Adaptive Hierarchy of Distributed Fuzzy Control: Application to Behavior Control of Rovers

To my parents and siblings who instilled confidence in me and showered me with respect; to my wife Jan and our children who remain a source of purpose and inspiration; and to my extended family whose pride has been a source of strength.

Acknowledgements

First and foremost I would like to acknowledge the spiritual and mental support of my wife for creating an environment conducive to the often difficult combination of family and academic lifestyles.

My gratitude goes out to my advisor, Professor Mohammad Jamshidi, for his valuable guidance, a professional relationship that has bore many fruit, and his confidence in my capabilities as a student and a researcher which he never ceased to reveal to myself and others. I acknowledge the intellectual input provided by other members of my distinguished Dissertation Committee including Dr. Nasir Ahmed, Associate Provost for Research (formerly Dean of Engineering and Chairman of Electrical and Computer Engineering), Dr. Gregory P. Starr, Professor and Graduate Advisor of Mechanical Engineering, Dr. Stephanie Forrest, Professor of Computer Science, and finally Dr. Paolo Fiorini, Member of Technical Staff of the Automation and Control Section at the NASA Jet Propulsion Laboratory (JPL). I am also grateful to my classmates and colleagues at UNM, the NASA Center for Autonomous Control Engineering, and NASA JPL who, perhaps unknowingly, contributed to the shaping of my ideas through meaningful discussions, both technical and otherwise. Excellent research assistance provided by Harrison Danny, Tanya Lippincott, and Fabian Lopez made it possible to generate meaningful experimental results. I am grateful to NASA for having the vision and insight to support the education of myself and other members of ethnic groups that are currently under-represented in the engineering and science professions. Such investments in people and their respective talents are at the core of what is necessary to insure that America remains at the forefront of developments and advancements in science and technology that benefit mankind.

I would also like to extend most sincere thanks to Dr. Yvonne B. Freeman, Provost and Vice President of Academic Affairs, Clark Atlanta University (formerly NASA Associate Administrator and, before that, Manager of the Minority Science and Engineering Initiatives Office at JPL). Dr. Freeman has supported and inspired me as a professional mentor and friend. Her influence has contributed to the will, discipline, and self-confidence that was necessary to achieve this goal. Finally, a special thanks is extended to my friends on the staff of the Minority Science and Engineering Initiatives Office at the Jet Propulsion Laboratory for their support and encouragement throughout my tenure as a Ph.D. student.

Adaptive Hierarchy of Distributed Fuzzy Control: Application to Behavior Control of Rovers

Adaptive Hierarchy of Distributed Fuzzy Control: Application to Behavior Control of Rovers

by

Edward W. Tunstel, Jr.

B.S., Mechanical Engineering, Howard University, 1986

M.E., Mechanical Engineering, Howard University, 1989

Ph.D. Electrical Engineering, University of New Mexico, 1996

Abstract

This dissertation addresses the synthesis of knowledge-based controllers for complex autonomous systems that interact with the real world. A fuzzy logic rule-based architecture is developed for intelligent control of dynamic systems possessing a significant degree of autonomy. It represents a novel approach to controller synthesis which incorporates fuzzy control theory into the framework of behavior-based control. The controller intelligence is distributed amongst a number of individual fuzzy logic controllers and systems arranged in a hierarchical structure such that system behavior at any given level is a function of behavior at the level(s) below. This structure addresses the combinatorial problem associated with large rule-base cardinality, as the totality of rules in the system are not processed during any control cycle. A method of computationally evolving fuzzy rule-bases is also introduced. It is based on the genetic programming paradigm of evolutionary computation and directly manipulates linguistic terminology of the system. This provides a systematic rule-base design method which is more

direct than current approaches that mandate numerical encoding/decoding of rule representations. Finally, a mechanism for multi-rule-base coordination is devised by generalization of fuzzy logic theoretic concepts. It is incorporated to endow the system with the capability to dynamically adapt its control policy in response to goals, internal system state, and perception of the environment.

The validity and practical utility of the approach is verified by application to autonomous navigation control of wheeled mobile robots, or rovers. Simulated and experimental navigation results produced by the adaptive hierarchy of distributed fuzzy control are reported. Results show that the proposed ideas can be useful for realization of autonomous rovers that are meant to be deployed in dynamic and possibly unstructured environments. This class of computer-controlled, wheeled mobile vehicles includes industrial mobile robots, automated guided vehicles, office or hospital robots, and in some cases natural terrain vehicles such as planetary rovers.

The proposed intelligent control architecture is generally applicable to autonomous systems whose overall behavior can be decomposed into a bottom-up hierarchy of increased behavioral complexity, or a decentralized structure of multiple rule-bases.

Contents

1 Introduction **1**

1.1 Challenges for Complex Fuzzy System Control 3

1.2 Facing the Challenge 5

1.2.1 Contributions 6

1.2.2 Roadmap 7

2 Theory and Principles of Fuzzy Control **9**

2.1 Mathematical Concepts 10

2.2 Operational Aspects of Fuzzy Reasoning 12

2.2.1 Fuzzification 15

2.2.2 Inference 16

2.2.3 Defuzzification 17

2.3 Rule-bases and Fuzzy Rules 20

2.3.1 Monolithic Rule Structures 21

2.3.2 Hierarchical Rule Structures 22

2.4 Behavior-based Control and Fuzzy Logic 23

3 Genetic Programming of Fuzzy Rule-Based Systems 25

3.1 Genetic Programming 26

3.2 Systematic Design of Fuzzy Controllers 27

3.3 Related Research 29

3.4 Genetic Programming for Rover Path Tracking 32

3.4.1 Rover steering control problem 32

3.4.2 GP Fuzzy Functions and Terminals 35

3.5 Syntactic Constraints and Structure-preserving Operators 36

3.6 Results and Discussion 40

3.6.1 Improved tracking and mean GP performance 44

3.6.2 Results with modified fitness measure 47

3.7 Summary and Conclusions 49

4 Adaptive Hierarchy of Distributed Fuzzy Control 52

4.1 Theoretical Extensions 54

4.1.1 Applicability-based Decision-making 57

4.1.2 Multiple Rule-base Coordination and Conflict Resolution 59

4.2 Behavior Modulation Theory 61

4.3 Issues of Stability Analysis 64

4.3.1 Supervisory control 64

4.3.2 Direct control 65

4.4 Conclusion 67

5 Fuzzy Behavior Control Systems **69**

5.1 Some Practical Concerns 70

5.2 Behavior-based Mobile Robot Control 73

5.2.1 Fuzzy-behaviors 73

5.2.2 Synthesis 75

5.3 Behaviors in the Adaptive Hierarchy 76

5.3.1 Composite behaviors 77

5.3.2 Primitive behaviors 78

5.3.3 Sensory fusion 80

5.3.4 An alternative example 81

5.4 Coordination by Behavior Modulation 82

5.5 Ethological Influences and Relationships 86

5.6 Conclusion 88

6 Navigation Simulation and Experiment **90**

6.1 Simulated Navigation Results 92

6.1.1 Goal-seeking 92

6.1.2 Effect of t-conorm on motion decisions 96

6.1.3 Route-following 97

6.2 Evolution of Intelligent Behavior Modulation 99

6.2.1 Steady-State GP 101

6.2.2 Behavior fitness evaluation 101

6.2.3 Evolved behavior modulation 103

6.3 Real World Experiments: Goal-seeking 106

6.4 Conclusion 109

7 CONCLUSIONS **112**

A Genetic Programming **131**

Procedure 131

Genetic operators 132

Closure and Sufficiency 134

List of Figures

2.1 Typical membership functions used in fuzzy control. 12

2.2 Canonical fuzzy control system configuration. 13

3.1 Control and error variables associated with a desired rover path. 33

3.2 Rover kinematic error categories. 34

3.3 Membership functions and hand-derived rule-base. 34

3.4 Rule-base tree satisfying syntactic constraints. 37

3.5 Performance comparison of rover path tracking: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle, (d) phase portrait of GP-evolved FLC. 43

3.6 Rover path tracking with 21 rules: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle, (d) standardized fitness curves. 45

3.7 Mean performance of GP with state-error norm as fitness. 46

3.8 Rover path tracking performance comparison using control effort as a cost: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle. 48

4.1 Possible hierarchy for turning behavior. 53

4.2 Hierarchical fuzzy behavior control architecture. 54

4.3 Intelligent supervisory control configuration. 55

4.4 Fuzzy behavior hierarchy. 55

4.5 Fuzzy primitive behavior. 57

4.6 Fuzzy coordination of primitive behaviors. 60

5.1 Hierarchical decomposition of mobile robot behavior. 76

5.2 Fuzzy decision and fuzzy control modules. 77

5.3 Sensory fusion operation. 81

5.4 Hypothetical behavior hierarchy for planetary rover navigation. 82

5.5 Partial behavior conflict. 84

5.6 Full behavior conflict. 85

6.1 UNM LOBOt. 91

6.2 Hierarchical decomposition of mobile robot behavior. 92

6.3 Simulation of goal-seeking behavior. 93

6.4 Behavior modulation and interaction during goal-seeking. 95

6.5 Goal-seeking using different t-conorms. 97

6.6 Route-following using waypoints. 98

6.7 Behavior modulation during route-following. 99

6.8 Example fitness cases. 102

6.9 Behavior fitness case scoring function. 103

6.10 Mean performance of GP and SSGP evolution. 104

6.11 Hand-derived coordination and behavior modulation. 105

6.12 SSGP-evolved coordination and behavior modulation. 105

6.13 Experiment: Short goal-seeking task. 108

6.14 Experiment: Behavior modulation during short goal-seeking task. 109

6.15 Experiment: Long goal-seeking task. 110

6.16 Experiment: Behavior modulation during long goal-seeking task. 111

List of Tables

6.1 Best evolved composite `goal-seek` behaviors. 104

Glossary

behavior-based control: distributed and decentralized control implemented using a collection of special-purpose task-achieving modules that execute concurrently.

behavior modulation: the autonomous act of regulating, adjusting or adapting the activation level of a behavior to the proper degree in response to a context, situation, or state perceived by an autonomous agent.

complex: too difficult or impractical for analysis using conventional quantitative techniques.

Degree of Applicability: a linguistic measure of the instantaneous level of activation of a behavior, expressed as a scalar in the closed unit interval, which determines the amount of influence the behavior will have on the control action corresponding to the situation prevailing during the current control cycle.

ethology: the scientific study of animal behavior patterns.

hand-derived: obtained based on expert knowledge, intuition, trial-and-error, or other non-automatic method.

soft computing: refers to intelligent systems methodologies which employ fuzzy logic, neural networks, probabilistic reasoning, evolutionary algorithms, or synergistic combinations of these.

Chapter 1

Introduction

Automatic control of complex dynamical systems is truly one of the greatest engineering challenges of today. Indeed the impact of this statement can only be assessed relative to current challenges in systems and control, as well as the present state-of-the-art. That is, the statement was perhaps also valid a half-century ago when "complex" control problems were among the class of problems we now consider mundane. It was about that time when classical control theory was formulated using frequency domain methods. From the 1930s to the 1960s, control engineers witnessed the development of optimal control theory and state-space analysis. As the "complexity" of systems continued to increase, it became necessary to consider questions of uncertainty which were responded to by developments in stochastic optimal control. This was followed by progressions made in the late 1960s in robust and adaptive control theory. And this progressive sophistication in control engineering continues today, ever driven by increasing technological demands and accelerated developments in computer technology.

Today, computer and electronics technology is pushing the envelope toward complex dynamic systems of increased *autonomy*. The demand for autonomy brings with it a host of requirements and control specifications which are not adequately addressed by conventional

control methods. To address the control needs of autonomous systems, yet another rise in the level of sophistication of control techniques is required. This brings us to the era of *intelligent control* [1]. An aim of intelligent control research is to develop autonomous systems that can dynamically interact with the real world. The notion of intelligence in systems dictates the need for cognitive, or knowledge-based control. System autonomy implies that the systems of interest are to be self-contained. That is, they must be capable of sensing the real world, reasoning about the real world, and physically influencing the real world. In this context, "real world" is meant to refer to physical environments that may be unstructured, unpredictable, dynamic, noisy, and/or unknown. As such, the real world presents an immense level of uncertainty which complicates the endeavor of developing and controlling autonomous systems.

In attempts to formulate approaches that can handle real-world uncertainty, researchers are frequently faced with the necessity to consider tradeoffs between developing complex cognitive systems that are difficult to control, or adopting a host of assumptions that lead to simplified models which are not sufficiently representative of the system or the real world. The latter option is a popular one which often enables the formulation of viable control laws. However, these control laws are typically valid only for systems that comply with the imposed assumptions, and furthermore, only in neighborhoods of some nominal state. Control laws can only be as accurate as the models they are based on. The option that involves complex systems has been less prevalent due to the lack of analytical methods that can adequately handle uncertainty and concisely represent knowledge in practical control systems. Recent research and applications employing non-analytical methods of soft computing such as fuzzy logic, evolutionary computation, and neural networks have demonstrated the utility and potential of these paradigms for intelligent control of complex systems [2, 3, 4, 5, 6, 7]. In particular, fuzzy logic has proven to be a convenient tool for handling real-world uncertainty and knowledge representation [8].

Fuzzy logic *control* is one of the more active areas of application of fuzzy logic and the

underlying fuzzy set theory introduced by Zadeh [9, 10]. A fuzzy logic controller (FLC) is an intelligent control system that smoothly interpolates between rules. In autonomous systems, tasks are generally performed based on evaluation of sensor data according to a set of rules/heuristics furnished by a human expert who has learned them from experience or training. More often than not, these rules are not crisp (based on binary logic), i.e. some common-sense reasoning or judgemental decision-making is necessary. The class of such problems can be addressed by a set of fuzzy variables and rules which, if suitably formulated, can be used to make expert decisions that approximate human reasoning. As pointed out by Lee [11], fuzzy logic controllers provide a means of transforming a linguistic control strategy that is based on expert knowledge into an automatic control strategy. The approach is very useful for handling problems that are too complex for analysis using conventional quantitative (analytical) techniques or when the available sources of information provide only qualitative, approximate, or uncertain data.

1.1 Challenges for Complex Fuzzy System Control

Most of the fuzzy controllers applied to industrial products and reported in the research literature utilize the monolithic rule-base structure, i.e. a single set of fuzzy rules. During any given control cycle all rules in the fuzzy rule-base are processed. Clearly the cardinality (total number of rules) of the fuzzy rule-base has a direct influence on the real-time performance of a fuzzy control system. This presents no problems for real-time control of systems requiring a relatively small number of rules (e.g. less than twenty). However, for more complex systems that require a significantly larger number of rules, this FLC architecture reveals a limitation in the form of degradations in real-time performance. This is a major concern for control of systems for which intelligence can be distributed throughout hierarchical or decentralized structures.

Examples of such systems are autonomous robotic agents, corporate decision-making entities, social systems, electric power systems, and other large-scale systems [12] in general. Raju et al [13, 14] have shown that for the conventional FLC, the cardinality of the fuzzy rule-base which increases exponentially with the number of system variables (inputs) can increase linearly if the rules are structured as a set of hierarchical expert levels. Jamshidi [12] has proposed a combined hierarchical-sensory fusion scheme characterized by a piecewise linear constant propagation of rules as a function of system variables. Thus, it is possible to overcome this source of computational complexity and facilitate practical implementations of complex FLCs by employing alternative hierarchical rule structures. One such hierarchical fuzzy approach is introduced in this dissertation.

Systematic design of FLCs in the absence of an expert, or sufficient knowledge of the problem domain, is currently an open problem. Various successful approaches that use *soft computing* methods have addressed this design problem. In many of the proposed approaches to automatically generating rule sets for FLCs, it is necessary to encode the linguistic rules as a numerical representation and subsequently decode them into the appropriate linguistic terminology of the problem. A more direct approach is proposed here — genetic programming[1] of fuzzy systems. Genetic programming eliminates numerical encoding/decoding of rule sets; it directly manipulates the linguistic terminology of the fuzzy system.

The monolithic rule-bases employed in many fuzzy control systems represent static nonlinear mappings from input to control output. Additional flexibility beyond the bounds of a particular nonlinear mapping is necessary for systems of significant autonomy. This issue has been addressed in some monolithic adaptive fuzzy controllers by an adaptive law for adjusting structural and linguistic parameters (membership functions and/or rules) of the system. Such adjustments effectively alter the nonlinear mapping of the controller. In multiple-rule-based

[1] For an overview of genetic programming, please see the Appendix.

controllers, the adaptive law can become unwieldy due to the greater number of nonlinearities that must be considered. A different approach is taken here in which adaptation is achieved by controlling interactions between multiple rule-bases such that an appropriate dynamic nonlinear mapping from situations to actions is achieved.

1.2 Facing the Challenge

Fuzzy logic is a powerful tool for use in control of dynamic systems. Proven advantages are robustness in the presence of system and external perturbations, ease of design and implementation, and efficiency of knowledge represention for systems of continuous variables [15]. The conventional FLC has been successfully used in a number of industrial plants and processes, and because of its advantages, it is sometimes the more favorable controller even when classical controllers (e.g. PID and its variants) are applicable. However, there are some limitations regarding the use of the FLC for more complex systems than those addressed to date. To reiterate, the limitations are: the potentially negative effect of large rule-bases on real-time performance, the enigma regarding FLC design in the absence of sufficient domain knowledge, and the lack of adaptation.

The purpose of this dissertation is to address these issues by advancing the state-of-the-art regarding synthesis of fuzzy controllers for complex distributed intelligent systems represented as hierarchical or decentralized structures. An intelligent fuzzy control architecture is developed and proposed in the chapters that follow. The controller intelligence is distributed amongst a number of individual fuzzy logic controllers and decision systems arranged in a hierarchical structure such that system behavior at any given level is a function of behavior at the level(s) below. Additional architectural structure is imposed by forming a hybrid between fuzzy logic control and behavior-based control, which is a product of artificial intelligence research in mo-

bile robotics. The practicality of the new approach is demonstrated by implementation of the proposed architecture on simulated and physical mobile robots and experiments with autonomous navigation in dynamic and non-engineered environments. An autonomous mobile robot, or rover, is a sufficiently complex plant for testing the validity of the approach. Development of such systems is important for automating activities in a variety of operating domains ranging from industrial environments to outer space. Examples include office settings, hospitals, factories, natural terrain, planetary surfaces, and environments deemed hazardous for humans. The scope of application is broad. However, the exposition provided here is written with applications to dynamical systems of the electro-mechanical, or mechatronic, variety in mind.

1.2.1 Contributions

During the course of developing theoretical and practical aspects of research presented here, several contributions have been made which advance the state-of-the-art in fuzzy controller synthesis. We list the contributions here.

A hierarchical structure that accommodates multivariable systems by distributing intelligence among multiple rule-bases.

A computational mechanism which provides adaptability to fuzzy control systems via multi-rule-base coordination.

An automatic approach to fuzzy rule-base design.

Based on these contributions, we will demonstrate how the current approach to fuzzy control can be extended to effectively deal with multivariable systems which require many rules. In

addition, a constrained syntactic structure is introduced which enables genetic programming to be used to evolve intelligent control rules for rover tracking and behavior coordination problems.

The research results have particular relevance to autonomous rover navigation research being conducted by NASA at the Jet Propulsion Laboratory in Pasadena, California. At JPL, behavior-based control schemes have been applied to planetary microrover navigation since 1990 [16, 17, 18, 19, 20]. This dissertation provides a slightly different approach that exploits the strengths of fuzzy logic for handling uncertainty in unstructured environments. This is an essential capability for planetary surface exploration by microrovers.

1.2.2 Roadmap

In Chapter 2 the terminology of fuzzy system theory which will be used throughout this dissertation is described. The relevant mathematics of fuzzy control systems, operational aspects of fuzzy reasoning, and rule structures are also covered. Chapter 3 demonstrates an approach to automatic discovery of fuzzy logic rule-bases based on artificial evolution. In particular, genetic programming is employed for systematic design of fuzzy controllers. These two early chapters lay the ideological foundation for the remainder of the research.

The main contribution is presented in Chapter 4 where theoretical extensions to conventional fuzzy control are introduced. The essential ingredients of the adaptive hierarchy of distributed fuzzy control are covered. Behavioral concepts which are natural by-products of incorporating fuzzy logic into the framework of behavior-based control are described. Finally, comments on stability analysis are given. Implications of applying the new approach to behavior control synthesis are discussed in Chapter 5. A behavior hierarchy for autonomous navigation is described, as well as the mechanisms responsible for adaptive behavior. Chapter 6 reports simulated and experimental results that verify the validity and practical utility

of the new approach in this problem domain. In addition, genetic programming is revisited through applications to evolution of high-level behaviors. The dissertation is concluded by Chapter 7.

Chapter 2

Theory and Principles of Fuzzy Control

Fuzzy control is one of the more active areas of application of fuzzy logic and the underlying fuzzy set theory introduced by Zadeh [9]. Fuzzy logic controllers are intelligent control systems that smoothly interpolate between rules, i.e. rules fire to continuous degrees and the multiple resultant actions are combined into an interpolated result. The capability of providing efficient control while processing uncertain information is the basis for fuzzy logic control. As a means of approximating or capturing the essence of human thinking, fuzzy logic is more flexible than traditional logic which is based on classical set theory. Fuzzy control theory provides a mechanism for incorporating human-like reasoning capabilities computationally in control systems. The candidate systems for fuzzy logic control can be characterized as systems that possess complex or unmodeled dynamics, high dimensionality, many interacting variables, system perturbations, or a combination of any of these.

At the heart of a fuzzy logic controller is a rule-base consisting of *if-then* rules. These are similar in form to production system rules of expert systems, however antecedents (*if*-part)

and consequents (*then*-part) of a fuzzy rule are fuzzy propositions expressed using fuzzy sets and linguistic variables. These and related terminology which will be used throughout this dissertation are described in this chapter.

2.1 Mathematical Concepts

A fuzzy set may be characterized by a mathematical formulation known as the *membership function*. This function assigns a numerical degree or grade of membership to a crisp (precise) number. More precisely, over a given universe of discourse X, the membership function of a fuzzy set $\tilde{A}$, denoted by $\mu_{\tilde{A}}(x)$, maps elements $x \in X$ into a numerical value in the closed unit interval, i.e.

$$\mu_{\tilde{A}}(x) : X \rightarrow [0, 1]. \tag{2.1}$$

Note that a membership function is a so-called *possibility* function and not a probability function. A fundamental distinction between the two is that, unlike probabilities, possibilities are not required to sum to one, nor does the integral of a possibility function have to equal one. Thorough expositions on the distinctions between possibility and probability can be found in [21]. In the context of control system applications, membership values are measures of causality in an input-output mapping. Within this framework, a membership value of zero corresponds to an element which is definitely not a member of the fuzzy set, while a value of one corresponds to the case where an element is definitely a member of the set [5]. Partial fuzzy set membership is indicated by intermediate membership values. Thus, a fuzzy set is a generalization of the notion of a classical set which takes on only two possible membership (Boolean) values — {0,1}, {FALSE, TRUE}, etc. The fuzzy set, $\tilde{A}$, can be represented as a crisp set of ordered pairs of $x \in X$ and $\mu_{\tilde{A}}(x)$, i.e.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}. \tag{2.2}$$

Alternative and commonly used notations for fuzzy sets are

$$\tilde{A} = \sum_{x \in X} \mu_{\tilde{A}}(x)/x \tag{2.3}$$

if X is a discrete or countable universe of discourse, and

$$\tilde{A} = \int_{x \in X} \mu_{\tilde{A}}(x)/x \tag{2.4}$$

if X is continuous or uncountable. In these equations, the function of each operator is different from its usual meaning in mathematics. In particular, $\sum$ indicates a countable enumeration rather than summation, $\int$ indicates an uncountable enumeration rather than integration, and / indicates an ordered pair rather than division. Throughout this dissertation notation (2.3) is used most since the subject is control systems which are invariably realized on digital computers. For all practical purposes, all fuzzy sets dealt with here are countable and discrete.

Typical membership functions used to express uncertainty in the system variables of fuzzy logic control systems take on triangular and trapezoidal shapes (or variants thereof) given by Equations (2.5) and (2.6), respectively, where a, b, c, and $d \in \Re$.

$$\mu_{tri}(x) = \begin{cases} 0 & ; \quad x < a \\ \frac{(x-a)}{(b-a)} & ; \quad a \leq x \leq b \\ \frac{(a-x)}{(b-a)} & ; \quad b \leq x \leq c \\ 0 & ; \quad x > c. \end{cases} \tag{2.5}$$

$$\mu_{trap}(x) = \begin{cases} 0 & ; \quad x < a \\ \frac{(x-a)}{(b-a)} & ; \quad a \leq x < b \\ 1 & ; \quad b \leq x \leq c \\ \frac{(c-x)}{(d-c)} & ; \quad c < x \leq d \\ 0 & ; \quad x > d. \end{cases} \tag{2.6}$$

These are illustrated in Figure 2.1. Smoother nonlinear variants of these typical membership

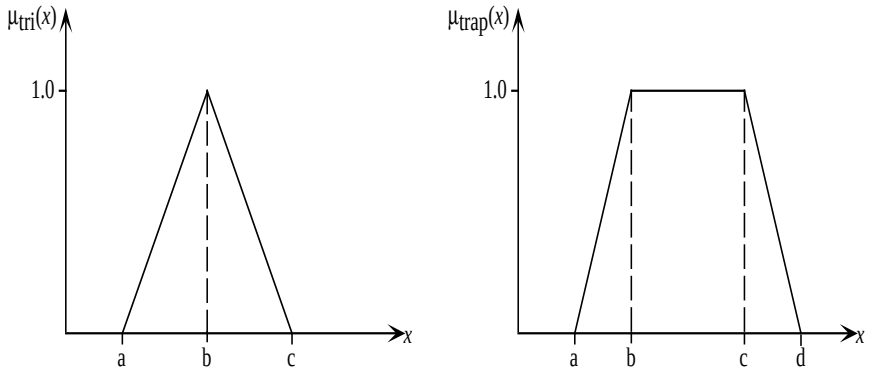


Figure 2.1: Typical membership functions used in fuzzy control.

function shapes have also been proposed. They are used primarily in fuzzy logic and find less practical utility in fuzzy control systems. This is due to the fact that minimal performance gains are achieved with the added expense of more complex mathematical operations that must be performed by a real-time fuzzy control algorithm. Piecewise linear functions such as Equations (2.5) and (2.6) are evaluated faster and more efficiently by digital computers and microcontrollers used in embedded applications.

A linguistic variable is a system variable whose definition can not be sufficiently specified using crisp sets. It takes on values that are words or sentences in a natural or artificial language which convey some ambiguous notion amenable to definition using fuzzy set theoretic concepts. Associated with a given linguistic variable (e.g. *speed*) are linguistic values, or fuzzy subsets (e.g. *slow*, *fast*, etc) expressed as membership functions which convey any uncertainty, vagueness, or imprecision of values of the linguistic variable.

2.2 Operational Aspects of Fuzzy Reasoning

The basic structure of a canonical FLC system architecture is depicted in Figure 2.2. The distinction of this architecture from those of classical linear feedback control systems is that

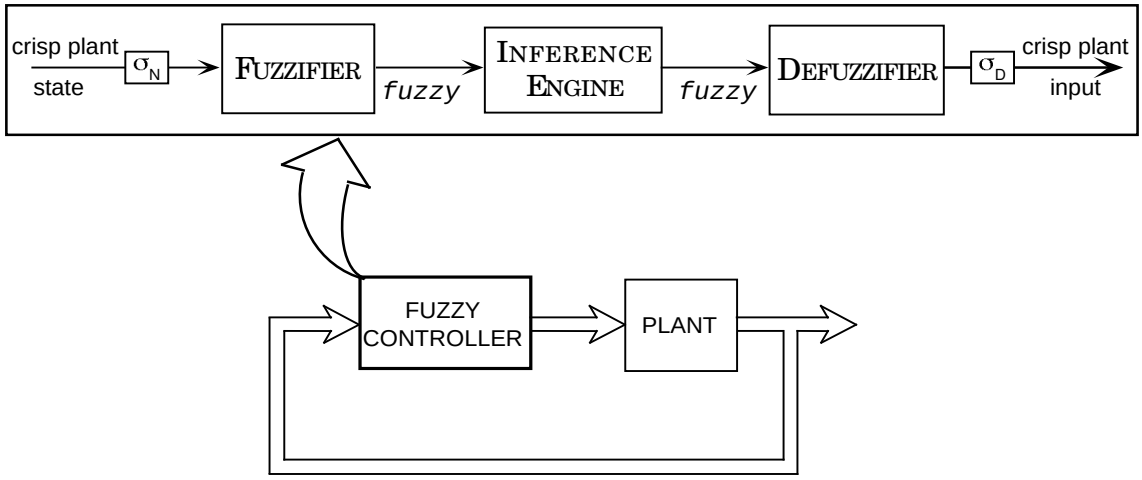


Figure 2.2: Canonical fuzzy control system configuration.

the controller block performs a nonlinear mapping from plant state or output information into plant control action(s). This mapping is characterized by a transformation of information from physical space to an abstract (fuzzy) or normalized space where decisions are made, and from this abstract space back to physical space. The symbols, σ_N and σ_D, represent normalization and denormalization scale factors respectively. Such transformations are common in soft computing techniques including neural networks and probabilistic reasoning systems based on Bayesian inference. It is interesting to note a conceptual analog in the general areas of signals and systems, where time domain data is transformed into the frequency domain, operated upon, and transformed back to the time domain using Fourier, Laplace, or z-transforms. Just as in these conventional approaches, the information to be processed in fuzzy systems is easier to handle in the intermediate abstract space.

Implementation of a fuzzy controller requires assigning membership functions for both inputs and outputs by partitioning the respective universes of discourse using fuzzy subsets. With knowledge of the membership functions in place, the FLC performs three primary operations: 1) *fuzzification* of input variables, 2) *inference* via a set of fuzzy rules that map fuzzy inputs

to fuzzy outputs, and 3) *defuzzification* of aggregated fuzzy outputs. The inference engine is responsible for fuzzy reasoning and corresponds to the abstract space mentioned in the analogy just given. Fuzzification (defuzzification) can be viewed as a transform (inverse transform) between crisp and fuzzy information. The introduction of some terminology at this point will be useful in discussions that follow. Recall that fuzzy set theory is a generalization of classical set theory. As such, it provides set operations which generalize classical set intersection and union. These belong to a class of fuzzy set aggregation operators called *triangular norms* and *triangular co-norms*[1] (t-norms and t-conorms), respectively, and are used to express fuzzy logical statements or propositions. Note that intersection and union correspond respectively to conjunction and disjunction in fuzzy (and crisp) logical propositions. The t-norms most commonly used in fuzzy control are the arithmetic minimum and algebraic product operations on membership values of fuzzy sets. Common t-conorms are arithmetic maximum and bounded-sum. Therefore, the intersection of two fuzzy sets, $\tilde{A}$ and $\tilde{B}$, with membership functions $\mu_{\tilde{A}}$ and $\mu_{\tilde{B}}$ is a fuzzy set $\tilde{C} = \tilde{A} \cap \tilde{B}$ whose membership function can be computed using one of the following expressions

$$\mu_{\tilde{C}} = \mu_{\tilde{A}\cap\tilde{B}}(x) = min(\mu_{\tilde{A}}, \mu_{\tilde{B}}) \tag{2.7}$$

$$\mu_{\tilde{C}} = \mu_{\tilde{A}\cap\tilde{B}}(x) = \mu_{\tilde{A}} \cdot \mu_{\tilde{B}} \tag{2.8}$$

Similarly, the union $\tilde{C} = \tilde{A} \cup \tilde{B}$ can be computed using one of the following expressions

$$\mu_{\tilde{C}} = \mu_{\tilde{A}\cup\tilde{B}}(x) = max(\mu_{\tilde{A}}, \mu_{\tilde{B}}) \tag{2.9}$$

$$\mu_{\tilde{C}} = \mu_{\tilde{A}\cup\tilde{B}}(x) = min(1, \mu_{\tilde{A}} + \mu_{\tilde{B}}) \tag{2.10}$$

The bounded-sum t-conorm, Equation (2.10), ensures that the result of a fuzzy set union is a normal (maximum height of 1) fuzzy set.

[1] Also referred to as *s-norms*

2.2.1 Fuzzification

Inputs to a fuzzy controller, $\mathbf{x}$, are usually measured variables associated with the state of the controlled plant. Since the inference engine processes fuzzy quantities, the inputs must first be assigned membership values (fuzzified), i.e. they must be transformed into a fuzzy domain to yield $\mu(\mathbf{x})$. This is achieved by evaluating the membership functions of each linguistic input variable with the crisp value as an argument. In many instances, the crisp value is first scaled into a normalized universe of discourse just prior to evaluation by membership functions.

As mentioned earlier, the heart of the controller inference engine is a set of if-then rules whose antecedents and consequents are made up of linguistic variables and associated fuzzy membership functions. If X and U are input and output universes of discourse for a rule-base of size N, the generic fuzzy if-then rule takes the following form

$$IF\ x\ is\ \tilde{A}_i\ THEN\ u\ is\ \tilde{B}_i \tag{2.11}$$

where x and u represent FLC input and output fuzzy linguistic variables, respectively, and $\tilde{A}_i$ and $\tilde{B}_i$ $(i = 1...N)$ are fuzzy sets representing linguistic values of x and u. Such rules define a causal relationship between the plant state and its control inputs [22]. That is, *IF* the current value of state x is a member of $\tilde{A}_i$, *THEN* this is a cause for control input u to take on values in $\tilde{B}_i$. In general, the rule antecedent consisting of the proposition "x is $\tilde{A}_i$" could be replaced by a compound fuzzy proposition consisting of a conjunction (and/or disjunction) of similar propositions. Similarly, the rule consequent "u is $\tilde{B}_i$" could include additional FLC output propositions. For each rule, state variables, x_j, of the plant are matched against fuzzy sets, $\tilde{A}_{ij}$, in each proposition of the antecedent yielding $\mu_{\tilde{A}_{ij}}(x_j)$. A degree of match is computed as the *rule strength*, $\rho_i \in [0, 1]$, by a conjunctive operation over all rule antecedents. For a rule with n propositions in the antecedent, the rule firing strength can be computed as

$$\rho_i = \min_j \mu_{\tilde{A}_{ij}}(x_j)\ ;\ j = 1, 2, \ldots, n \tag{2.12}$$

where the *min* t-norm (Equation (2.7)) has been used, or as

$$\rho_i = \prod_{j=1}^{n} \mu_{\tilde{A}_{ij}}(x_j) \tag{2.13}$$

using Equation (2.8).

2.2.2 Inference

Once the rule firing strength is determined, fuzzy subsets in the rule consequent are modified by fuzzy *implication* resulting in a possibility distribution (a fuzzy set) for the output of the rule. For a rule with m consequents the rule outputs are given by

$$\tilde{u}_i(u) = min(\rho_i, \tilde{B}_i) \tag{2.14}$$

or

$$\mu_{\tilde{u}_{ik}}(u_k) = min(\rho_i, \mu_{\tilde{B}_{ik}}(u_k)) \ ; \ k = 1, 2, \ldots, m. \tag{2.15}$$

In the fuzzy control literature Equation (2.14) is known as the Mamdani implication. An alternative implication, based on Larsen's product rule, can be used as well. It is given by

$$\tilde{u}_i(u) = \rho_i \cdot \tilde{B}_i \tag{2.16}$$

or

$$\mu_{\tilde{u}_{ik}}(u_k) = \rho_i \cdot \mu_{\tilde{B}_{ik}}(u_k) \ ; \ k = 1, 2, \ldots, m. \tag{2.17}$$

Many possible implication functions exist. From a practical point of view, Mamdani implication and Larsen's implication have been found to be well suited for approximate reasoning in fuzzy control due to their computational simplicity [23].

The inference engine computes an output fuzzy set, $\tilde{U} \in U$, representing the aggregated contribution (according to (2.9)) of all rules in the rule-base. This rule-base output is expressed

as

$$\tilde{U} = \bigcup_{i=1}^{N} \tilde{u}_i(u). \tag{2.18}$$

When $\tilde{u}_i(u)$ is determined using Mamdani implication, Equation (2.18) is called *max-min* inference. If Larsen's implication is used, Equation (2.18) is called *max-product* inference. In the research reported in this dissertation, max-product inference is used to compute rule-base outputs, and the arithmetic sum is used to aggregate fuzzy outputs from multiple rule-bases.

2.2.3 Defuzzification

The output of a fuzzy controller serves as the control input to the plant. In practical control systems plant inputs must be crisp values. Therefore, the the aggregated fuzzy output set resulting from rule-based inference must be defuzzified to yield a single real number output that serves as the control input signal for the plant. The output fuzzy set can be viewed as a possibility distribution over a range of crisp outputs. This range of outputs includes all control inputs recommended as desirable by the rule-base given the current input. Defuzzification determines the best value among the possibilities using a suitable functional expression. Two commonly used defuzzification formuli are the *Center-of-Area* and *Center-of-Sums* defuzzification methods. Each is a computation conceptually tailored after the centroid/center-of-gravity formula for a distributed load in two dimensions — a fundamental concept of engineering mechanics.

Let $U = \{u_1, u_2, \ldots, u_r\}$ be a discrete universe for the FLC output (plant input), and let u^* be the crisp control input. Then the general formula for defuzzification of a rule-base output fuzzy set, $\tilde{U}$, determines u^* as

$$u^* = \frac{\sum_{l=1}^{r} u_l \cdot \mu_{\tilde{U}}(u_l)}{\sum_{l=1}^{r} \mu_{\tilde{U}}(u_l)} \tag{2.19}$$

where, in our analogy, the numerator represents the resultant moment of the output fuzzy set (about the lower bound of U) and the denominator represents the area under the fuzzy set.

For Center-of-Area defuzzification the terms, $\mu_{\tilde{U}}(u_l)$, are computed as

$$\mu_{\tilde{U}}(u_l) = \max_i \mu_{\tilde{u}_i}(u_l), \tag{2.20}$$

and for Center-of-Sums defuzzification

$$\mu_{\tilde{U}}(u_l) = \sum_{i=1}^{N} \mu_{\tilde{u}_i}(u_l). \tag{2.21}$$

Given the size, and upper and lower bounds on any such U, the defuzzification can be computed efficiently using the theorem introduced below; it holds independent of whether the discrete output fuzzy set is computed using Equation (2.20) or Equation (2.21).

Theorem 2.1 *Let U_I be a finite universe of discourse defined over the closed interval, $I = [a, b]; a < b$, and discretized with resolution, δ_r, over equally spaced units. A discrete output fuzzy set, $\tilde{U}$, defined over U_I which expresses membership grades of a sequence, $u_l \in U_I \ni u_l = (a + l\delta_r), l = 0, 1, \ldots, r-1$, can be defuzzified to yield a crisp value, u^*, using the following* **shift defuzzification** *formula:*

$$u^* = a + \delta_r \frac{\sum_{l=0}^{r-1} l \cdot \mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)}, \quad \textit{where } \delta_r = \frac{b-a}{r-1}.$$

Proof: Let $\mu_{\tilde{U}}(u_l)$ denote the membership grades (function) of the elements $u_l \in U_I$. Then discrete defuzzification yields the crisp output u^* as

$$u^* = \frac{\sum_{l=0}^{r-1} u_l \cdot \mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)}.$$

Substituting $u_l = (a + l\delta_r)$ and expanding leads to

$$\begin{aligned} u^* &= \frac{\sum_{l=0}^{r-1} (a + l\delta_r)\mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)} \\ &= a\frac{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)} + \delta_r \frac{\sum_{l=0}^{r-1} l \cdot \mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)} \end{aligned}$$

or finally,

$$u^* = a + \delta_r \frac{\sum_{l=0}^{r-1} l \cdot \mu_{\tilde{U}}(u_l)}{\sum_{l=0}^{r-1} \mu_{\tilde{U}}(u_l)}. \qquad \blacksquare$$

Thus according to the theorem, a crisp fuzzy controller output, defined over a finite universe, can be computed from its associated discrete fuzzy output by defuzzifying over the discrete support (i.e. $\{u \in U | \mu_{\tilde{U}}(u) > 0\}$) of the fuzzy output, scaling by the resolution of the universe, and shifting by the lower bound of the universe. If input scaling or normalization is used by the FLC, a corresponding denormalization must be applied to the result of defuzzification to determine a control input in its appropriate non-normalized universe.

There is a practical and a philosophical issue to consider regarding the choice of defuzzification formuli. In practice, and in particular for real-time control applications, it is often desirable to minimize computation time. In what has been described thus far regarding computational aspects of FLCs, the defuzzification process is by far the most intensive. Therefore, it is a popular item to begin with when attempting to optimize a system for real-time control. The Center-of-Sums method is the faster of the two and is frequently chosen for real-time fuzzy control. Now, during the inference process each rule suggests a fuzzy control action ($\tilde{u}_i$ for the i-th rule). In the construction of $\tilde{U}$, the overall rule-base output, these individual fuzzy sets generally overlap one another. When Center-of-Area defuzzification is employed, any overlapping regions of two or more rule outputs are counted only once due to the *max* operation in Equation (2.20). If a majority of the rules suggested outputs in a common overlapping region, the persistence of these suggestions is diminished by the Center-of-Area method. As the number of rules in the rule-base increases, more information is lost. In the same situation, the Center-of-Sums method would sum (Equation (2.21)) outputs in the overlapping region, thus reflecting the persistence for outputs in that region. Philosophically, the latter method is closer in effect to the analogy of a distributed load. Moments from all loads in overlapping regions would be included in the centroid calculation. If a rule-base is meant to perform weighted (in the physical sense of the word) decision-making then the Center-of-Sums defuzzification seems more appropriate. The choice depends on the desired effect. As will be seen in later

chapters, Center-of-Sums defuzzification is employed for weighted decision-making amongst multiple rule-bases.

More detailed introductions to fuzzy control, fuzzy set operations, and the concepts of fuzzification, inference, aggregation, and defuzzification can be found in [5, 11, 22, 23, 24].

2.3 Rule-bases and Fuzzy Rules

Controllers that are based on fuzzy rule-based systems can be configured in a number of ways. The alternatives are governed by issues such as the fuzzy set resolution selected for system variables and the complexity of decision-making, or reasoning, demanded by the task environment. The fuzzy set resolution of the system variables (inputs) determines the total number of rules (i.e. the rule-base cardinality) necessary to cover all possible combinations of fuzzy controller inputs. Individual rule outputs for a given rule-base contribute to the shaping of a control/decision surface, the nonlinearity of which is a measure of the decision-making complexity. Thus, resolution of the state space and nonlinearity of the control surface are inter-related with regard to the interpolation necessary to produce desired behavior via approximate reasoning.

A rule-base that considers contingencies throughout the entire state space (given the fuzzy sets of the system variables) is said to be *complete*. The cardinality of a complete rule-base is given by the expression

$$R_c = \prod_{i=1}^{n} L_i \tag{2.22}$$

where n is the number of controller inputs and L_i is the number of linguistic values (e.g. *near*, *far*, etc) defined for the i-th input, i.e. the size of the i-th term set. When all n inputs have an

equal number of linguistic values, l, the cardinality grows exponentially, i.e.

$$L_i = l,\ \forall i \Rightarrow R_c = l^n$$

If R_c is relatively small (say, ≤ 20) then it is feasible to realize the fuzzy controller as a monolithic, or single-rule-base controller. Otherwise, alternatives such as hierarchical rule structures may be in order.

2.3.1 Monolithic Rule Structures

Most of the fuzzy controllers applied to industrial products and reported in the research literature use the monolithic rule-base structure. All of the precepts that govern the desired behavior of the system are encapsulated as a single collection of if-then rules. In most instances, the rule-base is designed to carry out a single control policy or behavioral goal. It is also possible to implement additional control policies within a single rule-base due to the fusion of information and conflict resolution achieved by aggregation and defuzzification processes respectively. Nevertheless, it becomes more difficult to design monolithic rule-bases that implement global behavior in pursuit of multiple interacting goals.

In order for an autonomous system such as a rover, or mobile robot, to operate in dynamic environments, it must be capable of achieving multiple goals whose relative priorities may change with time. Therefore, its controller should be designed such that a number of task-achieving *behaviors* can be realized and integrated to achieve different control objectives. This requires the formulation of a large and complex set of fuzzy rules. In this situation a potential limitation to the utility of the monolithic fuzzy controller becomes apparent. During any given control cycle all rules are sequentially processed. Clearly, the cardinality of the rule-base has a direct influence on real-time performance. As alluded to above, this presents no problem for real-time control of systems governed by relatively small rule-bases. However, more elaborate

controllers implemented as monolithic rule structures can potentially suffer from degraded real-time performance.

2.3.2 Hierarchical Rule Structures

We have seen that the cardinality of monolithic rule-bases increases exponentially with the number of system variables. An alternative rule structure was proposed by Raju et al [13, 14] that represents a rule-base as a hierarchy of rules for which the cardinality increases only linearly with the number of system variables. Koczy and Hirota [25] produced similar results by introducing hierarchically structured rules based on fuzzy partitions of the state space. A third approach is based on combining sensory fusion with the hierarchical rule structure [12, 26]. When it is possible to fuse some system variables (e.g. as a set of linear combinations [27]) such that a reduced set of inputs can be fed to the fuzzy controller, further reductions in overall rule-base cardinality can be achieved by applying the hierarchical structure to this reduced set. In fact, the cardinality increases in a piecewise-linear fashion and at a significantly slower rate than the linear increase reported in [13]. Hierarchical rule structures have also been proposed for fuzzy control of dynamic systems with interacting goals [28]. Thus, it is possible to overcome the aforementioned limitation of the monolithic rule structure and realize practical implementations of more complex behavior by employing hierarchical rule structures. We shall explore this possibility further in Chapter 4 where an approach to hierarchical distributed fuzzy control of system behavior is introduced. First, however, let us use Section 2.4 introduce what is meant here by system behavior and to relate it to fuzzy control.

2.4 Behavior-based Control and Fuzzy Logic

The behavior-based control paradigm emerged from an amalgamation of ideas in ethology, control theory, and artificial intelligence [29, 30, 31]. It has been described as a compromise between extremes of the agent control spectrum — traditional top-down deliberative strategies and purely reactive, bottom-up strategies [32, 33]. The former relies on a centralized world model to determine appropriate sequences of control actions. The latter is based on a collection of simple condition-action rules that map sensor readings into control actions with minimal use of internal models and internal state.

Behaviors, which are the fundamental unit of behavior-based control systems, have been defined in various ways by different researchers. A few of the proposed definitions are:

> A *behavior* is a control law that satisfies a set of constraints to achiev e and maintain a particular goal [33].

> A *behavior* is a trajectory through state space [34].

> A *behavior* is a regularity in the interaction dynamics between the agent and the environment [35]

Each of these definitions is valid; others have been proposed which are similar and sometimes contradictory. This reflects the subjectivity and lack of formal structure that currently prevails in the behavior-based control research community. For our purposes, it suffices to say that the notion of "behavior" in dynamical systems refers to a qualitative assessment of system activity in response to relevant stimuli in a particular problem domain or operating environment. We will refer to a behavior, or fuzzy-behavior, as any task/goal-oriented system response induced

by a combination of system purpose and the perceived state of the environment. Throughout this dissertation the term, "behavior," will be used in this context and is considered synonymous with rule-base, controller, etc.

A behavior-based control system is distinguished by a distributed and decentralized collection of (pseudo) parallel, concurrently active behaviors which achieve distinct tasks. Clever coordination of individual behaviors results in the emergence of more intelligent behavior(s) suitable for dealing with complex situations. Note that behavior-based control as studied herein is not to be confused with the "behavioral framework," a distinct research topic of analytical control theory concerned with modeling and system identification. The behavioral framework is based on Willems' theory of dynamical systems [36] and has not been influenced by ethological or artificial intelligence concepts.

Fuzzy logic control systems and behavior-based control systems share common developmental advantages. Namely, they both require short development times and are intrinsically flexible in their control structure and design. Individual fuzzy rules (and behaviors) can be formulated independently, and additional rules (and behaviors) can easily be added to a control system if necessary without altering or re-designing the existing system. We take advantage of these shared attributes to develop a natural hybrid approach to intelligent control of complex systems. Fuzzy logic theory is incorporated into the framework of behavior-based control by realizing behaviors as fuzzy controllers based on the mathematical and operational concepts described above. Increases in system complexity are handled by a novel approach to hierarchical behavioral decomposition accompanied by a systematic methodology for design of fuzzy rule-based control systems.

Chapter 3

Genetic Programming of Fuzzy Rule-Based Systems

A current research thrust in the area of intelligent control focuses on the development of autonomous systems that can dynamically interact with the real world. *Real world* is meant to refer to physical environments that might be unstructured, unpredictable, dynamic, noisy, and/or unknown. As such, the real world presents an immense level of uncertainty that complicates the endeavor of developing and controlling autonomous systems that are meant to interact with it. Recent research and applications employing non-analytical methods of soft computing including fuzzy logic and evolutionary computation has demonstrated the utility and potential of these paradigms for developing intelligent control systems [3, 5, 7]. Fuzzy logic control and evolutionary computation have proven to be convenient tools for handling real-world uncertainty and knowledge representation, and the design of intelligent control systems, respectively [8, 37]. In this chapter, we present an approach that exploits the combined attributes of these paradigms for the purpose of developing intelligent algorithms for controlling autonomous dynamic systems that interact with the real world. In particular, we apply the

genetic programming paradigm (GP) [7] to the problem of learning/discovering rules for use in a fuzzy rule-based control system.

3.1 Genetic Programming

The genetic programming paradigm computationally simulates the Darwinian evolution process by applying fitness-based selection and genetic operators to a population of parse trees (individuals). Each parse tree represents a computer program of a given programming language, and is a candidate solution to a particular problem. The programs are structured as hierarchical compositions of functions and terminals (arguments to functions) of various sizes and shapes. These individuals participate in a probabilistic evolutionary process wherein the population evolves over time in response to selective pressure induced by the relative fitnesses of the individuals in a particular problem environment. As applied here, each individual is coded as a LISP symbolic expression (S-expression) that implements condition-action statements which collectively serve as a rule-base to be embedded in a fuzzy-logic controller. The GP concept can be implemented in other programming languages as well, at both high and low levels [38, 39]. The approach developed in this chapter calls for the use of a *constrained syntactic structure* [7] for constructing each individual S-expression in the population. This, in turn, requires the definition of syntactic rules of construction and *structure-preserving operators* for breeding the resulting individuals.

The notion is accepted here that humans may not be the best designers of cognitive control systems that involve interactions between constituent parts [40]. At the same time, the issue of just how much design should be imposed and how much should be allowed to evolve is unresolved at this time [41]. It has been observed that the artificial evolution of computer programs may produce deterministic control strategies that have slightly different features than

those produced by humans. The existence of similar building blocks in strategies discovered by artificial evolution and those devised by humans has also been observed [42]. Having considered these, the following questions come to mind. What innovations in control strategies, if any, can we expect from artificial evolution for a given control problem for which a human-derived solution exists? What is the potential of genetic programming for the evolution of fuzzy controllers? In this chapter, results of an attempt to answer the latter question are presented, as its answer is related to that of the former. A particular tracking control problem is used here as the context in which to demonstrate the potential of GP. The problem is to determine a fuzzy linguistic rule-base for steering a rover onto a desired path. GP is used to evolve the rule-base, and its performance is compared to that of a fuzzy controller produced by the author via the usual trial-and-error design approach.

Before describing the control problem in more detail, let us consider motivations for seeking automatic design methods for fuzzy controllers, and why GP was considered as a possible means to that end. In Section 3.3 related research performed to date is recognized. The remainder of the chapter covers details of the GP implementation, results of the steering control application, conclusions and possible improvements to the approach.

3.2 Systematic Design of Fuzzy Controllers

Systematic design of fuzzy controllers in the absence of an expert, or sufficient knowledge of the problem domain, is currently an open problem. The approach often taken is an iterative one of trial-and-error. It typically involves tweaking of membership functions used to express the uncertainty in inputs and outputs, as well as modifications of the fuzzy rule-base. This process, which leads to a fuzzy controller that performs well according to the designer's subjective evaluation, can turn out to be quite lengthy depending on the complexity of the control

problem. This serves to weaken the scalability of one of the strongest attributes of fuzzy logic applications in control — fast development time. Various attempts have been made to address this design issue. These include the determination of fuzzy membership functions and rules by genetic algorithms (GAs) [37, 43, 44], and by learning using neural networks [45, 46]. When using these techniques to determine rule sets for fuzzy systems, it is often necessary to encode the rules in a numerical form suitable for processing by the GA or the neural network, and subsequently decode them into the appropriate linguistic terminology. For the GA, the chromosome representing a rule is typically a string of numerical genes. The alleles of these genes often belong to binary or n-ary alphabets, and/or the set of real or natural numbers, rather than the collection of linguistic variables, fuzzy sets, and fuzzy logic connectives that actually make up the rule. An exception is the representation proposed by Kinzel et al [47] where a chromosome is encoded as a matrix whose elements (alleles) are fuzzy sets. Booker [48] has also suggested ways around the "inadequacies of the binary encodings typically used with classifier systems" (a form of rule-based GA) that give learning classifier systems the ability to represent attributes as expressively as most symbolic systems. Furthermore, for approaches that use the simple GA [49, 50], the fixed-length chromosome restricts each individual to have the same pre-specified number of rules. The contention here is that the genetic programming paradigm offers a more direct approach to fuzzy controller design.

It is worthwhile to stress that the use of GP, rather than GA, as a means to evolve fuzzy rule-bases is a preferential design decision on the part of the author. Shedding some light on the subtle difference(s) between these approaches may provide some insight to the reader interested in applying one approach or the other. GP departs from its predecessor, the simple GA, primarily with regard to its genome[1] representation scheme. Structures undergoing adaptation in GP are executable hierarchical programs of dynamically varying size and structure,

[1]The structure operated upon by genetic operators.

rather than linear numerical strings [7]. Note, however, that tree-structured representations of computer programs are possible using GAs [51]. In the approach developed here for GP evolution of fuzzy rule-bases, the same fuzzy linguistic terms and operators that comprise the genes and chromosome persist in the phenotype[2]. Thus, the use of GP eliminates the need for encoding/decoding of the fuzzy linguistic rule set. Furthermore, the dynamic variability of the representation allows for rule-bases of various sizes and different numbers of rules. This enhances population diversity which is important for the success of the GP system, and any evolutionary algorithm for that matter. The dynamic variability also increases the potential for discovering rule-bases of smaller sizes than necessary for completeness, but sufficient for realizing desired behavior. No claims are made here about the relative performance of GAs versus GP as tools for search, optimization, or learning. After all, "GP is a GA where critical choices have been made to suit its goal of program discovery" [52]. The introduction of GP to the evolutionary computation research community merely provided a new perspective by demonstrating a flexible alternative to the numeric string genome used most GA applications. The advocacy of GP for evolving fuzzy rule-bases is rooted in its convenience of representation as it pertains to fuzzy system design. In the author's view, GP seems to be more appropriate for design of fuzzy rule-bases since it can facilitate the manipulation of linguistic variables directly associated with the problem.

3.3 Related Research

The artificial evolution of rules for systems control has been investigated by a number of researchers. Most have focussed on using models of Darwinian evolution. In recent years the body of related literature has become quite extensive. It is not the intention in this

[2]The target representation which the genome typically maps to.

chapter to provide an inclusive overview, but rather to acknowledge prior and ongoing research that most closely relates to the approach described herein. Interested readers may consult a recently compiled bibliography [53] for a broader overview. In [54], Goldberg demonstrated the effectiveness of classifier systems at learning rules to control position of a simple inertial object and to control a simulated natural gas pipeline. Classifier systems [55] use GAs to learn simple condition-action rules that are represented by fixed-length numerical strings. Like fuzzy rule-bases, classifier systems use parallel rule activation which allows simultaneous coordination of multiple actions. The main distinctions between fuzzy rule-based systems and classifier systems are that the former uses linguistic variables and fuzzy sets in its condition-action rules and does not have the luxury of a learning component. In this dissertation, a learning component is provided using GP which is better suited for manipulating symbolic rule representations. Grefenstette and Schultz have developed the SAMUEL system for learning control rules [56, 57, 58]. The system has proved successful at robot control problems (simulated and actual) as well as simulated control of evasive maneuvers for tactical aircraft. They introduced a restricted high-level rule language (and associated genetic operators) that distinguishes their GA approach from others that are based on the string representation of chromosomes. The resulting approach has strong similarities to classifier systems and the work described here. The fundamental differences lie in our use of GP instead of GA, fuzzy sets instead of crisp sets, and linguistic variables rather than numeric variables. Harvey et al [40, 59] have concentrated on evolving robot behavior using GAs in conjunction with neural networks. They suggest that building controllers by hand becomes prohibitively difficult for increasingly complex behavior. This view is shared by Feldman [44] who has developed a technique that encodes fuzzy control rules as a fuzzy *network*, a connectionist extension to fuzzy linguistic systems. That is, the GA is used to synthesize or modify the rules of the fuzzy network controller. Finally, Kinzel et al [47] deemed it necessary to modify the GA (using the matrix rule-base representation

mentioned earlier) by taking the properties of fuzzy controllers into account to facilitate fast convergence.

As a departure from the Darwinian approach Grefenstette [60] added Lamarckian mechanisms to the SAMUEL [56] system that improve the quality and computational cost of rule learning for control. The main Lamarckian feature is the incorporation of rule strengths that are modified as a direct result of the learning agent's experience. This is implemented using "generalization" and "specialization" operators triggered by specific conditions relating rule strengths and the outcome of the task being learned.

Koza [7] has applied genetic programming to a number of related control problems, namely, the truck backer-upper problem and the evolution of robot subsumption behaviors for wall-following and box-pushing. Shortly after the publication of Koza's text, applications of genetic programming to control problems of the type we focus on here have appeared in the literature. The most notable relation to this work is that of Reynolds [61] who has used GP to evolve corridor following behaviors for a simulated robot in the presence of noise. Similar work has been done by Fraser [62] in evolving multi-agent emergent behaviors, and Handley [63] in mobile robot path planning.

The differences between the current approach and that of SAMUEL [56] has already been pointed out. With the exception of that system and the work on fuzzy controller evolution, all of these applications of evolutionary computation result in evolved controllers that are deterministic computer programs based on binary logic reasoning. Each of the GA implementations that use string representations for chromosomes employ the binary encoding scheme. Each of the GP implementations make use of numeric values as terminals. Thus, the work described herein differs from the related work in either its focus on the evolution of *fuzzy* systems based on approximate reasoning, its use of GP with *linguistic* terminals and fuzzy logic operators as

functions, or both.

3.4 Genetic Programming for Rover Path Tracking

In the genetic programming paradigm, the program search space is contained in the set of all possible S-expressions (LISP symbolic expressions) that can be composed recursively from a set of *functions* and a set of *terminals*. Each function in the function set, F, takes a specified number of arguments. In general, functions may include arithmetic operators, mathematical functions (e.g. sine, cosine, absolute value), Boolean operators, conditionals, etc. Terminals in the terminal set, T, are typically either variables or constant atoms. For the sake of brevity, let us introduce the control problem addressed here before describing the details of the GP implementation. This will allow us to discuss the implementation issues within the context of the problem.

3.4.1 Rover steering control problem

The problem is to find a fuzzy rule-base that will properly steer a rover for path following in the plane. The problem is taken from Hemami [64, 65] where it is formulated for a class of low-speed (less than 2 m/s) tricycle-model vehicles. Hemami derived a state-space model, based on the robot kinematics, where the state vector consisted of measurable position (ε_d) and orientation (ε_θ) errors associated with path following (see Figure 3.1). The steering angle (δ) is the corrective control action that causes the error states to decay to zero, thus forcing the robot to follow the path. The position error is taken as the deviation from the nearest point on the desired path. The orientation error is the angular deviation of the robot's heading from the tangent to the desired path. The rule-base we wish to evolve is for a two-input-one-output fuzzy controller that will map the error states into a steering angle at each time step during

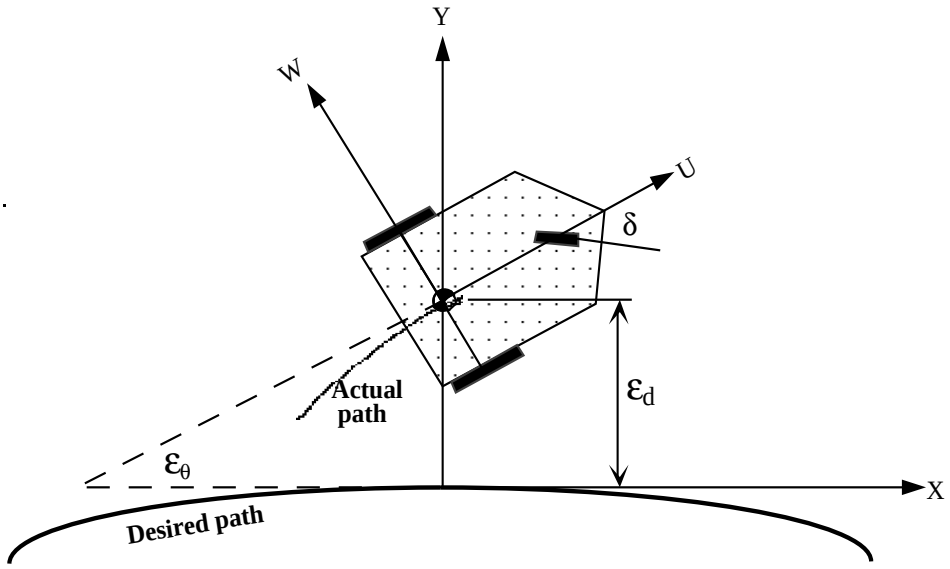


Figure 3.1: Control and error variables associated with a desired rover path.

the robot's attempt to follow a desired path. This is a fundamental motion capability that is often an integral part of more complex behavioral repertoires for autonomous mobile robots [66]. Based on the geometry of the problem as formulated in [64] the position/orientation errors fall into eight different categories that are pair-wise symmetric. Four of these are illustrated in Figure 3.2 (a–d); the remaining categories are symmetric to the four shown.

In this example the effort is focussed on evolving the rule-base and it is assumed that the membership functions are specified *a priori* and are fixed. The membership functions used for the inputs and output of the fuzzy controller are shown in Figure 3.3, along with a rule-base in the form of a fuzzy associative memory table. These are taken from an existing solution hand-derived and refined through trial-and-error by the author. There are five fuzzy sets each for input and output linguistic values. Thus, our hand-derived rule-base of $R_c = 25$ rules is complete, i.e. there is a rule for all combinations of input fuzzy sets taken two at a time. The linguistic notation of Figure 3.3 is as follows: $NB \equiv$ "negative big", $NS \equiv$ "negative small", $Z \equiv$ "zero", $PS \equiv$ "positive small", $PB \equiv$ "positive big" with the lowercase prefixes "p" and

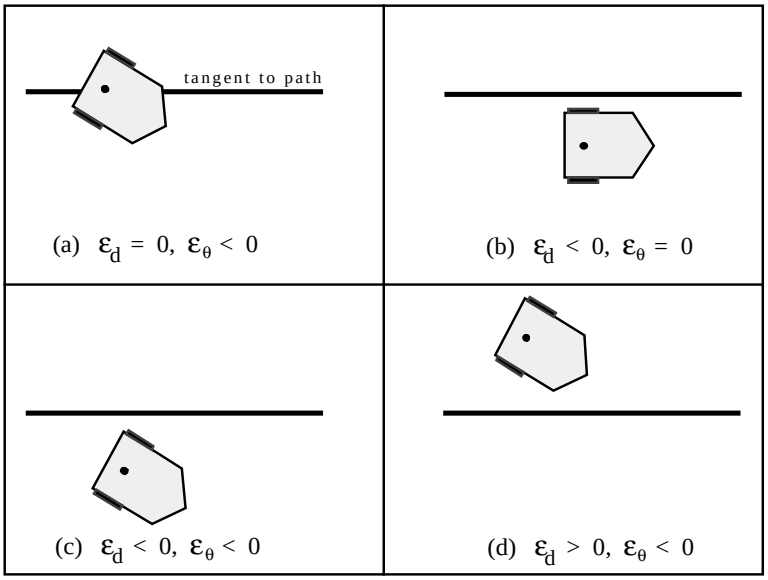
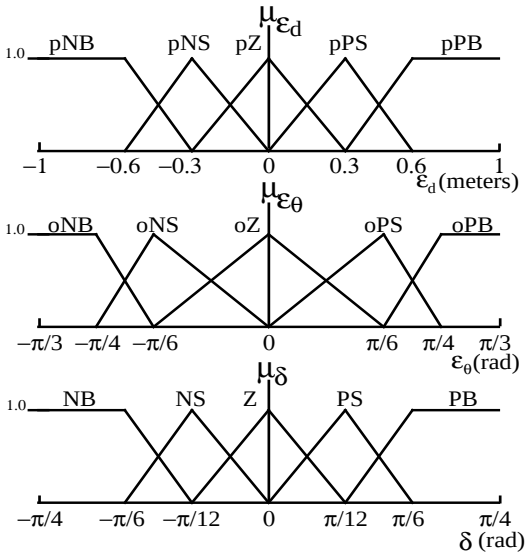


Figure 3.2: Rover kinematic error categories.

"o" designating fuzzy sets for position error and orientation error respectively. Fuzzy sets for the steering angle are labeled without a prefix.



ε_d \ ε_θ	oNB	oNS	oZ	oPS	oPB
pNB	PB	PB	PB	PB	Z
pNS	PB	PS	PS	Z	NS
pZ	PB	PS	Z	NS	NB
pPS	PB	Z	NS	NS	NB
pPB	Z	NB	NB	NB	NB

Figure 3.3: Membership functions and hand-derived rule-base.

3.4.2 GP Fuzzy Functions and Terminals

For the purpose of evolving fuzzy rule-bases (which are essentially programs of fuzzy conditional statements) the following function set was chosen,

$$F = \{\texttt{ANT}, \texttt{CONSQ}, \texttt{f\_AND}, \texttt{IF} - \texttt{THEN}, \texttt{f\_OR}\} \tag{3.1}$$

with `f_OR` (described below) taking a variable number of arguments (equal to the number of rules) and the remaining functions each taking two arguments. The function `ANT` represents a fuzzy proposition in the antecedent of a fuzzy rule. Its arguments are an input linguistic variable and an associated fuzzy membership function. For example, in the proposition, *error is LARGE*, *error* is a linguistic variable and *LARGE* designates a membership function expressing the "meaning" of the current value of *error*. `ANT` returns a numerical value in the closed interval [0, 1] representing the membership value, or degree of truth, of the proposition. Note that if a rule contains only one proposition in its antecedent the membership value represents the rule strength. `CONSQ` is defined in a similar manner for output linguistic variables and fuzzy sets except that it returns the output fuzzy set designated in the rule consequence. The `f_AND` function is simply the fuzzy intersection operator of fuzzy set theory. It performs the conjunction of two or more fuzzy propositions yielding a numerical value for the rule strength. The `f_AND` function can be defined using any t-norm [5]; recall from Chapter 2 that min and *product* are most commonly used in fuzzy control. Here we limit it to the conjunction of two propositions with the idea that conjunctive forms of higher order can be constructed by recursive calls to the function (the level of recursion is bound by a specified maximum depth of the rule tree). In addition, the current implementation restricts `f_AND` to occur only in rule antecedents. Therefore, its two arguments can be return-values of either `ANT` or a recursive call to itself. The function representing a rule is `IF-THEN`. Its first argument is the rule strength returned by either `ANT` or `f_AND`; its second argument is the fuzzy set returned by `CONSQ`.

Finally, the function `f_OR` serves as a fuzzy aggregation operator. It occupies the root node of every tree in the population of rule-bases. Each rule that fires in a fuzzy rule-base returns a fuzzy set as the result of the rule consequence. `f_OR` operates on the output fuzzy sets by taking their fuzzy union to produce a resultant fuzzy set representing the overall output of the rule-base. This overall output fuzzy set is the return-value of an individual rule-base in the population. Consequently, a wrapper (output interface) [7] for S-expressions is the Center-of-Area defuzzification operator which defuzzifies the fuzzy output to yield a real number for the control signal.

The terminal set is made up of the input and output linguistic variables and the corresponding fuzzy sets associated with the problem being solved. For the steering control problem the terminal set is defined as

$$T = \{\varepsilon_d, \varepsilon_\theta, \delta, pNB, pNS, pZ, pPS, pPB, oNB, oNS, oZ, oPS, oPB, \\ NB, NS, Z, PS, PB\} \tag{3.2}$$

Observe that the elements which make up the function and terminal sets are taken from the linguistic terminology of the problem at hand.

3.5 Syntactic Constraints and Structure-preserving Operators

In many genetic programming applications, unrestricted S-expressions are sufficient to solve a problem given a function set and a terminal set that satisfies the closure property [7]. That is, each function in F should be well defined and closed for any combination of arguments that it may encounter (see Appendix for more details). As a result, individuals may have any composition of elements from the combined set, $F \cup T$, occupy the nodes of the tree with the only restriction being that the root must be a function and the leaves must be terminals. This

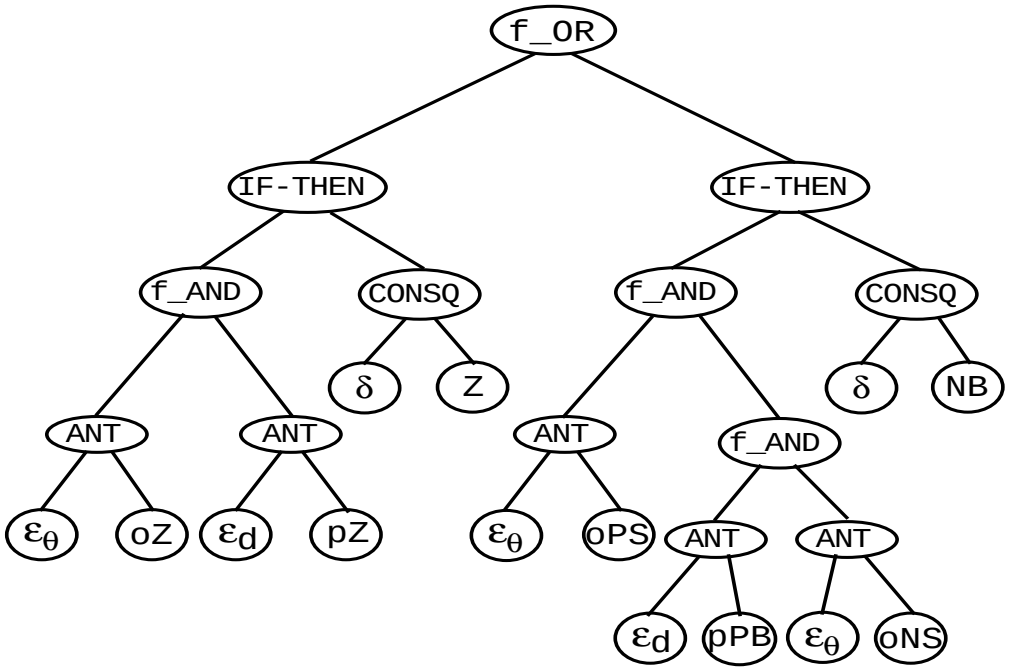


Figure 3.4: Rule-base tree satisfying syntactic constraints.

is not the case here. Instead, strong constraints are imposed on the syntax of a rule-base that are defined by special rules of construction.

A rule-base that could potentially evolve from the designated function set and terminal set can be expressed as a rooted, point-labeled tree with preordered branches. An example of a syntactically valid rule-base of two rules and a depth of five is depicted in Figure 3.4 along with its interpretation as a linguistic rule-base. From the figure one can imagine how arbitrary placement of functions and terminals in this tree could lead to severe syntactic violations. Valid rule-bases must conform to the following syntactic rules of construction:

- `f_OR` must occupy the root of the tree and cannot occur at non-root points.
- Only `IF-THEN` is allowed at the level immediately below the root.

- A left-child of `IF-THEN` can only be `ANT` or `f_AND`.
- A right-child of `IF-THEN` can only be `CONSQ`.
- A child of `f_AND` can be either `ANT` or `f_AND`.
- A child of `ANT` can only be input linguistic variables and input fuzzy sets.
- A child of `CONSQ` can only be output linguistic variables and output fuzzy sets.

Additional ramifications of these syntactic constraints are that full trees are not possible if the number of inputs and outputs is not equal, and extra care must be taken to ensure that linguistic variables are paired with appropriate fuzzy sets as children of `ANT` and `CONSQ` nodes. The minimum depth of a valid rule-tree is three; this corresponds to rules with a single antecedent. The maximum number of antecedents per rule is 2^{d-3}, where d is the maximum permissible depth of the rule tree specified as a control parameter of the GP run. The imposed syntactic structure, and the rules of construction, are similar to those of Koza's application to neural network design [67]; the constraints are stronger here.

All rule-bases in the initial population are randomly created using these rules, but descendant populations are created by the reproduction, crossover, and mutation operators. The offspring of rule-bases modified by crossover and mutation must also conform to the syntactic structure. There are eight types of points (for crossover or mutation) in the rule-base structure — one for each of the 5 functions, points with input linguistic variables, points with output linguistic variables, and points with fuzzy sets. Structure-preserving crossover is achieved by randomly selecting any non-root node as the crossover point in the first parent, and restricting the crossover point in the second parent to be a randomly-selected point of the same type. One exception is that `ANT` and `f_AND` make a valid pair of crossover points provided that one of the resulting offsprings do not violate the preset maximum depth for rule-base trees. The crossover

is completed in the usual way [7] by swapping the subtrees at (and including) the crossover points of the two parents. This crossover operator not only preserves the syntactic structure of the rule-base but it also preserves the context of subtrees, particularly when function nodes are selected as crossover points. This issue of context preservation in GP has been recently addressed by D'haeseleer [68], where he introduces two new crossover operators that provide a more flexible mechanism to decouple the evolution of different branches of an individual tree. Here, context preservation is a necessary by-product of the syntactic constraints imposed by the rule-base structure. Structure-preserving mutation is done by randomly selecting a non-root point in a rule-base tree, discarding the selected point and the nodes below it, and replacing the discarded portion with a randomly-generated (but syntactically valid) subtree at that point. Mutation points are chosen with uniform probability. The effect of mutation is controlled by a parameter that specifies the maximum depth for the randomly-created subtree that replaces the discarded portion. The root node is protected from both crossover and mutation. GP cycles through the current population performing fitness evaluation (as described below) and application of genetic operators to create a new population. The cycle repeats on a generation by generation basis until satisfaction of termination criteria (e.g. lack of improvement, maximum generation reached, or perfect hit percentage). The GP result is the best-fit rule-base that appeared in any generation.

Amidst all of the constraints on syntax and structure of the fuzzy rule-bases, there is room for some flexibility. In the creation of the initial population, the number of rules (number of arguments to `f_OR`) in each rule-base is assigned to be a random integer in the interval, $[R_{min}, R_{max}]$, specified before the run. The value for R_{min} is chosen as a lower bound on the size of a rule-base that the control engineer feels may be sufficient to control the system. The upper bound can be chosen such that $R_{max} \geq R_c$, the number of rules required for a complete rule-base. In the current example [10, 30] is used. As mentioned earlier, this feature of the

implementation is important for ensuring diversity in the population as it allows for rule-bases of different sizes. It also increases the potential for finding a rule-base of smaller size than necessary for completeness (although no selective pressure to evolve minimal rule-bases has been applied here). It is well-known to practitioners of fuzzy control that a number of dynamic systems exist that can be controlled using fewer rules than dictated by the value of R_c for the fuzzy rule-base. An example is the classic inverted pendulum problem for which a fuzzy controller with $R_c = 25$ performs optimally with ten or twelve rules [22]. Finally, it should be noted that some unusual circumstances regarding allowable rules result from the imposed structure. It is possible, for example, for an input linguistic variable to appear more than once in the antecedent of a rule (examine Figure 3.4). In fact, it is also possible for a given fuzzy proposition to have multiple occurrences in a rule (in this case redundant occurrences are deleted if they persist in the final solution). While one could argue that this unnecessarily enlarges the search space, such unusual possibilities are allowed to prevent restricting GP from discovering innovative control strategies that may be counter-intuitive to the human designer, and consequently overlooked.

3.6 Results and Discussion

In this section initial results of GP evolution of fuzzy rule-bases are reported. The GP system was run using small population sizes of 10–20 rule-bases for a number of generations ranging from 9–46. In GP, genetic diversity remains high even for very small populations due to the tree structure of individuals [7]. Twelve GP runs were executed for the steering control problem described above. Results from several representative runs in which the best-of-run rule-base performed well in comparison to the hand-derived rule-base are presented.

The simulated rover is based on Hemami's kinematic model [64] and approximate dimen-

sions are taken from the Hero-1 mobile robot — a 0.3m wheelbase, and a rear-axle-to-front-wheel offset of 0.2 m. In all of the simulations the robot travels at a constant speed of 1.5 m/s. Following Hemami's formulation, it is assumed that the error states are measurable. Thus, the robot has access to the error states (perhaps via odometry or other position sensing) at all times. It is also assumed here that the source of sensory information has practical uncertainties and imprecision associated with it. The simulations are conducted at 20 Hz (i.e. time steps of 0.05 seconds) for a maximum of 5 seconds. Eight fitness cases are used corresponding to initial conditions selected from each of the eight error categories mentioned in Section 3.4.1. The number of fitness cases used is problem dependent. In similar problems solved in [7] not more than twenty fitness cases were used. Ideally, the number should be chosen such that the fitness cases represent the search space sufficiently to allow the evolved control strategy to generalize (i.e. handle unforeseen initial conditions). Given the pairwise-symmetric error categories for the steering control problem, eight is a convenient choice. Moreover, in mobile robot problems involving time-consuming simulation of each fitness case per individual, small numbers of fitness cases and/or small population sizes are often necessary tradeoffs. During the GP evolution process, each rule-base in the current population is evaluated to determine its fitness for steering the robot onto the desired path (i.e. the goal is to force the error state vector to zero). This evaluation is achieved by simulating the robot's motion from each of the eight initial conditions until either the goal state is reached or time expires. The raw fitness of a rule-base is defined as the sum, over the fitness cases, of the Euclidian norms of the error state vector at the end of each fitness case, i.e.

$$Raw fitness = \sum_{i=1}^{8} \sqrt{(\varepsilon_d^2 + \varepsilon_\theta^2)_i} \tag{3.3}$$

Among the measures of fitness used in [7], standardized fitness (i.e. lowest numerical values imply best fit) is predominant particularly in problems for which the objective is to minimize costs such as error. In this problem standardized fitness is equivalent to raw fitness. Thus, a

perfect score is zero and lower raw fitnesses are associated with better rule-bases. In addition to scoring the best fitness, we would like candidate rule-bases to cause the error states to decay to final values within specified tolerances ($|\varepsilon_d| < 0.15$m and $|\varepsilon_\theta| < 0.26$ rad.) for each fitness case. Such an event is referred to as a *hit*. The conditions for a hit are imposed on the simulations as metrics for a successful trial. In other words, a simulation run through a given fitness case is considered successful if the error states decay to values within specified tolerances before the allocated time expires.

The GP control parameters set the maximum depth for rule-base trees in the initial population to six, the maximum depth of mutation subtrees to four, and the maximum rule-base depth after crossover to seven. At each generation, breeding of the population was performed using probabilities of 0.1 for reproduction, 0.5 for crossover (at any valid point), and 0.4 for mutation. The relatively high mutation rate (40%) was chosen as an attempt to compensate for any limitations that might stem from small population size. Tournament selection was used with a tournament size of two, i.e. the best fit of two randomly chosen rule-bases was selected for reproduction. In the run which yielded the best observed result (according to Equation (3.3)), GP discovered the best rule-base after 7 generations. The rule-base had 21 rules, a raw fitness of 1.58 and 8 hits. The hand-derived, complete rule-base had 25 rules and scored a raw fitness of 1.96 and 8 hits. The worst-case raw fitness for a given rule-base is 72.5. This was determined by evaluating Equation (3.3) with the largest possible error states that could accumulate over the duration of a fitness case while traveling at the specified constant speed. Based on this worst-case raw fitness these rule-bases correspond to 98% fitness and 97% fitness respectively.

For performance comparison, Figure 3.5(a–c) graphically illustrates the position error, orientation error, and control effort for the GP-evolved fuzzy controller and the hand-derived fuzzy controller. All results shown are for error category (d) of Figure 3.2 with $\varepsilon_d = 0.8$ m and

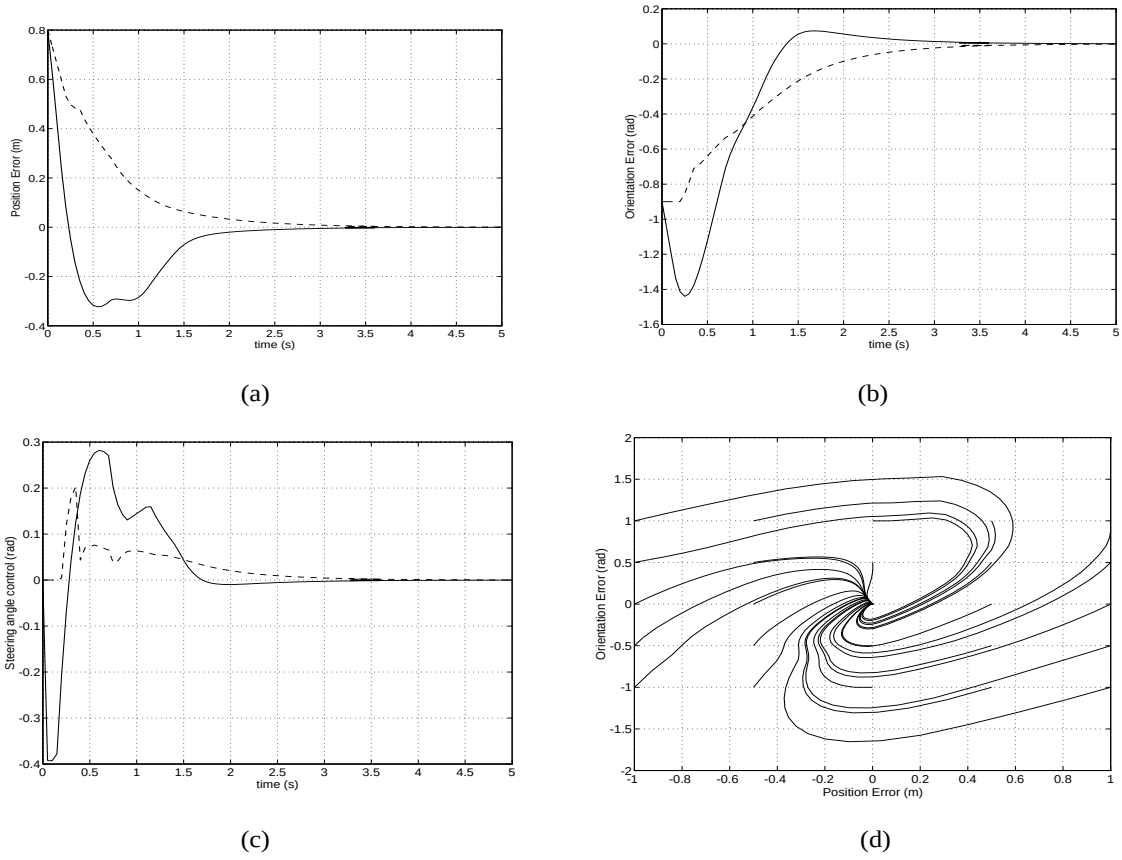


Figure 3.5: Performance comparison of rover path tracking: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle, (d) phase portrait of GP-evolved FLC.

ε_θ = -0.9 rad as initial conditions. Among the eight error categories, category (d) was shown [64] to be the most general for studying path tracking for tricycle-type vehicles. It is most general in the sense that in the process of correcting vehicle steering from initial error states in categories (a–c) (and corresponding symmetric cases), the vehicle error status ultimately reduces to category (d) or its counter-pair. Observing Figure 3.5a we note that the rise time of the GP-evolved controller is fastest and results in an overshoot of the goal by about 0.3 m ($\approx$ 1 ft.) before it hones in on the path about 2 seconds later. Its response time in reaching the goal, however, is practically the same as for the hand-derived controller. The hand-derived controller forces the errors to zero in a smoother manner and without overshoot. However, it

was explicitly designed to exhibit this type of behavior. Recall that the fitness measure used to drive the evolution of the GP-evolved rule-base favors rule-bases that result in small final errors. There was no selective pressure for rule-bases to exhibit smooth response without overshoot. That is, the fitness measure has no component which will penalize overshoot or other undesirable response characteristics for that matter. Nonetheless, the results compare favorably with those of the hand-derived fuzzy controller. Observing the control efforts (Figure 3.5c), we see that the steering angle for the GP-evolved controller spans a wider range of motion, and as a result, expends more energy in achieving the goal. Although the evolved controller learned the steering control rules using only 8 pre-selected fitness cases, it was able to generalize when started from initial conditions throughout the error state space. This is shown in the phase portrait of Figure 3.5d which reveals that the origin is a stable node of the system. In other problems this may not be the case. In such situations it may be necessary to use random initial conditions in each fitness case to avoid overfitting pre-selected initial conditions. It should be noted that in this particular run the evolved rule-base resulted from the anomalous presence of a highly fit rule-base in the initial population whose genetic material persisted in the early generations and was only slightly improved by GP in generation 7. More dramatic evolutionary improvements over the generations were shown in other GP runs.

3.6.1 Improved tracking and mean GP performance

GP runs with different random seeds, population sizes, and numbers of generations yielded comparable performance results. In one instance, the GP discovered a rule-base of 30 rules that exhibited results very similar to those shown in Figure 3.5. Less control effort was expended and, consequently, the overshoot amplitudes were reduced. Other rule-bases of 21 rules were also found that can be considered to have performed as well as the hand-derived controller if a steady-state position error of 0.2 m (≈ 8 inches) was acceptable according to the control system

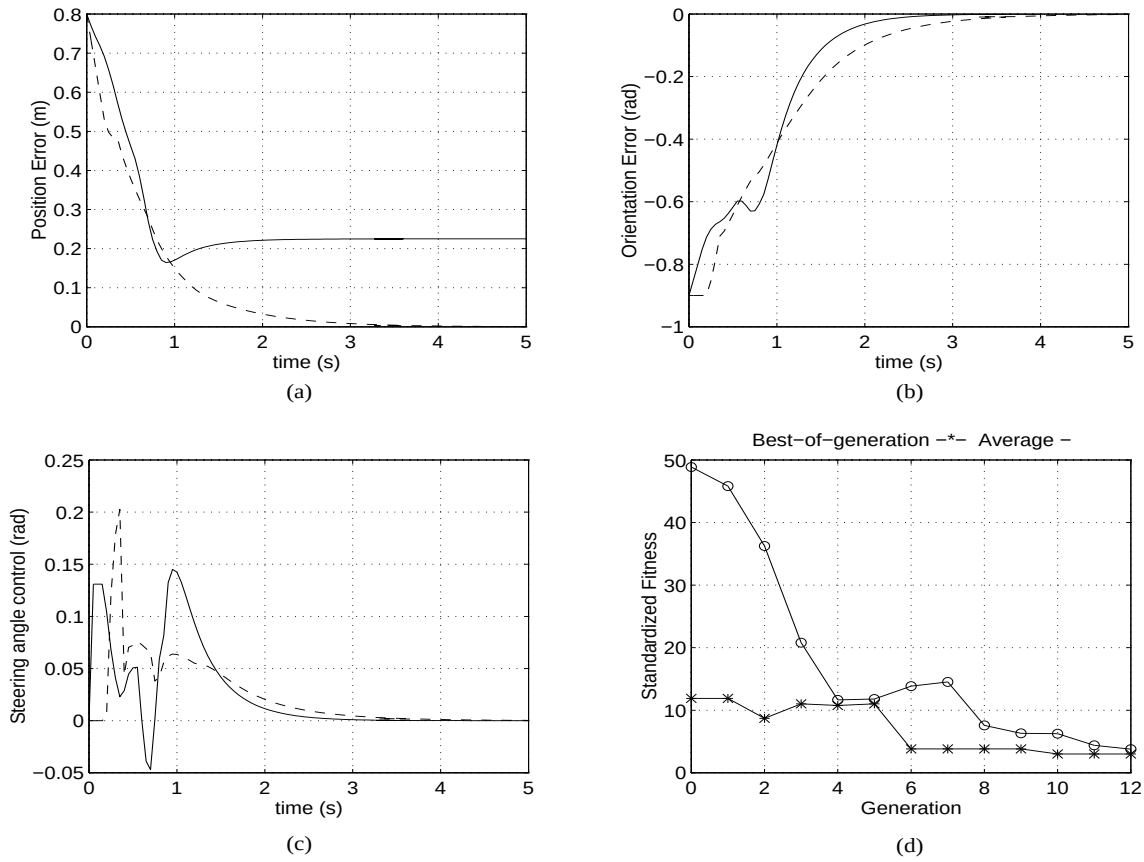


Figure 3.6: Rover path tracking with 21 rules: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle, (d) standardized fitness curves.

specifications. Results from one of these 21-rule controllers are shown in Figure 3.6. Generally, slightly faster settling times were observed with rule-bases of 21 rules. Figure 3.6d depicts a typical progression of the evolution process as a plot of standardized fitness vs. generations for the population average (–o–) and best-of-generation (–*–) rule-base. The response curves in Figures 3.6(a–c) are for the best-of-run rule-base which GP found in generation 10; it had a raw fitness of 2.8 and 7 hits. The result shown in the figure is for the only failed fitness case in which the steady-state position error exceeds the specified tolerance by about 0.05m ($\approx$ 2 inches).

A consolidated idea of the performance of GP for the steering control problem can be ob-

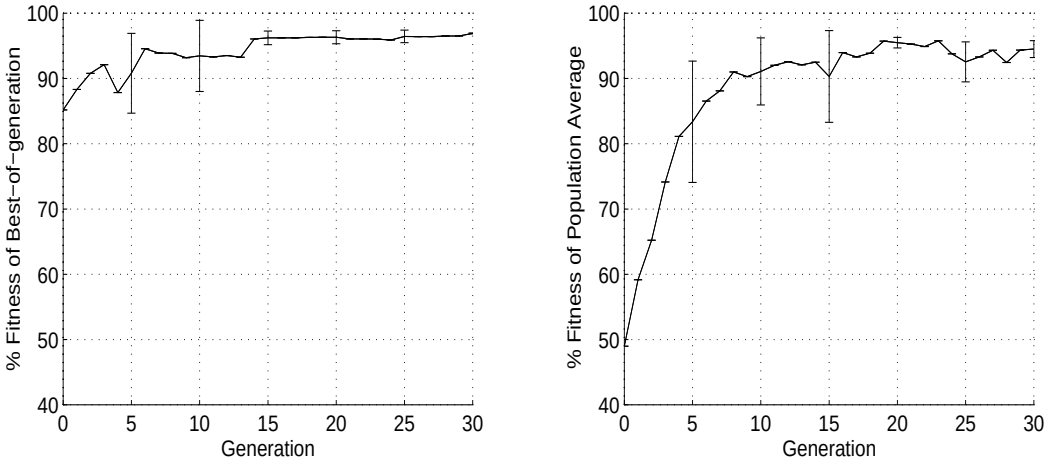


Figure 3.7: Mean performance of GP with state-error norm as fitness.

tained from Figure 3.7. The figure shows the mean GP performance over six independent runs, each starting with distinct random number generator seeds. Mean performance is plotted as a fitness percentage (based on the worst-case raw fitness) for the best rule-base and population average rule-base in each generation. Error bars are shown every five generations indicating the performance variance one standard deviation from the mean. On average, GP was capable of improving the current best rule-base in the population (Figure 3.7a) throughout the evolution; the average rule-base in the population itself (Figure 3.7b) also improves. The best fuzzy rule-base in the initial population is 85% fit for path following in the plane. After 30 generations the best fuzzy rule-base is 97% fit. The presence of at least one 85% fit solution in the initial population suggests that finding a set of fuzzy rules (given the membership functions in Figure 3.3) for this problem is not very difficult. As such, methods like hill-climbing or even random search might do just as well. In any case, the problem reveals potential utility of GP for fuzzy rule-base evolution.

3.6.2 Results with modified fitness measure

The smallest rule-base discovered by GP had 11 rules and a raw fitness of 1.99 according to Equation (3.3). However, the control signal was unacceptable due to fast oscillations of the steering angle during the first second or so of control. Such oscillations could cause damage to the robot's steering mechanism. This controller scored a fitness close to that of our hand-derived rule-base due to the "blindness" of the fitness measure to events taking place before the end of each fitness case. This revealed a necessity to modify the fitness measure in subsequent runs to include control effort as a cost. Several runs were executed after modifying the fitness measure to determine whether or not it would induce the desired effect of minimizing the control effort. The following fitness function was used,

$$Raw fitness = \sum_{i=1}^{8} \sqrt{(\varepsilon_d^2 + \varepsilon_\theta^2 + \bar{\delta}^2)_i} \tag{3.4}$$

where $\bar{\delta}_i$ is the average corrective control effort expended for fitness case i. Since lower raw fitness is associated with better rule-bases, this fitness function favors rule-bases that expend the least average control effort over the fitness cases. The best observed results after modifying the fitness measure are shown in Figure 3.8 along with the response curves for the complete rule-base. The GP evolved a new rule-base that had 18 rules, a raw fitness (according to Equation (3.4)) of 1.37, and 8 hits. The fuzzy controller with the complete rule-base scored a raw fitness of 2.43 using Equation (3.4). We see that there is indeed a reduction in the control effort (Figure 3.8c) due to the selective pressure induced by Equation (3.4). This reduction in control effort prevailed in *all* of the fitness cases. Thus, the new controller is the fittest despite the borderline, but acceptable, steady-state position error of 0.149m ($\approx$ 6 inches) for this single case.

The importance of the fitness function used in evolutionary algorithms is evident in these examples. It is important that the fitness function map observable parameters of the problem

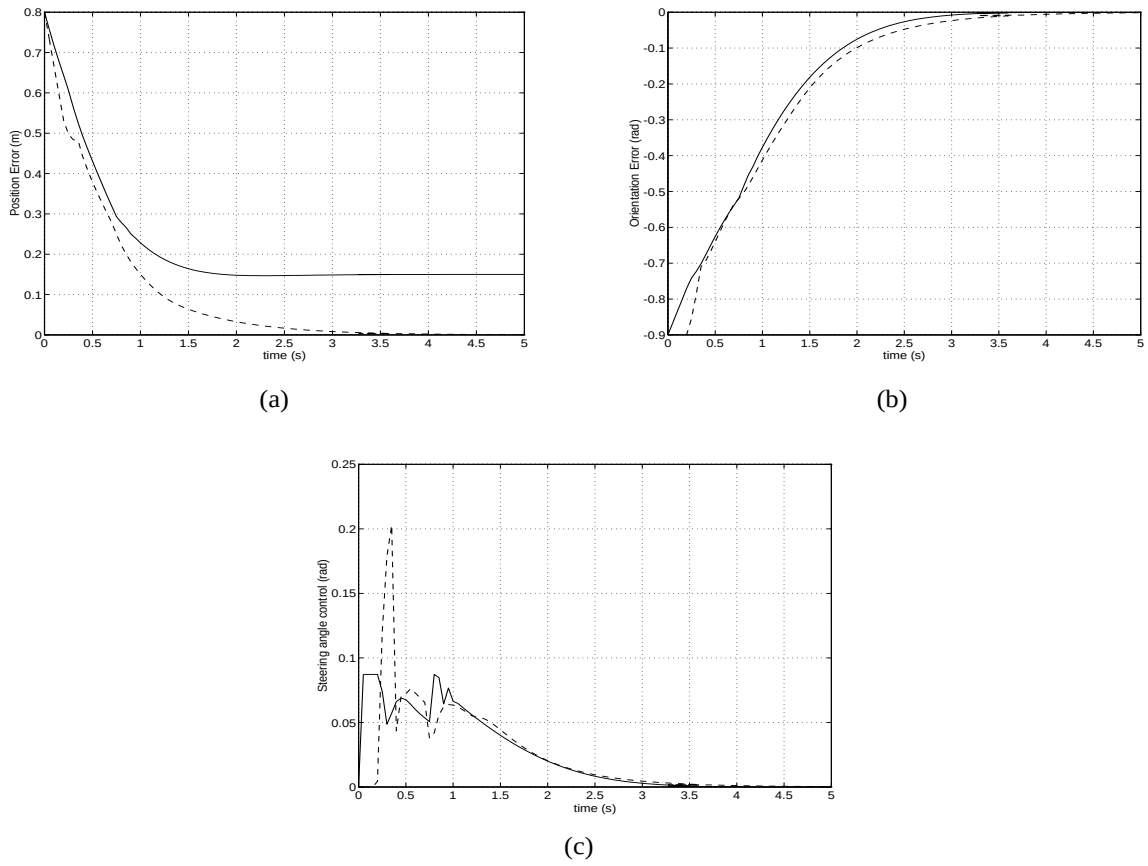


Figure 3.8: Rover path tracking performance comparison using control effort as a cost: GP-evolved FLC —, Hand-derived FLC - -; (a) position error, (b) orientation error, (c) steering angle.

into a spectrum of values that differentiate the performance of individuals in the population. If the spectrum of fitness values is not sufficiently rich, the fitness function may not provide enough information to guide GP toward regions of the search space where improved solutions might be found. The most common types of fitness functions used in GP are error measures and problem-specific payoffs. For problems involving simulation of controlled behavior, a variety of performance attributes can be considered for inclusion in the fitness measure. Examples include a maximum number of time steps, explicit error tolerances, terminating physical events such as task success or failure, and penalties/rewards thereof. In general, selected performance attributes can be weighted to emphasize their importance in the search for candidate solutions.

The fitness function is analogous to the performance measure of optimal control theory, or more generally, the objective function of optimization theory.

3.7 Summary and Conclusions

In summary, given suitable function and terminal sets, GP proceeds by randomly generating an initial population of rule-bases. This is followed by evaluating each rule-base in the current population and applying genetic operators to rule-bases selected with probability based on fitness. Genetic operators are applied to produce the next generation such that proper syntax is preserved. This process repeats until satisfaction of some termination criteria. The GP result is the best-fit rule-base that appeared in any generation.

The investigation reported in this chapter has revealed the potential of genetic programming as a tool for designing rule-bases for fuzzy logic controllers. For the purpose of evolving rule-bases, the GP implementation has some advantages over the simple GA and neural networks. Namely, it facilitates manipulation of the linguistic variables directly associated with the problem, and it allows for populations of rule-bases of various sizes. An additional feature of the syntactic structure is that it provides for context preservation as a by-product of structure-preserving crossover.

GP was applied to the problem of evolving a fuzzy behavior for controlling a mobile robot to steer onto a desired path. Good results have been obtained using small populations of rule-bases and the constrained syntactic structure for S-expressions. A number of fuzzy rule-bases have been evolved whose performances have been found to compare favorably with that of a complete rule-base derived by the author. Several evolved rule-bases performed better than the human-derived solution according to respective fitness measures imposed on the simulated

evolution. GP evolution was able to produce fuzzy controllers that required fewer rules than necessary for rule-base completeness. As with alternative evolutionary algorithms, the fitness function can be tailored to emphasize desired performance attributes. It was found that the best runs were those that used tournament selection as opposed to fitness-proportionate selection.

Regions of the search space with favorable rule-bases were consistently found using GP. In many cases suboptimal solutions with respect to the objective fitness function were found, suggesting that GP performs well as a global adaptive search method. Possible improvements toward optimal solutions can be made by synthesizing a hybrid between GP and a localized search method such as hill-climbing [52]. From the vantage point of fuzzy rule-based systems design, initial results suggest that seeding initial rule-bases with prior knowledge (e.g. rules ensuring stability), and perhaps, additional tuning of fuzzy membership functions may be necessary to improve the robustness of the GP solutions. Additional modifications that may improve on the results reported here are: adding different inference and defuzzification methods as options to be selected by the evolutionary process, and using random initial conditions in each fitness case to avoid overfitting pre-selected initial conditions.

As an alternative to the generational process of GP, a "steady-state" evolution could be applied as in the Steady-State Genetic Algorithm (SSGA) [69]. In this GA variant a few offspring of well fit parents, in a population of fixed size, replace the least fit individuals in the population on each iteration. This has the desirable side effect that good individuals tend to rise to the top of the fitness ranks where they are protected from deletion. Conversely, the lesser fit individuals tend to sink to the bottom of the fitness ranks where they are more likely to be deleted. This idea is easily applied to GP as well without a need to alter the constrained syntactic structure established above. Later in Chapter 6, we will apply GP and a steady-state GP to the evolution of fuzzy coordination rules in a more complex hierarchical fuzzy control system to be introduced in the next chapter. This upcoming application will challenge

genetic programming to scale up from evolving low-level regulatory and tracking types of fuzzy controllers (such as the steering controller) to higher-level coordination behaviors.

Chapter 4

Adaptive Hierarchy of Distributed Fuzzy Control

This chapter introduces a novel intelligent control architecture that employs a hierarchical rule-base structure enabling distribution of intelligence amongst a finite number of task-achieving *fuzzy-behaviors*. The formulation of such hierarchies is facilitated by incorporation of ethological ideology supporting an inherent hierarchical nature of behavior in animals. As such, it is a conceptual model of an intelligent system and its behavioral inter-relationships. It should be noted that the reference to hierarchy here is not implied as in the classical computational sense (i.e. bidirectional flow of information between levels). The network of distributed behaviors is hierarchical in the sense that overall system behavior is decomposed into a bottom-up hierarchy of increasing behavioral complexity in which behavioral activity at a given level is a function of behavioral activities at the level(s) below. A collection of *primitive* behaviors resides at the lowest level which is referred to as the primitive level. Primitive behaviors are encoded as fuzzy rule-bases with distinct control policies governed by fuzzy inference. They are typically simple and self-contained behaviors that serve a single purpose while operating

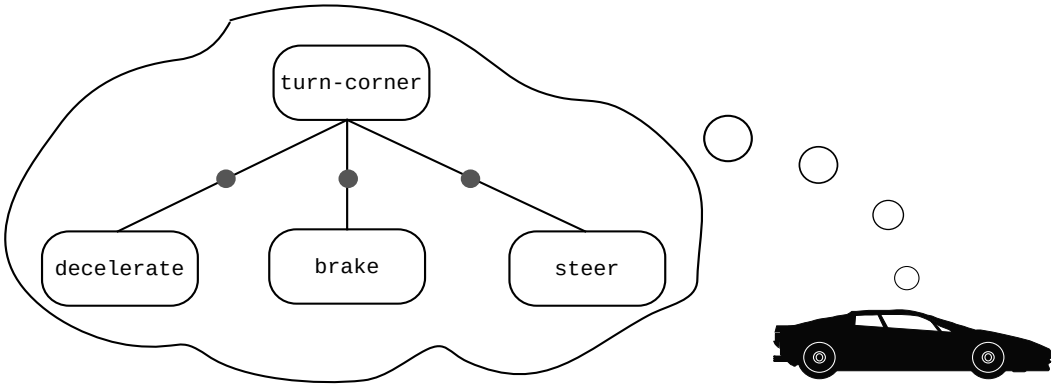


Figure 4.1: Possible hierarchy for turning behavior.

in a reactive (non-deliberative) or reflexive (memoryless) fashion. Primitive behaviors perform nonlinear mappings from different subsets of the available sensor suite to (typically, but not necessarily) common actuators. Each exists in a state of solipsism, and alone, would be insufficient for performing complex tasks. Such primitive behaviors are building blocks for more intelligent *composite behaviors*. That is, their capabilities can be combined through synergistic coordination to produce composite behavior(s) suitable for goal-directed operations. In the autonomous systems research community this property is often referred to as *emergent* behavior. As an example, consider the driver-automobile system as an autonomous system operating in a dynamic environment. Consider further the act of turning a corner at a moderate speed (say, 56 km/h [$\approx$ 35 mph]) as a composite behavior, and the individual acts of decelerating, braking, and steering as primitive behaviors. Then one can interpret the turning behavior as a synergism of the behaviors in the primitive level, each operating at varying levels of activation — see Figure 4.1. The remainder of this chapter explains how such systems can be realized as fuzzy logic controllers. The theoretical basis of the approach is presented followed by issues of analysis.

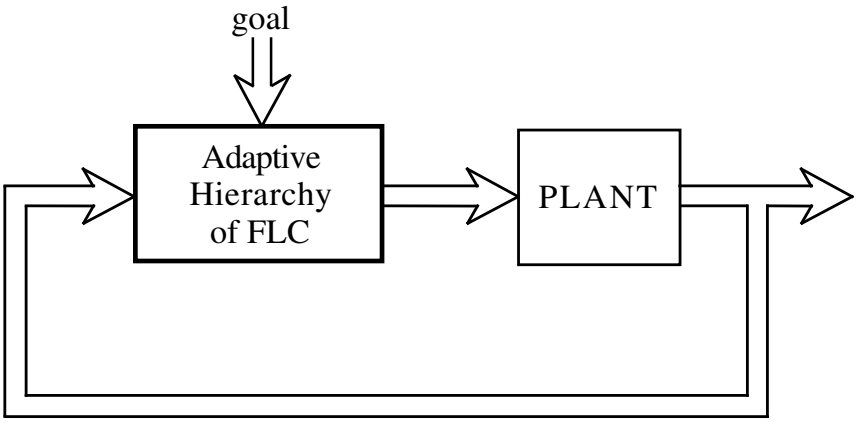


Figure 4.2: Hierarchical fuzzy behavior control architecture.

4.1 Theoretical Extensions for Complex Intelligent Systems

The hierarchical architecture can be represented by the control system block diagram shown in Figure 4.2. It differs from the canonical FLC described in Chapter 2 in that a multi-level structure of fuzzy rule-bases is employed and an adaptive mechanism is provided. Note that this architecture permits the fuzzy control hierarchy to assume the role of an intelligent supervisory controller over a conventional linear controller as depicted in Figure 4.3. In such a supervisory role, the hierarchy generates control set-points as input to the low-level controller (which is designed for regulation and/or tracking) in support of some higher level task-oriented control mission. Hence, its purpose in this configuration is to provide autonomy as opposed to parameter-tuning or gain scheduling operations for the conventional controller. Similar roles for fuzzy supervisory controllers have been reported in the recent literature [70, 71].

Figure 4.4 is a more detailed conceptual view of the hierarchy of distributed FLCs consisting of a primitive level of individual system behaviors, β_i, coordinated by higher-level system behaviors, B_j, via a weight-adaptive scheme introduced below. Each behavior in the hierarchy is similar to the canonical FLC in that it performs a mapping from some input space to some

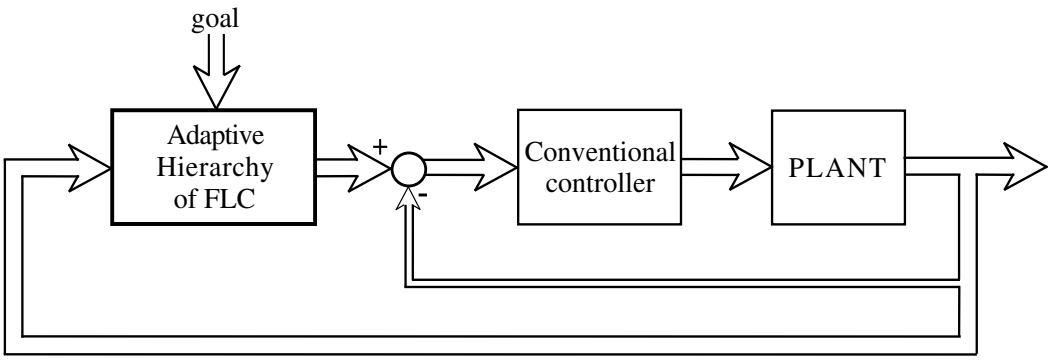


Figure 4.3: Intelligent supervisory control configuration.

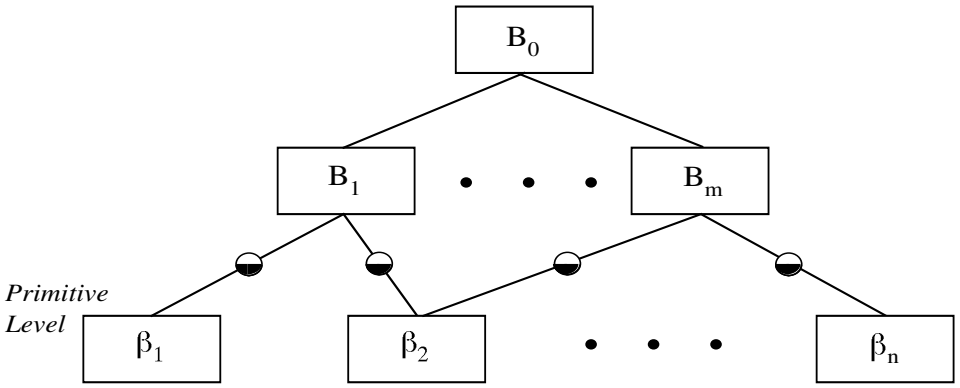


Figure 4.4: Fuzzy behavior hierarchy.

output space. The primitive behaviors map inputs to control outputs, while higher-level behaviors act as fuzzy decision systems which map goal information and other input to dynamically adaptive weights associated with each primitive behavior. Aspects of the architecture that depart from, and/or augment, canonical FLC implementations are discussed below. These are related to an enhancement of the conventional rule structure, and methods of coordination and conflict resolution among competing fuzzy logic-based behaviors.

Let X and U be input and output universes of discourse of a primitive behavior with a rule-base of size N. In Chapter 2 we described the generic fuzzy if-then rule as follows

$$IF\ x\ is\ \tilde{A}_i\ THEN\ u\ is\ \tilde{B}_i \tag{4.1}$$

where x and u represent input and output fuzzy linguistic variables, respectively, and $\tilde{A}_i$ and $\tilde{B}_i$ $(i = 1, 2, \ldots, N)$ are fuzzy subsets representing linguistic values of x and u. Typically, x refers to sensory data or goal information and u to actuator control signals. Formally, the i-th fuzzy if-then rule of the behavior is represented by a fuzzy relation (implication), $\tilde{u}_i \in X \times U$, which is a fuzzy set itself. Moreover, an entire fuzzy rule-base can be characterized as a single fuzzy relation, $\tilde{\beta}$, which is a union of fuzzy relations $\tilde{u}_i$, $i = 1, 2, \ldots, N$.

$$\tilde{\beta} = \bigcup_{i=1}^{N} \tilde{u}_i \tag{4.2}$$

This equation is essentially the same as Equation (2.18). A fuzzy rule-base, then, can also be represented as a fuzzy set. Thus, the mathematical operations of fuzzy inference in FLCs are closed for fuzzy sets. This fact serves as the basis for extending fuzzy set and logic operations used for monolithic fuzzy control to multi-rule-based fuzzy control.

In conventional fuzzy control, the aggregated result given by Equation 4.2 undergoes defuzzification to yield a crisp FLC output. As such, the defuzzification process is a form of coordination and conflict resolution among conflicting rule recommendations. In applications of the new architecture to complex autonomous systems, coordination and conflict resolution among *rule-bases* that recommend different control actions is a frequent concern. Discussions in the next chapter validate this. For these multi-rule-based fuzzy systems, we address such concerns by extending the mechanism of rule conflict resolution to *rule-base* conflict resolution via generalization of fundamental fuzzy logic concepts. That is, in the same way that individual fuzzy rule outputs are aggregated to yield a resultant fuzzy output set, outputs from multiple primitive behaviors are aggregated to yield a resultant fuzzy set. However, in the case of multiple behaviors this resultant fuzzy set represents the output of the overall behavior hierarchy. In order for this to work effectively, defuzzification of primitive behavior outputs must be deferred until after the aggregation takes place. Therefore, in the hierarchy of distributed fuzzy control the output of each primitive behavior is a fuzzy set. This is illustrated in Figure 4.5.

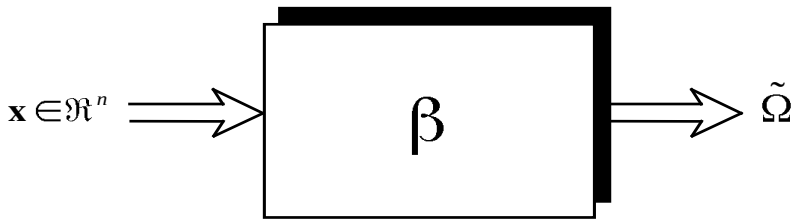


Figure 4.5: Fuzzy primitive behavior.

4.1.1 Applicability-based Decision-making

Regarding the structure of the rules, the proposed architecture advocates a control philosophy based on weighted rule-base (FLC) decision-making and rule-base selection. Weighted decision-making implies the incorporation of meta-rule-bases in which individual rules have weighting consequents. Consider two fuzzy system behaviors, B_1 and B_2, offering different or conflicting fuzzy control recommendations given, as in Equation (4.2), by $\tilde{\beta}_1$ and $\tilde{\beta}_2$. One way to coordinate, or resolve a conflict between, these two recommendations is by aggregating them via fuzzy union and defuzzifying the result to yield a crisp control action. However, in many cases this fusion of recommendations does not provide sufficient decision-making flexibility for autonomous control. What is needed is a mechanism for controlling the amount of influence a particular behavior has on the control action in a context-dependent way. The architecture provides this flexibility by introducing a scheme embodied in a concept referred to here as the *degree of applicability* which we will now define.

Definition 4.1 (Degree of Applicability (DOA)) *The DOA is a linguistic measure of the instantaneous level of activation of a behavior, B, expressed quantitatively as a scalar, $\alpha_B \in [0, 1]$, which determines the amount of influence that B will have on the control action corresponding to the situation prevailing during the current control cycle.*

The degree of applicability can be thought of in ethological terms as a motivational tendency of the behavior. Fuzzy rules of composite behaviors are formulated to include weighting consequents which govern the degree of applicability of primitive behaviors at a lower level. Thus, the canonical fuzzy controller rule structure is enhanced by the incorporation of meta-rule structures which are referred to here as *applicability rules*. Let B_c be a composite behavior comprised of N_p primitive behaviors. Then the degree of applicability, α_p, of primitive behavior p ($p = 1, 2, \ldots, N_p$) is specified in the consequent of applicability rules of the form

$$IF\ x\ is\ \tilde{A}_i\ THEN\ \alpha_p\ is\ \tilde{D}_i \tag{4.3}$$

where $\tilde{A}_i$ is defined as in Equation (4.1). $\tilde{D}_i$ is a fuzzy set specifying the linguistic value (e.g. "*high*") of α_p for the situation prevailing during the current control cycle. This feature allows certain system behaviors to influence the overall behavior to a greater or lesser degree depending on the current situation and system goal. It serves as a form of motivational adaptation since it causes the control policy to change in response to goals, sensory input, and internal state. Thus, behavior coordination is accomplished using meta-rules that provide a form of the ethological concepts of inhibition and dominance observed in animal behavior. Behaviors with partial applicability ($0 \leq \alpha < 1$) can be said to be inhibited by a dominant behavior with maximal applicability, i.e. with DOA equal to $\alpha_{\max}$ such that $\alpha < \alpha_{\max} \leq 1$.

As described here, the degrees of applicability are analogous to neuronal activation levels associated with artificial neural networks. Effects similar to neuron threshold activation are implemented with the use of λ-cuts of fuzzy sets. An λ-cut of a fuzzy set, $\tilde{C}$, defined over a universe U, is a crisp set, $\tilde{C}_\lambda$, containing all the elements of U with membership grade in $\tilde{C}$ greater than or equal to λ [72]. Formally,

$$\tilde{C}_\lambda = \{x \in X | \mu_{\tilde{C}}(x) \geq \lambda\} \tag{4.4}$$

where $\mu_{\tilde{C}}(\cdot) : X \rightarrow [0,1]$ is a membership function of fuzzy set $\tilde{C}$. To utilize λ-cuts for implementing thresholding behavior activation, we consider the output fuzzy set resulting from an inference evaluation of a rule-base. If the λ-cut of the inferred fuzzy set is null, then the system recommends that the level of activation for the associated behavior is zero. Thus if a threshold, $\theta \in [0,1]$, is imposed on a particular behavior, that behavior will be activated only when its DOA equals or exceeds its activation threshold, i.e. $\alpha \geq \theta$.

Behavior *selection* is a special case of this approach and occurs when the DOA of a primitive behavior is non-zero and above its activation threshold while others are zero or below threshold. When this occurs, the total number of rules to be consulted on a given control cycle is reduced. The reduction in rule evaluations is not as dramatic or static as in the strict rule hierarchies proposed in [13, 26] since we are dealing with *behavior* hierarchies that achieve interacting goals. As such, the number of rules consulted during each control cycle varies dynamically as governed by the DOAs and thresholds of the behaviors involved. When the state of the system's operating environment satisfies the conditions for activation of a single behavior, or several, there is no need to process rules from behaviors that do not apply (as is done in the conventional FLC architecture). Processing rules from irrelevant behaviors would result in unnecessary consumption of computational resources and possible introduction of "noise" into the decision-making process. In the proposed approach, there are no conceivable circumstances under which the totality of rules in the system will be consulted during a single control cycle.

4.1.2 Multiple Rule-base Coordination and Conflict Resolution

The coordination of fuzzy-behaviors is done in the framework of fuzzy logic theory via operations on fuzzy sets. In order to describe the process we will concentrate on a single composite behavior, B_c, which can be decomposed into primitive behaviors β_1 and β_2 ($N_p = 2$). We do

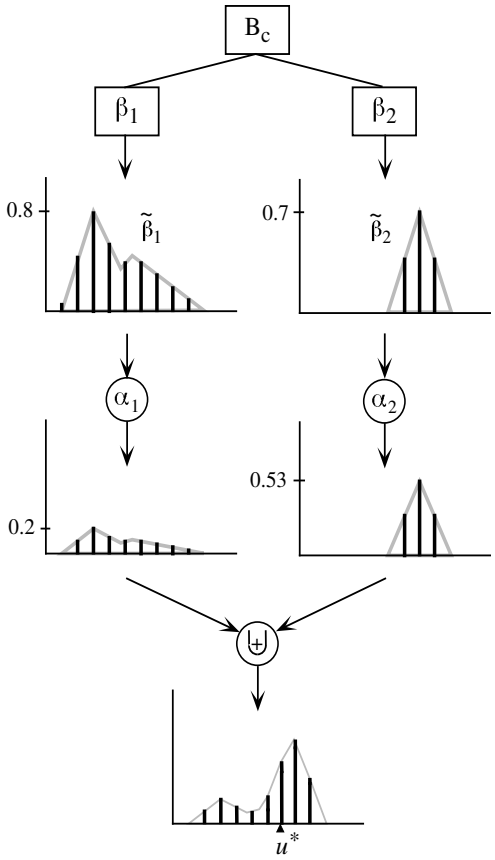


Figure 4.6: Fuzzy coordination of primitive behaviors.

this without loss of generality since the following description also applies when $N_p > 2$, and across additional composite behaviors. Consider a given control cycle during which B_c dictates that the applicabilities of β_1 and β_2 are say, $\alpha_1 = 0.25$ and $\alpha_2 = 0.75$ respectively. Fuzzy rules of each applicable primitive behavior are processed yielding output fuzzy sets, $\tilde{\beta}_1$ and $\tilde{\beta}_2$ (see Figure 4.6). Recall that these output fuzzy sets are equivalent to the result produced by rule-base evaluation in conventional FLCs *before* applying the defuzzification operator.

Following the consultation of individual primitive behaviors, each fuzzy behavior output is weighted by the corresponding degree of applicability. Thus, β_1 and β_2 are activated to degrees α_1 and α_2. Here, the behavior activation is accomplished via scalar multiplication of the output

fuzzy sets by the appropriate degree of applicability. This is equivalent to the conjunction of a crisp set of height α_i ($i \in [1, N_p]$) with the output fuzzy set. The resulting fuzzy sets are then aggregated using an appropriate t-conorm operator, and defuzzified to yield a crisp output, u^*, that is representative of the intended coordination of behavior. The illustration of this hypothetical example, shown in Figure 4.6, reveals that the output of B_c is influenced more by its dominant primitive behavior (in this case β_2) as intended. This procedure is expressed in a more formal manner below.

In the framework of fuzzy logic theory, conflict resolution is handled implicitly in the mechanics of the conventional fuzzy inference and defuzzification processes. That is, conflicting fuzzy outputs from individual rules are aggregated and defuzzified (resolved) to yield non-fuzzy control actions. In a similar manner as for coordination, we use fuzzy set theory to generalize this concept to resolve conflicts among conflicting rule-bases, i.e. behaviors. However, the manner in which this is done depends on whether there is partial or full conflict among competing behaviors. We will return to the conflict resolution issue in Chapter 5 during our discussion of applications to mobile robot behavior control.

4.2 Behavior Modulation Theory

The coordination procedure described above is a generalization of the idea of *rule* weighting in a single rule base to *rule-base* weighting among multiple rule-bases. Control recommendations from each applicable behavior are considered in the final decision. In the general case, the resultant control action can be thought of as a consensus of recommendations offered by multiple experts. In some instances, it may be evident from current sensory data that only one particular system behavior is fully applicable ($\alpha = 1$). In such cases, coordination simply reduces to the aforementioned behavior selection as is done in conventional behavior-based systems

and alternative approaches to fuzzy adaptive control [73, 74]. We refer to this as switching coordination since behaviors are alternately switched on and off.

Complex interactions in the form of behavioral cooperation or competition occur when more than one primitive behavior is active. These forms of behavior are not perfectly distinct; they are extremes along a continuum [75]. Instances of behavior throughout the continuum can be realized using *behavior modulation* which we define as follows.

Definition 4.1 (Behavior Modulation) *The autonomous act of regulating, adjusting or adapting the activation level of a behavior to the proper degree in response to a context, situation, or state perceived by an autonomous agent.*

In a given implementation the "proper degree" is governed by the desired behavioral response of the agent. Behavior modulation is achieved by dynamic adaptation of the DOAs of active primitive behaviors. Thus, the DOA concept and behavior modulation are intimately related.

We have established Equation (4.2) as an expression for the output fuzzy set of a primitive behavior. Let us denote the fuzzy output of primitive behavior p as $\tilde{\beta}_p$, and its corresponding DOA as α_p. Let P be the set of all primitive behaviors in a given adaptive hierarchy of distributed fuzzy control. Then the modulated fuzzy output of p is given by $\alpha_p \cdot \tilde{\beta}_p$. At this point the use of an appropriate t-conorm will take care of aggregating individual modulated fuzzy outputs to produce a resultant output of the behavior hierarchy. The arithmetic sum t-conorm has been chosen for this purpose since it facilitates the enforcement of the *weighted* decision-making intended in the philosophy of the adaptive hierarchy of distributed fuzzy control. The arithmetic sum will be denoted by the symbol, $\uplus$. Finally, if we denote the output fuzzy set of the behavior hierarchy as $\tilde{\beta}_H$ then its computation is performed using the following *fuzzy*

behavior hierarchy equation

$$\tilde{\beta}_H = \biguplus_{p \in P} \alpha_p \cdot \tilde{\beta}_p \tag{4.5}$$

or in the notation of membership functions,

$$\mu_{\tilde{\beta}_H}(u) = \sum_{p \in P} \alpha_p \cdot \mu_{\tilde{\beta}_p}(u). \tag{4.6}$$

The crisp control output, $u^* \in U = \{u_1, u_2, \ldots, u_r\}$, which serves as the input to the plant follows from the discrete Center-of-Sums defuzzification of $\tilde{\beta}_H$. That is,

$$u^* = \frac{\sum_{m=1}^{r} u_m \cdot \mu_{\tilde{\beta}_H}(u_m)}{\sum_{m=1}^{r} \mu_{\tilde{\beta}_H}(u_m)} \tag{4.7}$$

$$= \frac{\sum_{m=1}^{r} u_m \sum_{p \in P} \alpha_p \cdot \mu_{\tilde{\beta}_p}(u_m)}{\sum_{m=1}^{r} \sum_{p \in P} \alpha_p \cdot \mu_{\tilde{\beta}_p}(u_m)} \tag{4.8}$$

Of course, the shift defuzzification theorem holds as well. In this procedure, multiplication by α_p expresses the relative *applicability* of a primitive behavior to the current situation, while the scalar α_p itself represents the *weight* of the behavior in the aggregated control decision. Operators other than multiplication can be used to achieve a similar effect. Yager [76] refers to such operators as importance transformations and suggests a general class of them for both t-norm and t-conorm aggregations.

Note that using the arithmetic sum as an aggregation operator will often lead to resultant fuzzy sets which are supernormal (of height greater than 1). This is due to the summation of a number of membership values (in [0,1]) in regions where output fuzzy sets of several behaviors overlap. This presents no problems since the supernormal fuzzy set is defuzzified in the usual way as a quotient of "moment" and "area" (recall the analog to computing the centroid of a distributed load). Since both the moment and area of the set are equally effected by the summation, the effect cancels out in the quotient. Many successful applications of fuzzy control have employed arithmetic sum as a t-conorm, including Kosko's Standard Additive Model [6, 77] and Mizumoto's product-sum-gravity method of reasoning [78, 79]. In contrasting this approach

with the *max* t-conorm, Kosko [6] states that *max* "tends to produce a uniform distribution ... as the number of combined fuzzy sets increases. A uniform distribution always has the same mode and centroid. So, ironically, as the number of ... rules increases, system sensitivity decreases. The additive combination technique [on the other hand] tends to invoke the fuzzy version of the central limit theorem. The added fuzzy waveforms pile up to approximate a symmetric unimodal, or bell-shaped, membership function." Thus is the desired effect sought by the hierarchy when combining multiple fuzzy-behavior outputs.

4.3 Issues of Stability Analysis

While the focus of this dissertation is primarily one of control system synthesis, control system analysis is important enough to warrant some dedicated space. In any dynamic system, the questions of guaranteed stability and controllability arise. These are structural properties of control systems, the acceptable meanings of which are defined in the mathematical language of analytic control theory. It is not clear whether the analytical tools of conventional control theory are the most suitable for analyzing the structural properties of fuzzy controllers or other control systems based on soft computing techniques. As such, many researchers are currently concentrating on developing theoretical approaches to the problem as it relates to fuzzy systems. For example, Mamdani [15] argues that fuzzy control provides an alternative paradigm to the analytic control theory that consists of non-analytic approaches to control and are based on decision-making approaches from artificial intelligence.

4.3.1 Supervisory control

Thus far, the adaptive hierarchy has only been applied in a supervisory mode (see Figure 4.3). This is its intended mode of operation. That is, it is meant to be applied to systems as a

high-level or task-level controller atop conventional controllers (e.g. PID or its variants). As such, stability issues are assessed using the well established analysis methods of linear systems. Outputs of the adaptive hierarchy serve as control set-points for the low-level controller(s). If it is the case that low-level controller specifications impose bounds on its input, say $u \in U_c$, then compliance of the hierarchy is easily ensured by imposing the same bounds on U_H, the behavior output universe of discourse, such that $U_H \subseteq U_c$. In this way, any defuzzification operation on behavior hierarchy outputs is guaranteed to yield a crisp control $u^* \in U_c$. Therefore, in the supervisory mode of operation, a closed-loop system influenced by the adaptive hierarchy of distributed fuzzy behaviors is stable in the same sense as its underlying conventional controller.

4.3.2 Direct control

In the absence of a stable low-level conventional controller, the adaptive hierarchy can be used for direct control of the plant (see Figure 4.2). If this is the case, stability and controllability need to be addressed in some convincing way. Robustness to system parameter perturbations is typically had for free since this is a known characteristic of fuzzy control systems.

One of the earliest approaches to stability analysis of fuzzy controllers was developed by Braae and Rutherford [80]. The approach is known as the fuzzy phase plane approach (or state space approach) and is based on the relationship between the phase plane and the fuzzy rule-base. It is a graphical approach that is useful for predicting stability as well as other dynamic phenomena. Fuzzy phase plane analysis is limited to two-dimensional systems due to difficulties in the interpretation of higher-dimensional graphical representations of the phase plane [22]. The fact that fuzzy behavior-based systems typically do not comply with the dimensionality restrictions of the approach precludes its use for these systems.

To date, much of the research on stability analysis techniques has viewed fuzzy control

systems as nonlinear dynamic systems. Analysis methods based on Lyapunov stability and input-output (the small-gain theorem) stability can be applied *if* a dynamic model of the plant is known [22]. An alternative approach is to formulate behaviors as Takagi-Sugeno fuzzy systems [81] which are characterized by functional expressions in the rule consequents. When these functional expressions are linear each rule consequent corresponds to a linear controller with constant coefficients. These forms of the Takagi-Sugeno FLC are supported by a convincing stability theory based on Lyapunov's direct method [81, 12]. In applications to mobile vehicle control, a collision avoidance behavior is typically is most important and will tend to dominate other system behaviors at all times. In a system designed with this property, it should suffice to show that the collision avoidance behavior is stable in some sense in order to claim stability of the behavior hierarchy. The collision avoidance behavior (and other primitive behaviors) used for direct control can be formulated as a Takagi-Sugeno behavior. This would enable stability analysis of the adaptive behavior hierarchy for the direct control case. One problem with this approach, however, is that the condition for global stability [81] of a behavior requires the existence of a common positive-definite matrix that satisfies the Lyapunov Equation for every linear rule consequent. In most instances this common matrix would be difficult to determine, particularly for a large set of rules. In addition, if new rules are added to the behavior the search for a common matrix becomes more difficult. More conservative stability criteria are available for Takagi-Sugeno FLCs using the Interval Matrix Method [12]. These are based on recent stability results for time-varying discrete interval matrices [82], and are not as computationally intense as the Lyapunov approach.

Recent developments in stability analysis provide additional alternatives for fuzzy systems without requiring functional expressions in the rule consequents. Kosko [77] recently reported several theorems related to global asymptotic stability of so-called additive fuzzy systems. The results are applicable to fuzzy systems designed according to his Standard Additive Model

which differs only slightly from the model presented in this and the previous chapter. The main differences are in the calculation of rule strengths and the aggregation of rule outputs. The Standard Additive Model uses product and sum, respectively. Kosko also provides a corollary to Tanaka's main result [81] which gives a sufficient condition for the choice of the *identity* matrix as the common positive-definite matrix mentioned above. Tso and Fung [83] proposed a new technique for stability analysis of autonomous vehicles controlled by FLCs. Their "equivalent transformation" algorithm is based on the idea of analytical determination of a control surface that is equivalent to an arbitrary desired control surface for an FLC. Conditions on this equivalent control surface that are obtained in accordance with Lyapunov and Hurwitz stability criteria leads to global asymptotic stability. Finally, Cao et al [84] proposed a Lyapunov-like approach to stability analysis that is applicable to both the Mamdani and the Takagi-Sugeno types of FLCs. They develop their approach based on a theory of generalized dynamic systems. As such, the method can be used for model-free *and* model-based fuzzy control design.

4.4 Conclusion

This chapter has covered the essential ingredients of the proposed approach to hierarchical fuzzy control of complex autonomous systems. Overall system behavior is decomposed into a bottom-up hierarchy of increasing behavioral complexity with primitive behaviors at the lowest level serving as fuzzy controllers, and composite behaviors at higher levels serving as fuzzy decision systems. Capabilities of primitive behaviors are combined to produce composite behavior(s) suitable for goal-directed operations. That is, their fuzzy outputs are modified by their respective DOAs and intelligent behavior emerges through behavior modulation. The underlying theory consists of relatively simple fuzzy set operations across multiple rules and

multiple rule-bases which serve as a basis for extending the monolithic FLC approach to multi-rule-based control. In the next chapter we discuss the implications of this approach in the context of applications to navigation control of autonomous mobile vehicles.

Chapter 5

Fuzzy Behavior Control Systems

In order to implement the proposed ideas and to test their validity, a sufficiently complex plant and associated environment should be chosen. An ideal choice is the problem of controlling autonomous mobile vehicles or rovers that are to be deployed in dynamic and possibly unstructured environments (e.g. office settings, factories, natural terrain, planetary surfaces, etc). This is currently a very active and challenging area of research of concern to a multidisciplinary community of engineers and scientists. Electrical and mechanical engineers, computer scientists, and more recently, biologists and cognitive scientists are all contributing to the area of autonomous mobile robot research.

Traditional methods which address mobile robot control issues have relied upon strong mathematical modeling and analysis. Various approaches proposed to date are suitable for control of automated guided vehicles which operate in *structured* environments and perform relatively simple tasks that require only motion along fixed paths. For this class of mobile vehicles, classical control techniques for tracking or teleoperation can be applied without great difficulty. Motion control research in mobile robotics is now steering toward advances beyond fixed-path tracking and teleoperation and closer to true autonomy. As environmental structure

and task constraints are removed from the problem domain, the need for increased autonomy mandates the development of higher-level intelligent controllers. Unfortunately, operations in unstructured environments require robots to perform complex tasks for which analytical models for control can often not be determined. Furthermore, in cases where models are available, it is questionable whether or not uncertainty and imprecision are sufficiently accounted for. Robust behavior requires that uncertainty be accommodated in the robot controller, especially where autonomy is desired. The very nature of autonomy dictates the need for some capacity of adaptive behavior. Under such conditions fuzzy logic control is shown to be an attractive solution that can be successfully implemented on real-time autonomous systems.

The remainder of this dissertation describes how the application of fuzzy logic in the framework of behavior control can contribute to the realization of autonomous rovers. In this chapter, implications of applying the new approach to behavior control synthesis are discussed. Simulated and experimental results that verify its validity and practical utility in this problem domain are reported in the next chapter. First, let us discuss some issues of practical concern in mobile robotics.

5.1 Some Practical Concerns

To say the least, a significant amount of progress must be made to achieve the sophistication necessary for true autonomy. A current limitation is due to the processing capabilities of state-of-the-art microprocessors and other computational resources which must be carried on-board the vehicle. Mobile systems with modest computational resources are typically equipped with inexpensive range sensors such as infrared proximity detectors, sonar, and/or position sensors such as optical encoders. The problems encountered with these sensors are well known [85]. The reliability of sensory data collected from infrared sensors is affected by the reflective prop-

erties of obstacles in the environment. As a result, the quality of the range data operated on by an associated fuzzy controller is suspect. When using sonar ranging to reason about the state of the world, we must be prepared to handle the inaccurate sensory information which is inherent in sonar-based systems. Ideally, when sonar pulses are emitted against a surface they are reflected from that surface to a receiver, thus allowing for range calculation based on the speed of sound and time of flight of the pulse. In practice, however, it is quite common for the sonar pulse to be reflected from the initial surface, to other surfaces and finally back to the receiver, yielding incorrect range calculations. Such specular reflections make obstacles appear to be closer to, or farther away from, the robot. Specular reflection is a predominant source of error, particularly outdoors where surfaces are generally rough. The specular reflection problem inherent in sonar range finders is similar to the problem humans face when inside of a house of mirrors. In such environments, the human vision system is quite susceptible to false positive readings despite its complexity and otherwise robust performance. As such, the potential for false positive readings in the world directly impacts the algorithms used to navigate throughout. Dead reckoning, which is the procedure of calculating or measuring vehicle heading and distance travelled, and adding these to a known initial location, is another source of error and uncertainty. Systematic errors caused by unequal wheel diameters, uncertainty about the vehicle's wheelbase, or other mechanical imperfections are common [86]. In addition, non-systematic errors such as irregularities (bumps, cracks, etc) in the terrain contribute significantly to the problem. Since these errors effect both the heading and distance calculation, errors in position of the vehicle become increasingly large as it travels. Frequent updating of position based on known references or landmarks in the environment is often necessary to maintain accurate localization over long traverses. In the face of these formidable practical concerns, an approach to autonomous navigation must be both robust and adaptable.

To endow mobile vehicles with adaptability sufficient for autonomous navigation, the ap-

proximate reasoning capability provided by fuzzy logic can be exploited as a resource for intelligence. The hierarchical fuzzy control architecture proposed in the previous chapter facilitates this by providing an efficient framework for control of such complex distributed-intelligence based systems. In general, fuzzy logic controllers provide robustness to perturbations, design simplicity, and efficiency in dealing with continuous variables [15]. The successful management of uncertain, unreliable, and/or unmodeled data is a proven attribute of fuzzy logic based inference engines. In the context of mobile robot control, a fuzzy logic-based system has the advantage that it allows the intuitive nature of sensor-based navigation to be easily modeled using linguistic terminology. The computational loads of typical fuzzy inference systems are relatively light. As a result, reactive fuzzy control systems permit intelligent decisions to be made in real-time, thus allowing for smooth and uninterrupted motion. Reactive behavior-based control systems require less computation than traditional systems and still produce robust autonomous performance [19, 87, 88].

At some point in the development of mobile robot control architectures, the software must be interfaced with the physical robot hardware for validation of simulation results or final deployment in the target environment. This constitutes the physical embodiment that instantiates the intelligent mobile robot. There are a number of realization options for physically embodied mobile robots given an assortment of computational resources, communications devices, sensors and actuators. The options can be generally classified according to whether the control mode is semi-autonomous or autonomous. By semi-autonomous we mean that the main information processing responsible for the robot's intelligence is resident on some remote processor(s) and not carried onboard the vehicle. In addition, the remote processor communicates with the robot via either physical tether or radio frequency (RF) signals. Autonomous control is meant to refer to situations in which all computational resources are carried onboard. We will focus on the latter hereafter.

5.2 Behavior-based Mobile Robot Control

In recent years we have witnessed a shift in ideology regarding problem decomposition for mobile robot control. Traditional architectures are modularized according to a functional decomposition of tasks into an overall sense-plan-act cycle. The robot control system executes functional modules sequentially, and will malfunction if any module is missing. Due to this sequential processing computational bottlenecks are common, particularly in the planning modules. Mobile robot research is now leaning towards control approaches that are modeled after natural processes, and that advocate a behavioral decomposition of tasks with quasi-parallel execution. In principal, behavior-based control can be applied in situations where classical control is not feasible, usually due to tremendous modeling difficulties. The behavior control paradigm was initially proposed in the seminal paper by Brooks [30] where it was realized in the subsumption architecture. In this architecture, a number of behaviors (implemented as finite state automata) execute in parallel in response to instantaneous sensory data. The behaviors comprise a distributed system that controls a mobile robot through arbitrated competition and collaboration. Since the introduction of this architecture a number of variants have been proposed for behavior control. This chapter presents one of the most recent variants among those that employ approximate reasoning via fuzzy logic.

5.2.1 Fuzzy-behaviors

Some of the earliest attempts at applying fuzzy logic to the control of mobile robot vehicles were made by Uragami et al [89] and Sugeno and Murakami [90]. Uragami and colleagues [89] implemented a fuzzy program to control a simple inchworm robot to simulate a person wandering through a town. They concluded that robots could be used to explore spatial regions, with humans issuing commands in the form of fuzzy instructions. Later, Sugeno and

Murakami [90] conducted successful experiments in embedded fuzzy control of a model car for parking based on an operator's control actions. This early work was done before the initial reports on the subsumption architecture became available, and thus, was not influenced by the behavioral decomposition proposed by Brooks [30].

Following the introduction of the subsumption architecture and the emergence of the behavior control paradigm, a host of groups recognized the advantages to be gained by incorporating fuzzy logic into the framework of behavior control for mobile robots. Maeda et al [91] proposed a modification of Zadeh's fuzzy algorithm which includes adjustable thresholds which govern rule firing. Saffiotti et al [92, 93] have developed fuzzy-behaviors for complex navigation tasks demonstrating the robustness of fuzzy control in blending reactive and goal-oriented behavior. Pin and Watanabe [94] developed qualitative reasoning schemes for autonomous navigation in unknown environments with emphasis placed on embedded control using VLSI fuzzy chips. Badreddin [95] proposed a unique alternative to fuzzy behavior fusion based on fuzzy analogical gates. A heterogeneous network of fuzzy controllers for reactive behavior-based control was implemented by Goodridge and Luo [96]. In this network, control actions are generated by outputs of independent fuzzy controllers that are linked together through a qualitative rule-base. Li [97] emphasizes weighting of reactive behaviors, achieved by implicit mechanisms of fuzzy inference, as an improvement in efficiency over priority-based arbitration. Tunstel and Jamshidi [98, 99] have proposed strategies for fuzzy behavior-based mapping and fuzzy spatial map representation for navigation. Research is also being pursued in the area of motion control for path execution [100]. These are but a few of the research activities in fuzzy-behavior control of mobile robots. The approaches of each of these research activities are mutually similar and have some things in common with the architecture described in the previous chapter. The approach proposed here differs from other approaches mainly in the hierarchical structure of the fuzzy rules, and the incorporation of a dynamic behavior arbitration mechanism as a source

of adaptive behavior.

5.2.2 Synthesis

Advantages are gained in a fuzzy-behavior control hybrid with regard to representation and handling of uncertain and imprecise knowledge about the robot's environment, and in behavior coordination and conflict resolution. Non-fuzzy behavior controllers explicitly account for real-world uncertainty by augmenting crisp reasoning with heuristics in a manner akin to production systems of artificial intelligence. Fuzzy control, on the other hand, implicitly accounts for uncertainty by virtue of the approximate reasoning capability of fuzzy logic.

In many of the non-fuzzy behavior control implementations, behaviors are synthesized as finite state automata or augmented finite state machines. In contrast to this, fuzzy-behaviors are synthesized as fuzzy rule-bases. Each behavior is encoded as a fuzzy rule-base with a distinct mobile robot control policy governed by fuzzy inference. The procedure for fuzzy-behavior synthesis consists of first defining linguistic terminology for the behavior inputs and outputs, partitioning the sensor space and actuator space using appropriate fuzzy sets, and formulating fuzzy rules that satisfactorily govern the desired response of the behavior in all practical situations. This is the same procedure that is used for fuzzy controller synthesis. It is an iterative procedure of trial-and-error which, in practice, involves fine tuning of the shapes of membership functions used to express uncertainty in inputs and outputs, as well as modifications to the fuzzy rule-base.

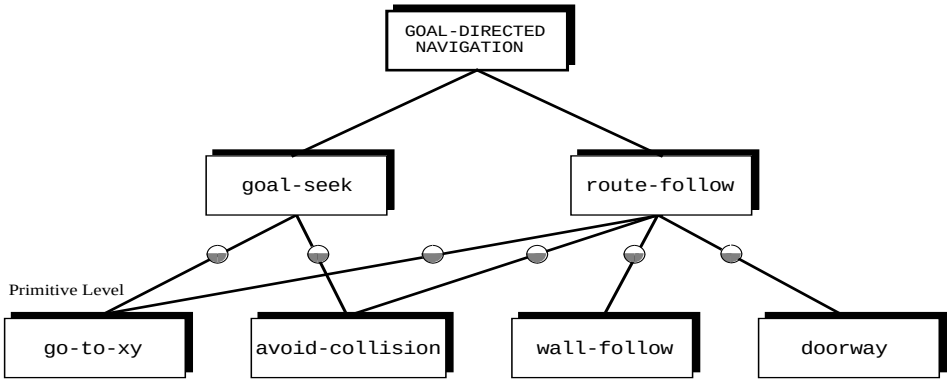


Figure 5.1: Hierarchical decomposition of mobile robot behavior.

5.3 Behaviors in the Adaptive Hierarchy

Mobile robots operating in non-engineered domains must be capable of reliable navigation in the presence of static and dynamic obstacles (e.g. humans and/or other moving robots). It is preferred that such robots be designed to navigate autonomously, in an equally effective manner, in both sparsely populated environments (e.g. an overnight security robot patrolling an office building) and in cluttered environments (e.g. a robot transporting material on a busy factory floor). Several capabilities are necessary to achieve this, including collision avoidance, self and goal localization, and traversal through indoor features such as halls, doorways, and densely cluttered spaces. A behavior hierarchy encompassing these capabilities is shown in Figure 5.1. It implies that goal-directed navigation can be decomposed as a behavioral function of `goal-seek` (collision-free navigation to some location) and `route-follow` (assuming some direction is given in the form of waypoints or a path plan). These behaviors can be further decomposed into the primitive behaviors shown, with dependencies indicated by the adjoining lines. The circles represent weights and activation thresholds of associated primitive behaviors. As described in the previous chapter, fluctuations in these weights are at the root of the intelligent coordination of primitive behaviors which leads to adaptive system behavior.

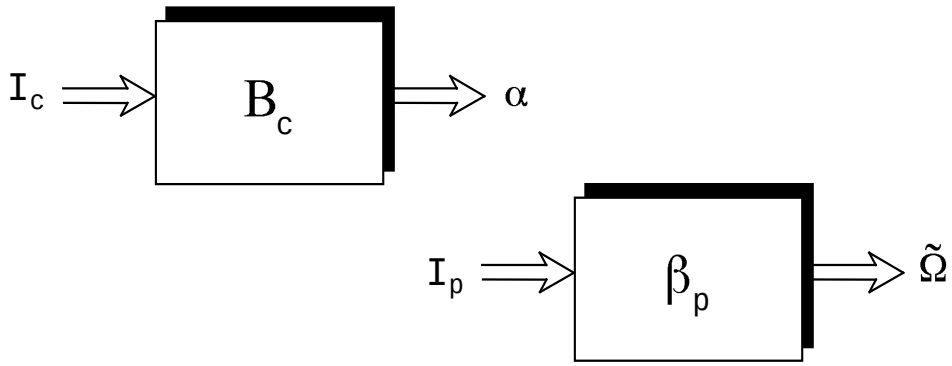


Figure 5.2: Fuzzy decision and fuzzy control modules.

The hierarchy facilitates decomposition of complex problems as well as run-time efficiency by avoiding the need to evaluate rules from behaviors that do not apply. The composite behaviors are fuzzy decision systems and the primitive behaviors are fuzzy controllers (see Figure 5.2). That is, the output(s) of composite behaviors are used in control decision-making, while the output(s) of primitive behaviors are applied as control inputs to the plant.

5.3.1 Composite behaviors

The `goal-seek` behavior is a composite behavior serving as a fuzzy decision system. Its inputs consist of the range to the nearest obstacle ($r_{\min}$), the distance from the goal (d_{goal}), and the angular heading to the goal (θ_{goal}). Its outputs are DOAs, α_{gt} and α_{ac}, which correspond to the DOAs of `go-to-xy` and `avoid-collision` respectively. Its purpose is to coordinate the activation of its underlying primitive behaviors through behavior modulation. This is achieved in the current implementation with 11 rules.[1]

The `route-follow` behavior also serves as a fuzzy decision system. Its inputs are the same as those of `goal-seek` except that it takes additional route information in the form of a list of waypoints leading to a goal/sub-goal. Outputs of `route-follow` consist of α_{gt}, α_{ac}, α_{wf} and

[1]Note that the number of rules may vary from system to system.

α_{dw}. The latter two outputs correspond to DOAs of the `wall-follow` and `doorway` behaviors respectively. The purpose of `route-follow` is to coordinate the activation of its underlying primitive behaviors such that navigation via the specified waypoints is achieved. In the current implementation this is done using 18 rules.

5.3.2 Primitive behaviors

The `avoid-collision` behavior is perhaps the most important primitive behavior in an autonomous mobile system. Here, it is implemented as a fuzzy *control* behavior since it is comprised of a set of fuzzy control rules, i.e. rules with control inputs as consequents. Its three inputs convey information about the obstacle situation relative to the front and both sides of the robot. These inputs are determined from a sensory fusion operation (see below) on available sensor range data. The outputs of the behavior, and all other primitive behaviors, are fuzzy control outputs from which crisp controller inputs are computed by defuzzification of the behavior hierarchy equation (Equation 4.5). These are typically desired velocities of each wheel (in the case of a differential-drive mechanism) or the desired linear velocity and heading of the robot. As its name implies, the purpose of `avoid-collision` is to steer a robot away from obstacles. When there are no obstacles to avoid, the behavior exhibits a tendency to move the robot in a forward direction. Left alone, this behavior displays a wandering activity. The current implementation uses 11 fuzzy control rules to avoid collisions.

The `go-to-xy` primitive behavior takes the current Euclidean position error between the robot and the goal, and the corresponding heading error as its inputs. Therefore, it relies on information about its relative pose with respect to its current goal, as opposed to sensor range data. Hence, this behavior is not cognizant of any concept associated with obstacles; its "knowledge" is purely Cartesian. The sole purpose of `go-to-xy` is to steer a robot in the

direction of its goal, force the robot to move to its goal, and stop the robot's motion when it has arrived. In effect, it merely directs motion along a straight line trajectory to a particular location. A set of 18 rules are used.

Like `avoid-collision`, the `wall-follow` behavior takes information about the obstacle situation relative to the front and both sides of the robot. However, the information sampled by each of these behaviors are defined over universes of discourse that are specified differently, and partitioned using different sets of membership functions. The behavior's purpose is to follow walls by causing the robot to move parallel to walls at a specified distance. It currently operates using 8 fuzzy control rules. Finally, the `doorway` behavior takes similar input information. Its purpose is to guide a robot through narrow passageways in walls. It uses 8 rules.

These brief descriptions reveal that behaviors at the primitive level (of the current implementation) use a total of 11 inputs and 45 rules. Each input universe is partitioned using *at least* 2 linguistic values, or fuzzy sets. More than 2 fuzzy sets actually span the universes of most of the inputs. However, if we make the very optimistic assumption that each input universe is partitioned using only 2 fuzzy sets, then a complete rule-base for a monolithic FLC realization would require $2^{11} = 2048$ rules! Distribution of the controller intelligence among four primitive behaviors in this case results in a dramatic reduction in the number of rules. Furthermore, if all four primitive behaviors are not concurrently active at a given instant, then less than 45 rules will be consulted during the corresponding control cycle.

It is important to point out the solipsist view that each of the primitive behaviors have of the "world". Operating alone, each would be insufficient for autonomous navigation. The functionality of the system depends on a combined effect of the behavioral functionality of each primitive (and or course, the competence of the composite behaviors which coordinate them).

5.3.3 Sensory fusion

Sensor suites used for autonomous navigation usually consist of a considerable number of individual sensors. The total number of relevant sensors varies from system to system. However, it is typically large enough to make feeding each measurement to an FLC infeasible due to the combinatorial effect that many inputs have on rule-base cardinality. Before delivery to behaviors, relevant sensor data are routed through a preprocessing stage which combines measurements into a reduced, but useful, amount of information. We refer to this idea as sensory fusion. This is distinguished from the fairly common use of the term "sensor fusion," which refers to determining the most proper interpretation of inconsistent or conflicting data from multiple sensor sources.

Behaviors act on sensor data from different subsets of the sensor suite. These data include so-called virtual sensor data which are derived from actual sensor readings (e.g. d_{goal}, which is computed from position encoder data and knowledge of goal coordinates). If we let S_I denote the set of all available inputs to behaviors, and S_p denote the non-empty set of inputs considered by behavior p, then $S_p \subset S_I$ for all p. Furthermore, if I_p is the set of inputs actually used by behavior p then

$$I_p = \ fuse(S_p) \tag{5.1}$$

where $fuse(\cdot)$ is some operator or function appropriate for the desired combination of sensor information. The idea is expressed in Figure 5.3 where sensor measurements $\sigma_i \in S_p$, $\iota_j \in I_p$, and $n < m$. Common fusion operations used for $fuse(\cdot)$ are $min(\cdot)$, as used here, and linear combinations of inputs in S_p [26].

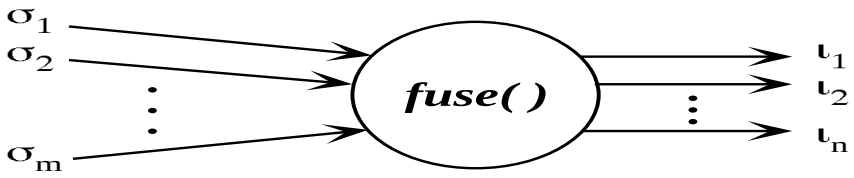


Figure 5.3: Sensory fusion operation.

5.3.4 An alternative example

As an additional example of a behavior hierarchy, consider the outdoor navigation problem encountered by natural terrain vehicles such as planetary rovers. Autonomous rovers designed for natural terrain must be capable of point-to-point navigation in the presence of varying obstacle (rocks, boulders, dense vegetation, etc.) distributions, surface characteristics, and hazards. Often the task is facilitated by knowledge of a series of waypoints, furnished by human, operators which lead to designated goals. In some cases, such as exploration of the surface of Mars [20, 101], this supervised autonomous control must be achieved without the luxury of continuous remote communication between the mission base station and the rover.[2] Considering these and other constraints associated with rover navigation, suitable behavior hierarchies similar to the hypothetical one shown in Figure 5.4 could be constructed. In this figure the behavioral functions of `goal-seek`, `route-follow`, and an additional composite behavior, `localize` are decomposed into a slightly different suite of primitive behaviors. The design of behaviors at the primitive level would be tailored to the navigation task and an environment with characteristics of natural terrain.

Note that decomposition of behavior for a given mobile robot system is not unique. Consequently, suitable behavior repertoires and associated hierarchical arrangements are arrived at following a subjective analysis of the system and the task environment. The total number,

[2]Time delays between Earth and Mars can be anywhere between 6 and 41 minutes.

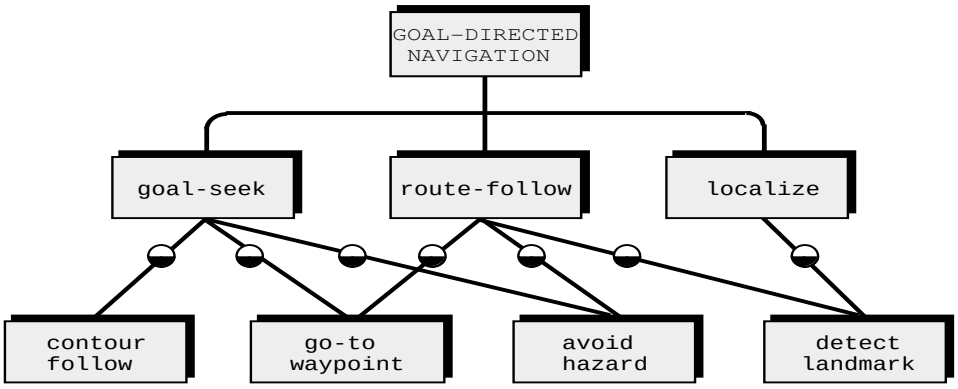


Figure 5.4: Hypothetical behavior hierarchy for planetary rover navigation.

and individual purpose, of fuzzy-behaviors in a given behavior hierarchy is indicative of the problem complexity and can be conveniently modified as required.

5.4 Coordination by Behavior Modulation

In the last chapter, we described mechanisms for multiple-behavior coordination and conflict resolution as generalizations of fuzzy set theoretic concepts, namely, rule weighting and rule conflict resolution (via aggregation and defuzzification). In mobile robot navigation based on multiple-behavior systems, behaviors compete for control of the robot by recommending different, and possibly conflicting, control actions. This occurs frequently during any sufficiently complex navigation task. In the face of such competition, a decision must be made to determine the resultant control action given the bids from individual behaviors. In the jargon of behavior-based control the decision-making process is referred to as *behavior arbitration*. While there are a number of possible approaches to behavior arbitration, the most common approach employs a prioritization scheme wherein the control recommendation of only one behavior among several competing behaviors is taken; recommendations from the remaining (lower priority) behaviors are ignored [19, 30].

In contrast to this switching type of arbitration, we advocate controlling mobile robots using the more comprehensive arbitration scheme of behavior modulation proposed in the previous chapter. This arbitration scheme permits more than one behavior to influence the control action to the extent governed by respective degrees of applicability. The resultant control action in this case is a consensus of controls recommended by applicable behaviors. This facilitates a more natural and smoother control performance which leads an observer to describe the resulting emergent activity as behavior *fusion* [96, 102] or behavior *blending* [92]. This strategy for multiple behavior coordination was developed to enable *robust* autonomous performance. It represents an approach that is particularly suitable in the context of fuzzy-behavior hierarchies. We refer to it as behavior modulation due to its additional responsibilities of regulating, adjusting, and/or adapting individual robot behaviors to degrees conducive for realizing the current navigation goal. Several instances of independent research have converged to similar ways of approaching autonomous mobile robot navigation [93, 103, 104, 105].

In mobile robot navigation we are often concerned with behavior conflicts. For example, we can imagine the necessity to resolve conflicts between avoiding obstacles and following a wall when navigating in a cluttered corridor. Such a situation is illustrated in Figure 5.5a where the navigation objective requires following the right-wall. Assuming the robot travels at a constant speed, a representative partition of the universe of discourse for its steering control is as shown in Figure 5.5b. Based on this partition, plausible fuzzy steering output recommendations are shown in Figure 5.5c for wall-following and collision avoidance. The `wall-follow` behavior recommends proceeding in the current direction (recall that primitive behaviors serve their single purpose; they are not cognizant of the purpose of other behaviors). The `avoid-collision` behavior opts to turn left (or hard-left depending on the frontal proximity of the obstacle). Also shown is a region of overlap between these fuzzy outputs indicating a partial behavioral conflict. Due to such overlap, situations of partial conflict are not significantly distinguish-

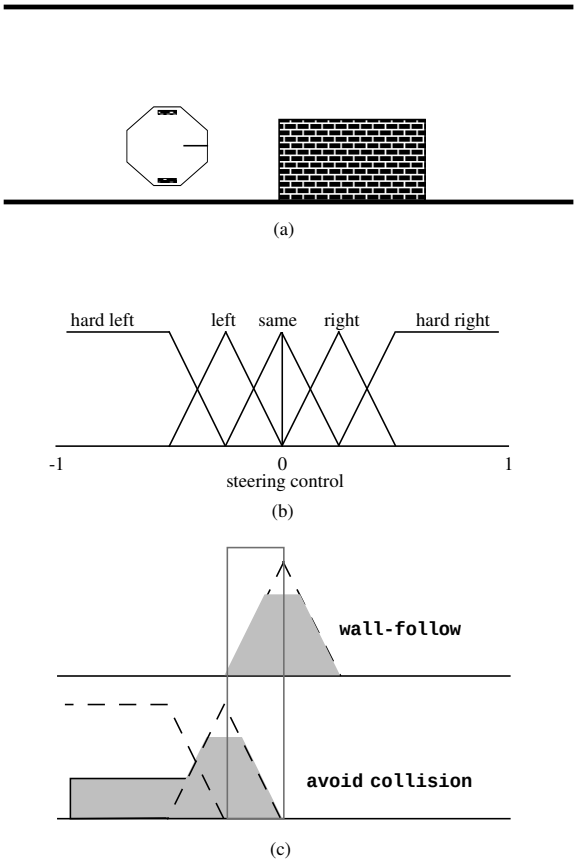


Figure 5.5: Partial behavior conflict.

able from general cases requiring behavior coordination. Therefore, partial behavior conflict is handled as described in the previous chapter.

Alternatively, consider the situation shown in Figure 5.6 in which go-to-goal and collision avoidance behaviors are interacting. In this case, the objective is to navigate to a specified goal (indicated by the 'X'). However, the robot's direct path is blocked and a decision must be made to proceed by turning right or left. Based on the same steering control partition of Figure 5.5b, plausible fuzzy control outputs are indicated in Figure 5.6b. Here, the `go-to-xy` behavior urges the robot to proceed straight towards the goal. The `avoid-collision` behavior suggests going left or right. In this case there is no overlap and the behaviors are fully conflicting. If we

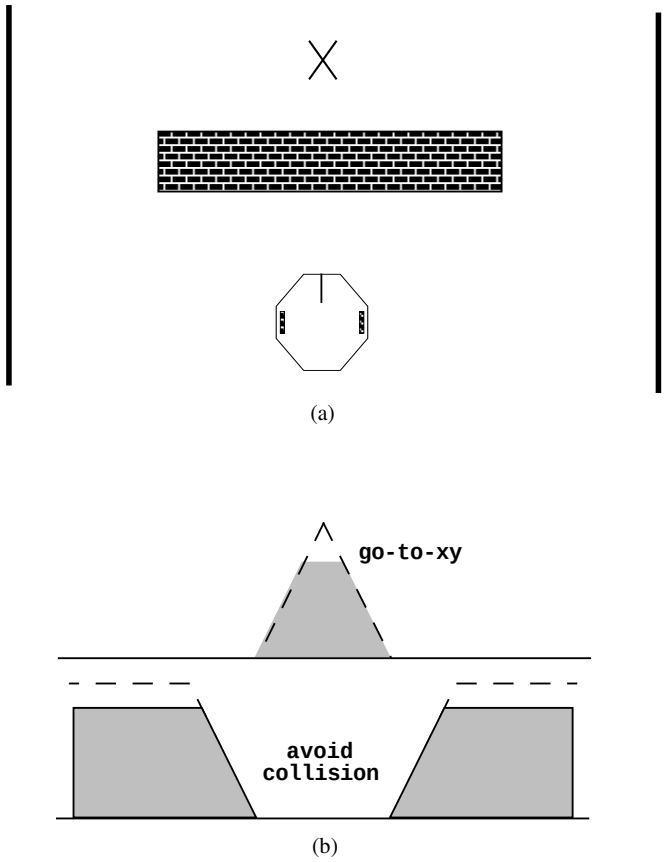


Figure 5.6: Full behavior conflict.

aggregate and defuzzify these fuzzy recommendations using common defuzzification methods (e.g. center-of-area) the result would be to proceed straight, thus leading to collision with the obstacle. This will occur even if a higher degree of applicability was assigned for obstacle avoidance. This is a limitation of defuzzification methods commonly used in fuzzy control when faced with full conflicts, but not necessarily a limitation of the proposed fuzzy control approach.

The problem just described is important not only in navigation tasks but in most tasks requiring reasoning among fully conflicting alternatives. A common solution is to select a designated *default* alternative (such as "turn right"), or to randomly select one action from

the set of conflicting alternatives. A control action could also be selected based on some criteria such as minimum required control effort. Currently, we deal with this problem by turning in the direction of most free space, or turning right if there is equal free space on either side. Thus far, default reasoning has proven to be sufficient. More flexible approaches based on fuzzy logic defuzzification methods have been proposed in the literature. Yager [106] introduced a method for defuzzification that utilizes nonmonotonic conjunction of fuzzy sets representing allowed control actions and control actions recommended by the rule-base. In the example given here, the allowed control actions are left or right; the recommended action is straight. The nonmonotonic conjunction is coupled with the random generation defuzzification method [76] to achieve defuzzification under constraints. Pfluger et al [107] also proposed a solution based on their centroid-of-largest-area defuzzification strategy. Such methods should be adopted in situations where commonsense reasoning approaches like the right-turn policy prove to be insufficient for achieving desired performance.

5.5 Ethological Influences and Relationships

Interesting parallels can be drawn between behavior modulation in the control of electromechanical systems, and ethological theories of action/behavior selection. These and other aspects of behavioral control in natural systems have influenced the development of the hierarchy of distributed fuzzy logic control in one way or another. The idea of behavior hierarchies is supported by a host of similar conceptual ethological models of motivational control of behavior such as those proposed by Tinbergen [108], Baerends [109], and MacLean [110]. Neural correlates of behavior that possess similar attributes have also been proposed [111, 112]. In fact, ideas originally expressed via theories of animal behavior are finding increasing application in approaches to robot and artificial agent control [31, 96, 113, 114, 115, 116].

The DOA, defined in Chapter 4, can be thought of in ethological terms as a *motivational tendency* of an associated behavior. It serves as a form of motivational adaptation since it causes the control policy to dynamically change in response to goals, sensory input, and internal state. As described in Chapter 4 the DOA, $\alpha \in [0, 1]$, of a given primitive behavior governs the relative amount of influence the behavior has on the instantaneous behavior of the system. As an economic interpretation, the DOAs of a set of primitive behaviors convey their respective *utilities*. The evaluation of these utilities performed by applicability rules endows the robot with motivational autonomy [117]. Behavior activation levels (DOAs), behavior modulation, and threshold activation are all concepts related to characteristics of behavior in biological systems. As implemented in the hierarchy, these mechanisms collectively allow robots to exhibit behavioral responses throughout the continuum. As mentioned earlier, this is in contrast to non-fuzzy behavior arbitration which typically employs fixed priorities that allow only one activity to influence the robot's behavior during a given control cycle. Regarding animal behavior, Lorenz [118] notes that "only a few instances are known in which the activation of one behavior system excludes absolutely the activation of any other."

During the course of any sufficiently complex navigation task, the applicability of each primitive behavior undergoes continuous nonlinear variation reflecting the level of activation recommended by the behavior control system. We will get a glimpse of this in the next chapter, where we will see that behavioral interactions caused by these concurrent variations leads to the emergence of intelligent navigation behavior via cooperation and competition. Observation of the interaction dynamics among multiple behaviors reveals bouts of cooperation and competition expressed as rapid overlapping and non-overlapping oscillations in graphs of DOA versus time. In interpreting such "fast dynamics", Varela [119] writes, "...these oscillations are the symptoms of—very rapid—reciprocal cooperation and competition between distinct agents that are activated by the current situation, vying with each other for differing modes of

interpretation for a coherent cognitive framework and readiness for action." He also points to recent brain studies [120] revealing evidence of similar phenomena. In describing animal behavior, Staddon [75] refers to competition as "reciprocal inhibition", described as "the primary principle of reflex interaction ... [which] holds for incompatible behavioral units at any level of complexity." In the context of fuzzy-behaviors, the term *soft* reciprocal inhibition seems more appropriate since competing fuzzy-behaviors are rarely fully inhibited or fully dominant.

5.6 Conclusion

The hierarchy of fuzzy-behaviors provides an efficient approach to synthesis of adaptive behavior capabilities necessary for robust autonomous navigation by mobile robots. Its practical utility lies in the decomposition of overall behavior into sub-behaviors that are activated only when applicable. When conditions for activation of a single behavior (or several) are satisfied, there is no need to process rules from behaviors that do not apply. This would result in unnecessary consumption of computational resources and possible introduction of "noise" into the decision-making process. The approach also allows filtering of undesirable inter-behavioral influences through the use of thresholds. The modularity and flexibility of the approach, coupled with its mechanisms for weighted decision-making, makes it a suitable framework for modeling and controlling situated adaptation in autonomous robots.

For many years, ethologists have developed theories and models to explain aspects of animal behavior. They are addressing a more difficult problem than the behavior synthesis problem addressed herein, namely, a behavior analysis problem based on external observations of behavior. Ideas and results of their work are quite useful as foundations for developing intelligent *robot* behavior. While they continue to focus on analysis of behavior from the outside, we concentrate on synthesis from the inside. Perhaps we will arrive at a midpoint with some unified

understanding of intelligence.

Chapter 6

Navigation Simulation and Experiment

Validation of the intelligent control architecture was done through a combination of simulation and experimentation on a real mobile robot. The emphasis is on embedded applications since most mobile robots must carry the bulk of their computational resources onboard. The simulated mobile robot is modeled after LOBOt, a custom-built robot driven by a 2-wheel differential configuration with two passive casters for support. The independent drive motors on LOBOt are geared DC motors. As shown in Figure 6.1, it is octagonal in shape, stands about 75 cm tall and measures about 60 cm in width. Range sensing is achieved using a layout of 16 ultrasonic transducers (arranged primarily on the front, sides, and forward-facing obliques); optical encoders on each driven wheel provide position information used for dead-reckoning. In the simulations, ideal pose (position and orientation) information $(x\ y\ \theta)^T$ is assumed and is computed using a kinematic model of the differential-drive mechanism. Its maximum speed was limited to $0.3m/s$. The sensor model generated range readings with errors as large as $\approx 100mm$ and lower angular resolution than the actual sonar. The output of the primitive behaviors are

Figure 6.1: UNM LOBOt.

right and left wheel speeds; the inputs to the hierarchy are the goal location and subsets of sensor readings. LOBOt is controlled using a 75MHz Pentium-based master processor (laptop PC) and Motorola MC68HC11 microprocessor slaves for sonar processing and low-level motor control functions. The low-level motor control is of the conventional PI (proportional-integral) type.

The structure of the behavior hierarchy was determined based on a subjective assessment of the motion capabilities necessary for goal-directed navigation. Goal-seeking and route-following capabilities are demonstrated for which underlying fuzzy-behaviors at the primitive level have been hand-derived. Behavior evolution has been applied to composite behaviors (Section 6.2) to discover applicability rule-bases responsible for appropriately modulating primitive behaviors.

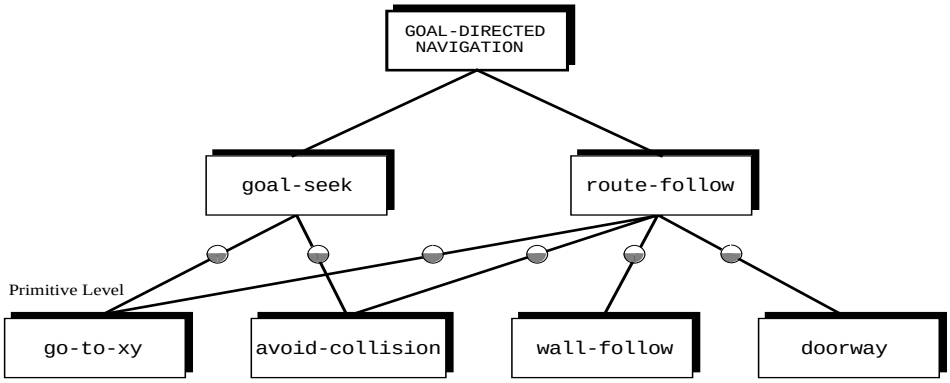


Figure 6.2: Hierarchical decomposition of mobile robot behavior.

6.1 Simulated Navigation Results

In this section, we examine representative simulations of adaptive behavior controlled by the fuzzy-behavior hierarchy. The simulated "world" is a hypothetical indoor layout not unlike a warehouse or office building. The robot is not provided with an explicit map, however, it is cognizant of the notion of a two-dimensional Cartesian coordinate system. Its path is not pre-planned; it is executed in response to instantaneous sensory feedback.

6.1.1 Goal-seeking

In order to demonstrate the operational aspects of the controller in the simplest manner possible we concentrate on navigating to a specified goal utilizing the composite behavior — `goal-seek`. It's place in the overall hierarchy is illustrated in the left portion of Figure 6.2 which shows that its effect arises from synergistic interaction between `go-to-xy` and `avoid-collision` behaviors. These primitive behaviors (and others shown) have been independently developed and tested in simulation to predict their individual performances in indoor spaces with various obstacle arrangements. When more behaviors are involved the approach proceeds in a straightforward manner by appending additional DOAs and any necessary antecedents to applicability rules

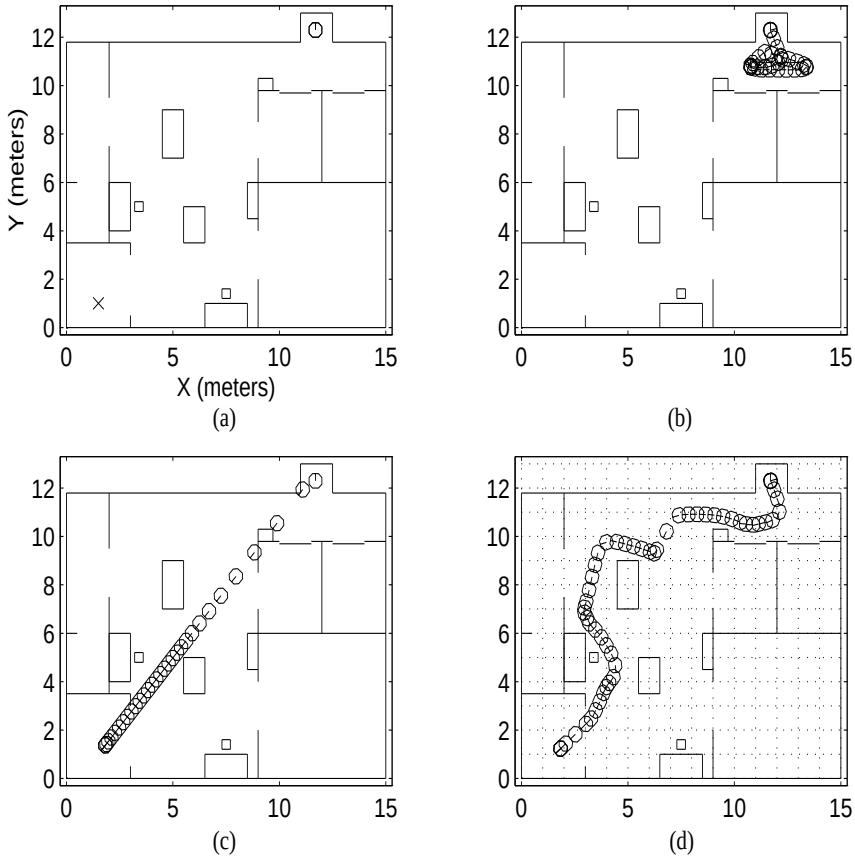


Figure 6.3: Simulation of goal-seeking behavior.

accordingly.

The initial state of the simulation is shown in Figure 6.3a with LOBOt located at a docking/charging station with pose $(x\ y\ \theta)^T = (11.7\ 12.3\ \frac{\pi}{2})^T$. Its task is sensor-based navigation to a goal located at, $(1.5,\ 1)$ and marked by the X. The primitive behaviors are each shown acting alone in Figure 6.3 b and c. Recall that these behaviors are only capable of exhibiting their particular primitive roles, lacking awareness of the other behaviors in the system and the stimuli that drive them. Thus, `avoid-collision` merely displays cyclic collision-free behavior in the immediate vicinity of the robot's initial location, while `go-to-xy` displays a taxic reaction that propels the robot toward the goal irregardless of obstacles in its path. Success-

ful completion of the task, resulting from adaptive coordination of the primitive behaviors, is shown in Figure 6.3d. In the current implementation, applicability rules used by `goal-seek` to modulate the underlying primitive behaviors consider three instantaneous input states — the range to the nearest obstacle ($r_{\min}$), the distance from the goal (d_{goal}), and the angular heading to the goal (θ_{goal}). Thus, the fuzzy rules which assign DOAs to primitive behaviors are of the form of the following examples:

IF $r_{\min}$ *is* DZONE *and* d_{goal} *is* NOTSMALL
THEN α_{ac} *is* HIGH; α_{gt} *is* ZERO

IF $r_{\min}$ *is* FAR *and* d_{goal} *is* MED
THEN α_{ac} *is* LOW; α_{gt} *is* HIGH

IF $r_{\min}$ *is* NEAR *and* θ_{goal} *is* RIGHT
THEN α_{ac} *is* HIGH; α_{gt} *is* LOW

where uppercase symbols are linguistic values represented by fuzzy sets defined over appropriate input/output universes of discourse. The lingusitic label, $DZONE$, of the first example rule refers to a "danger zone" within which obstacles are too close to the robot and the situation is perhaps unsafe. The consequent linguistic variables, α_{gt} and α_{ac}, correspond to the DOAs of `go-to-xy` and `avoid-collision` respectively. The linguistic values in the rule antecedents adequately convey the uncertainty and imprecision that is characteristic of sensors used on mobile robots. Linguistic values in the consequents are fuzzy partitions defined over the universe [0, 1].

In Figure 6.4, the behavioral modulation during the simulation is shown as the temporal evolution of the DOAs, or motivational tendencies, of each primitive behavior. The interaction dynamics shows evidence of brief bouts of competition (overlapping oscillations) and

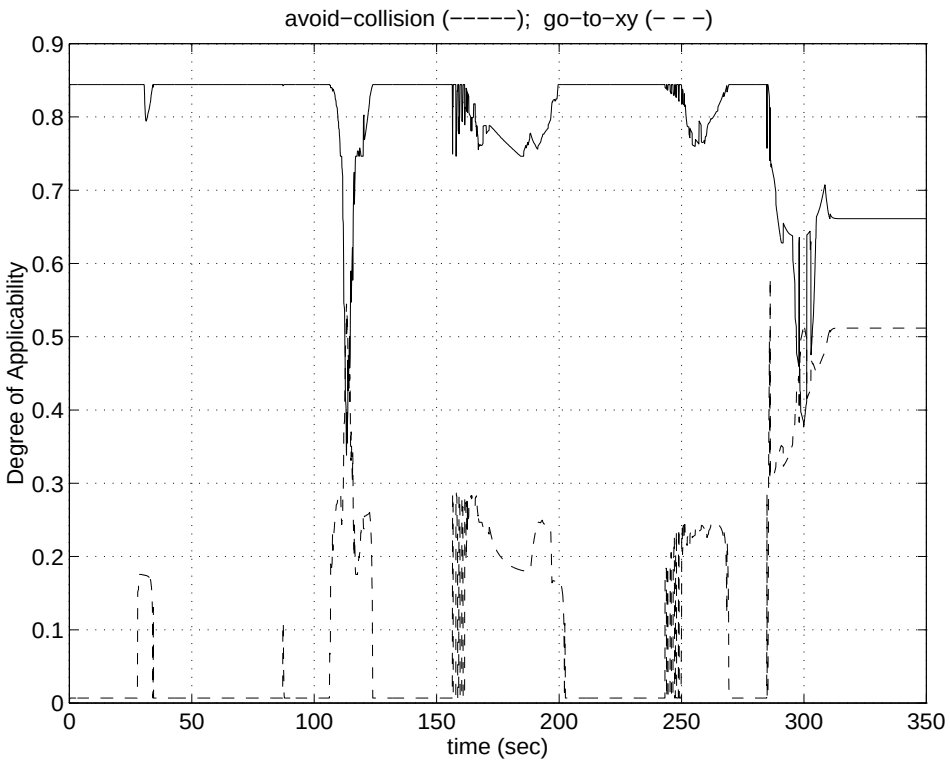


Figure 6.4: Behavior modulation and interaction during goal-seeking.

cooperation with varying levels of dominance. These are characteristic of the "fast dynamics" mentioned in Section 5.5 and referred to by Varela [119]. Initially, `avoid-collision` has the dominant influence over the robot's motors due to the close proximity of walls in the docking/charging station. It virtually maintains dominance throughout the task due to the relative clutter in the environment. The first bout of competition corresponds to the robot's approach towards the obstacle located at $(5, 8)$; the second bout occurs as it enters the goal room. Elsewhere, the applicabilities vary continuously reflecting the levels of activation recommended by the behavior control system.

6.1.2 Effect of t-conorm on motion decisions

An additional flexible feature of the architecture lies in the choice of an appropriate operator for consolidating multiple control recommendations. We focus on the t-conorm, or generalized fuzzy union operator of fuzzy set theory. Recall that primitive rule-base outputs are fuzzy sets, and an aggregation across rule-bases must be performed to produce an overall control output. As the selection of the t-conorm used for *rule-base* aggregation dictates how anything approaching a consensus will be made, available options should be considered.

We consider the following t-conorms: bounded sum, arithmetic maximum, probabilistic sum, and the Sugeno S_λ family ($\lambda \geq -1$) which is one of a variety of parameterized families of aggregation operators [72]. Their definitions follow respectively, where $a \in [0,1]$, $b \in [0,1]$ and $U(a,b)$ denotes the t-conorm operator.

$$U(a,b) = \min(1, a+b) \tag{6.1}$$

$$U(a,b) = \max(a,b) \tag{6.2}$$

$$U(a,b) = a + b - ab \tag{6.3}$$

$$U(a,b,\lambda) = \min(1, a+b+\lambda ab) \tag{6.4}$$

The selection of the above set of t-conorms was based on their computational simplicity (i.e. no division or exponent operations required). The simulated navigation was run using each of the operators defined above to examine the relative impact that each has on motion decisions made during the run. That is, the fuzzy outputs of `go-to-xy` and `avoid-collision` were aggregated using Equations (6.1)–(6.4). The resulting path taken by the robot using the bounded sum is shown in Figure 6.5a. This is the same path taken in the previous example — Figure 6.3d where the arithmetic sum, $\uplus$, was used as the t-conorm. The robot simultaneously achieves the goals of reaching the target location *and* avoiding collisions. The paths

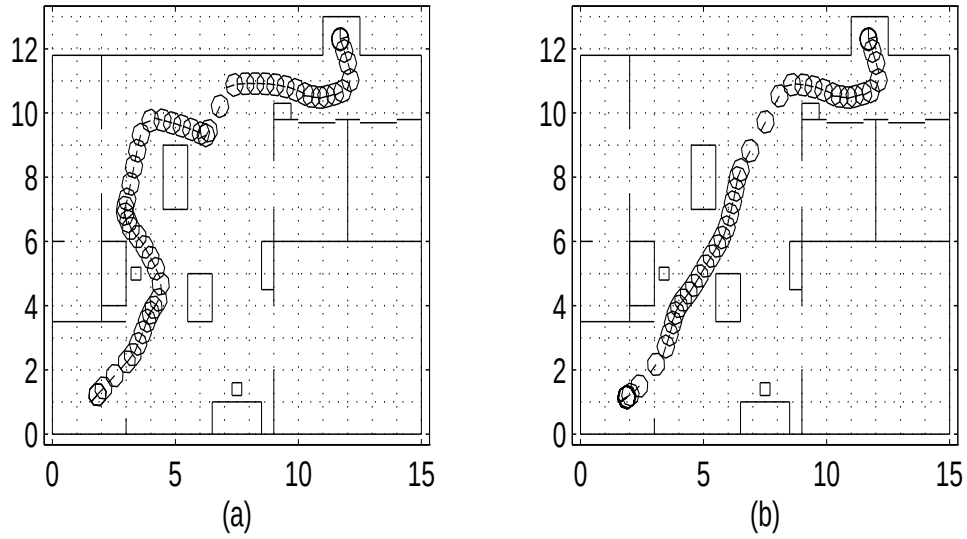


Figure 6.5: Goal-seeking using different t-conorms.

resulting from using maximum and probabilistic sum were very similar to the bounded sum case. However, the decisions made as a result of applying the S_λ family for $\lambda \geq 1$ were clearly different as revealed by the alternative path shown in Figure 6.5b for $\lambda = 1$. The ensemble of control decisions made over the course of this run led to a more direct path to the goal. The results were similar for $\lambda > 1$. Thus, possible variations in system behavior can be determined through examination of the effects of the chosen fuzzy union operator on multi-behavior decision-making.

6.1.3 Route-following

In the hierarchy of Figure 6.2 `route-follow` employs capabilities of several primitive behaviors. We demonstrate its performance in a navigation task utilizing the same primitives with the addition of `wall-follow`. For this example different start and goal locations are used and a designated route is specified by three additional waypoints to the goal. The initial state is $(x\ y\ \theta)^T = (10m\ 5m\ -\frac{\pi}{2}rad)^T$; the goal is located at $(1.2m, 5.2m)$. Waypoints between these

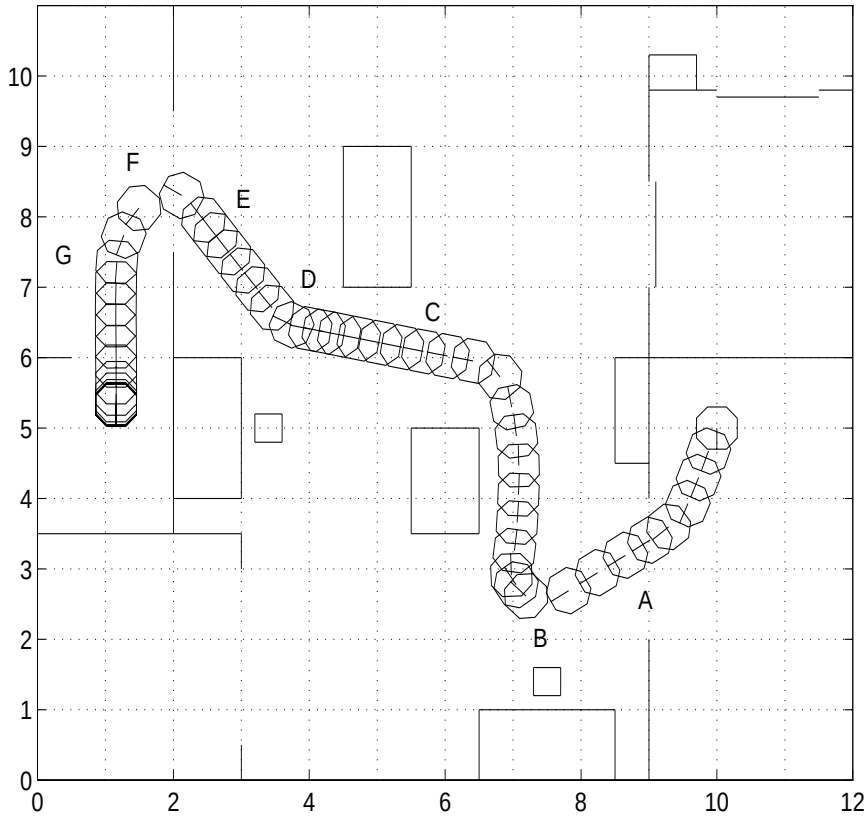


Figure 6.6: Route-following using waypoints.

locations are

$$(7.5, 2.5) \rightarrow (3.5, 6.5) \rightarrow (2.0, 8.5)$$

The resulting route is shown in Figure 6.6 and the corresponding DOAs for each primitive behavior are shown separately in Figure 6.7. Labels A–G in each figure indicate a correlation between robot position along the route and the DOAs applied at that instant.

At point A as the robot exits the start room all three primitive behaviors compete for control. At B α_{ac} takes over as the dominant behavior while approaching the first waypoint. After avoiding an obstacle, α_{gt} becomes dominant at C on approaching the second waypoint. Dominance modulates between α_{ac} and α_{gt} through competition while traversing through D and E where the robot adjusts its heading towards the goal room. Interactions among the

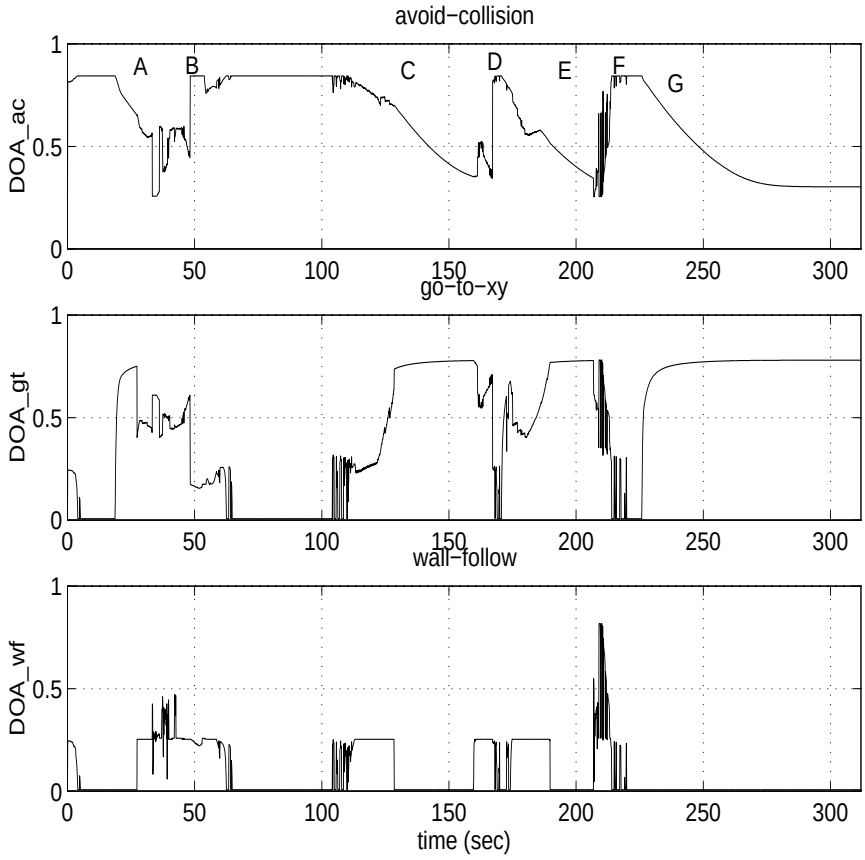


Figure 6.7: Behavior modulation during route-following.

three primitive behaviors resurfaces at F where α_{wf} briefly takes over. It becomes inactive at G giving way to α_{ac}, and finally to α_{gt} on direct approach to the goal. During the majority of the task each primitive behavior is active to varying degrees influencing the overall robot behavior in response to goals, sensory input, and internal state.

6.2 Evolution of Intelligent Behavior Modulation

At this point is it clear that controlling goal-directed behavior in an autonomous vehicle is possible using the adaptive hierarchy of distributed fuzzy control. Additional primitive and composite behaviors can be added to a system to increase its functionality as long as associ-

ated applicability rules can be formulated that relate the composite behaviors to behaviors in the primitive level. In order to formulate suitable coordination rules for behavior modulation, one must first decide what the DOAs of low-level primitive behaviors should be in all practical situations perceived from sensory input. Formulation of such rules is not entirely intuitive, and expert knowledge about how to concurrently coordinate primitive behaviors is not readily available. Humans often find it difficult to design knowledge-based control systems with interacting rule-bases, particularly in the absence of experts or sufficient knowledge of the problem. Moreover, practical experience has revealed that fuzzy control alone is sometimes insufficient for addressing complex intelligent control problems of robotics. It is often necessary to adopt hybrid solutions [121, 122].

In this section, we address the problem of automatic discovery/learning of coordination, or applicability, rules for use at the composite behavior level of the hierarchy. This problem has been previously approached in the contexts of other coordination schemes by using reinforcement learning [123] and hybrids of reinforcement and neural networks [124, 125]. In Chapter 3 the potential of the genetic programming paradigm was demonstrated for learning fuzzy rule-bases for low-level regulation and tracking types of problems. Building on that foundation, we apply the approach here to higher-level behavior modulation.

Recall from Chapter 3 that in the process of learning fuzzy rules, GP manipulates the linguistic variables directly associated with the fuzzy-behaviors. The function set consists of components of the generic fuzzy if-then rule and common fuzzy logic connectives, i.e. functions for antecedents, consequents, fuzzy intersection, rule inference, and fuzzy union. Each behavior (rule-base) is an executable program that evaluates to an output fuzzy set resulting from fuzzy inference. The terminal set is made up of the input and output linguistic variables and pre-specified membership functions associated with the desired behavior.

6.2.1 Steady-State GP

In addition to the generational GP process, we also employ non-generational Steady-State Genetic Programming (SSGP). SSGP has been successfully applied by Reynolds [61], as well as Nordin and Banzhaf [126], to the robot behavior evolution problem. More recently, it has been applied to the evolution of behavior coordination and action selection [127, 128].

In the SSGP approach the concept of "generations" does not exist. Instead, on each iteration following creation of the initial population only a few offspring are produced. The offspring replace the worst few individuals in the population, and the cycle repeats until termination criteria are satisfied. This is the general idea. However, methods for selecting parents to breed, creating new offspring, determining worst individuals, and replacing worst individuals tend to vary across applications. In the variant applied here, two parent behaviors are selected by tournament (of size 3) to produce two offspring. The fitnesses of the two offspring are evaluated and they are added to the population. Behaviors to be removed are chosen randomly from the set of below-average behaviors in the current population. Syswerda [69] suggests that steady-state evolution provides automatic elitism, and allows for an aggressive learning rate without jeopardizing what is good in the population.

6.2.2 Behavior fitness evaluation

During evolution each behavior in the current population is evaluated via simulation in a number of indoor *fitness cases* subject to an upper time limit of 200 seconds. In this work, $n_f = 5$ fitness cases are used; the simplest and most difficult of these are illustrated in Figure 6.8(a) and 6.8(b) respectively. Goal locations in the figure are indicated by an X, the robot is depicted as an octagonal icon with a radial line designating its initial heading, and its range sensor horizon is indicated by the shaded regions of Figure 6.8(a). In each case, the dimension

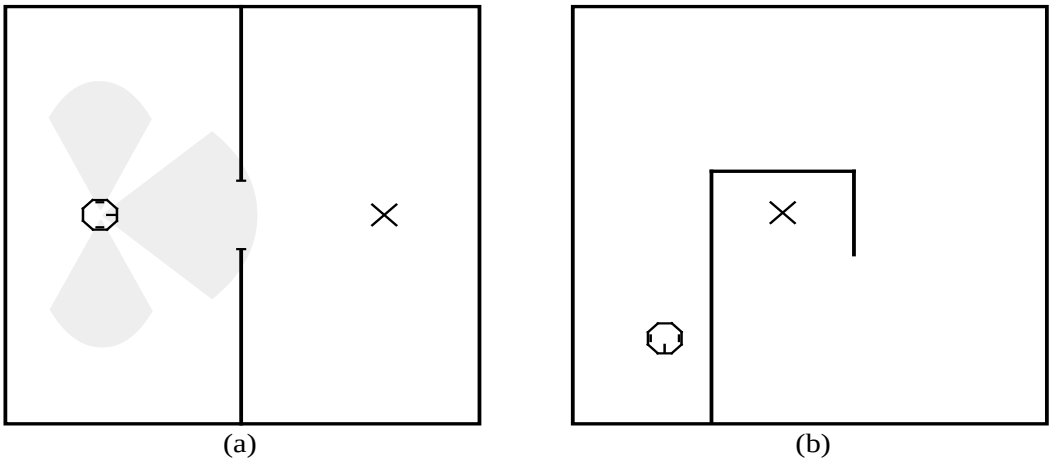


Figure 6.8: Example fitness cases.

of the indoor space is $10m \times 10m$. Each fitness case was chosen to represent situations likely to be encountered in indoor environments.

For a given behavior, the score of a trial run through fitness case i is given by

$$S_i = \begin{cases} 100 & ; \quad \textit{goal reached} \\ \frac{100}{\gamma(1+10e_N)} & ; \quad \textit{otherwise} \end{cases} \tag{6.5}$$

where e_N is the normalized residual distance to the goal in the case of a time-out or collision. The parameter $\gamma = 2$ if a collision occurs; otherwise $\gamma = 1$. That is, the score for an unsafe trial is half of that for a collision-free trial with all else being equal (see Figure 6.9). The overall fitness of the behavior is the average score over all n_f fitness cases:

$$F = \frac{1}{n_f} \sum_{i=1}^{n_f} S_i \tag{6.6}$$

Thus, the highest possible score, and hence fitness, is 100.

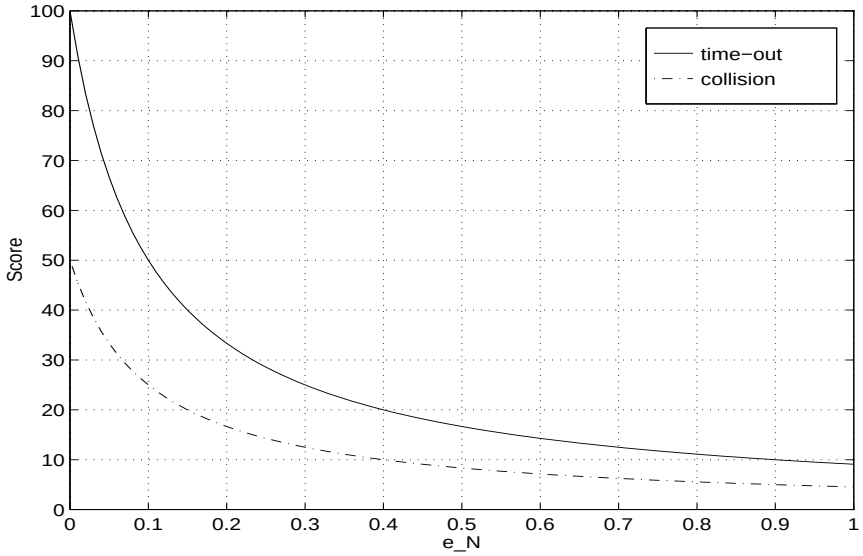


Figure 6.9: Behavior fitness case scoring function.

6.2.3 Evolved behavior modulation

The simulated world is considerably more general than any one of the fitness cases used during the evolution process and, thus, provides a suitable environment to test the generalization capability of the evolved behaviors. The GP system was run using population sizes of 10–20 rule-bases for a number of generations ranging from 10–15. Recall that in GP, genetic diversity remains high even for very small populations due to the tree structure of individuals [7]. Steady-state GP was also applied using a population size of 20. Results of runs using both approaches are summarized graphically in Figure 6.10. The mean performance of GP over five runs is shown, in the left half of Figure 6.10, as the progression of the population average fitness during the first ten generations. The right half of Figure 6.10 shows the progression of the average fitness of the current population at each iteration. Twenty behaviors were processed in the initial population; thereafter, two new behaviors evolved at each iteration. A trend towards higher fitness is evident for both GP and SSGP. Table 6.2.3 lists some quantitative details about the best behavior evolved by each approach. The run which produced the best

	Population size	#Evaluations	#Rules	Best fitness	Success rate
GP	10	150	11	86.5	70%
SSGP	20	108	9	87.8	83%

Table 6.1: Best evolved composite `goal-seek` behaviors.

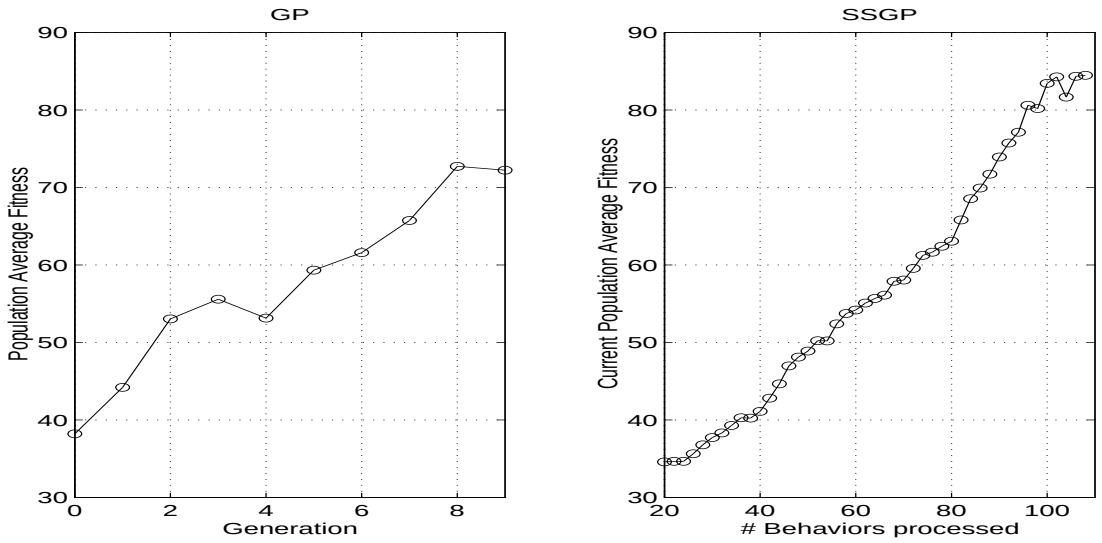


Figure 6.10: Mean performance of GP and SSGP evolution.

GP-evolved behavior was terminated after 15 generations. Since SSGP is non-generational, a corresponding number of "generations" could not be listed for comparison. Instead, the amount of processing done by each approach is listed as the total number of fitness evaluations performed. The success rate was determined from navigation runs in three different simulated domains not included in the set of fitness cases. For this problem good regions of the search space were discovered with less processing by SSGP.

Having pointed out some operational details of the behavior hierarchy, let us compare the performance of the hand-derived `goal-seek` behavior to a behavior evolved for the same purpose. We will consider an arbitrary point-to-point navigation task from initial state $(1\ 11\ -\frac{\pi}{2})^T$ to a goal located at (13.5 4.5). The successful path executed by the hand-derived behavior is shown in Figure 6.11 along with the corresponding behavior modulation history. We compare

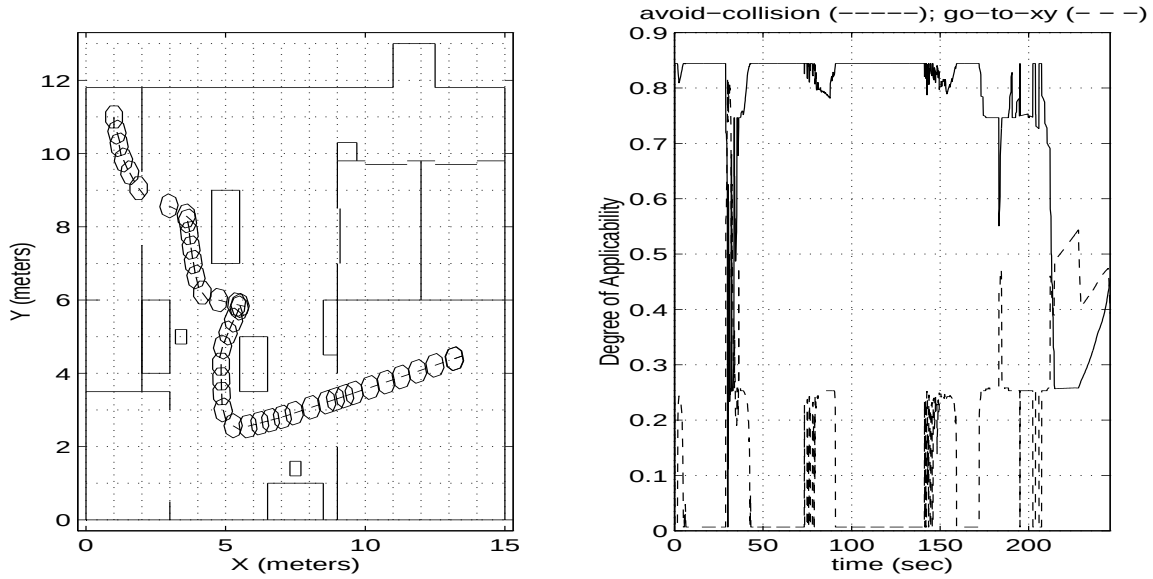


Figure 6.11: Hand-derived coordination and behavior modulation.

this with the same task executed by the best SSGP-evolved behavior shown in Figure 6.12. The evolved behavior coordination results in a more direct path to the goal due to higher motivation applied to `go-to-xy`. The resulting path in this example is executed about 20% faster than

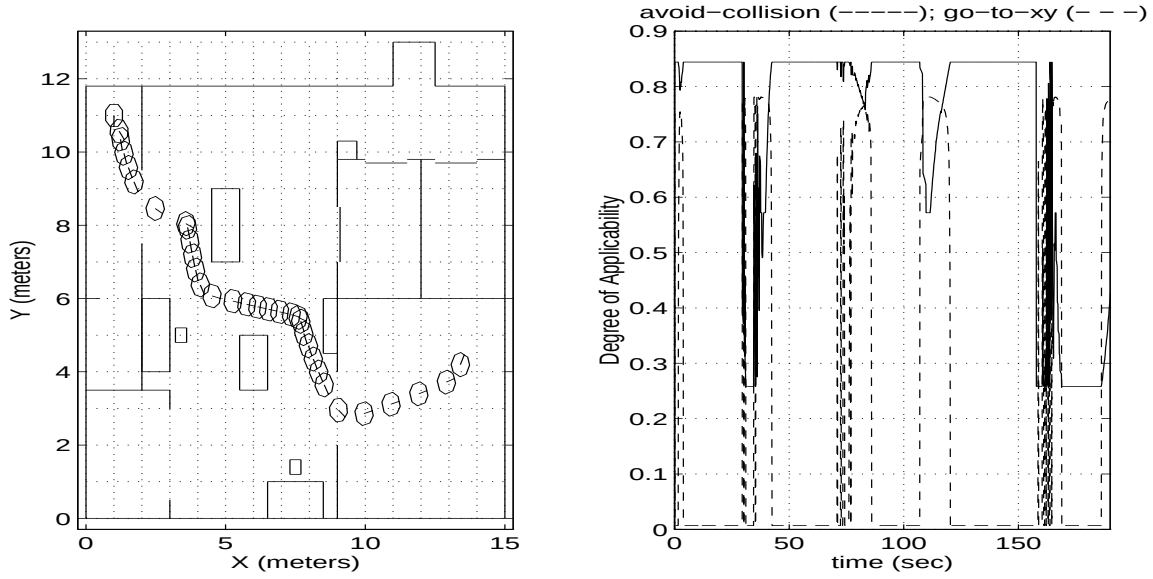


Figure 6.12: SSGP-evolved coordination and behavior modulation.

the path taken via hand-derived coordination. We also note that the behavior modulation demonstrated by the evolved behavior is more complex. Near uniform bouts of competition and cooperation throughout the task are evident in the decision-making, thus leading to similar amounts of behavioral influence for each primitive behavior. As listed in Table 6.2.3, this was achieved using less applicability rules than both the hand-derived behavior and the best GP-evolved behavior. For an identical navigation task, the relative levels of activation induced by the SSGP-evolved `goal-seek` behavior more closely resembles a consensus.

6.3 Real World Experiments: Goal-seeking

LOBOt was built by students who were not formally trained in machine design techniques. As a result, the rover was assembled without strict regard to proper mechanical tolerances or precision. The rover is plagued with misalignments. Furthermore, the wheelbase (the distance between the points of contact of the two driven wheels and the floor) is not fixed as it should be. This is due to a loose fit of the left wheel on its motor shaft which allows the wheel to slide outward along the shaft in an unpredictable manner. A change in the effective wheelbase of about half an inch results. Given the host of mechanical imperfections built (inadvertently) into the vehicle, it represents a major challenge for fuzzy logic-based navigation control. The results of actual experiments on LOBOt demonstrate tolerance for imprecision and uncertainty in the adaptive hierarchy of distributed fuzzy control.

In the current implementation, the cycle time of the intelligent controller is 0.15 seconds (7 Hz). This time includes the time spent acquiring and preprocessing sonar data, and commanding the motors. Acquisition of sonar data is the major bottleneck of the control loop. These data (16 range readings) are acquired serially from a microcontroller external to the master processor at a transmission rate of 9600 baud. Motor commands are issued via the master

processor parallel port which consumes less than a millisecond. Wheel encoder readings are also acquired via the master processor parallel port (it is bi-directional) within one millisecond. Without these control interface functions, the overall inference of the adaptive behavior hierarchy takes about 0.05 seconds. That is, the hierarchy itself can run at a rate of 20Hz.

As in the simulations, the robot is not provided with a map. However, it is cognizant of the notion of a two-dimensional Cartesian coordinate system. Its paths are not pre-planned; they are executed in response to instantaneous sensory feedback from the environment. Therefore, we are essentially dealing with a local navigation problem as opposed to a global navigation problem which relies on a global map that is either provided a priori, or is acquired via exploration. Results presented here show that the fuzzy logic-based local navigation control is useful in situations where maps are not available or are perhaps unreliable. Fuzzy control also lends itself well to global navigation and map-based path planning [129].

Two representative results of goal-seeking by LOBOt are presented here. The experiments were conducted in an indoor environment consisting of corridors and doors.[1] The rover's task is to navigate from one location to another on the same floor of the building. Relatively short traverses are presented since the poor dead-reckoning of the vehicle prohibited long traverses without large accumulations of pose errors.

The first experimental result is shown in Figure 6.13. LOBOt was commanded to navigate from one hallway to an adjacent hallway. The total path length is approximately 15 meters. The `avoid-collision` and `go-to-xy` behaviors were modulated as depicted in Figure 6.14 where it is clear that `avoid-collision` dominates throughout the task. This is due to the ever-present corridor walls which are in close proximity to the rover, thereby causing the DOA of `avoid-collision` to remain high. The actual start and goal locations are: $(x\ y\ \theta)^T =$

[1] The first floor of the Electrical and Computer Engineering building at the University of New Mexico.

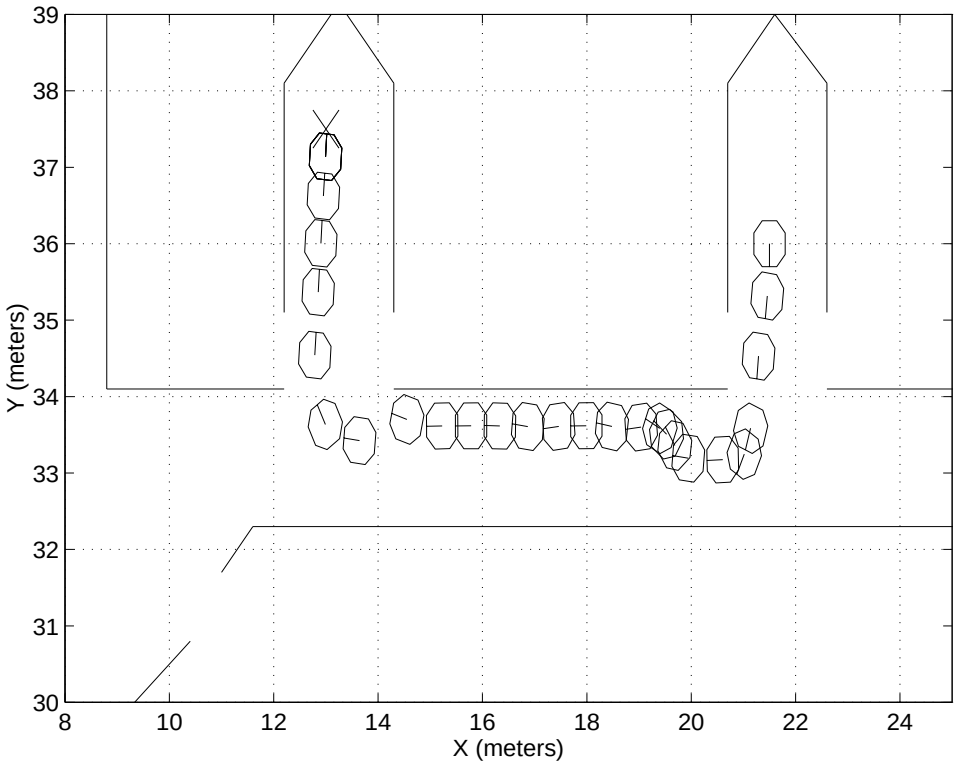


Figure 6.13: Experiment: Short goal-seeking task.

$(21.5m\ 36m\ -\frac{\pi}{2}rad)^T$ and $(13m, 37.5m)$.

The last experimental result is a longer traverse which is almost twice as long as the path just discussed. The start and goal locations are: $(x\ y\ \theta)^T = (9.5m\ 22m\ 3.0rad)^T$ and $(21.5m, 37.5m)$. As shown in Figure 6.15, LOBOt successfully navigates to the goal. Note that this experiment required human intervention at three points along the path to assist in updating the actual position of the vehicle. Dead-reckoning errors accumulated during the traversal were too large for LOBOt to have successfully reached the goal alone. Figure 6.16 shows the behavior modulation history during the navigation task. As expected, `avoid-collision` dominates throughout with only one bout of competition with `go-to-xy` as LOBOt approaches the goal hallway.

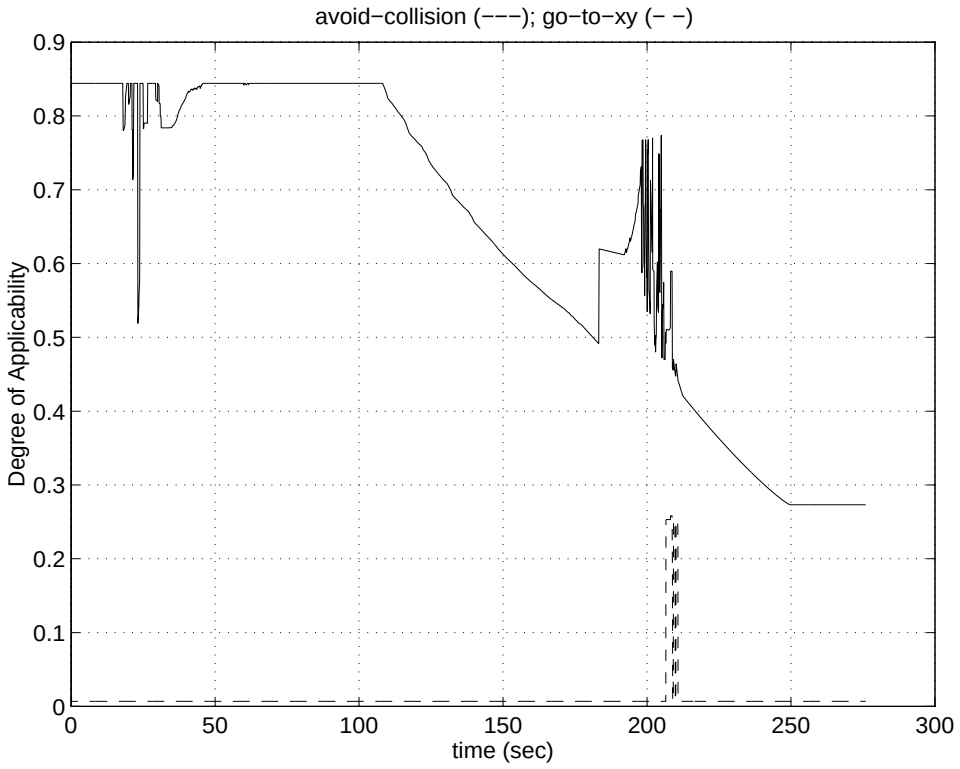


Figure 6.14: Experiment: Behavior modulation during short goal-seeking task.

6.4 Conclusion

Simulation and experimental results both show the utility of the adaptive hierarchy of distributed fuzzy control for autonomous local navigation. The success of the representative real-world results is noteworthy given the mechanical imperfections of the actual rover which add uncertainty and imprecision to an already difficult problem.

Observation of operational aspects of the approach reveals interesting properties that support phenomena formerly observed in the behavior of natural systems. This approximate reasoning-based approach to behavior control appears to have potential as a conceptual model of intelligent behavior and behavioral relationships. It is useful for autonomous sensor-based navigation involving both goal-seeking and route-following.

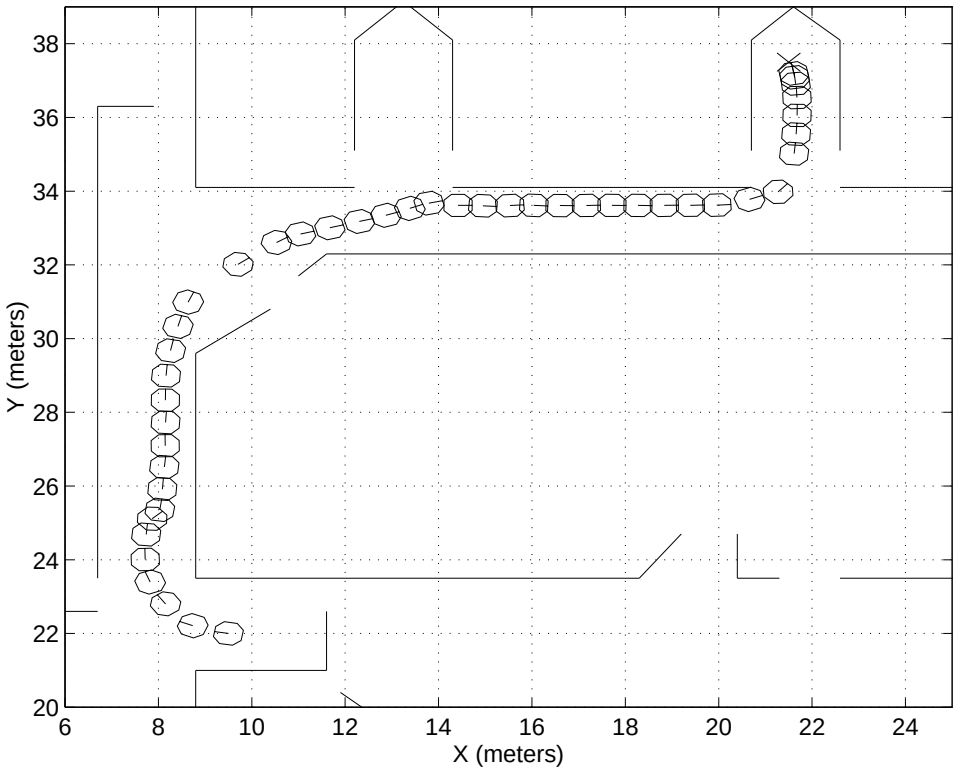


Figure 6.15: Experiment: Long goal-seeking task.

Genetic programming proved useful for learning fuzzy-behaviors at the coordination level of the hierarchy. In particular, rules of composite behaviors were evolved for coordinating low-level fuzzy-behaviors which reside at the primitive level. Conventional GP and steady-state GP were applied, each yielding good results for small populations. Overall, SSGP yielded slightly better results for goal-seeking coordination. Using only five fixed fitness cases during behavior evolution modest generalization capabilities were exhibited by the highest fit behaviors. An SSGP-evolved behavior showed a better behavior modulation capability than both the hand-derived behavior and the best GP-evolved behavior. It is expected that additional improvement can be achieved using a richer set of fitness cases.

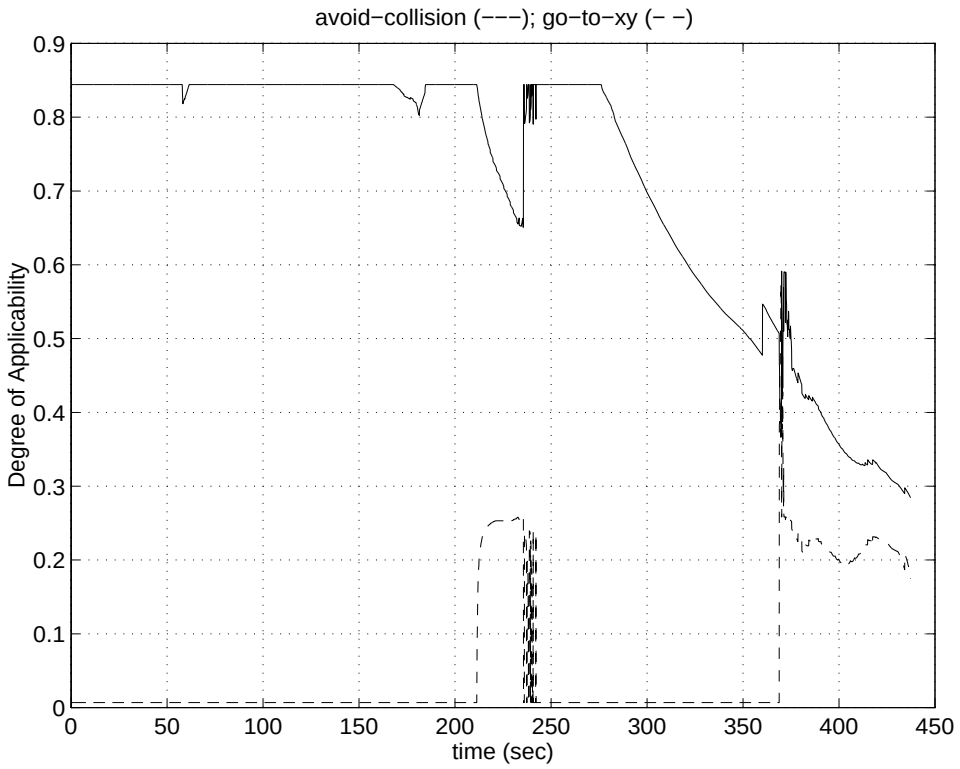


Figure 6.16: Experiment: Behavior modulation during long goal-seeking task.

Chapter 7

CONCLUSIONS

In the preceding chapters, the use of fuzzy logic has been advocated for developing intelligent autonomous control systems that interact with the real world. Several limitations of the canonical FLC were discussed. Specifically, the combinatorial effect of large rule bases degrades real-time performance; systematic approaches that have been recently proposed as FLC design methodologies are indirect; and the canonical FLC has no provision for adaptability. In response to these limitations, this dissertation makes the following contributions:

A hierarchical structure that accommodates multivariable systems by distributing intelligence among multiple rule-bases.

A computational mechanism which provides adaptability to fuzzy control systems via multi-rule-base coordination.

An automatic approach to fuzzy rule-base design.

In order to exploit the power of fuzzy logic for controlling complex autonomous systems that interact with the real world, an intelligent control architecture was proposed. The proposed

architecture is a hierarchy of distributed FLCs that accommodates large rule-base cardinality; it employs a mechanism for controller adaptation, and the genetic programming paradigm for direct rule-base design. The application domain was behavior control of rovers (mobile robots). A theoretical framework has been described that forms a hybrid of fuzzy logic, genetic programming, and behavior control. As such, this dissertation research combines possibilistic and probabilistic approaches to the control of behavior in dynamic systems, and is based on considerations from approximate reasoning, evolutionary computation, and ethology. The goal is to enable the realization of truly autonomous systems.

The preceding chapters have shown that the current approach to fuzzy control can be extended to, more effectively, deal with multivariable systems which require many rules. In addition, the research has demonstrated that genetic programming, supported by a suitable constrained syntactic structure, is a flexible method for automatic design of fuzzy rule-bases. Genetic programming is used off-line to learn suitable coordination rules for modulating underlying primitive behaviors. When the rules are incorporated into the behavior hierarchy, the primitive level adapts dynamically in real-time due to fluctuations in the DOAs from cycle to cycle. This allows the system to compensate for local changes in the environment as perceived by instantaneous sensory feedback. The adaptive hierarchy of distributed fuzzy control provides an efficient approach to synthesis of adaptive behavioral capabilities necessary for robust autonomous control. Its practical utility lies in the decomposition of overall behavior into sub-behaviors that are activated only when applicable. This facilitates practical application to real-time control. When conditions for activation of a single behavior (or several) are satisfied, there is no need to process rules from behaviors that do not apply. Processing rules from irrelevant behaviors would result in unnecessary consumption of computational resources and possible introduction of "noise" into the decision-making process. Furthermore, for multi-input systems, immediate benefits result from distributing intelligence among multi-

ple rule-bases. Namely, the resulting modularity facilitates the design of intelligent controllers for such complex systems, and dramatic reductions in total number of rules in the system is realized.

The theoretical foundation of the approach was developed based on generalizing fuzzy set theoretic concepts of rule weighting and conflict resolution to *rule-base* weighting and conflict resolution. This was facilitated by the fact that mathematical operations of fuzzy inference in FLCs are closed for fuzzy sets. This fact served as the basis for extending fuzzy set and logic operations used for monolithic fuzzy control to multi-rule-based fuzzy control. The underlying theory enabled the introduction of the concepts of behavioral degree of applicability and behavior modulation. Results of autonomous rover navigation have verified the theoretical approach in both simulation and the real world. Autonomous navigation in unstructured and non-engineered environments is a complex control problem which involves achieving multiple goals, conflict resolution, and multiple interacting behaviors. The adaptive hierarchy proved to be an effective intelligent control solution to this problem. The navigation results presented in this dissertation apply only to autonomous navigation in indoor environments. The architecture can be applied for outdoor navigation as well. In future research, implementations for outdoor and rugged terrain vehicles such as wheeled and legged planetary rovers should be considered.

A significant feature of the architecture which could be the focus of future extensions is behavior threshold activation. Thresholds imposed on degrees of applicability would allow filtering of undesirable inter-behavioral influences. Threshold activation of behaviors has not been fully exploited in the reseaarch reported here. The feature remains as an additional degree of freedom of the architecture which deserves further attention. In general, thresholds for behavior activation are difficult to choose. The problem is similar to that of specifying degrees of applicability for behavior modulation. This has been addressed here using genetic program-

ming. If activation thresholds for behaviors are meant to vary, they can also be determined using fuzzy rules evolved by genetic programming. On the other hand, if static thresholds are employed, genetic algorithms or reinforcement learning could be used. In any case, the threshold activation feature coupled with the mechanism for weighted decision-making provided by the degree of applicability leads to a strong framework for situated adaptation in autonomous systems.

Observation of operational aspects of the controller reveals interesting properties that support phenomena formerly observed in the behavior of natural systems. That is, the rapid overlapping oscillations in the degrees of applicability of active behaviors resembles measured signal activity in the cerebral cortex of animals during transitions from one activity to another [120, 130]. This is an interesting coincidence which suggests that the proposed approach to system behavior control has potential as a conceptual model of intelligent behavior and behavioral relationships. For many years, ethologists have developed theories and models to explain aspects of animal behavior. They are addressing a more difficult problem than the behavior synthesis problem addressed herein, namely, a behavior analysis problem based on external observations of behavior. Ideas and results of their work are quite useful as foundations for developing intelligent autonomous system behavior. While they continue to focus on analysis of behavior from the outside, we concentrate on synthesis from the inside. Perhaps we will arrive at a midpoint with some unified understanding of intelligence.

Bibliography

[1] Panos J. Antsaklis and Kevin M. Passino (Eds.). *An Introduction to Intelligent and Autnonomous Control.* Kluwer Academic Publishers, Boston, MA, 1993.

[2] F. Aminzadeh and M. Jamshidi (Eds.). *Soft Computing with Fuzzy Logic, Neural Networks and Distributed Artificial Intelligence*, volume 4 of *Environmental and Intelligent Manufacturing Systems.* Prentice Hall, Englewood Cliffs, NJ, 1994.

[3] Yuval Davidor. *Genetic Algorithms and Robotics: A Heuristic Strategy for Optimization*, volume 1 of *Robotics and Automated Systems.* World Scientific Publishing Co., NJ, 1991.

[4] C.J. Harris, C.G. Moore, and M. Brown. *Intelligent Control: Aspects of Fuzzy Logic and Neural Nets*, volume 6 of *Robotics and Automated Systems.* World Scientific Publishing Co., NJ, 1993.

[5] Mohammad Jamshidi, Timothy Ross, and Nader Vadiee (Eds.). *Fuzzy Logic and Control: Software and Hardware Applications*, volume 2 of *Environmental and Intelligent Manufacturing Systems.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[6] Bart Kosko. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[7] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA, 1992.

[8] E.H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7:1–13, 1975.

[9] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 12:338–353, 1965.

[10] Lotfi A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC–3(1):28–44, 1973.

[11] Cheun Chien Lee. Fuzzy logic in control systems: Fuzzy logic controller, part I. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):404–418, March/April 1990.

[12] Mohammad Jamshidi. *Large-Scale Systems: Modeling, Control, and Fuzzy Logic*, volume 8 of *Environmental and Intelligent Manufacturing Systems.* Prentice Hall PTR, Upper Saddle River, NJ, 1997.

[13] G.V.S. Raju, J. Zhou, and R.A. Kisner. Hierarchical fuzzy control. *International Journal of Control*, 54(5):1201–1216, 1991.

[14] G.V.S. Raju and J. Zhou. Adaptive hierarchical fuzzy controller. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(4):973–980, 1993.

[15] E.H. Mamdani. Twenty years of fuzzy control: Experiences gained and lessons learnt. In *IEEE International Conference on Fuzzy Systems*, pages 339–344, 1993.

[16] David P. Miller. Mini-rovers for mars exploration. In *Proceedings of the Vision-21 Workshop*, NASA Lewis Research Center, Cleveland, OH, 1990.

[17] Erann Gat, Marc G. Slack, David P. Miller, and R. James Firby. Path planning and execution monitoring for a planetary rover. In *IEEE International Conference on Robotics and Automation*, pages 20–25, May 1990.

[18] David P. Miller, Rajiv Desai, Erann Gat, Robert Ivlev, and John L. Loch. Reactive navigation through rough terrain: Experimental results. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 823–828, July 1992.

[19] Erann Gat, Rajiv Desai, Robert Ivlev, John Loch, and David Miller. Behavior control for robotic exploration of planetary surfaces. *IEEE Transactions on Robotics and Automation*, 10(4):490–503, Aug. 1994.

[20] L. Matthies et al. Mars microrover navigation: Performance evaluation and enhancement. *Autonomous Robots*, 2(4):291–311, 1995.

[21] James C. Bezdek, editor. *IEEE Transactions on Fuzzy Systems – Special Issue on Fuzziness vs. Probability – The N-th Round*. IEEE, February 1994. not actually a book.

[22] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An Introduction to Fuzzy Control*. Springer-Verlag, Berlin, Germany, 1993.

[23] Cheun Chien Lee. Fuzzy logic in control systems: Fuzzy logic controller, part II. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):419–435, March/April 1990.

[24] Timothy J. Ross. *Fuzzy Logic With Engineering Applications*. McGraw Hill, New York, 1995.

[25] L.T. Koczy and K. Hirota. Interpolation in structured fuzzy rule bases. In *IEEE International Conference on Fuzzy Systems*, pages 402–406, 1993.

[26] J. Bruinzeel, M. Jamshidi, and A. Titli. A sensory fusion-hierarchical real-time fuzzy control approach for complex systems. Technical Report No. 95347, LAAS-CNRS, Toulouse, France, August 1995.

[27] Takeshi Yamakawa. A fuzzy inference engine in nonlinear analog mode and its application to a fuzzy logic control. *IEEE Transactions on Neural Networks*, 4(3):496–522, May 1993.

[28] Hamid Berenji, Y.Y. Chen, C.C. Lee, J.S. Jang, and S. Murugesan. A hierarchical approach to designing approximate reasoning-based controllers for dynamic physical systems. In *6th Conference on Uncertainty in Artificial Intelligence*, pages 362–369, 1990.

[29] David J. McFarland. *Feedback Mechanisms in Animal Behavior*. Academic Press, New York, 1971.

[30] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA–2(1):14–23, 1986.

[31] T.L. Anderson and M. Donath. Animal behavior as a paradigm for developing robot autonomy. *Journal of Robotics and Autonomous Systems*, 6:145–168, 1990.

[32] Maja J. Matarić. Behavior-based control: Main properties and implications. In *IEEE International Conference on Robotics and Automation: Workshop on Architectures for Intelligent Control Systems*, pages 46–54, May 1992.

[33] Maja J. Matarić. Behavior-based control: Examples from navigation, learning, and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence, Special Issue: Software Architectures for Physical Agents*, 9(2–3):46–54, 1997.

[34] T.L. Skillman and K. Hopping. Dynamic composition and execution of behaviors in a hierarchical control system. In *Mobile Robots IV*, pages 349–360. SPIE, 1989.

[35] T. Smithers. Are autonomous agents information processing systems? In Luc Steels and Rodney A. Brooks, editors, *The Artificial Life Route to Artificial Intelligence: Building embodied, situated agents*, pages 123–162. Lawrence Erlbaum Associates, Hillsdale, NJ, 1995.

[36] Jan C. Willems. Paradigms and puzzles in the theory of dynamical systems. *IEEE Transactions on Automatic Control*, 36(3):259–294, March 1991.

[37] C.L. Karr. Design of an adaptive fuzzy logic controller using a genetic algorithm. In *4th International Conference on Genetic Algorithms*, pages 450–457, 1991.

[38] Andy Singleton. Genetic programming with C++. *BYTE Magazine*, pages 171–176, February 1994.

[39] P. Nordin. A compiling genetic programming system that directly manipulates the machine code. In K.E. Kinnear, editor, *Advances in Genetic Programming*, pages 311–331. MIT Press, Cambridge, MA, 1994.

[40] Inman Harvey, P. Husbands, and D. Cliff. Issues in evolutionary robotics. Technical Report CSRP 219, University of Sussex, England, July 1992.

[41] M. Matarić and D. Cliff. Challenges in evolving controllers for physical robots. Technical Report CS-95-184, Computer Science Department, Brandeis University, Waltham, MA, November 1995.

[42] John R. Koza. Hierarchical automatic function definition in genetic programming. In L.D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 297–318, San Mateo, CA, 1993. Morgan Kaufmann.

[43] Abdollah Homaifar and Ed McCormick. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3(2):129–139, May 1995.

[44] D.S. Feldman. Fuzzy network synthesis with genetic algorithms. In *5th International Conference on Genetic Algorithms*, pages 312–317, 1993.

[45] D.T. Pham and D. Karaboga. Design of neuromorphic fuzzy controllers. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 103–108, LeTouquet, France, October 1993.

[46] S. Huang and R.M. Nelson. Artificial neural networks used as a rule selector for fuzzy logic controllers. In *ASME Conference Computers in Engineering*, pages 445–454, 1993.

[47] J. Kinzel, F. Klawonn, and R. Kruse. Modifications of genetic algorithms for designing and optimizing fuzzy controllers. In *1st IEEE Conference on Evolutionary Computation*, pages 28–33, Orlando, FL, 1994.

[48] L.B. Booker. Representing attribute-based concepts in a classifier system. In G. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 115–127, San Mateo, CA, 1991. Morgan Kaufmann.

[49] John H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[50] David E. Golberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[51] Nichael Cramer. A representation for the adaptive generation of simple sequential programs. In *1st International Conference on Genetic Algorithms*, pages 183–187, 1985.

[52] Una-May O'Reilly. *An Analysis of Genetic Programming.* PhD dissertation, School of Computer Science, Carleton University, Ottawa, Ontario, September 1995.

[53] F. Herrera O. Cordón and M. Lozano. A classified review on the combination fuzzy logic-genetic algorithms bibliography. Technical Report DECSAI 95129, Dept. of Computer Science and AI, University of Granada, Spain, October 1995.

[54] David E. Goldberg. Dynamic system control using rule learning and genetic algorithms. In *9th International Joint Conference on Artificial Intelligence*, pages 588–592, 1985.

[55] J.H. Holland. Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In J.G. Carbonell R.S. Michalski and T.M. Mitchell, editors, *Machine Learning II*, Los Altos, 1986. Morgan Kaufmann.

[56] John J. Grefenstette. A system for learning control strategy with genetic algorithms. In *3rd International Conference on Genetic Algorithms*, pages 183–190, 1989.

[57] Alan C. Schultz and John J. Grefenstette. Improving tactical plans with genetic algorithms. In *2nd International Conference on Tools for AI*, pages 328–334, 1990.

[58] Alan C. Schultz. Learning robot behaviors using genetic algorithms. In *1st World Automation Congress*, pages 607–612, 1994.

[59] Inman Harvey. *The Artificial Evolution of Adaptive Behaviour*. PhD dissertation, University of Sussex, England, April 1995.

[60] John J. Grefenstette. Lamarckian learning in multi-agent environments. In *4th International Conference on Genetic Algorithms*, pages 303–310, 1991.

[61] Craig W. Reynolds. Evolution of corridor following behavior in a noisy world. In D. Cliff, Husbands, Meyer, and Wilson, editors, *From Animals to Animats 3: 3rd International Conference on Simulation of Adaptive Behavior*, pages 402–410. MIT Press, 1994.

[62] Adam P. Fraser, J.R. Rush, and D.P. Barnes. Evolving multiple agent behaviors for biologically inspired robots. In R.A. Brooks and Pattie Maes, editors, *Artificial Life IV*. MIT Press, 1994.

[63] S. Handley. The genetic planner: The automatic generation of plans for a mobile robot via genetic programming. In *IEEE International Symposium on Intelligent Control*, pages 190–195, 1993.

[64] Ahmad Hemami. Steering control problem formulation of low speed tricycle-model vehicles. *International Journal of Control*, 61(4):783–790, 1995.

[65] Ahmad Hemami, M.G. Mehrabi, and R.M.H. Cheng. Optimal kinematic path tracking control of mobile robots with front steering. *Robotica*, 12(6):563–568, Nov.–Dec. 1994.

[66] Edward Tunstel. Coordination of distributed fuzzy behaviors in mobile robot control. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 4009–4014, Vancouver, BC Canada, October 1995.

[67] John R. Koza and J.P. Rice. Genetic generation of both the weights and architecture for a neural network. In *IEEE International Joint Conference on Neural Networks*, pages 397–404, 1991.

[68] Patrick D'haeseleer. Context preserving crossover in genetic programming. In *1st IEEE Conference on Evolutionary Computation*, pages 256–261, Orlando, FL, 1994.

[69] G. Syswerda. A study of reproduction in generational and steady-state genetic algorithms. In G. Rawlins, editor, *Foundations of Genetic Algorithms*, San Mateo, CA, 1991. Morgan Kaufmann.

[70] Reza Langari and Wei Li. Analysis and efficient implementation of fuzzy logic control algorithms. In *Biennial Conference of the North American Fuzzy Information Processing Society*, pages 1–4, University of California, Berkeley, June 1996.

[71] B. Ewers, J. Bordeneuve-Guibé, J.-P. Garcia, and J. Piquin. Expert supervision of conventional control systems. In *IEEE International Conference on Fuzzy Systems*, pages 143–149, New Orleans, LA, Sep 1996.

[72] George J. Klir and Tina A. Folger. *Fuzzy Sets, Uncertainty, and Information.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[73] S.W. Kim and M. Park. Fuzzy compliance robot control using multi rule-base. In *IEEE International Conference on Fuzzy Systems*, pages 1343–1348, 1992.

[74] S.K. Tso and Y.H. Fung. Intelligent fuzzy switching of control strategies in path control for autonomous vehicles. In *IEEE International Conference on Robotics and Automation*, pages 281–286, 1995.

[75] J.E.R. Staddon. *Adaptive Behavior and Learning.* Cambridge University Press, New York, 1983.

[76] Ronald R. Yager and D.P. Filev. *Essentials of Fuzzy Modeling and Control.* Wiley and Sons, New York, 1994.

[77] Bart Kosko. Stability in feedback additive fuzzy systems. In *IEEE International Conference on Fuzzy Systems*, pages 1924–1930, New Orleans, LA, Sep 1996.

[78] M. Mizumoto. Product-sum-gravity method = fuzzy singleton-type reasoning method = simplified fuzzy reasoning method. In *IEEE International Conference on Fuzzy Systems*, pages 2098–2102, New Orleans, LA, Sep 1996.

[79] M. Mizumoto. Fuzzy controls under product-sum-gravity methods and new fuzzy control methods. In A. Kandel and G. Langholz, editors, *Fuzzy Control Systems*, pages 276–294. CRC Press, 1994.

[80] M. Braae and D. Rutherford. Theoretical and linguistic aspects of fuzzy logic control. *Automatica*, 15(5):553–577, 1979.

[81] Kazuo Tanaka and Michio Sugeno. Stability anaysis and design of fuzzy control systems. *Fuzzy Sets and Systems*, 45:135–156, 1992.

[82] H.-S. Han and J.-G. Lee. Necessary and sufficient conditions for stability of time-varying discrete interval matrices. *International Journal of Control*, 59:1021–1029, 1994.

[83] S.K. Tso and Y.H. Fung. Development of stability analysis for fuzzy-logic controller of autonomous mobile vehicles. In *2nd World Automation Congress*, Montpellier, France, May 1996.

[84] S.G. Cao, N.W. Rees, and G. Feng. Model-free stability analysis for continuous-time fuzzy control systems. In *IEEE International Conference on Fuzzy Systems*, pages 1532–1538, New Orleans, LA, Sep 1996.

[85] H.R. Everett. *Sensors for Mobile Robots: Theory and application.* A K Peters, Ltd., Wellesley, MA, 1995.

[86] J. Borenstein and L. Feng. Umbmark — a method for measuring, comparing, and correcting dead-reckoning errors in mobile robots. Technical Report UM-MEAM-94-22, The University of Michigan, Ann Arbor, MI, Dec. 1994.

[87] Rodney A. Brooks and Anita M. Flynn. Fast, cheap, and out of control: A robot invasion of the solar system. *Journal of British Interplanetary Society*, 42(10):478–485, October 1989.

[88] Erann Gat. Robust low-computation sensor-driven control for task-directed navigation. In *IEEE International Conference on Robotics and Automation*, pages 2484–2489, Sacramento, CA, 1991.

[89] M. Uragami, M. Mizumoto, and K. Tanaka. Fuzzy robot controls. *Journal of Cybernetics*, 6:39–64, 1976.

[90] M. Sugeno and K. Murakami. An experimental study on fuzzy parking control using a model car. In M. Sugeno, editor, *Industrial Applications of Fuzzy Control*, pages 125–138. Elsevier Science Publishers, 1985.

[91] Y. Maeda, M. Tanabe, M. Yuta, and T. Takagi. Hierarchical control for autonomous mobile robots with behavior-decision fuzzy algorithm. In *IEEE International Conference on Robotics and Automation*, pages 117–122, Nice, France, May 1992.

[92] Alessandro Saffiotti, E.H. Ruspini, and K. Konolige. Blending reactivity and goal-directedness in a fuzzy controller. In *IEEE International Conference on Fuzzy Systems*, pages 134–139, 1993.

[93] A. Saffiotti, K. Konolige, and E.H. Ruspini. A multi-valued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1–2):481–526, 1995.

[94] Francois G. Pin and Y. Watanabe. Navigation of mobile robots using a fuzzy behaviorist approach and custom-designed fuzzy inferencing boards. *Robotica*, 12(6):491–504, Nov.–Dec. 1994.

[95] E. Badreddin. Fuzzy relations for behavior-fusion of mobile robots. In *IEEE International Conference on Robotics and Automation*, pages 3278–3283, 1994.

[96] Stephen G. Goodridge and R.C. Luo. Fuzzy behavior fusion for reactive control of an autonomous mobile robot: Marge. In *IEEE International Conference on Robotics and Automation*, pages 1622–1627, 1994.

[97] Wei Li. Fuzzy-logic-based reactive behavior control of an autnomous mobile system in unknown environments. *Engineering Applications of Artificial Intelligence, Special Issue: Control Applications of Fuzzy Logic*, 7(5):521–531, October 1994.

[98] Edward Tunstel and Mo Jamshidi. Fuzzy logic and behavior control strategy for autonomous mobile robot mapping. In *3rd IEEE International Conference on Fuzzy Systems*, pages 514–517, Orlando, FL, June 1994.

[99] Edward Tunstel. Fuzzy spatial map representation for mobile robot navigation. In *10th Annual Symposium on Applied Computing*, pages 586–589, Nashville, TN, February 1995. ACM.

[100] John Yen and N. Pfluger. A fuzzy logic based robot navigation system. In *AAAI Symposium on Applications of Artificial Intelligence to Real-World Autonomous Mobile Robots*, pages 195–199, October 1992.

[101] J. Matijevic and D. Shirley. The mission and operation of the mars pathfinder microrover. In *IFAC 13th Triennial World Congress*, San Francisco, CA, 1996.

[102] J.L. Jones and A.M. Flynn. *Mobile Robots: Inspiration to Implementation*. Robotics and Automated Systems. A K Peters, Ltd., Wellesley, MA, 1993.

[103] L. Moreno, E. Moraleda, M.A. Salichs, J.R. Pimentel, and A. de la Escalera. Fuzzy supervisor for behavioral control of autonomous systems. In *International Conference on Industrial Electronics, Control, and Instrumentation IECON '93*, pages 258–261, 1993.

[104] F. Michaud, G. Lachiver, and C.T. Le Dinh. A new control architecture combining reactivity, planning, deliberation and motivation for situated autonomous agent. In *IEEE International Conference on Fuzzy Systems*, pages 258–264, 1996.

[105] L. Correia and A. Steiger-Garçao. A useful autonomous vehicle with a hierarchical behavior control. In J.J. Merelo F. Móran, A. Moreno and P. Chácon, editors, *Advances in Artificial Life, 3rd European Conference on Artificial Life*, pages 625–639. Springer, Granada, Spain, 1995.

[106] Ronald R. Yager. Nonmonotonic set theoretic operations. *Fuzzy Sets and Systems*, 42:173–190, 1991.

[107] N. Pfluger, John Yen, and Reza Langari. A defuzzification strategy for a fuzzy logic controller employing prohibitive information in command formulation. In *IEEE International Conference on Fuzzy Systems*, pages 717–723, 1992.

[108] Niko Tinbergen. *The Study of Instinct*. Oxford University Press, Oxford, 1951.

[109] G.P. Baerends. A model of the functional organization of incubation behaviour in the herring gull. *Behaviour, An International Journal of Comparative Ethology, Suppl. 17*, pages 261–310, 1970.

[110] P. D. MacLean. *A Triune Concept of the Brain and Behavior*. University of Toronto Press, Toronto, Ontario, 1973.

[111] J.J. Wine and F.B. Krasne. The cellular organisation of crayfish escape behaviour. In D.C. Sandeman and H.L. Atwood, editors, *The Biology of Crustacea*, volume 4, pages 242–292. Academic Press, New York, 1982.

[112] J. R. P. Halperin. *A Connectionist Neural Network Model of Aggression*. PhD dissertation, Department of Ethology, Toronto University, Toronto, Ontario, 1990.

[113] T. Tyrell. *Computational Mechanisms for Action Selection*. PhD dissertation, Center for Cognitive Science, University of Edinburgh, Edinburgh, UK, 1993.

[114] Marco Dorigo and U. Schnepf. Genetics-based machine learning and behavior-based robotics: A new synthesis. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(1):141–154, Jan.–Feb. 1993.

[115] Bridget E. Hallam, J. R. P. Halperin, and J. C. T. Hallam. An ethological model for implementation in mobile robots. *Adaptive Behavior*, 3(1):51–79, Summer 1994.

[116] B. Blumberg. Action-selection in hamsterdam: Lessons from ethology. In Dave Cliff, P. Husbands, J-M. Meyer, and S. W. Wilson, editors, *3rd International Conference on Simulation of Adaptive Behavior*, pages 108–117. MIT Press, Cambridge, MA, 1994.

[117] David J. McFarland and Thomas Bösser. *Intelligent Behavior in Animals and Robots.* MIT Press, Cambridge, MA, 1993.

[118] Konrad Z. Lorenz. *The Foundations of Ethology.* Springer-Verlag, New York, 1981.

[119] F. J. Varela. The re-enchantment of the concrete. In Luc Steels and Rodney A. Brooks, editors, *The Artificial Life Route to Artificial Intelligence: Building embodied, situated agents*, pages 11–22. Lawrence Erlbaum Associates, Hillsdale, NJ, 1995.

[120] Walter Freeman and C. Skarda. Spatial eeg patterns, nonlinear dynamics, and perception: The neo-sherrington view. *Brain Research Reviews*, 10:145–175, 1985.

[121] E. Tunstel, M.R. Akbarzadeh-T, K. Kumbla, and M. Jamshidi. Hybrid fuzzy control schemes for robotic systems. In *10th IEEE International Symposium on Intelligent Control*, pages 171–176, Monterey, CA, August 1995.

[122] E. Tunstel, M.-R. Akbarzadeh-T., K. Kumbla, and M. Jamshidi. Soft computing paradigms for learning fuzzy controllers with applications to robotics. In *Biennial Conference of the North American Fuzzy Information Processing Society*, pages 355–359, University of California, Berkeley, June 1996.

[123] Andrea Bonarini. Learning to coordinate fuzzy behaviors for autonomous agents. In *2nd European Congress on Intelligent Techniques and Soft Computing EUFIT '94*, pages 475–479, 1994.

[124] K.C. Koh H.R. Beom and H.S. Cho. Behavioral control in mobile robot navigation using fuzzy decision making approach. In *IEEE International Conference on Intelligent Robots and Systems*, pages 1938–1945, 1994.

[125] D. Gachet, M.A. Salichs, L. Moreno, and J.R. Pimentel. Learning emergent tasks for an autonomous mobile robot. In *IEEE International Conference on Intelligent Robots and Systems*, pages 290–297, 1994.

[126] P. Nordin and W. Banzhaf. Genetic programming controlling a miniature robot in real time. Technical Report 4/95, Computer Science Department, University of Dortmund, Germany, 1995.

[127] Edward Tunstel and Tanya Lippincott. Genetic programming of fuzzy coordination behaviors for mobile robots. In *International Symposium on Soft Computing for Industry, 2nd World Automation Congress*, Montpellier, France, May 1996.

[128] Markus Olmer, P. Nordin, and W. Banzhaf. Evolving real-time behavioral modules for a robot with gp. In *2nd World Automation Congress*, Montpellier, France, May 1996.

[129] Hartmut Surmann, Jörg Huser, and Jens Wehking. Path planning for a fuzzy controlled autonomous mobile robot. In *IEEE International Conference on Fuzzy Systems*, pages 1660–1665, 1996.

[130] Walter J. Freeman. *Mass Action in the Nervous System*. Academic Press, New York, 1975.

Appendix A

Genetic Programming

Procedure

This appendix includes definitions and/or elaborated concepts of genetic programming as described in the definitive text by Koza [7]. There are five preparatory steps to applying genetic programming:

(1) Determine a suitable function set, F.

(2) Determine a terminal set, T.

(3) Define a fitness function or measure.

(4) Set control parameters for the run (e.g. population size, maximum number of generations, genetic operator probabilities, etc).

(5) Determine method of designating a result and termination criteria.

Genetic programming breeds computer programs to solve problems according to the following steps:

(1) Randomly generate an initial population of programs.

(2) Execute each program in population and assign a fitness value according to a specified fitness function.

(3) Create a new generation by applying reproduction, crossover, and mutation to programs selected with some probability based on fitness.

(4) If termination criteria is not met, go to step (2).

(5) The best program from any generation is the result (which may be a solution or an approximation).

Genetic operators

Reproduction

The reproduction operation is a two-step process. First a single program is selected from the population according to some selection method based on fitness.

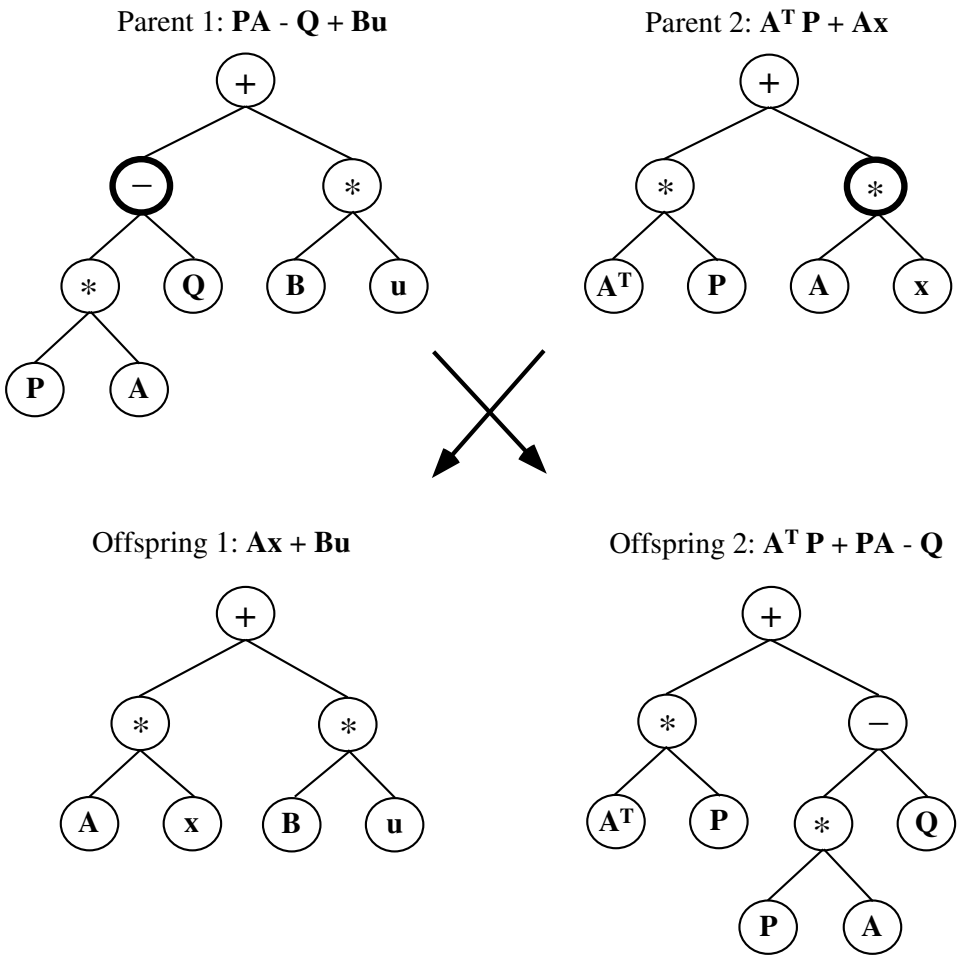
In *fitness-proportionate selection*, the probability that program P_i will be copied into the next generation as a result of one reproduction operation is

$$\frac{f(P_i)}{\sum_{j=1}^{M} f(P_j)}$$

where $f(P_i)$ is the fitness of program P_i, and M is the size of the population.

In *Tournament selection* a number of programs (often two) are chosen at random from the population and the program with best fitness is selected.

Second, the selected program is copied from the current population into the next, i.e. the new generation.



Crossover

The crossover operation starts with two parental programs and produces two offspring programs that are added to the new generation. The parental programs are chosen from the population according to the same selection method used for reproduction. The operation begins by independently selecting one random point (using uniform probability distribution) from each parent as the respective crossover point. The subtrees subtended from crossover points are then swapped between the parents to produce the two offspring. In the illustration, the crossover points in each parent are indicated by the dark circles.

Closure and Sufficiency

Function and terminal sets (F and T) chosen for a given genetic programming problem must satisfy the closure and sufficiency properties.

Let S_F be the set of all values and data types that may possibly be returned by any function $\phi \in F$; let S_{arg} be the set of all arguments acceptable by all $\phi \in F$. Finally, let S_T be the set of all values and data types that may be assumed by any terminal $\tau \in T$. Then the *closure* property requires that

$$\forall \phi \in F,\ S_F \subseteq S_{arg}\ and\ S_T \subseteq S_{arg}.$$

In other words, each $\phi \in F$ should be well defined and closed for any combination of arguments that it may encounter. Koza points out that this property is required only for functions and terminals that may actually be encountered. However, if the structures undergoing adaptation are *constrained syntactic structures* (as is the case in this dissertation), then closure is only required over the values $v_F \in S_F$ and $v_T \in S_T$ that will actually be encountered.

The *sufficiency* property requires that the collection of elements of the set $F \cup T$ be capable of expressing a solution to the problem. Identification of an adequate collection of functions and terminals with sufficient explanatory power to solve a given problem may not be possible in some domains. The definitive text provides numerous illustrative examples, of how to select F and T, which may be used as guides.