CDB on the robot MARGE. And Tunstel [124] has defined a form of iterated CDB called "behavior modulation," which has been tested on the robot LOBOT [122].

4.4 Discussion

The problem of how to coordinate the activity of a set of behaviors remains the Achilles' heel of behavior-based robotics. This problem has two facets: how to decide which behavior(s) should be activated, and how to fuse the output of concurrent, possibly conflicting behaviors. Fuzzy logic offers mechanisms for both tasks.

Fuzzy context rules provide a flexible means to encode behavior arbitration strategies. Like fuzzy control rules, context rules allow us to write complex strategies in a modular way using a logical format. The fact that the same format is used for the control rules and the arbitration rules makes it easy to write increasingly complex behaviors in a hierarchical fashion. Fuzzy command fusion can then be used to combine recommendations from concurrent behaviors. The resulting scheme, called CDB, is strictly more general than other coordination schemes commonly used in robotics, including behavior switching and (weighted) vector summation. CDB has been used, in various flavors, in many autonomous robots.

While fuzzy logic gives us a valuable tool for writing coordination strategies, it does not give a solution to the general problem of behavior coordination. For example, we still do not know how to discriminate a situation where different commands proposed by different behaviors should be averaged, from one where they should be regarded as a conflict to be resolved in some way. These problems are inherent to any form of local combination, and can be seen as instances of the general problem of relating local computation (or action) to global results (or goals). As such, these problems can only be solved by a careful integration between local and global reasoning. Understanding this integration is currently a major challenge of autonomous robotics; we shall come back to this issue in Section 6.

5 Perception and modeling

Both the high-level and the low-level processes indicated in Figure 2 need to consider perceptual information in order to reduce uncertainty. In this section, we turn our attention to the way in which perceptual information can be represented and processed.

The issue of perception and modeling is amazingly vast, and its different facets constitute the subject of many research fields, including sensor processing, computer vision, object and scene interpretation, knowledge representation, and belief revision – to mention but a few. In this note, we shall do no more than touch on a few issues that are particularly relevant to the problem of autonomous robot navigation, and for which solutions based on fuzzy logic have been proposed. We structure our discussion in two parts: (i) how to use sensor data to form a local view of the current state of the environment, as needed by the execution layer; and (ii) how to use sensor data and prior knowledge to build a global model of the environment, or *map*, as needed by the planning layer.

5.1 Local perception

At the execution level, both sensor-based behaviors and context rules may need environment information coming from the sensors. In many cases, the data used are rather simple, typically consisting in distance measurements obtained by infrared or sonar sensors. For instance, a typical fuzzy control rule for reactive obstacle avoidance might look like

IF (right-distance IS close) AND (left-distance IS far)
THEN (turn IS sharp-left),

where "right-distance" and "left-distance" directly refer to sensor measurements. This approach is effective if the dimensionality of the input space is small. In autonomous robots, however, the opposite is often the case, as robots tend to include many heterogeneous and redundant sensors. For example, MARGE [44] has two video cameras, 13 narrow angle sonars, 3 wide angle sonars, tactile sensors, and wheel encoders; and FLAKEY [24] takes range information from a stereo vision system, from a laser range-finder, and from a buffer recording the last 50 measurements taken by the sonars. Unfortunately, the number of fuzzy rules is $O(K^n)$ in the general case, where K is the number of fuzzy predicates and n is the number of input variables. Clearly, the complexity of the rule-set becomes unmanageable, from both a design and a computational viewpoints, when n is large.

In general, we can use two (non exclusive) strategies to cope with this complexity, as shown in Figure 6. The first strategy is to decompose the control problem into small behaviors, each considering only a small portion of the input space. We have already become familiar with this strategy in the previous sections. (See [43] for a discussion of behavior-based approaches from the perspective of reducing the complexity of the input space.) The second solution is to introduce a limited number of intermediate variables, meant to classify the different perceptual situations that are relevant to the robot's behavior. The input from the sensors is fed to a situation classification module that sets the value of these intermediate variables. A typical intermediate variable can be "facing obstacle," or "wheels skidding." This approach is sometimes referred to as "bipartite" in the literature.

The qualitative information provided by intermediate variables can be used in fuzzy control rules. For instance, the fuzzy rules for the obstacle avoidance behavior used in Flakey [106] look like

IF obstacle-close-right AND NOT(obstacle-left)
THEN turn sharp-left

where "obstacle-close-right" and "obstacle-left" are two fuzzy variables whose value is computed from the last 50 sonar readings, and possibly from data from the camera. Pre-processing the input to a fuzzy behavior is also proposed by Pin and Watanabe [92], who collapse the readings from 24 sonars into three variables "left", "center" and "right"; and by Braunstingl *et al.* [20], who summarize the readings from 12 sonar sensors into one "general perception" vector.

The sort of pre-processing discussed in this section is by no means a new idea, nor it is peculiar to autonomous robotics or to fuzzy logic. Control engineers are used to decompose their problems into two components: the state estimation problem, inferring the value of the interesting variables of the controlled system from the observations; and the input regulation problem, generating control actions to bring the system to a desired state. What we have called "situation classifier" could also be considered as a state estimator. However, the perspective we take here is not to recover the value of a system variable, but to classify the state of the system among several possible (fuzzy) situations.

The traditional approach to writing a state estimator is based on the availability of mathematical models of the observed system and of the sensors. When linear models cannot be easily obtained, while heuristic knowledge about the system can, fuzzy rules may give us a convenient alternative. Examples of this are provided by Maeda *et al.* [69], who use fuzzy rules to classify the shape of a road from image data; and by Altrock *et al.* [125], who use fuzzy rules to diagnose situations in which their model car is sliding or skidding. The latter authors note that the two-level partition has made a difficult problem (the dynamic stabilization of a model car at high speed) manageable. Some authors have also proposed the use of learning techniques for situation classification (e.g., [50]).

The situation classification module can be used to perform other types of data pre-processing. For example, Saffiotti *et al.* [105] use this module to store "object descriptors" whose role is to integrate sensor and prior knowledge about relevant objects in the environment; Arrúe *et al.* [4] use a similar technique to overcome the problems due to temporary loss of perceptual information. Another way in which the situation classification module can be used is by combining the data coming from different but redundant sensors. This is an instance of the general problem of *data fusion*:

Data fusion consists in combining several pieces of information issued from different sources about the same phenomenon in order to take a better decision on this phenomenon. ([17])

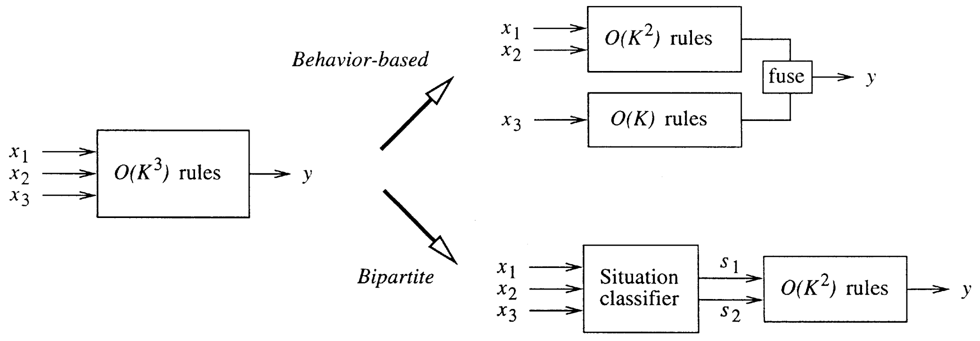

Fig. 6. Two strategies for reducing the dimensionality of the input space in a fuzzy controller

An important feature of data fusion is the possibility to exploit redundancies to improve the reliability of the data. This is one of the motivations of the work reported by Ollero *et al.* [83], who combine the data from different sonars that cover the same area (left, front or right) into one fuzzy variable, which is interpreted as a measurement from a virtual sensor looking at the area. We direct the reader to [16,17] for more on the use of fuzzy logic for data fusion.

Finally, we note that the output produced by the pre-processing module may be in general a fuzzy variable, while the input used by fuzzy control rules is, in most cases, a crisp number. (Some authors have proposed the use of "fuzzy sensors," which also produce fuzzy sets as output [37].) We then have two alternatives for using these data: we can extend the computation of the fuzzy rules to accept a fuzzy antecedent by using some form of generalized modus ponens; or we can convert (defuzzify) the fuzzy variable to a crisp value to be used in the antecedents. The first solution may entail a significant increase in the computational complexity, while the second one may incur in a loss of information. Demirli and Türkşen [26] and Saffiotti [104] discuss these options in the context of robot navigation.

5.2 Global modeling

Low-level execution needs local sensor information in order to react to the relevant events in the environment. Higher-level reasoning processes need a more global representation of the environment in order to generate abstract plans of action. At this level, the main aim of sensor processing is to obtain an accurate and complete picture of the environment. To do this, we need to synchronously fuse the data from different sensors, and to diachronically accumulate data over time. The result of this process is usually to build a *map* of the environment.

Various map representations have been used in the robotics literature, whose relative adequacy depends on the task, and on the characteristics of the robot and of the environment. In this subsection, we focus on two popular types of representations: occupancy-based representations, and feature-based representations. In the first type, the basic objects are small areas (or "cells") tesselating the space, and the basic property is the fact that a cell is occupied or not. In the second type, objects are environmental features of some type, and properties include their spatial location.

An important aspect of a map representation is the way in which it can account for the uncertainty in its properties. In most cases, uncertainty is represented by probabilistic means. For example, Moravec and Elfes' occupancy grids [76] associate to each cell a probability distribution over the set $\{\mathsf{occupied}, \mathsf{free}\}$. And Smith and colleagues' stochastic maps [112] associate with each feature a probability distribution about its position and orientation in a Cartesian frame. Some researchers, however, have proposed fuzzy-set based representations of map uncertainty.

Poloni *et al.* [94, 84] and Tunstel [121] have proposed variants of Moravec and Elfes' occupancy grids in which each cell c is associated with a possibility distribution π_c over the set $\{\mathsf{occupied}, \mathsf{free}\}$ (or, equivalently, the sets of free and occupied cells are fuzzy sets). Several reasons justify the use of fuzzy sets to represent occupancy information. First, the stochastic method behind Moravec and Elfes' grids relies on the assumption that a large number of well distributed data is available, which is rarely the case during robot navigation. Second, to correctly apply probabilistic techniques, we must pay a strong attention to the independence assumptions we make: as shown by Berler and Shimony [13], this may result in a great complexity, both conceptual and computational. Third, the fuzzy approach only needs a qualitative model of the sensors, as opposed to the stochastic model needed by probability-based techniques. Finally, a possibility distribution can distinguish between the state of a cell c being uncertain (e.g., $\pi_c(\mathsf{occupied}) = \pi_c(\mathsf{free}) = 0.5$), or being unknown ($\pi_c(\mathsf{occupied}) = \pi_c(\mathsf{free}) = 1$) – that is, c has not been explored. This information can be used to plan further exploration [85].

Feature-based representations of maps can also profit from the ability of fuzzy logic to represent and reason with weak knowledge, and to distinguish between different facets of uncertainty [59, 60]. For example, we may wish to distinguish between the vagueness or inaccuracy in the position of the feature, and the uncertainty in its very existence – e.g., the map may be wrong, the feature may have been removed from the environment, or its existence may have been inferred from a spurious sensor reading [64]. Recall the block location example used in the Introduction. Figure 7 shows how we represent different types of weak information about this location by using a fuzzy subset $B(x)$ of the set X of possible positions (taken here in one dimension for graphical clarity). In (a) the position of B is crisp and certain; item (b) tells us that B is located at approximately 5 (vagueness); in (c), B can possibly be located anywhere between 5 and 10 (imprecision); in (d), it can be either at 5 or at 10 (ambiguity); in (e) we are told that B is at 5, but the information may be invalid, so we put a small "bias" of possibility that it be located just anywhere (unreliability); finally, (f) combines vagueness, ambiguity and unreliability. Note that the case of total ignorance is represented by $B(x) = 1$ for all $x \in X$.

Fuzzy sets have been used to represent approximate spatial information in a general setting by Dutta [35]. They have later been used by several authors to represent imprecise spatial properties of features in the context of robot navigation [58, 1, 40, 107]. Some of these proposals are based on low-level, uninterpreted features, like segments or polyhedra; others are based on high-level, semantic features, like doors, walls, and corners. For instance, Kim *et al.* [58] consider sets of geometric primitives (points and lines) whose parameters are given by fuzzy numbers. Gasós and Martín [40] and Amat *et al.* [1] take maps to be sets of fuzzy segments, i.e., segments whose width and length are trapezoidal fuzzy sets. The last proposal also uses fuzzy operators to combine partial maps built by several, cooperating robots; the cooperation aspect is further developed in [67]. Finally, Saffiotti and Wesley [107] represent maps by sets of high level features, each one associated with a fuzzy position in a global Cartesian frame.

Whether a map representation based on low-level features is better than one based on high-level features depends on the type of environment and on the available sensors. The use of high-level features may make the map more robust, as these features are more stable over time. However, high-level

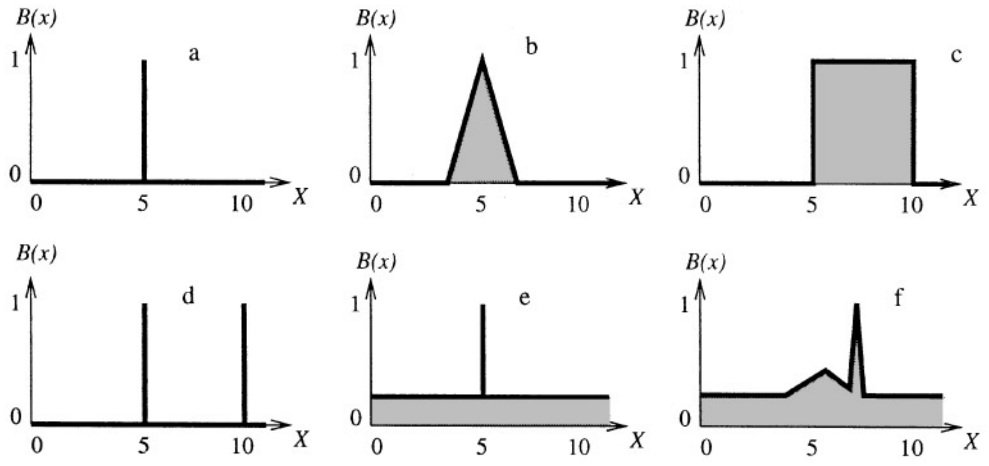

Fig. 7. Representing different types of positional uncertainty for an object B by fuzzy sets: (a) crisp; (b) vague; (c) imprecise; (d) ambiguous; (e) unreliable; (f) combined. (Adapted from [107].)

features may be difficult to extract, and some environments (especially outdoor) may contain only a small number of them.

5.3 Discussion

The choice of an adequate representation for uncertainty is crucial to perceptual interpretation and environment modeling. Most of the approaches in the robotic domain are based on a probabilistic representation of uncertainty. This representation is adequate when two conditions hold: (i) the underlying uncertainty can be given a probabilistic interpretation; and (ii) the data required by probabilistic techniques is available. Both conditions may be violated in the case of autonomous robots, and techniques based fuzzy set theory may then offer a valuable alternative.

Fuzzy sets include crisp sets as a special case, hence they can represent precise and complete information when available. However, they can also represent "weak" types of information, including imprecise, vague or unreliable information, at the level of detail which is available and without requiring a precise quantification of the uncertainty. Paraphrasing a motto from the domain of knowledge representation ([65], p. 53), we might say that

> The expressive power of fuzzy logic determines not so much what can be said, but what can be left unsaid.

In robotics, this expressive power can be used to perform sensor interpretation based on a qualitative, approximate model of the sensors, as opposed to the stochastic models required by probabilistic techniques. We have seen applications of this to the detection and classification of relevant events, and to the building of approximate maps.

These advantages come to a price: perhaps due to this weak and qualitative nature, the foundations underlying the representations and operations of fuzzy logic are not fully understood yet. While we have well defined operational semantics for a probability value, the same cannot be said of a membership degree or a possibility value. Perhaps more annoyingly for practical applications, we still have little guidance in the choice of the operators to be used for information aggregation and for reasoning – in fact, most authors choose their operators on the ground of intuitive considerations or empirical testing. Many researchers share the feeling that this is a natural situation for a field still in its puberty; progress towards a stronger foundational theory of fuzzy logic and its semantics is a major concern in the field (e.g., [101, 52, 14]).

6 Layer integration

The last issue we touch on is the problem of how to link, or *integrate*, the different levels of representation and reasoning that must be present in an autonomous agent. The study of this link has been attracting attention from philosophers and scientists for centuries; in robotics, the integration problem mainly consists in connecting the low-level and high-level layers in the architecture in Figure 2. More precisely, we distinguish two facets of this problem. The first one, that we call the *registration* problem, deals with representations. It consists in maintaining the right correspondence between the local representation of the environment used at the lower level, and the global representation used at the higher level. The second aspect, that we call the *task integration* problem, deals with tasks. It consists in maintaining the right correspondence between the execution of behaviors at the lower level, and the achievement of the goals considered by the higher level.

Curiously enough, relatively few works have been reported in the literature that use fuzzy logic to address the integration problem. Yet, layer integration, in its multiple facets, is at the hart of autonomous robotic applications; in what follows, we show that fuzzy logic may provide convenient mechanisms to address some of the problems posed by layer integration.

6.1 Registration

Perception happens locally, in the egocentric frame of reference of the robot. In order to ensure a correspondence between the local representations of the environment built by the perceptual processes, and the global representation contained in a map, the robot must be able to estimate its own position with respect to this map. This is often referred to as the *self-localization* problem. Self-localization can be performed by using the information from the odometric sensors to update the robot's position as this moves. Unfortunately, odometry has cumulative errors, and the robot's odometric estimate of its position can diverge from reality without bounds. Landmark-based techniques are commonly used to correct this problem: the robot compares the observed position of perceptual landmarks with their expected position, given the information in the map and the current location estimate, and uses the result of the comparison to correct this estimate. Note that these techniques require that prior information about the position of the landmarks is available.

Most existing approaches to landmark-based self-localization are based on a probabilistic representation of spatial uncertainty, and use some form of Kalman filter [55, 112, 77, 64] to update the robot's position estimate. These approaches can be very effective, provided that: (i) the underlying uncertainty can be given a probabilistic interpretation; (ii) the initial estimate is good enough; and (iii) the required data is available. In particular, the last requirement means that (iii-a) we have an accurate dynamic model of the robot; (iii-b) we have an accurate stochastic model of the sensors; and (iii-c) these systems do not change in unpredictable ways with time. These conditions are pretty demanding, and they may easily be violated in the case of autonomous robotics. When this happens, fuzzy logic may offer valuable alternatives which require less demanding assumptions: for example, fuzzy-based localization methods typically need only qualitative models of the system and of the sensors.

Some of the proposers of the fuzzy map representations discussed in the last section have also given a corresponding fuzzy self-localization algorithm. Gasós and Martín [40] perform self-localization by comparing the partial fuzzy map built by the robot during navigation with a pre-existing global map, usually built in a preceding exploration phase. Saffiotti and Wesley [107] take a fusion-oriented perspective: each perceived feature is seen as a potential source of information about the robot's position, based on the comparison between the feature's observed position and the map; the pieces of information obtained from different features are then combined, by fuzzy intersection, into a new estimate of the robot's position. Both algorithms have shown good performance on real robots even in situations of high uncertainty. For example, [107] reports an experiment where the robot manages to self-localize on a given map starting from a situation of total positional ignorance – a difficult task for probability-based methods.

Other authors have proposed self-localization techniques based on fuzzy triangulation with perceptual landmarks. Demirli and Türkşen [27] propose a technique based on a fuzzy model of the sonar sensors, which is derived from experimental data relative to a specific surface type (a dry wall). Bison *et al.* [15] use possibilistic logic [32] for representing and combining data from the sensors. The latter approach is peculiar in its using a syntactical calculus to perform data fusion; the use of a logical formalism allows the authors to equip this calculus with a clear, similarity-based semantics.

6.2 Task integration

The aim of planning is to connect action, locally controlled by the execution process, to the overall goals of the agent, globally analyzed by the planning process. The heart of this connection is the *plan*: the connection will be successful if the execution layer can make good use of the plan. As noted by Hanks and Firby [47], an essential aspect of the integration problem is the definition of a *shared plan representation*, that is, a representation that can be reasoned about and generated by the high-level processes, and can be effectively used by the low-level processes to control execution.

The definition of a shared plan representation is complicated by the fact that the processes and the representations used at the different layers are essentially different. For example, high-level reasoning processes typically manipulate symbolic representations that are based on an abstract model or reality, while low-level processes manipulate numerical data that are grounded in the physical world through the robot's sensors and effectors. These two levels call for different types of tools, and pose different requirements on a plan representation. For example, many path planners generate paths with sharp turns, which are not adequate for execution; and many robot execution systems are based on plans written in elaborated reactive languages (e.g., [47, 31]), which are too complex to be generated by current planners.

Fuzzy logic has an extremely attractive feature in this respect: its ability to represent both the symbolical and the numerical aspects of reasoning. Fuzzy logic can be embedded in a full logical formalism, endowed with a symbolic reasoning mechanism; but it is also capable of representing and processing numerical data. (This double nature is the key to the success of fuzzy control: the designer gives a symbolic description of a control policy, and fuzzy logic provides an interpretation of this description as a non-linear control mapping.) Hence, fuzzy logic can be used as a tool to represent plans that can be shared between the higher and lower level.

To see this, recall that fuzzy rules can be used to express complex arbitration strategies. These strategies can be seen as specifications of plans of action for the robot, in the form of *situation* → *action* rules. For example, the set of context rules listed in Figure 8 constitutes a plan to reach room 5 as follows: when in corridor-1, the robot follows it, until it reaches corridor-2; when the robot is in corridor-2, it follows it until it gets close to door-5; and so on. If executed in an environment with the right topological relations, this plan will get the robot into room-5 from anywhere in the corridors 1 and 2. The plan also incorporates an avoidance behavior to go around possible obstacles.

Plans expressed by fuzzy context rules satisfy the requirements for a shared plan representation. On the one hand, these plans can be executed using the CDB mechanism, for example by a hierarchical fuzzy controller like the one in Figure 5. On the other hand, these plans have a simple and logical format, and thus lend themselves to be automatically generated by symbolic processes. The last property has been demonstrated by Saffiotti *et al.* [105], who use a standard goal regression planner, together with a topological map of the environment, to generate navigation plans for the robot Flakey. These plans are constituted by sets of fuzzy arbitration rules as the one shown in Figure 8. A similar approach has been implemented by Surmann and colleagues on the robot Moria [117].

There is a second advantage resulting from the use of a uniform formalism across different layers: the behavior of the resulting, multilevel system can be analyzed as a whole. For example, Saffiotti *et al.* [105] prove a few composition theorems that relate the CDB composition of two different fuzzy behaviors, which individually achieve two different goals, to the achievement of the corresponding composite goal. These theorems effectively link the goal decomposition performed by a planner to the behavior composition performed by CDB. A different analysis is offered by Wang [128]. Wang considers a three-level architecture where the bottom level is designed using standard control theory techniques, while fuzzy rules are used at the two upper levels. By interpreting the fuzzy rules as a non-linear mapping, Wang can mathematically analyze the system as a whole. However, Wang does not consider the problem of automatically generating the rules used in the upper layers.

6.3 Discussion

Probably the most peculiar feature of fuzzy logic is its intrinsic ability to integrate numeric ("fuzzy") and symbolic ("logic") aspects of reasoning. This double nature suggests fuzzy logic as a natural tool to address the problem of integration between high-level layers, which typically perform symbolic computations, and low-level layers, which typically perform numeric manipulations. The integration has two main advantages: (i) it allows us to use symbolic planners to automatically generate complex behavior coordination strategies that achieve a given goal; and (ii) it allows us to perform a mathematical analysis of the overall behavior resulting from coordinating several fuzzy behaviors. This potential of fuzzy logic seems to have been scarcely noticed in the robotic literature to this date. Given the key role played by integration issues in autonomous robotics, we speculate that this feature will be essential to the future exploitation on fuzzy logic in this domain.

7 Concluding remarks

Fuzzy logic has features that are particularly attractive in light of the problems posed by autonomous robot navigation. Fuzzy logic allows us to model different types of uncertainty and imprecision; to build robust controllers starting from heuristic and qualitative models; and integrate symbolic reasoning and numeric computation in a natural framework. In these pages, we have illustrated these points by examples taken from the literature, and have outlined some pros and cons of solutions based on fuzzy logic.

Fuzzy logic is not the philosopher's stone, and many of the problems in autonomous robotics may not benefit from it. First, there are cases in which fuzzy logic should not be used. For example, when the assumptions required by more classical techniques (e.g., classical control theory) are met, we may prefer to use these techniques due to the large amount of formal tools available. And when the type of uncertainty we must deal with is inherently probabilistic, e.g. it originates in a stochastic process, and we have a good estimate of the probability distributions, then probability theory should be used. Second, even in those cases in which the weaker assumptions required by fuzzy logic would make it a better choice, its application may be problematic due to a few as yet unresolved problems. We have already noted a couple of these in the above discussions: the relative lack of formal tools for the design and analysis of fuzzy controllers; and the still unsatisfactory semantics for the numbers and the operators used. Current research in the field endavours to fill both gaps.

Needless to say, there are far more many things in the realm of autonomous robot navigation than we have touched on in this short overview. Our choice of issues was biased towards those problems for which the usefulness of fuzzy logic seems to be most evident. Two omissions are particularly worth mentioning. First, we did not consider learning. Learning techniques have been extensively used in the field of autonomous robotics (see [29, 38] for representative samples). As noted in Section 3.4, learning has also been used to infer the structure or the parameters of fuzzy behaviors. Second, we did not mention planning under uncertainty. Several planning techniques that explicitly take uncertainty into account have been proposed in the robotics literature, including some based on fuzzy logic (e.g., [85, 25]).

We close this survey by a note from the robotic folklore. Robots based on fuzzy behaviors have demonstrated excellent performance in the occasion of public robotics events. The team of Pioneer robots from SRI International won the "Call a Meeting" event at the 1996 AAAI robotic competition [45]; Moria, from the German GMD, was given the Intelligence Award at the 1995 Fuzz'IEEE robot competition [117]; Marge, from NCSU, won the office rearrangement event at the 1993 AAAI competition [82]; and Flakey, from SRI International, placed second at the first AAAI competition in 1992 [24, 108].[5] The use of fuzzy logic often resulted in extremely smooth motion and reliable reactivity, as it is best illustrated by a judge's comment about Flakey: "Only robot I felt I could sit or lie down in front of." (What he actually did!)

IF obstacle	THEN Avoid
IF ($\neg$obstacle $\wedge$ in(Corr-1) $\wedge$ $\neg$in(Corr-2))	THEN Follow(Corr-1)
IF ($\neg$obstacle $\wedge$ in(Corr-2) $\wedge$ $\neg$near(Door-5))	THEN Follow(Corr-2)
IF ($\neg$obstacle $\wedge$ near(Door-5) $\wedge$ $\neg$in(Room-5))	THEN Cross(Door-5)
IF in(Room-5)	THEN Still

Fig. 8. A set of context rules can be used to represent a full plan. When executed in the right environment, this plan will bring the robot into room 5 starting anywhere in corridor 1 or 2

[5] Differently from Flakey, the winning entry, Carmel, was a highly engineered robot who required some modification of the environment. These two robots illustrate the fundamental tradeoff in robotics between using sophisticated engineering or sophisticated software [24].

References

1. **Amat, J.; López de Mantaras, R.; Sierra, C.:** Cooperative autonomous low-cost robots for exploring unknown environments. In *Procs. of the 4th Int. Symp. on Experimental Robotics* (*ISER'95*), pages 28–33, Stanford, CA, 1995
2. **Arkin, R.C.:** Motor schema based navigation for a mobile robot. In *Procs. of the IEEE Int. Conf. on Robotics and Automation*, pages 264–271, 1987
3. **Arkin, R.C.:** Integrating behavioral, perceptual and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, 6:105–122, 1990
4. **Arrúe, B.C.; Cuesta, F.; Braunstingl, R.; Ollero, A.:** Fuzzy behaviors combination to control a non-holonomic mobile robot using virtual perception memory. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 1239–1244, Barcelona, SP, 1997. IEEE Press
5. **Balzarotti, M.; Ulivi, G.:** The fuzzy horizon obstacle avoidance method for mobile robots. In *Procs. of the World Automation Congress* (*WAC*), volume 3, pages 51–57, Montpellier, FR, 1996. TSI Press
6. **Barfoot, C.W.:** A new approach to behavioral control: fuzzy behaviors. In *Procs. of the 1st On-Line Workshop on Soft Computing* (*WSC1*), *special session on fuzzy logic in autonomous robotics*, 1996. On-line at http://iridia.ulb.ac.be/FLAR
7. **Baxter, J.W.; Bumby, J.R.:** Fuzzy logic guidance and obstacle avoidance algorithms for autonomous vehicle control. In *Procs. of the Int. Workshop on Intell. Autonomous Vehicles*, pages 41–52, Southampton, UK, 1993
8. **Bellman, R.E.; Zadeh, L.A.:** Decision making in a fuzzy environment. *Management Science*, 17:141–164, 1970
9. **Benreguieg, M.; Maaref, H.; Barret, C.:** Fuzzy helps to control a motion of a mobile robot in a partially-known environment. In *Procs. of the European Congress on Fuzzy and Intelligent Technologies* (*EUFIT*), pages 905–909, Aachen, DE, 1995
10. **Beom, H.R.; Cho, H.S.:** A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning. *IEEE Trans. on Systems, Man, and Cybernetics*, 25(3):464–477, 1995
11. **Berenji, H.; Chen, Y-Y.; Lee, C-C.; Jang, J-S.; Murugesan, S.:** A hierarchical approach to designing approximate reasoning-based controllers for dynamic physical systems. In *Procs. of the 6th Conf. on Uncertainty in Artificial Intelligence*, pages 362–369, Cambridge, MA, 1990
12. **Berenji, H.R.:** The unique strength of fuzzy logic control. *IEEE Expert*, page 9, August 1994. Response to Elkan's "The paradoxical success of fuzzy logic", same issue
13. **Berler, A.; Shimony, S.E.:** Bayes networks for sensor fusion in occupancy grids. In *Procs. of the Conf. on Uncertainty in Artif. Intell.*, 1997
14. **Bilgiç, T.; Türkşen, I.B.:** Measurement of membership functions: theoretical and empirical work. In H. Prade D. Dubois and H.J. Zimmermann, editors, *International Handbook of Fuzzy Sets and Possibility Theory*. Kluwer Academic, Norwell, MA, 1998. Forthcoming
15. **Bison, P.; Chemello, G.; Sossai, C.; Trainito, G.:** A syntactical approach to data fusion. In *Procs. of the European Conf. on Symbolic and Quantitative Approaches to Reasoning and Uncertainty* (*ECSQARU*) , number 1244 in LNAI, pages 58–70, Bad Honnef, DE, 1997. Springer-Verlag
16. **Bloch, I.:** Information combination operators for data fusion: A comparative review with classification. *IEEE Trans. on Systems, Man, and Cybernetics*, A-26(1):52–67, 1996
17. **Bloch, I.; Maître, H.:** Fusion of image information under imprecision. In B. Bouchon-Meunier, editor, *Aggregation of Evidence under Fuzziness*, Series Studies in Fuzziness. Physica Verlag, Springer, 1998. Forthcoming
18. **Blöchl, B.:** Fuzzy control in real-time for vision guided autonomous mobile robots. In *Procs. of the 8th Austrian Conf. on Fuzzy Logic in AI* (*FLAI'93*), *LNAI* 695, pages 114–125. Springer-Verlag, 1993
19. **Bonarini, A.; Basso, F.:** Learning to to coordinate fuzzy behaviors for autonomous agents. *Int. J. of Approximate Reasoning*, 17(4):409–432, 1997
20. **Braunstingl, R.; Sanz, P.; Ezkerra, J.M.:** Fuzzy logic wall following of a mobile robot based on the concept of general perception. In *Procs. of the 7th Int. Conf. on Advanced Robotics* (*ICAR'95*), pages 367–376, Sant Feliu de Guixols, SP, 1995
21. **Brooks, R.A.:** A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, 1986
22. **Brooks, R.A.:** Intelligence without reason. In *Procs. of the Int. Joint Conf. on Artificial Intelligence*, pages 569–595, San Mateo, CA, 1991. Morgan Kaufmann
23. **Castellano, G.; Attolico, G.; Stella, E.; Distante, A.:** Automatic generation of rules for a fuzzy robotic controller. In *Procs. of the Int. Conf. on Intelligent Robots and Systems* (*IROS*), Osaka, JP, 1996
24. **Congdon, C.; Huber, M.; Kortenkamp, D.; Konolige, K.; Myers, K.; Ruspini, E.H.; Saffiotti, A.:** CARMEL vs. Flakey: A comparison of two winners. *AI Magazine*, 14(1):49–57, Spring 1993
25. **Da Costa Pereira, C.; Garcia, F.; Lang, J.; Martin-Clouaire, R.:** Planning with graded nondeterministic actions: a possibilistic approach. *Int. J. of Intelligent Systems*, 1997. Forthcoming
26. **Demirli, K.; Türkşen, I.B.:** Mobile robot navigation with generalized modus ponens type fuzzy reasoning. In *Procs. of the IEEE Conf. on System, Man and Cybernetics*, pages 3724–3729, 1995
27. **Demirli, K.; Türkşen, I.B.:** Sonar based mobile robot localization by using fuzzy triangulation. *IEEE Trans. on Robotics and Automation*, 1997. Submitted
28. **Dorigo, M.; Colombetti, M.:** *Robot shaping: an experiment in behavior engineering*. MIT Press / Bradford Books, 1997
29. **Dorigo, M.:** (Editor) Special issue on: Learning autonomous robots. *IEEE Trans. on Systems, Man, and Cybernetics*, B-26(3), 1996
30. **Driankov, D.; Hellendoorn, H.; Reinfrank, M.:** *An introduction to fuzzy control*. Springer-Verlag, Berlin, DE, 1993
31. **Drummond, M.:** Situated control rules. In *Procs. of the 1st Int. Conf. on Principles of Knowledge Representation and Reasoning*, pages 103–113, 1989
32. **Dubois, D.; Lang, J.; Prade, H.:** Fuzzy sets in approximate reasoning, part 2: logical approaches. *Fuzzy Sets and Systems*, 40:203–244, 1991
33. **Dubois, D.; Prade, H.:** Criteria aggregation and ranking of alternatives in the framework of fuzzy set theory. *TIMS Studies in the Management Sciences*, 20:209–240, 1984
34. **Dubois, D.; Prade, H.; Ughetto, L.:** Checking the coherence and redundancy of fuzzy knowledge bases. *IEEE Trans. on Fuzzy Systems*, 5(5):398–417, 1997
35. **Dutta, S.:** Approximate spatial reasoning: integrating qualitative and quantitative constraints. *Int. J. of Approximate Reasoning*, 5(3):307–330, 1991
36. **Firby, J.R.:** An investigation into reactive planning in complex domains. In *Procs. of the AAAI Conf.*, pages 202–206, 1987
37. **Foulloy, L.; Galichet, S.:** Fuzzy sensors for fuzzy control. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2(1):55–66, 1994
38. **Franklin, J.A.; Mitchell, T.M.; Thrun, S.:** (Editors). Special issue on: Robot learning. *Machine Learning*, 23(2-3), 1996
39. **Garnier, Ph.:** *Contrôle d'exécution réactif de mouvements de véhicules en environnement dynamique structuré*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, FR, Dec 1995
40. **Gasós, J.; Martín, A.:** Mobile robot localization using fuzzy maps. In T. Martin and A. Ralescu, editors, *Fuzzy Logic in A.I.-Procs. of the IJCAI '95 Workshop. Number* 1188 *in Lecture Notes in AI, pages* 207–224. *Springer-Verlag*, 1997
41. **Gat, E.:** Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Procs. of the AAAI Conf.*, pages 809–815, San Jose, CA, 1992
42. **Ghanea-Hercock, R.; Barnes, D.P.:** An evolved fuzzy reactive control system for co-operating autonomous robots. In *Procs. of*

the Int. Conf. on Simulation and Adaptive Behavior (*SAB*), Cap Cod, 1996

43. Goodridge, S.G.; Kay, M.G.; Luo, R.C.: Multi-layered fuzzy behavior fusion for reactive control of an autonomous mobile robot. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 579–584, Barcelona, SP, 1997

44. Goodridge, S.G.; Luo, R.C.: Fuzzy behavior fusion for reactive control of an autonomous mobile robot: MARGE. In *Procs. of the IEEE Int. Conf. on Robotics and Automation*, pages 1622–1627, San Diego, CA, 1994

45. Guzzoni, D.; Cheyer, A.; Julia, L.; Konolige, K.: Many robots make short work. *AI Magazine*, 18(1):55–64, Spring 1997

46. Hacking, I.: *The emergence of probability*. Cambridge University Press, Cambridge, UK, 1975

47. Hanks, S.; Firby, R.J.: Issues and architectures for planning and execution. In *Procs. of the DARPA Workshop on Innovative Approaches to Planning, Sheduling and Control*, San Diego, CA, 1990

48. Hasemann, J.M.: Fuzzy logic for path and grasp planning. VTT Machine Automation, on-line report at http://www.ele.vtt.fi/aut/jmhfuzgr.htm, Oulu, FI, 1996

49. Hirota, K.; Arai, Y.; Hachisu, S.: Fuzzy controlled robot arm playing two-dimensional ping-pong game. *Fuzzy Sets and Systems*, 32:149–159, 1989

50. Hoffmann, F.: Soft computing techniques for the design of mobile robot behaviours. *Information Sciences*, 1998. To appear

51. Hoffmann, F.; Pfister, G.: Evolutionary learning of a fuzzy control rule base for an autonomous vehicle. In *Procs. of the Conf. on Information Processing and Management of Uncertainty* (*IPMU*), pages 1235–1238, Granada, SP, 1996

52. Höhle, U.: A survey on the fundamentals of fuzzy set theory. In Kreith, editor, *Handbook of Mechanical Engineering*. CRC Press, 1997. To appear

53. Huser, J.; Surmann, H.; Peters, L.: Automatic behaviour adaption for mobile robots with different kinematics. In *Procs. of the European Congress on Fuzzy and Intelligent Technologies* (*EUFIT*), pages 1095–1099, Aachen, DE, 1996

54. Isik, C.: Identification and fuzzy rule-based control of a mobile robot motion. In *Procs. of IEEE Int. Symp. on Intelligent Control*, pages 94–99, 1987

55. Jazwinski, A.H.: *Stochastic processes and filtering theory*. Academic Press, 1970

56. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986

57. Kiendl, H.; Rüger, J.J.: Stability analysis of fuzzy control systems using facets functions. *Fuzzy Sets and Systems*, 70:275–285, 1995

58. Kim, W.J.; Ko, J.H.; Chung, M.J.: Uncertain robot environment modelling using fuzzy numbers. *Fuzzy Sets and Systems*, 61:53–62, 1994

59. Klir, G.J.; Folger, T.A.: *Fuzzy sets, uncertainty, and information*. Prentice-Hall, 1988

60. Krause, P.; Clark, D.: *Representing uncertain knowledge*. Kluwer Academic Press, Dordrecht, NL, 1993

61. Kruse, R.; Gebhardt, J.; Klawonn, F.: *Foundations of Fuzzy Systems*. Wiley and Sons, 1994

62. Latombe, J.C.: *Robot Motion Planning*. Kluver Academic Publishers, Boston, MA, 1991

63. Lee, C.C.: Fuzzy logic in control systems: fuzzy logic controller (Parts I and II). *IEEE Trans. on Systems, Man, and Cybernetics*, 20(2):404–435, 1990

64. Leonard, J.J.; Durrant-Whyte, H.F.; Cox, I.J.: Dynamic map building for an autonomous mobile robot. *Int. J. of Robotics Research*, 11(4):286–298, 1992

65. Levesque, H.J.; Brachman, R.J.: A fundamental tradeoff in knowledge representation and reasoning. In R.J. Brachman and H.J. Levesque, editors, *Readings in Knowledge Representation*, pages 41–70. Morgan Kaufmann, Los Altos, CA, 1985

66. Li, W.: Fuzzy logic-based 'perception-action' behavior control of a mobile robot in uncertain environments. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 1626–1631, Orlando, FL, 1994

67. López-Sánchez, M.; López de Mántaras, R.; Sierra, C.: Incremental map generation by low const robots based on possibility/necessity grids. In *Procs. of the Conf. on Uncertainty in Artif. Intell.*, 1997

68. De Luca, A.; Oriolo, G.; Samson, C.: Feedback control of a nonholonomic car-like robot. In J.-P. Laumond, editor, *Planning robot motion*, chapter 4. Springer-Verlag, Berlin, DE, 1998. To Appear

69. Maeda, M.; Maeda, Y.; Murakami, S.: Fuzzy drive control of an autonomous mobile robot. *Fuzzy Sets and Systems*, 39:195–204, 1991

70. Maeda, M.; Shimakawa, M.; Murakami, S.: Predictive fuzzy control of an autonomous mobile robot with forecast learning function. *Fuzzy Sets and Systems*, 72:51–60, 1995.

71. Makita, Y.; Hagiwara, M.; Nakagawa, M.: A simple path planning system using fuzzy rules and a potential field. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 994–999, Orlando, FL, 1994

72. Martínez, J.L.; Ollero, A.; García-Cerezo, A.: Fuzzy strategies for path tracking of autonomous vehicles. In *Procs. of the European Congress on Fuzzy and Intelligent Technologies* (*EUFIT*), pages 24–30, Aachen, DE, 1993

73. McDermott, D.: Planning reactive behavior: a progress report. In *Procs. of the DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control*, pages 450–458, San Diego, CA, 1990

74. Michaud, F.: Selecting behaviors using fuzzy logic. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 585–592, Barcelona, SP, 1997

75. Michaud, F.; Lachiver, G.; Le Dinh, C.T.: A new control architecture combining reactivity, deliberation and motivation for situated autonomous agent. In *Procs. of the Int. Conf. on Simulation and Adaptive Behavior* (*SAB*), pages 245–254, Cap Cod, 1996

76. Moravec, H.P.; Elfes, A.: High resulution maps from wide angle sonar. In *Procs. of the IEEE Int. Conf. on Robotics and Automation*, pages 116–121, 1985

77. Moutarlier, P.; Chatila, R.: Stochastic multisensory data fusion for mobile robot location and environment modeling. In *5th Int. Symp. on Robotics Research*, pages 207–216, Tokyo, JP, 1989

78. Murphy, R.R.; Hawkins, D.K.: Behavioral speed control based on tactical information. In *Procs. of the Int. Conf. on Intelligent Robots and Systems* (*IROS*), pages 1715–1721, Osaka, JP, 1996

79. Nilsson, N.J.: SHAKEY the robot. Technical Note 323, SRI Artificial Intelligence Center, Menlo Park, California, 1984

80. Nilsson, N.J.: Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1:139–158, 1994

81. Nishimori, K.; Tokutaka, H.; Hirakawa, S.; Kishida, S.; Ishihara, N.: Comparison of several fuzzy reasoning methods on driving control of a model car. In *Procs. of the Int. Conf. on Fuzzy Logic and Neural Networks*, pages 421–424, Iizuka, JP, 1992

82. Nourbakhsh, I.; Morse, S.; Becker, C.; Balabanovic, M.; Gat, E.; Simmons, R.; Goodridge, S.; Potlapalli, H.; Hinkle, D.; Jung, K.; Van Vactor, D.: The winning robots from the 1993 robot competition. *AI Magazine*, 14(4):51–62, Winter 1993

83. Ollero, A.; García-Cerezo, A.; Martínez, L.; Mandow, A.: Fuzzy tracking methods for mobile robots. In M. Jamshidi, A. Titli, L. Zadeh and S. Boverie, editors, *Applications of fuzzy logic: Towards high machine intelligence quotient systems*, chapter 17. Prentice-Hall, New Jersey, 1997. To appear

84. Oriolo, G.; Ulivi, G.; Vendittelli, M.: Real-time map building and navigation for autonomous robots in unknown environments. *IEEE Trans. on Systems, Man, and Cybernetics*, 1998. To appear

85. Oriolo, G.; Vendittelli, M.; Ulivi, G.: Path planning for mobile robots via skeletons on fuzzy maps. *Intelligent Automation and Soft Computing*, 2(4):355–374, 1996

86. Pan, J.; Pack, D.J.; Kosaka, A.; Kak, A.C.: FUZZY-NAV: a vision-based robot navigation architecture using fuzzy inference for uncertainty reasoning. In *Procs. of the World Congress on Neural Networks*, pages 602–607, Washington, DC, 1995

87. **Panti, G.:** Multi-valued logics. In D. Gabbay, Ph. Smets, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*. Kluwer Academic Press, 1998. To appear
88. **Payton, D.W.:** An architecture for reflexive autonomous vehicle control. In *Procs. of the IEEE Int. Conf. on Robotics and Automation*, pages 1838–1845, San Francisco, CA, 1986
89. **Pereira, J.; Bowles, J.B.:** A comparison of PID and fuzzy control of a model car. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 849–854, Orlando, FL, 1994
90. **Pfluger, N.; Yen, J.; Langari, R.:** A defuzzification strategy for a fuzzy logic controller employing prohibitive information in command formulation. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 717–723, San Diego, CA, 1992. IEEE Press
91. **Pin, F.G.; Bender, S.R.:** Adding memory processing behavior to the fuzzy behaviorist approach (FBA): resolving limit cycle problems in autonomous mobile robot navigation. *Intelligent Automation and Soft Computing*, 3, 1997. to appear
92. **Pin, F.G.; Watanabe, Y.:** Navigation of mobile robots using a fuzzy behaviorist approach and custom-designed fuzzy inference boards. *Robotica*, 12:491–503, 1994
93. **Pin, F.G.; Watanabe, Y.:** Automatic generation of fuzzy rules using the fuzzy behaviorist approach: the case of sensor-based robot navigation. *Intelligent Automation and Soft Computing*, 1(2):161–178, 1995
94. **Poloni, M.; Ulivi, G.; Vendittelli, M.:** Fuzzy logic and autonomous vehicles: experiments in ultrasonic vision. *Fuzzy Sets and Systems*, 69:15–27, 1995
95. **Raju, G.V.S.; Zhou, J.; Kisner, R.A.:** Hierarchical fuzzy control. *Int. J. Control*, 54(5):1201–1216, 1991
96. **Rausis, K.; Myszkorowski, P.; Longchamp, R.:** Obstacle avoidance control of a mobile robot. In *Procs. of the 3rd IFAC Symp. on Intelligent Components and Instruments for Control Applications*, pages 263–268, Annency, FR, 1997
97. **Rescher, N.:** Semantic foundations for the logic of preference. In N. Rescher, editor, *The Logic of Decision and Action*. Univ. of Pittsburgh Press, Pittsburgh, PA, 1966
98. **Rosenblatt, J.K.; Payton, D.W.:** A fine-grained alternative to the subsumption architecture for mobile robot control. In *Procs of the IEEE Int. Conf. on Neural Networks*, volume 2, pages 317–324, Washington, DC, 1989
99. **Rueda, A.; Pedrycz, W.:** A design method for a class of fuzzy hierarchical controllers. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 196–199, San Francisco, CA, 1993
100. **Ruspini, E.H.:** Fuzzy logic in the Flakey robot. In *Procs. of the Int. Conf. on Fuzzy Logic and Neural Networks* (*IIZUKA*), pages 767–770, Iizuka, JP, 1990
101. **Ruspini, E.H.:** On the semantics of fuzzy logic. *Int. J. of Approximate Reasoning*, 5:45–88, 1991
102. **Ruspini, E.H.:** Truth as utility: A conceptual synthesis. In *Procs. of the 7th Conf. on Uncertainty in Artificial Intelligence*, pages 316–322, Los Angeles, CA, 1991
103. **Saffiotti, A.:** An AI view of the treatment of uncertainty. *The Knowledge Engineering Review*, 2(2):75–97, 1987. See the panel discussion about this paper on KER 3(1) 59–86
104. **Saffiotti, A.:** Robot navigation under approximate self-localization. In F. Pin M. Jamshidi and P. Dauchez, editors, *Robotics and Manufacturing vol. 6 – Proc of the 6th Int. Symp.* (*ISRAM*), pages 589–594. ASME Press, New York, NY, 1996
105. **Saffiotti, A.; Konolige, K.; Ruspini, E.H.:** A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995
106. **Saffiotti, A.; Ruspini, E.H.; Konolige, K.:** Blending reactivity and goal-directedness in a fuzzy controller. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 134–139, San Francisco, California, 1993. IEEE Press
107. **Saffiotti, A.; Wesley, L.P.:** Perception-based self-localization using fuzzy locations. In M. van Lambalgen L. Dorst and F. Voorbraak, editors, *Reasoning with Uncertainty in Robotics – Procs of the 1st Int. Workshop*, number 1093 in LNAI, pages 368–385. Springer-Verlag, Berlin, DE, 1996
108. **Saffiotti (Maintainer), A.:** Fuzzy logic in the autonomous mobile robot Flakey: on-line bibliography. http://iridia.ulb.ac.be/saffiotti/flakeybib.html. ftp://iridia.ulb.ac.be/pub/saffiotti/robot/
109. **Shin, K.G.; Cui, X.:** Design of a knowledge-based controller for intelligent control systems. *IEEE Trans. on Systems, Man, and Cybernetics*, 21(2):368–375, 1991
110. **Skubic, M.; Graves, S.; Mollenhauer, J.:** Design of a two-level fuzzy controller for a reactive miniature mobile robot. In *Procs. of the 3rd Int. Conf. on Industrial Fuzzy Control and Int. Systems*, Houston, TX, 1993
111. **Smets, Ph.:** Probability, possibility, belief: which for what? In *Procs. of the Wp. on Foundations and Applications of Possibility Theory*, pages 20–40, Ghent, BE, 1995. World Scientific
112. **Smith, R.C.; Cheeseman, P.:** On the representation and estimation of spatial uncertainty. *Int. J. of Robotics Research*, 5(4):56–68, 1986
113. **Sugeno, M; Griffin, M.F.; Bastian, A.:** Fuzzy hierarchical control of an unmanned helicopter. In *Procs. of Int. Fuzzy System Association Conference* (*IFSA*), pages 179–182, Seoul, KR, 1993
114. **Sugeno, M.; Morofushi, T.; Mori, T.; Tatematsu, T.:** Fuzzy algorithmic control of a model car by oral instructions. *Fuzzy Sets and Systems*, 32:207–219, 1989
115. **Sugeno, M.; Nishida, M.:** Fuzzy control of model car. *Fuzzy Sets and Systems*, 16:103–113, 1985
116. **Surmann, H.:** *Automatischer Entwurf von Fuzzy Systemen*. PhD thesis, Univ. Dortmund, Düsseldorf, DE, 1995. VDI Reihe 8 Nr. 452, VDI Verlag
117. **Surmann, H.; Huser, J.; Peters, L.:** A fuzzy system for indoor mobile robot navigation. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 83–86, Yokohama, JP, 1995. IEEE Press
118. **Takeuchi, T.; Nagai, Y.; Enomoto, N.:** Fuzzy control of a mobile robot for obstacle avoidance. *Information Sciences*, 43:231–248, 1988
119. **Tanaka, K.; Sano, M.:** Trajectory stabilization of a model car via fuzzy control. *Fuzzy Sets and Systems*, 70:155–170, 1995
120. **Tanaka, K.; Sugeno, M.:** Stability analysis and design of fuzzy control systems. *Fuzzy Sets and Systems*, 45:135–156, 1992
121. **Tunstel, E.:** Fuzzy spatial map representation for mobile robot navigation. In *Procs. of the ACM 10th Annual Symp. on Applied Computing*, pages 586–589, Nashville, TN, 1995
122. **Tunstel, E.; Danny, H.; Lippincott, T.; Jamshidi, M.:** Autonomous navigation using an adaptive hierarchy of multiple fuzzy behaviors. In *Procs. of the IEEE Int. Sym. on Computational Intelligence in Robotics and Automation*, Monterey, CA, 1997
123. **Tunstel, E.; Lippincott, T.:** Genetic programming of fuzzy coordination behaviors for mobile robots. In *Procs. of the World Automation Congress* (*WAC*), *ISSCI Track*, pages 647–652, Montpellier, FR, 1996. ASME Press
124. **Tunstel, E.W.:** *Adaptive Hierarchy of Distributed Fuzzy Control*: *Application to Behavior Control of Rovers*. PhD dissertation, University of New Mexico, Department of Electrical & Computer Engineering, 1996
125. **von Altrok, C.; Krause, B.; Zimmermann, H.J.:** Advanced fuzzy logic control of a model car in extreme situations. *Fuzzy Sets and Systems*, 48:41–52, 1992
126. **Voudouris, C.:** *Fuzzy hierarchical control for autonomous mobile robots*. MSc Thesis, Dept. of Computer Science, Univ. of Essex, Esssex, UK, Sept 1993
127. **Voudouris, C.; Chernett, P.; Wang, C.J.; Callaghan, V.L.:** Hierarchical behavioural control for autonomous vehicles. In A. Halme and K. Koskinen, editors, *Procs. of the 2nd IFAC Conf. on Intelligent Autonomous Vehicles*, pages 267–272, Helsinki, FI, 1995
128. **Wang, L.-X.:** A mathematical formulation of hierarchical systems using fuzzy logic systems. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 183–188, Orlando, FL, 1994
129. **Yager, R.R.:** On a hierarchical structure for fuzzy modeling and control. *IEEE Trans. on Systems, Man, and Cybernetics*, 23(4):1189–1197, 1993

130. **Yager, R.R.:** Knowledge-based defuzzification. *Fuzzy Sets and Systems*, 80:177–185, 1996
131. **Yen, J.; Pfluger, N.:** A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation. *IEEE Trans. on Systems, Man, and Cybernetics*, 25(6):971–978, 1995
132. **Zadeh, L.A.:** Why the success of fuzzy logic is not paradoxical. *IEEE Expert*, pages 43–46, August 1994. Response to Elkan's "The paradoxical success of fuzzy logic", same issue
133. **Zhang, J.; Raczkowskyan A. Herp, J.:** Emulation of spline curves and its applications in robot motion control. In *Procs. of the IEEE Int. Conf. on Fuzzy Systems*, pages 831–836, Orlando, FL, 1994.