

5 USING FUZZY LOGIC FOR MOBILE ROBOT CONTROL

Alessandro Saffiotti

Enrique H. Ruspini

Kurt Konolige

Abstract: The development of techniques for autonomous operation in real-world, unstructured environments constitutes one of the major trends in the current research on mobile robotics. In spite of recent advances, a number of fundamental difficulties remain. In this chapter, we discuss how fuzzy logic techniques can be used to address some of these difficulties. To illustrate the discussion, we describe the fuzzy-logic solutions developed on Flakey, the mobile robot of SRI International.

5.1 INTRODUCTION

The operation of an autonomous mobile robot in a real-world unstructured environment requires consideration of multiple issues.

First, the controller must be able to operate under conditions of imprecision and uncertainty. For example, prior knowledge about the environment is, in general, incomplete, uncertain, and approximate. Perceptually acquired information is also typically noisy and incomplete. Furthermore, the execution of control commands is not completely reliable while the dynamics of real-world environments is complex and unpredictable. To cope with these difficulties, the controller must be able to respond *reactively* to unforeseen events as soon as they are perceived.

Second, the controller must be able to reach explicit goals such as the objectives specified by users or the subgoals generated by the high-level deliberation

processes of automated intelligent planners. The controller must be able to incorporate these goals into *purposive* procedures that promote their attainment.

Finally, the controller must be able to consider multiple concurrent requirements. For example, a mobile robot may need to reach the end of a hallway, while keeping its load well balanced, and avoiding static and moving obstacles. It is important to note that these requirements may demand the activation of both reactive and goal-achieving procedures. A major problem in the design of an effective controller for mobile autonomous devices is the *combination* of goal-specific strategies by resolution of conflicts between multiple objectives.

The autonomous robotics literature contains numerous examples of approaches that seek to deal with these problems by decomposition of the overall control function into a number of simple units, called *behaviors*, (Brooks, 1986). Each behavior is responsible for producing certain actions (typically, motion and perception actions), aimed at achieving or maintaining a particular goal such as staying away from obstacles.

Behavior-based approaches to mobile robot control have gained increasing popularity, being supported by considerations arising from the study of animal behavior (Arbib, 1981), by architectural concerns (Brooks, 1986), and by their implementation convenience (Gat, 1992). These approaches face, however, two major problems: the combination of simple behaviors to form more complex ones, and the integration of behaviors with higher-level processes, like a strategic planner.

Fuzzy logic provides tools that are of potential interest to mobile robot control. Most applications of fuzzy logic in this field concern the use of fuzzy control techniques to implement individual behavior units. Fuzzy controllers are a convenient choice when an analytical linear model of the system to be controlled cannot be easily obtained; they have shown a good degree of robustness in face of large variability and uncertainty in the parameters; and they lend themselves to efficient implementations, including hardware solutions. These characteristics fit well the needs of autonomous navigation 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 since Sugeno and Nishida's model car (Sugeno and Nishida, 1985), and it is still the most common one. (See (Saffiotti, 1997b) for a critical survey.)

Although less noticed until recently, fuzzy logic can also provide useful tools to address another important difficulty in behavior-based approaches to autonomous robotics: how to coordinate the concurrent execution of several behaviours aimed at the achievement of different, possibly conflicting objectives; and how to link this coordinated activity to the global analysis performed by the higher-level reasoning processes. This chapter presents a high-level discussion of a fuzzy-logic based approach to autonomous navigation that addresses these fundamental difficulties.

We will describe techniques to control an autonomous mobile robot based on the use of fuzzy logic for trading off conflicting goals. The formal bases for these techniques have been set forth by Ruspini (Ruspini, 1990; Ruspini, 1991a) after the seminal works by Zadeh (e.g., (Zadeh, 1978)). In a nutshell, each goal is associated with a function that maps each perceived situation to a measure of desirability of possible actions from the point of view of that goal. Behaviors induced by many simultaneous goals can be smoothly blended into a dynamic sequence of control actions. In particular, reactive and goal-oriented behaviors are blended into a single sequence of control actions.

These techniques have been first implemented on Flakey, the mobile robot platform of the Artificial Intelligence Center of SRI International (Saffiotti et al., 1993; Konolige et al., 1997). The illustrative examples that we present in this paper are drawn from our experience with this robot. The scope of these techniques extends, however, beyond this particular test-bed, as they effectively deal with basic problems in the development of intelligent autonomous systems such as the attainment of multiple, possibly conflicting, objectives; the integration of numerical control and symbolic planning; and the construction and utilization of approximate and incomplete models of the world.

5.2 FUZZY BEHAVIORS

In our approach to robot control, we express desirable behavioral traits as quantitative *preferences*, defined over the set possible control actions, from the perspective of the goal associated with that behavior. For example, a behavior for avoiding obstacles could map configurations of sonar readings that correspond to the presence of an obstacle on the left of the robot into a function that prefers actions that steer the robot to the right.

Following the formal semantic characterization of Ruspini (Ruspini, 1991a; Ruspini, 1991b), we describe each behavior B in terms of a desirability function

$$Des_B : \text{State} \times \text{Control} \rightarrow [0, 1],$$

that measures, for each state vector x and control vector c , the desirability $Des_B(x, c)$ of applying the control c when the state is x *from the point of view of attaining goals associated with B* . Equivalently, we can say that Des_B associates each situation x with the fuzzy set \mathbf{C} of control values characterized by the membership function $\mu_{\mathbf{C}}(c) = Des_B(x, c)$.

In general, c is an n -dimensional vector of control values (the nature of the input vector x is further characterized in Sect. 5.4). In the case of our test-bed robot, Flakey, c is a pair of set points (v, θ) , where v is a linear velocity and θ is a turning angle. An internal conventional motor controller is employed to achieve the set-points recommended by the fuzzy controller.

Desirability functions are less committing than classical control functions or similar formal objects employed in other approaches to robot control, like Khatib's pseudo-potential fields (Khatib, 1986) or Arkin's motor schemas (Arkin, 1990a). In fact, desirability functions do not map input states to controls but to a measure quantifying preferences over the space of these controls. This

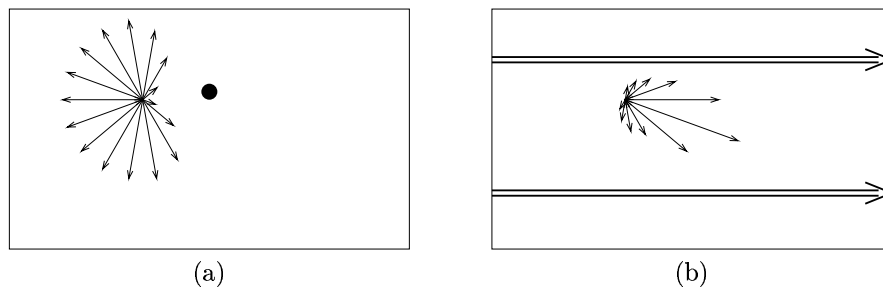


Figure 5.1 Desirability functions for two behaviors: (a) Keep-Off; (b) Follow.

approach recognizes that many alternative controls can generate, to a greater or lesser extent, the same *type* of behavior. The control that should actually be applied in a given situation depends not only on that situation but also on the interaction with other concurrent behaviors. We will see in Sect. 5.7 that the major difference between desirability functions and potential-field approaches lays in the way that behaviors are merged.

To better understand the role of desirability functions, consider a behavior named **KeepOff** seeking to keep the robot away from a point P with coordinates (P_x, P_y) (e.g., P may correspond to a sensed obstacle). We may define a desirability function for **KeepOff** by considering, for each control $c = (v, \theta)$, how close the robot would come to P after the next control cycle, if c were to be actually applied.¹ If the new position of the robot after application of the control c is denoted by $\hat{R} = (\hat{R}_x, \hat{R}_y)$, then the desirability of employing that control might be evaluated by means of a ramp function defined on the distance between the predicted position and the obstacle:

$$Des_{\text{KeepOff}}(x, c) = \min \left(1, \frac{1}{d_0} \sqrt{(P_x - \hat{R}_x)^2 + (P_y - \hat{R}_y)^2} \right)$$

where d_0 is a threshold denoting the safe distance.²

Figure 5.1 (a) plots this desirability function for a fixed value of v and different values of θ . In this figure, the position of the robot is indicated by the smaller dot while that of the obstacle is shown by the larger one. Each vector indicates a possible turning control θ with its length proportional to the corresponding value of the desirability function. Figure 5.1 (b) shows a possible desirability function for the **Follow** behavior, which seeks to proceed along a given lane (indicated by the double lines).

Simple behaviors may be composed into more complex sets of actions by combining their desirability functions by means of the logical operations of fuzzy logic. In particular, we utilize the min, max, and complement to one operations to compute the truth value of logical conjunction (\wedge), disjunction (\vee), and negation (\neg), respectively.³ We will also utilize \oslash , the quasi-inverse

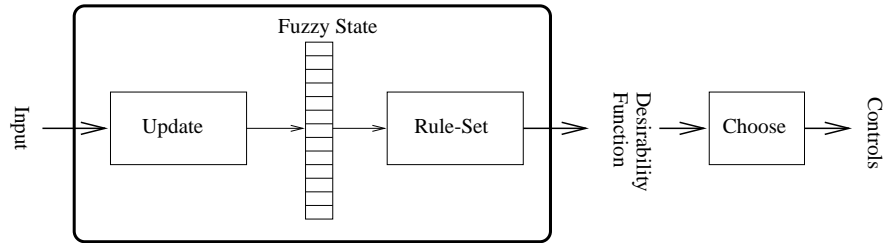


Figure 5.2 Implementation of a behavior in the fuzzy controller.

of \wedge , to compute the truth value of a logical implication from those of its antecedent and consequent.

5.3 IMPLEMENTATION OF A FUZZY BEHAVIOR

The definition and implementation of desirability functions is a major aspect of our approach. Proper characterization of these functions is important to assure efficient performance and to facilitate effective combination of multiple behaviors.

We have already suggested a procedure to define such functions in the context of the **KeepOff** behavior. This type of procedure starts from an analytical specification of the intended behavior in terms of desirable states building then the desirability function from this specification and from a model of the robot. In general this is a complex procedure, which we are currently seeking to simplify and systematize. Currently, however, we follow a more empirical path, and implement desirability functions by a set of fuzzy rules that are experimentally tuned until the produced behavior is judged to be a satisfactory approximation of the intended one.

We implement each behavior by a fuzzy machine having an architecture illustrated in Figure 5.2. The **Fuzzy State** is a vector of fuzzy variables, each representing the truth value, on the $[0, 1]$ scale, of a fuzzy proposition of interest (e.g., “obstacle-close-on-left”). At every cycle, the **Update** module produces a new fuzzy state based on the current input. The **Rule-Set** module contains a set R of fuzzy rules

$$\text{IF } A_i \text{ THEN } C_i, \quad i = 1, \dots, n,$$

where A_i is a propositional formula in fuzzy logic built from the fuzzy propositions in **Fuzzy State**, plus the logical connectives \wedge , \vee and \neg , evaluated as explained above, and where C_i is a fuzzy set of control values. For each possible control value c , $C_i(c)$ quantifies the extent by which c is a good instance of C_i . From these fuzzy rules, a desirability function Des_R is computed by

$$Des_R(x, c) = [A_1(x) \wedge C_1(c)] \vee \dots \vee [A_n(x) \wedge C_n(c)]. \quad (5.1)$$

IF (lane-too-right \wedge \neg lane-angled-left) THEN turn-medium-right IF (lane-too-left \wedge \neg lane-angled-right) THEN turn-medium-left IF (lane-angled-right \wedge \neg centerline-on-left) THEN turn-smooth-right IF (lane-angled-left \wedge \neg centerline-on-right) THEN turn-smooth-left
--

Figure 5.3 Fuzzy rules for the Follow behavior.

Intuitively, the equation (5.1) characterizes a control c as being desirable in the state x , if there is some rule in R that supports c and whose antecedent is true in x . This interpretation of a fuzzy rule set is that of a classical (Mamdani type) fuzzy controller, generalized so as to allow each antecedent A_i to be an arbitrary fuzzy-logic formula.

Figure 5.3 shows the rules that implement the previously discussed Follow behavior in our robot test-bed, Flakey. These rules keep the robot within the lane boundaries (first two rules) and lined up with them (last two). The **Fuzzy State** is composed of six propositional variables, whose truth value depends on the the current position of the lane with respect to the robot. Flakey’s controller incorporates a dozen of behaviors such as this one—including behaviors for avoiding obstacles, for facing a given object, for crossing a door, and for reaching a near location. Each such behavior typically consists of four to eight rules involving up to a dozen fuzzy predicates.

Regardless of its implementation approach, a desirability function specifies, for each input variable value, a ranking over possible controls rather than a unique control value to apply in that situation. The robot eventually employs this ranking to *choose* one specific control \hat{c} that is sent to the plant. A possible mechanism to accomplish that selection is centroid defuzzification:

$$\hat{c} = \frac{\int c \text{Des}_R(x, c) dc}{\int \text{Des}_R(x, c) dc}, \quad (5.2)$$

which computes the mean of possible control values, weighted by their degree of desirability.

Centroid defuzzification has been found satisfactory in our experiments whenever the rules in a rule set do not suggest dramatically opposite actions. In those cases, centroid defuzzification obviously does not work since averaging of multi-modal desirability measures might result in selection of a very undesirable choice (e.g., the best trade-off between avoiding an incoming train by jumping to the left or to the right is hardly to stay on the track!). In our development of mobile robot controllers, our empirical strategy has been to design the rule sets so as to avoid production of multi-modal desirability functions—roughly, we insist that rules that propose opposite controls have mutually exclusive antecedents. Other authors (Yen and Pfluger, 1992) have relied, however, on alternative defuzzification functions.

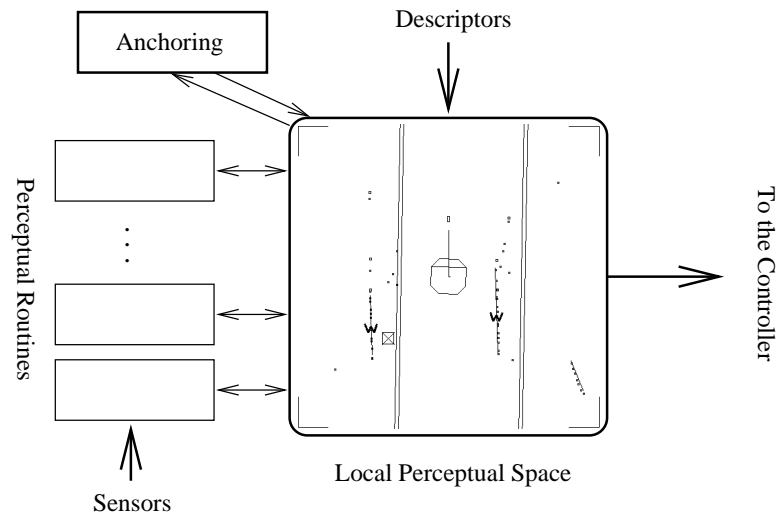


Figure 5.4 Local perceptual space and controller input.

5.4 TAKING GOALS INTO CONSIDERATION

We have been, so far, rather vague about the nature of the input to the controller's behaviors. The simplest form that might be taken by such input is direct sensor observations, e.g., readings from the sonar sensors indicating the presence of obstacles. Behaviors that only use the current perceptual data, possibly after some perceptual interpretation, are often called *reactive*. The amount of perceptual interpretation that is processed by such behaviors is usually small, in the interest of short response time, but it need not be.

When explicit *goals* are sought, however, it is often necessary to resort to some internal representation that does not originate in the sensor data. For example, suppose that the robot is trying to reach a landmark that is out of the range of its sensors. The mobile device must then rely on assumptions about the landmark position based, for example, on a map of the environment. Consequently, goal-achieving behaviors generally require some non-perceptual input to relate relevant properties of the goal with the adequacy of control actions.

Our strategy to give explicit goals to purposive behaviors is based on the utilization, as the input source for the controller, of a global data structure, called the *Local Perceptual Space* (LPS), which fuses both sensor and prior information. This approach is illustrated in Figure 5.4, where the LPS is represented in a Cartesian plane centered on the robot, itself drawn in the center of the figure as seen from above. The LPS includes sensor data at different levels of abstraction and interpretation. The small dots represent, for example, readings from the sonar sensors while the long segments labeled "W" are

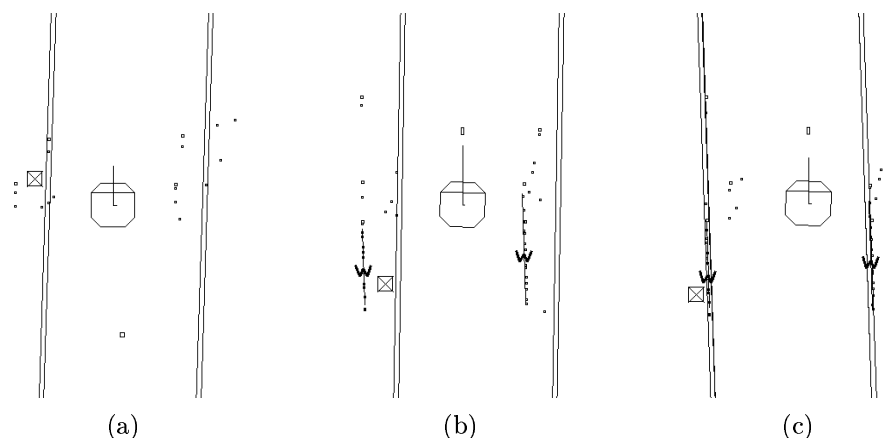


Figure 5.5 Anchoring a corridor descriptor to sensor readings.

linear features extracted from these readings, assumed to be walls. The LPS also includes *descriptors* of various objects used by the purposeful behaviors. For example, the two double lines in the picture indicate the descriptor of a corridor to follow.

Descriptors are a general, flexible mean of representing the abstract data required by goal-achieving behaviors. Descriptors are typically specified by a planner or some other higher-level deliberation process on the basis of prior information such as that contained on a map of the environment. For example, a lane descriptor might be employed to specify the expected position of a specific corridor that we want to follow.

Descriptors permit the implementation of goal-achieving behaviors using an approach similar to that employed to specify reactive behaviors (see Fig. 5.2). In general, however, the **Fuzzy State** specification for the former class of behaviors depends on properties of a descriptor (e.g., its position) rather than on direct perceptual inputs. The Follow behavior above is an example of a goal-achieving behavior.

Descriptors do not necessarily correspond to physical objects in the operational environment of the robot. For example, we may use an abstract lane descriptor to implement a “foray” behavior that takes the robot out a few meters from a given home position and then back again (this behavior was actually implemented in the controller employed to guide Flakey in the 1992 AAI Robotics Competition (Congdon et al., 1993)). Whenever the descriptor corresponds to an actual environmental feature, however, it is important to maintain an accurate correspondence between this feature and its descriptor since control rules are based solely on the properties of the descriptor while the robot must perform the intended behavior with respect to the real feature.

The process employed in our approach to maintain the correspondence between descriptors and environmental features is called *anchoring* (Saffiotti,

1994). Figure 5.5 shows an example of anchoring during the execution of the **Follow** behavior. Initially (a), the robot is given a descriptor—produced from prior map information—of a corridor to follow (double lines). After a while (b), enough sonar readings are gathered to allow the perceptual routines to recognize the existence of two parallel walls, marked by “W.” Note that the descriptor does not match precisely the position of the real corridor as there may be errors either in the robot’s estimate of its own location or in the map of the environment. Finally (c), the perceived position of the walls is used to update the descriptor, i.e., the descriptor is *anchored* to the sensor data. As the rules in **Follow** are based on the properties of the descriptor, this anchoring entails that the robot will now follow the actual (sensed) corridor.

In control theory terms, we could say that the controller in the example above initially operates in an open-loop fashion with respect to prior information, switching then to a closed-loop, sensor-based regime when the walls are perceived. The corridor descriptor is normally re-anchored at each control cycle whenever the walls are visible. The ability to recover from vague or imprecise assumptions is particularly important in practice since our maps typically contain only approximate metric information. The descriptor serves as a source of credible assumptions when perceptual data is not available, as is the case when first engaging a new corridor, or when the walls are momentarily occluded by obstacles.

5.5 BLENDING OF BEHAVIORS

In previous sections we have discussed the implementation of reactive and purposive behaviors by means of fuzzy rules. As it was noted in the Introduction, several behaviors of either type may be active at any time during the operation of the intelligent mobile agent. Correspondingly, several instances of the fuzzy machine shown in Fig. 5.2 are simultaneously active in the fuzzy controller. At any given moment, the overall behavior of the robot is the result of the combination of all the behaviors that are active at that moment. In this section, we discuss the combination or *blending* of fuzzy behaviors.

The simplest type of behavioral combination is that where the desirability functions are simply conjoined to indicate that, regardless of context, we aim to attain all the objectives that they represent. In this case, for every state x , the combined desirability of a control value is given by the minimum of the degrees of desirability assigned by each behavior to that value:

$$(Des_1 \wedge Des_2)(x, c) = Des_1(x, c) \wedge Des_2(x, c) . \quad (5.3)$$

We call the expression $(Des_1 \wedge Des_2)$ the *conjunctive combination* of Des_1 and Des_2 . A disjunctive combination can be defined in a similar way using the operator \vee .

Note that we could use triangular norms different from \min in Equation (5.3). The nature of the triangular norm being employed essentially characterizes the tradeoff between goals that underlies their combination (Dubois and Prade, 1985). Utilization of the \min triangular norm means that, in order for the

conjunction to be deemed desirable to the level α , each of the conjuncts must be also desirable at least to the level α . An alternative conjunction operator, the product T-norm, results in a different notion of tradeoff: a decrease in the level of desirability of one conjunct can be compensated by a corresponding increase in the other.

Conjunctive combination works well in situations where the behaviors being combined do not conflict—that is, for every state there is some control that is desirable from the perspective of each behavior. When behaviors conflict, however, the combined desirability function would be identically zero. Unfortunately, this is situation is often the norm rather than the exception.

We may want to combine, for example, a behavior for traveling along the center of a corridor and another one for avoiding obstacles. When the robot is approaching an obstacle ahead, the first behavior would prefer controls that go forward while the second would favor controls that turn the robot, say, right. The key observation here is that each behavior has its own *context* of applicability and that each desirability function should be considered only when that context is appropriate. In our example, the first behavior can be sensibly applied only in situations where the space in front of the robot is free. When the obstacle is detected, this behavior is outside its area of competence and we should (at least partially) disregard its preferences.

We represent contexts of applicability as fuzzy sets of the set of possible states. We define the *context restriction* of a desirability function Des to a context Cxt by the expression

$$Des \downarrow Cxt(x, c) = Des(x, c) \circledast Cxt(x),$$

where Cxt is a fuzzy subset of state space.

Utilization of the quasi-inverse \circledast to compute truth values of implications might be interpreted as follows. When the state x is in the core of the context Cxt , i.e., $Cxt(x) = 1$, then the desirability of a control c is measured by $Des(x, c)$. When the state x is fully outside of the context of applicability, i.e., $Cxt(x) = 0$, then any control is admissible (as far as the particular goal associated with Des is concerned). Intermediate values of Cxt make fully admissible those control values c for which $Des(x, c) \geq Cxt(x)$.⁴ In other words, context restriction leads to a less committed desirability measure that only expresses a definite preference when the agent is in one of the context states, but (partially) allows other possibilities when it is (partially) outside these states.

We can combine context restriction and conjunctive combination into a general combination pattern, called *context-dependent blending*, where each behavior is associated with a context, which identifies the states where the behavior is competent. The blended desirability function is defined by

$$Des = Des_1 \downarrow Cxt_1 \wedge Des_2 \downarrow Cxt_2. \tag{5.4}$$

The previous definition characterizes context-dependent blending in terms of a desirability function that agrees with Des_1 to the extent that the current

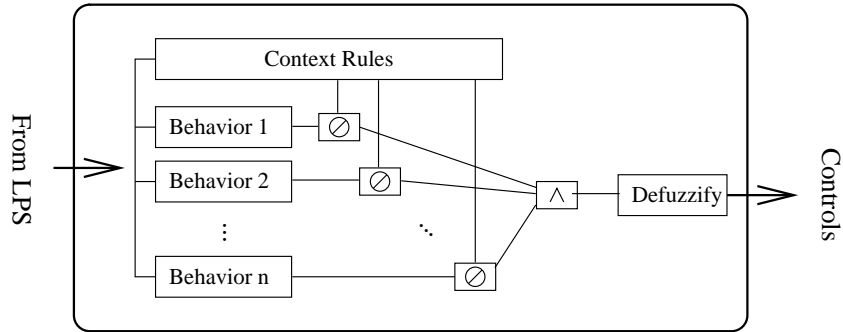


Figure 5.6 A hierarchical fuzzy controller for context-dependent blending.

situation satisfies Cxt_1 , and that agrees with Des_2 to the extent that it satisfies Cxt_2 . Multiple behaviors may be combined by context-dependent blending. The commutativity and associativity of the \wedge operator assure that the generalization of the right hand side of Equation 5.4 does not depend on the order of the behaviors being combined.

Context-dependent blending may be implemented in a hierarchical fuzzy controller as shown in Fig. 5.6. In this configuration, each behavioral module is a fuzzy machine as illustrated in Fig. 5.2. The context rules are fuzzy meta-rules of the form

$$\text{IF } A_j \text{ THEN apply } B_j, \quad j = 1, \dots, m, \quad (5.5)$$

where A_j is a fuzzy-logic formula describing a context, and B_j is a behavior. The truth values, given the current state, of the formulas defining the contexts are used to discount the desirability function produced by the corresponding behavior. The discounted values are then merged together according to (5.4). Finally, the resulting blended desirability function is defuzzified by (5.2) to produce a crisp control. As we shall see in Section 5.7, the fact that defuzzification is performed *after* combination is essential.⁵

Figure 5.7 shows a simple example of context-dependent blending where FOLLOW is employed to follow a corridor, and KEEP-OFF is activated to avoid obstacles. The meta-rules used to encode the contexts in this example are:

```
IF approaching_obstacle THEN apply Keep-Off
IF NOT(approaching_obstacle) THEN apply Follow
```

The bars (a) and (b) show the activation level and the preferred controls (turn and acceleration) for each behavior at two instants while the bottom line in each graph shows the result of the blending. In (a) an obstacle has been detected and the preferences of Keep-Off dominate, thus making the robot abandon the mid-line of the corridor. Later, when the path is clear, the goal-oriented preferences expressed by Follow regain importance (b), and the robot recovers its mid-corridor course. The graph at the bottom of the picture (c)

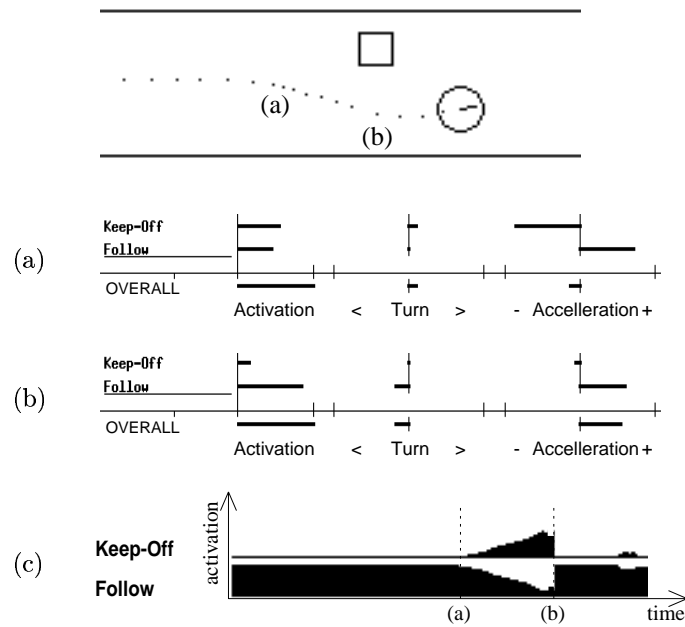


Figure 5.7 Blending corridor following and obstacle avoidance.

shows the evolution of the activation level of each behavior (i.e., the truth value of the corresponding context) over time.

5.6 EXPERIMENTS

The approach presented in this paper has been implemented on the SRI International mobile robot Flakey having been tested extensively in indoor robot navigation experiments. In this section, we present two experimental runs that illustrate the performance of Flakey.

Flakey is a custom-built mobile robot, with a height of 1 meter and a diameter of 0.6 meters, driven by two independent wheels at maximum linear velocity of about 0.5 meters/sec. Sensors include a ring of 12 sonars, bumpers, wheel encoders, and video and audio equipment (not used in the experiments reported here). On-board computation capabilities include a 2-processor Sparc 10-51. The control software is distributed over several processes that are daisy-chained with a basic cycle time of 100 msecs. Most of the interpretation and control routines have execution times ranging from 1 to 10 msecs on the Sparc 10 while the fuzzy controller requires about 2 msecs with one behavior active and approximately 10 msecs when 8 behaviors are active (on average, 2-3 behaviors are active at any given time during a typical navigation task). Al-

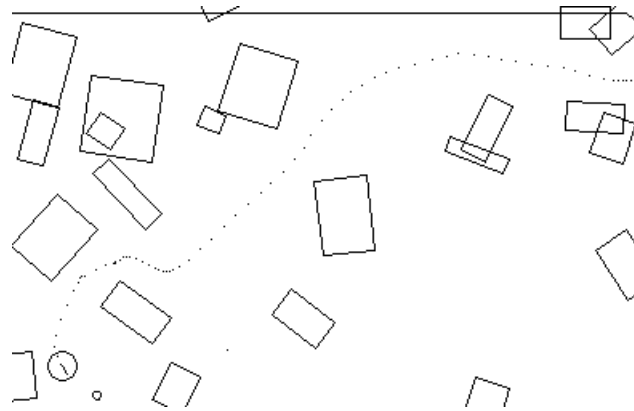


Figure 5.8 A run of the Wander behavior in an unknown environment.

though Flakey is capable of running all required computational processes on its on-board computers, we typically execute, for programming convenience, most functions remotely, communicating with Flakey via a radio-link.

Our first experimental result was obtained while the robot wandered around in an unknown environment avoiding static and moving obstacles. This task was performed by Flakey at the First AAAI Robotics Competition (Congdon et al., 1993). In this competition Flakey placed second in overall rankings and was particularly commended by the judges for its reliable reactivity. Flakey did not make use of maps of the environment, simply roaming around at random while trying to stay away from obstacles as they were perceived. Figure 5.8 shows a run of the Wander behavior in an environment similar to that of the competition. The smoothness of motion, both in turning and acceleration, can be seen in the wake of small dots that indicate Flakey's trajectory (one dot per second).

The Wander behavior is built by blending three basic behaviors:

1. The *Go-Forward* behavior keeps a constant cruising speed *when* the way is clear;
2. The *Keep-Off* behavior keeps the robot a safe distance away from obstacles *when* an obstacle is detected; and
3. The *Avoid-Collisions* behavior activates emergency maneuvers *when* an immediate collision danger is sensed.

All these behaviors are purely reactive, simply monitoring sonar readings to determine the presence of obstacles. The operation of these behaviors, however, does not require object descriptors.

Our second example, shown in Fig. 5.9, illustrates how context-dependent blending of behaviors may be employed to perform full navigation plans. Each

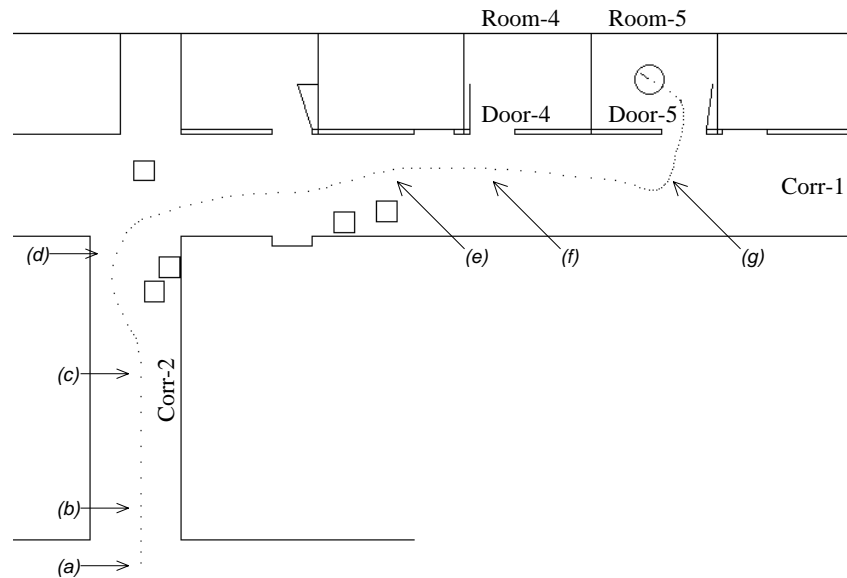


Figure 5.9 Execution of a full navigational plan.

action in the navigation plan is represented as a behavior having a context of applicability. This view of plans as sets of “situation \rightarrow action” rules has been advocated by several authors as a most useful perspective when dealing with situated agents (Suchman, 1987), (Schoppers, 1987), (Payton et al., 1990). In our experiment, Flakey was initially at the lower left intersection in the environment shown in the picture. The ultimate navigation goal was to guide Flakey to Room-5.

The following behaviors and contexts constitute a plan to reach this goal:

IF $near(Obstacle)$	THEN Keep-Off
IF $\neg near(Obstacle) \wedge at(Corr2) \wedge \neg at(Corr1)$	THEN Follow(Corr2)
IF $\neg near(Obstacle) \wedge at(Corr1) \wedge \neg near(Door5)$	THEN Follow(Corr1)
IF $\neg near(Obstacle) \wedge near(Door5)$	THEN Cross(Door5)

where we have explicitly indicated the descriptors employed by the goal-achieving behaviors, like in Follow(Corr1). Intuitively, the robot must Follow Corr-2 until it reaches Corr-1; then Follow Corr-1 until it is in the neighborhood of Door-5; and finally Cross this door. The Keep-Off behavior may be activated at any moment to avoid unexpected obstacles (the *Og* descriptor is an ‘occupancy grid’ used for sensor-based obstacle avoidance). Note that Follow(Corr1) and Follow(Corr2) denote two distinct behaviors that share the same rules but differ in the descriptor they use.

We have shown elsewhere (Saffiotti et al., 1995) that plans of this type can be generated by classical AI planning techniques. For example, the above plan has

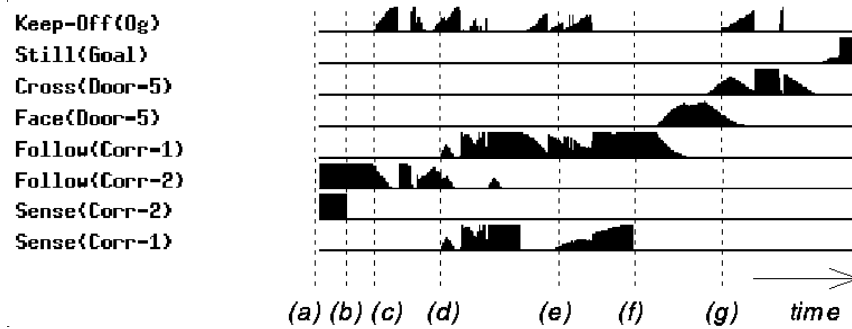


Figure 5.10 Behavior activations during the execution shown in Figure 5.9.

been automatically generated by Flakey from knowledge of the navigation goal of the plan by a simple goal-regression planner employing a sparse topological map annotated with approximate metric information as the source of prior environmental information.⁶

Figure 5.9 shows an actual execution of this plan, while Figure 5.10 plots the temporal evolution of the level of activation of each behavior—that is, of the truth level of the corresponding context. Total execution time was approximately 120 sec at maximum speeds of 400 mm/sec. Each behavior is activated when and to the extent by which its contextual conditions are verified. Although the plan does not specify a pre-determined order of behavior execution, this sequencing may be implicit in the definitions of the contexts of applicability as execution of one behavior may be necessary to reach situations where another might be activated as is the case in the vicinity of (d) and (g).

It is important to remark that behaviors do not “terminate” in the usual sense, being simply deactivated when their context become false and being possibly reactivated if this context becomes true again. This particular aspect of our approach is exemplified by the Sense behavior, which helps Flakey to detect walls by preferring straight and slow motions. This behavior is activated when Flakey enters a new corridor, and remains operational until the walls are anchored (a, d); the Sense behavior may be reactivated later on if anchoring is lost, e.g., if the walls are occluded by obstacles for too long (e).

Finally, note the interaction between purposeful behaviors and reactive obstacle avoidance after (c) and around (d), (e), and (g). In particular, the interaction immediately following (g) deserves further comment. The Cross behavior relies on prior information about the position of the door. In our experiment, this prior knowledge was rather imprecise and the robot was required to activate the Keep-Off behavior to avoid colliding with the edge of the wall. The combined effect of the blending of these two behaviors was to lead Flakey into the opening that is “more or less at the estimated position,” correcting its motion to find the true entrance. The elasticity of fuzzy rules thus guarantees

a smooth degradation of performance when the prior knowledge employed to characterize the descriptors is partly incorrect.

5.7 DISCUSSION

The majority of the current implementations of behavior-based architectures for mobile robot control use on-off schemas to combine behaviors with behaviors being enabled or inhibited based on some a priori hierarchy. Furthermore, only one behavior at a time has control over effectors. The limitations of this approach have been noticed by several authors who have advocated a weighted form of combination where the outputs of different behaviors are merged and the overall control is the result of a tradeoff computation (Maes, 1989; Payton et al., 1990; Arkin, 1998).

Our approach offers a solution to the behavior combination problem that is based on two distinguishing principles:

1. Behaviors are viewed as modules that express goal-specific preferences as to the suitability of control actions rather than as low-level procedures that compete for the control of the robot's effectors.
2. Behavior blending relies on a general context-dependent mechanism based on sound interpretations (Ruspini, 1991a) of fuzzy-logic concepts and structures rather than on ad-hoc forms of combination.

One of the major advantages of weighted combination with respect to crisp activation-inhibition is a better integration between reactive and goal-achieving behaviors (Payton et al., 1990; Saffioti, 1997a). Weighted combination permits to consider the preferences of purposive behaviors during reactive maneuvers thus biasing the control choices toward the achievement of explicit goals.

It is interesting to compare context-dependent blending with the *artificial potential field* methods, first introduced by Khatib (Khatib, 1986) and now extensively employed in motion planning and plan execution (Latombe, 1991; Arkin, 1998). In this approach, vector fields are employed to model the nature of desired interactions between robots and environmental features.

For example, obstacles are represented as generators of repulsive fields that keep the robot away from collisions. Correspondingly, attractive fields are utilized to steer the robot towards desirable locations in its workspace. At each point, the robot responds to a pseudo-force proportional to the gradient of the field. Multiple potential fields are combined by linear superposition of these vectors thus leading to a result based on blending of the "most desirable" motion choice for each goal given the current situation. In contrast, when combining two desirability functions, we *first* combine the component desirability functions, effectively forming a full preference function, choosing *then* the preferred control by defuzzification of the blended function (see Fig. 5.6). In this respect, our combination schema is close to the one proposed by Payton and Rosenblatt (Payton et al., 1990). That schema, however, is an ad-hoc mechanism which is not grounded in any mathematical formalism.

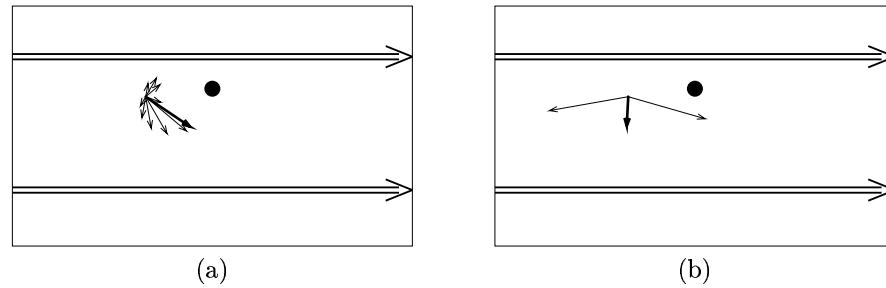


Figure 5.11 Combination of desirability functions (a) and of pseudo-forces (b).

The difference between potential-field and desirability-function combination is illustrated in Fig. 5.11. The left side (a) of the figure illustrates the result of blending, with equal weights, the two desirability functions shown in Fig. 5.1 above; the thick arrow indicates the preferred control obtained by applying centroid defuzzification to the blended function. On the right side (b), it is shown the vector composition of the two preferred controls that would be individually selected by separate consideration of each desirability function. These vectors may also be thought of as pseudo-forces generated by the Follow-Corridor and by the Keep-off behaviors, respectively. As it appears, the two methods of composition may lead to different results. It is intuitively clear that desirability functions carry more information than pseudo-forces since they also measure the desirability of suboptimal controls.

The second important problem that we have also addressed is that of relating control-level processing with the high-level abstract deliberative tasks required by plan production and evaluation. Our approach in this regard relies on two major conceptual tools:

1. *Object descriptors* employed to guide the activity of basic behaviors towards attainment of abstract goals, specified by higher-level modules and anchored to low-level perceptual data.
2. *Context-dependent activation*, in which the conditions that render a particular behavior applicable are expressed by formulae of fuzzy logic.

Fuzzy-logic methods have been of essential importance to take advantage of these conceptual devices. The elasticity inherent in fuzzy formalisms facilitates, for instance, the utilization of object descriptors even when there is not an exact correspondence between the position of the descriptor and the one of the object that it describes. Reliance on the language of fuzzy logic permits, on the other hand, to define complex activation conditions more expressively than is the case for alternative approaches. Furthermore, the ability to integrate abstract controller descriptions and to specify patterns of behavior combination as logical formulae facilitates integration with classical symbolic planning techniques.

Our fuzzy-logic based approach to behavior combination has been shown to be amenable to a formal description and analysis (Saffiotti et al., 1995). We have been able to derive formal results, for example, showing that, under certain assumptions, two behaviors that promote independent goals may be combined by context-dependent techniques to produce a blended behavior that promotes a combination of those goals. Our approach leads, therefore, to sound formalisms for the study of controller properties.

It is also clear that our approach, like similar treatments, has its share of methodological problems, which we are currently seeking to understand and overcome. Fuzzy behaviors, for example, have been so far based primarily on local “greedy” descent gradient methods, which are highly reactive and simple to compute. As is the case with other local techniques this approach to behavior definition cannot guarantee that the device will not be trapped on local minima or infinite loops. More elaborate formulations should emphasize integration with techniques such as dynamic programming that are based on a more thorough analysis of the eventual consequences of control actions. Our treatment of this type of problems has relied, so far, on higher-level deliberation processes only to limit behavior instantiation and combination to those cases where local techniques are appropriate (e.g., on the basis of a coarser global analysis). We have also relied on execution-monitoring approaches to detect local minima and failure conditions.

Our experience has also shown that the process of tuning the parameters of fuzzy rules may be rather difficult. Although the ability to decompose complex behaviors into simpler ones facilitates the identification of rules, we have found that some behaviors—such as obstacle avoidance—demanded several days of experimental debugging. We are currently studying methods for the automatic synthesis of behaviors from specifications and the possible role of learning techniques to generate behaviors as well as to improve behavioral performance.

Finally, much still remains to be done to develop effective analytical formalisms that permit to determine the degree by which composite behaviors attain a variety of control objectives.

5.8 CONCLUSIONS

Fuzzy logic methods have been proved to be effective tools to design highly responsive controllers for autonomous mobile agents. These controllers are capable of implementing motion and perception behaviors so as to attain multiple, possibly conflicting, goals. Fuzzy-logic approaches have also been useful as the bases of formal results that facilitate the analysis of two important problems inherent in behavior-based approaches: the coordination of multiple behaviors and the incorporation of prior knowledge into the controller.

Our experiments have shown that fuzzy behaviors are able to operate relying only on approximate maps and imprecise sensing: a most important requirement for an autonomous vehicle intended for use in unstructured environments. In related work (Saffiotti and Wesley, 1996), we have also shown how fuzzy logic may be used to address the self-localization problem in robot navigation. The

major problems that we have found with our approach concern the empirical nature of the definition and tuning of the fuzzy rules. We have hinted above at some of the directions that we are exploring to mitigate these problems.

The focus of this chapter was on the application of fuzzy logic techniques to the design and implementation of basic navigation behaviors, and to the combination of basic behaviors to form complex behaviors to execute full navigational plans. While these issues are pivotal to any autonomous navigation capability, there are many other important problems in autonomous robotics for which solutions based on fuzzy logic can be, and have been sought. These include the perception and modelling of the robot's environment; the ability to self-localize with respect to a given (partial) map; the ability to recognize and recover from failures due to unexpected environmental changes or hardware failures; and the ability to learn from experience. The interested reader is addressed to (Saffiotti, 1997b; Hoffmann, 1998; Driankov and Saffiotti, 1999) for some up-to-date surveys of these applications.

Acknowledgments. Work by the first author was partly supported by the BELON project founded by the *Communauté Française de Belgique*. Enrique H. Ruspini was partially supported by the U.S. Air Force Office of Scientific Research under Contract No. F49620-91-C-0060. Additional support was provided by SRI International. The authors benefitted from discussions with H. Berenji, P. Bonissone, D. Driankov, N. Helft, O. Khatib, J. Lowrance, K. Myers, D. Ruspini, L. Valverde, and L. Zadeh.

Notes

1. This example clearly oversimplifies matters for the sake of clarity. A more realistic treatment should consider more factors, e.g., the velocity obtained by applying c .
2. This approach to the definition of a desirability measure is known as *predictive fuzzy control* (Yasunoby and Miyamoto, 1985).
3. These operators are those actually used in our experiments on the robot Flakey. See (Saffiotti et al., 1995) for a generalization of our treatment to include arbitrary continuous triangular norms, triangular co-norms, and generalized negation operators (Weber, 1983).
4. This characterization is valid for the inverse of the minimum T-norm. Other T-norms have inverses that lead to a more gradual relaxation of the inadmissibility of control values outside the core of the context.
5. A similar approach to the treatment of multi-objective control problems utilizing fuzzy-logic methods has been previously proposed by (Berenji et al., 1990). Context-dependent blending generalizes and extends that approach by allowing dynamic modification of the degrees of importance of each goal.
6. The actual plan has more behaviors (Saffiotti et al., 1995), including some for perceptual actions such as the *Sense* behavior discussed below, and more complex contexts than the simplified account given in the text.

References

- Arbib, M. A. (1981). Perceptual structures and distributed motor control. In Brooks, V., editor, *Handbook of Physiology – The Nervous System II*, pages 1449–1465. American Physiological Society, Bethesda, MD.

- Arkin, R. C. (1990a). The impact of cybernetics on the design of a mobile robot system: a case study. *IEEE Trans. on Systems, Man, and Cybernetics*, 20(6):1245–1257.
- Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press, Cambridge, MA.
- Berenji, H., Chen, Y.-Y., Lee, C.-C., Jang, J.-S., and Murugesan, S. (1990). A hierarchical approach to designing approximate reasoning-based controllers for dynamic physical systems. In *Proc. of the Conf. on Uncertainty in Artif. Intell.*, pages 362–369, Cambridge, MA.
- Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23.
- Congdon, C., Huber, M., Kortenkamp, D., Konolige, K., Myers, K., Ruspini, E. H., and Saffiotti, A. (1993). CARMEL vs. Flakey: A comparison of two winners. *AI Magazine*, 14(1):49–57.
- Driankov, D. and Saffiotti, A., editors (1999). *Fuzzy logic techniques for autonomous vehicle navigation*. LNCS. Springer, Berlin, DE. Forthcoming.
- Dubois, D. and Prade, H. (1985). A review of fuzzy set aggregation connectives. *Information Sciences*, 36:85–121.
- Gat, E. (1992). Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Proc. of the AAAI Conf.*, pages 809–815, San Jose, CA.
- Hoffmann, F. (1998). Soft computing techniques for the design of mobile robot behaviours. *Information Sciences*. To appear.
- Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98.
- Konolige, K., Myers, K., Ruspini, E., and Saffiotti, A. (1997). The Saphira architecture: A design for autonomy. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1):215–235.
- Latombe, J. C. (1991). *Robot Motion Planning*. Kluwer Academic, Boston, MA.
- Maes, P. (1989). The dynamics of action selection. In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, pages 991–997, Detroit, MI.
- Payton, D. W., Rosenblatt, J. K., and Keirse, D. M. (1990). Plan guided reaction. *IEEE Trans. on Systems, Man, and Cybernetics*, 20(6):1370–1382.
- Ruspini, E. H. (1990). Fuzzy logic in the Flakey robot. In *Proc. of the Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA)*, pages 767–770, Iizuka, JP.
- Ruspini, E. H. (1991a). On the semantics of fuzzy logic. *Int. J. of Approximate Reasoning*, 5:45–88.
- Ruspini, E. H. (1991b). Truth as utility: A conceptual synthesis. In *Proc. of the Conf. on Uncertainty in Artif. Intell.*, pages 316–322, Los Angeles, CA.
- Saffiotti, A. (1994). Pick-up what? In Bäckström, C. and Sandewall, E., editors, *Current Trends in AI Planning — Proc. of EWSP '93*, pages 166–177. IOS Press, Amsterdam, NL.
- Saffiotti, A. (1997a). Fuzzy logic in autonomous robotics: behavior coordination. In *Proc. of the 6th IEEE Int. Conf. on Fuzzy Systems*, pages 573–578, Barcelona, Spain. IEEE Press.

- Saffiotti, A. (1997b). The uses of fuzzy logic for autonomous robot navigation. *Soft Computing*, 1(4):180–197. On-line at <http://iridia.ulb.ac.be/FLAR/>.
- Saffiotti, A., Konolige, K., and Ruspini, E. H. (1995). A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526.
- Saffiotti, A., Ruspini, E. H., and Konolige, K. (1993). Blending reactivity and goal-directedness in a fuzzy controller. In *Proc. of the 2nd IEEE Int. Conf. on Fuzzy Systems*, pages 134–139, San Francisco, California. IEEE Press.
- Saffiotti, A. and Wesley, L. P. (1996). Perception-based self-localization using fuzzy locations. In Dorst, L., van Lambalgen, M., and Voorbraak, F., editors, *Reasoning with Uncertainty in Robotics*, number 1093 in LNAI, pages 368–385. Springer-Verlag, Berlin, DE.
- Schoppers, M. J. (1987). Universal plans for reactive robots in unpredictable environments. In *Proc. of the Int. Joint Conf. on Art. Int.*, pages 1039–1046.
- Suchman, L. (1987). *Plans and situated actions: the problem of human machine communication*. Cambridge University Press, Cambridge, MA.
- Sugeno, M. and Nishida, M. (1985). Fuzzy control of model car. *Fuzzy Sets and Systems*, 16:103–113.
- Weber, S. (1983). A general concept of fuzzy connectives, negations and implications based on t-norms and t-conorms. *Fuzzy sets and systems*, 11:115–134.
- Yasunoby, S. and Miyamoto, S. (1985). Automatic train operation by predictive fuzzy control. In Sugeno, M., editor, *Industrial Applications of Fuzzy Control*, pages 1–18. North-Holland, Amsterdam, NL.
- Yen, J. and Pfluger, N. (1992). A fuzzy logic based robot navigation system. In *Proc. of the AAAI Fall Symp. on Mobile Robot Navigation*, pages 195–199, Boston, MA.
- Zadeh, L. A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28.