BEHAVIOR-BASED ROBOT CONTROL USING FUZZY DISCRETE EVENT SYSTEM

MD. RAJIBUL HUQ

BEHAVIOR-BASED ROBOT CONTROL USING FUZZY DISCRETE EVENT SYSTEM

by

© Md. Rajibul Huq

A thesis Submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

Faculty of Engineering and Applied Science
Memorial University of Newfoundland

December, 2006

St.John's, Newfoundland, Canada

Abstract

The key issue associated with behavior-based architecture is the development of a behavior coordinator. A behavior coordinator should possess a number of important properties. The literature suggests that (a) both behavior arbitration and command fusion techniques should be combined in order to coordinate competitive and cooperative behaviors, (b) the coordinator should possess adequate means of modeling the current state of the world, (c) sensory uncertainties should be modeled using multi-valued logic, (d) the coordinator should be capable of making persistent behavior selection, (e) it should provide the opportunity to accommodate hierarchical decision-making for reactive action generation, (f) it should possess predictive decision-making capability for handling future environmental uncertainties, (g) it should provide satisfactory means of decision analysis, and finally (h) it should be modular to achieve robustness with a larger number of behaviors. This thesis attempts to address some of the issues mentioned above while exploiting fuzzy logic (FL) based methodologies for behavior coordination.

The initial investigation includes experimentation of different methodologies in the literature. A FL-based controller is designed for motor schema based mobile robot navigation. Fuzzy meta rules have been used to adaptively generate weights for each motor schema. It uses human reasoning to model the deterministic uncertainty of sensory data, which in turn helps to reduce the possibility of generating incorrect weights for motor schemas. Experimental results demonstrate that the fuzzy logic based approach overcomes some of the common problems of schema-based navigation. It has been observed that a pure FL-based method has several disadvantages, such as it imposes a scalability problem when the system consists of both competitive and cooperative behaviors and it does not provide any feedback measures for decision

analysis. Moreover, the rule-based knowledge representation becomes complex in cases where previous state information is incorporated for persistent decision-making.

While understanding the shortcomings of each technique, this thesis present a novel behavior coordination architecture using Fuzzy Discrete Event System (FDES). This architecture addresses the shortcomings of the FL-based technique using complimentary properties of Discrete Event System (DES). The DES-based method provides supervisory control techniques for behavior arbitration and has a suitable framework for decision analysis in terms of observability and controllability. Furthermore, it supports multi-level behavioral decomposition and uses previous state information of the system for persistent decision-making. As a result, the combination of FL and DES provides the opportunity to integrate several key features to achieve a high performance behavior coordinator.

Finally the proposed architecture is experimentally tested using three different robotic applications, namely robotic navigation, robotic box pulling and robotic visual attention. The navigational experiments demonstrate that the performance measures of the FDES-based system are unaffected even under changing or complex environments. The FDES-based system is able to produce oscillation-free and collision-less navigation in the experiments. The FDES based concept is then extended to robotic object-pulling operation. These experiments demonstrate the multi-level behavioral decomposition feature of the proposed architecture. The object-pulling task is divided into anchoring and navigation subtasks. The robot, first, anchors the object and then navigates to the target location. The experimental results demonstrate that the FDES-based approach provides reliable execution of anchoring task and produces collision-free navigation to the target location. Finally, the FDES-based application is implemented to model visual attention system of a robot. This method integrates bottom-up bias (sensory feedback) with top-down influence (previous experience of

the robot) to control pan-tilt motion of a camera mounted on a robot. The FDES-based system is able to prevent abrupt changes in focus of attention and produces smooth transition in motion commands when visual attention changes between the objects.

Overall the proposed FDES-based approach outperforms several other common techniques due to its key characteristics, such as combination of arbitration and command fusion, FL-based world state modeling, persistent decision-making, multilevel behavioral decomposition, decision analysis capability, and modularity of the system.

Acknowledgement

It is my great pleasure to finally have a chance to show my gratitude to peoples who accompanied me during the four years of this doctoral program.

First, I would like to thank my supervisors, Dr. Raymond Gosine and Dr. George Mann, for giving me the opportunity to carry out my doctoral research under their supervision. It was absolutely a wonderful experience to work with Ray who gave me the inspiration to work on 'how to get along with a robot'. My heart-felt gratitude to George for rescuing me everytime I got into trouble with the robots. Without George's guidance, comments and close supervision I could not have brought this thesis to its conclusions.

I am highly grateful to Memorial University of Newfoundland, C-CORE, Natural Science and Engineering Research Council of Canada (NSERC) and Canada Foundation for Innovation (CFI) New opportunity program to provide financial supports to carry out this research.

My special thanks to different members of Intelligent System group at C-CORE for all the insightful conversations, specially to Dilan for always being a reliable source of solution for any kind of problem I faced with the robots.

I would like to show my profound gratitude to my parents. They raised me well and have made me who I am. Finally, I would like to thank Momotaz, for her comments and help about this research as a fellow researcher and continuous support and encouragement as a loving wife. I could not have done this without her.

Contents

List of Tables

List of Figures

List of Abbreviations

AASBA	:	Affective Action Selection and Behavior Arbitration
ART2	:	Adaptive Resonance Theory 2
CLA	:	Centroid of Largest Area
DAMN	:	Distributed Architecture for Mobile Navigation
DES	:	Discrete Event System
EPF	:	Electrostatic Potential Fields
FL	:	Fuzzy Logic
FIS	:	Fuzzy Inference System
FDES	:	Fuzzy Discrete Event System
FSA	:	Finite State Automata
GA	:	Genetic Algorithm
GAPPS	:	Goals as Parallel Program Specifications
HA	:	Hybrid Automata
HSI	:	Hue-Saturation-Intensity
LTM	:	Long Term Memory
MF	:	Membership Function
NN	:	Neural Network
RAP	:	Reaction Action Package
RGB	:	Red-Green-Blue
SSS	:	Servo-Subsumption-Symbolic

WM : Working Memory

List of Symbols

A	:	the combined action
A_i	:	the action associated with the i^{th} behavior
$A_{i,pan}$, $A_{i,tilt}$	:	the pan-tilt motion command associated with B_i
A_{pan}, A_{tilt}	:	the overall pan-tilt motion command
$\hat{B}$	:	the highest priority behavior in the $\hat{n}^{th}$ state
$[\hat{B}]$	:	the set of cooperative behaviors of $\hat{B}$
B_i	:	the i^{th} behavior
c_i	:	the environmental state caused by the execution of action A_i
C	:	the state-based controllability of a FDES
C_a	:	average controllability
C_{ij}	:	the state-based controllability of F_{ij}
COL	:	total number of collisions
d_e	:	the degree of observability of the e^{th} event
d_e^c	:	the degree of unobservability of the e^{th} event
D_1, D_2	:	the distances from the current robot's position to the nearest and the second nearest subgoal
$D_{i,pan}$, $D_{i,tilt}$	:	the distances from the center of the visual field to the centroid of i^{th} object along x-axis and y-axis
D_{obs}	:	average distance to the nearest obstacle
$D_{obs}(k)$	:	the distance to the nearest obstacle in the k^{th} decision cycle

D_{tra}	:	total traveled distance by the robot
$D(x, y)$	:	Euclidean distance between (x, y) and (x_r, y_r)
E	:	the total number of events
$f^*_{jn}(\cdot)$	:	the MF associated with $\mathcal{X}^*_{jn}$ that maps z^*_i to a graded membership value, where $* \in \{\mathcal{R}, \mathcal{T}, \mathcal{O}, \mathcal{W}\}$ and $\mathcal{R}$, $\mathcal{T}$, $\mathcal{O}$, $\mathcal{W}$ indicate route, target, obstacles, and tangent to obstacles respectively.
$f^*_q(\cdot)$	:	the MF associated with $\mathcal{Y}^*_q$ that maps β_* to a graded membership value
$f^e_{ij}(\cdot)$	:	the e^{th} MF of F_{ij}. It maps z_{ij} to a graded membership value.
F_{ij}	:	the FDES that determines the activity of B_i with respect to z_{ij}
F_i	:	the FDES that combines the outputs of F_{ij}
FS_*	:	fuzzy inference systems
FV	:	view of the camera
G	:	a fuzzy automaton
$h(k)$	:	the frustration level at the k^{th} time step
HFV	:	height of the image captured by the camera
I	:	Idenity matrix
J	:	the total number of sensory data used by the behaviors
J^*	:	the total number of sensory data associated with a FIS
J_i	:	the total number of sensory data associated with B_i
K	:	the total number of decision cycles
$l_{nn\acute{n}}$	:	the degree of inconsistency between the n^{th} and $\acute{n}^{th}$ state in a FDES
L	:	the inconsistency matrix
L_{ij}	:	the inconsistency matrix associated with F_{ij}

$\mathcal{N}_j^*$	:	the total number of fuzzy sets defined over z_j^*
M	:	the total number of behaviors
N	:	the total number of elements in a fuzzy state vector
O	:	the state-based observability of a FDES
O_a	:	average observability
O_{ij}	:	the observability of F_{ij}
p	:	positions in a 2D space used with integer subscripts
$P_i(\cdot)$	:	input condition of the i^{th} fuzzy rule in a FIS
$Q_i(\cdot)$	:	output condition of the i^{th} fuzzy rule in a FIS
R_{curr}	:	average radius of curvature
s_o	:	the initial fuzzy state vector
s	:	the current fuzzy state vector
$\acute{s}$	:	the expected fuzzy state vector
$\acute{s}^e$	:	the expected fuzzy state vector due to the occurrence of the e^{th} event
s_i	:	the current fuzzy state vector of F_i
$\acute{s}_i$	:	the expected fuzzy state vector of F_i
s_{ij}	:	the current fuzzy state vector of F_{ij}
$\acute{s}_{ij}$	:	the expected fuzzy state vector of F_{ij}
$\acute{\tilde{s}}_i$	:	the modified state vector produced by $\tilde{\Sigma}_i$ in F_i
$\acute{s}_i(k)$	:	the expected fuzzy state vector of F_i at the k^{th} decision cycle
$\acute{s}_{ij}(k)$	:	the expected fuzzy state vector of F_{ij} at the k^{th} decision cycle
$\bar{s}$	:	average of the output fuzzy state vectors of F_{ij}
$[state_n]$	:	the set of behaviors having the highest memberships in the n^{th} state
$[state_{\hat{n}}]$	:	the set of behaviors having the highest activity state, which is

		the $\hat{n}^{th}$ state
S	:	a set of fuzzy state vectors,
$SUP(\cdot)$	:	a function that maps the sensory feedback to the desired sets of events
t_p	:	the length of a decision cycle
$t_p(k)$	:	the length of the k^{th} decision cycle
t_{nav}	:	total navigation time of the robot
T	:	transpose operator
$U(\cdot)$	:	utility function
v	:	translational velocity command
v_{max}	:	maximum translational velocity
Δv	:	average rate of change of velocity
$\hat{\mathbf{V}}_*$	:	unit vector representing motor schema
$\mathbf{V}$	:	the coordinated schema
$\mathbf{V}_i$	:	the i^{th} force vector
$w_{n\acute{n}}$	:	the degree of restriction of state transition between the n^{th} and $\acute{n}^{th}$ state
W	:	a matrix that describes the undesired state transition between states in a FDES
W_i	:	describes the undesired state transitions in F_i
W_{ij}	:	describes the undesired state transitions in F_{ij}
WFV	:	width of the image captured by the camera
(x, y)	:	a location in 2D Cartesian Coordinate
(x_r, y_r)	:	current location of the robot
(x_o, y_o)	:	location of obstacle
(x_1, y_1)	:	location of the nearest subgoal with respect to the current robot's

	:	location
(x_2, y_2)	:	location of the second nearest subgoal with respect to the current robot's location
$(x_r(k), y_r(k))$	:	the location of the robot at the the k^{th} decision cycle
$[xE, yE, \theta E]^{\mathrm{T}}$	:	estimated odometry error in the current robot's position
$[x\acute{E}, y\acute{E}, \theta\acute{E}]^{\mathrm{T}}$	:	estimated odometry error in the previous robot's position
$[x_{odo}, y_{odo}, \theta_{odo}]^{\mathrm{T}}$	:	the current robot's position obtained using odometry
$[\hat{x}_{odo}, \hat{y}_{odo}, \hat{\theta}_{odo}]^{\mathrm{T}}$	:	the corrected robot's position obtained using odometry
$[x_{loc}, y_{loc}, \theta_{loc}]^{\mathrm{T}}$	:	the current robot's position obtained using localization module
$\mathcal{X}^*_{jn}$	:	the n^{th} fuzzy set defined over z^*_j
$\mathcal{Y}^*_q$	:	the q^{th} fuzzy set defined over β_*
μ	:	graded membership value
z^*_j	:	the j^{th} sensory data associated with a FIS
Z^*	:	a set of sensory data associated with a FIS
Z_i	:	the set of sensory data associated with B_i
z_{ij}	:	the j^{th} sensory data associated with B_i
$z_{ij}(k)$	:	the j^{th} sensory data of B_i at the k^{th} decision cycle
α^e	:	the e^{th} fuzzy event matrix
$\acute{\alpha}$	:	the unobservable form of the e^{th} event
α^e_{ij}	:	the e^{th} event of F_{ij}
α^e_i	:	the e^{th} event of F_i
$\tilde{\alpha}^e_i$	:	the desired event matrix produced by the controller
β_*	:	weight of the unit vector $\hat{\mathbf{V}}_*$
β_i	:	the weight of the i^{th} behavior
δ	:	a threshold (scalar or matrix) used to compare the actions of behaviors

$\delta_{i\acute{i}}$	:	a scalar value, which is an element of the matrix δ, used to compare the actions of the i^{th} and $\acute{i}^{th}$ behavior
$\lambda_{n\acute{n}}$	:	the overlap between the n^{th} and $\acute{n}^{th}$ states in terms of the intersected membership of their corresponding MFs
$\mu^{e}_{n\acute{n}}$	:	the state transition possibility from the n^{th} state to the $\acute{n}^{th}$ state when the e^{th} event occurs
μ_n	:	the degree of possibility of being in the n^{th} state
$\acute{\mu}^{e}_{n}$	:	the expected possibility of being in the n^{th} state due to the occurrence of the e^{th} event
$\acute{\mu}_n$	:	the expected possibility of being in the n^{th} state
$\tilde{\mu}^{e}_{n\acute{n}}$	:	the state transition possibility from the n^{th} state to the $\acute{n}^{th}$ state when $\tilde{\alpha}^{e}_{i}$ occurs
$\acute{\tilde{\mu}}_n$	:	the n^{th} element of the state vector $\acute{\tilde{s}}_i$
Σ	:	a set of fuzzy event matrices
$\tilde{\Sigma}_i$	:	the desired set of events produced by the controller
γ^{e}_{i}	:	the state transition possibility due to the occurrence of α^{e}_{i} in F_i
κ	:	a proportionality constant having the unit per second
$\bigcirc$	:	on the order of
τ, ν	:	weighting factors used with integer subscripts
Ω	:	rotational velocity command
$\Omega(k)$	:	the rotational velocity at the k^{th} decision cycle
θ_{pan}, θ_{tilt}	:	the pan-tilt angles of associated with A_{pan} and A_{tilt}
$\theta_{\mathcal{O}}$	:	orientation of the obstacle
$\theta_{\mathcal{T}}$	:	orientation of the target
θ	:	expected robot orientation with respect to current sensory data
$\tilde{\theta}$	:	previous robot orientation

ξ	:	a state transition function from $S \times \Sigma$ to S
ζ_{att}, ζ_{ref}	:	attractive and repulsive potential fields
$\tilde{\cup}$	:	fuzzy OR (max) operator
$\otimes$	:	max-min fuzzy composition
$\equiv_{co}$	:	an equivalence relation, which determines the cooperative behaviors of $\hat{B}$

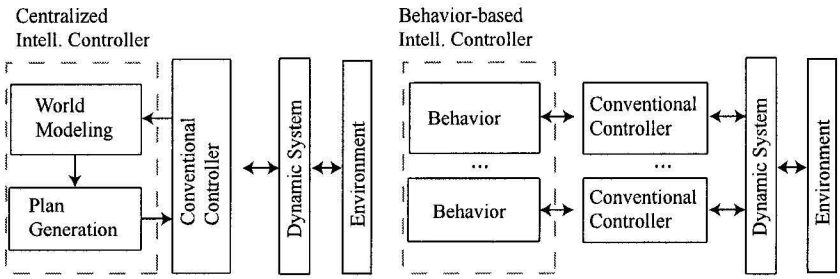
Chapter 1

Introduction

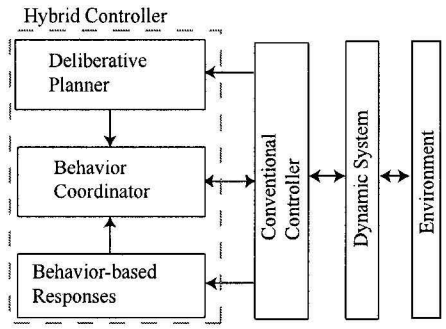
1.1 Background

A major challenge in autonomous robotics is to devise an intelligent control system, which can execute an appropriate control action while considering uncertainties associated within its own sensor system and the surrounding dynamic environment. Currently, there exist three broad categories of intelligent architectures: centralized (deliberative), behavior-based (reactive), and hybrid (deliberative-reactive). Centralized architectures [1,2,3,4,5,6,7,8] create a complete model of the static environment by combining all available sensory data. Then it employs deliberative planning in order to generate a series of actions within the context of the static model to accomplish a given task (see Fig.1.1(a)). After successful execution of an action, the robot stops, gathers more information, and repeats the process. An important aspect of this architecture is the top-down approach of planning, where high level constraints are integrated into low level control commands. Centralized intelligent controller can coordinate multiple goals and constraints within a complex environment. However, planning a series of actions by sensor fusion in a centralized architecture introduces a potentially harmful delay [9]. Moreover, the system may fail entirely if any sin-

gle part fails, e.g., sensor fusion or planning is not functioning properly [10]. This leads to inappropriate use of centralized controller for a real time system, where the environment is dynamic or uncertain. On the other hand, behavior-based architec-



(a) Centralized architecture (b) Behavior-based architecture



(c) Hybrid architecture

Figure 1.1: Intelligent control architectures, adapted from [9]

tures [11,12,13,14,15,16,17,18] are composed of independent task-achieving modules, or behaviors (see Fig.1.1(b)). Each behavior receives a particular sensory perception which is related to a given task, thus avoiding the need for sensor fusion. Moreover, task-achieving aspect of each behavior leads to distributive control process, which reduces planning complexity and increases responsiveness to a dynamic environment. Behavior-based architectures possess bottom-up approach of decision-making as they do not integrate high level constraints in action generation process. They are also more robust because if any behavioral unit of the system fails, the other units continue to function independently. However, a completely distributed system does not reflect

the multiple objectives and constraints that the system is subjected to at any given moment, thus leading to significantly suboptimal performance [19, 9] and unreliable decision-making [20]. Additionally, the interactions, both between the behavioral units and between the system and its environment, are less predictable and more difficult to understand and modify as compared to a purely centralized system.

Hybrid architectures [21, 22, 23, 24] attempt to combine the goal-directed solution of centralized architecture and responsiveness of behavior-based architecture (see Fig.1.1(c)). The top level of a hybrid architecture is a deliberative planner, which assimilates all available information and creates long-term global plans. The lowest level consists of a behavior-based architecture, which recommends real-time responses to sensory stimuli. The most important part of a hybrid architecture is the intermediate level, called behavior coordinator or sequencer [25]. A behavior coordinator takes into account the high level constraints of a deliberative planner and real-time responses of individual behaviors. As a result, it can generate an action, which satisfies the objectives of both the planner and behaviors.

1.2 Problem statement

Existing taxonomy of behavior coordination mechanisms indicates two major categories [19, 26]: *Learning* to coordinate behaviors and *Supervisory* techniques for behavior coordination. Conventional *Learning* approaches [27, 28, 29, 30, 31, 32] employ reinforcement techniques that maintain a repertoire of behaviors and search the best candidate behavior with respect to a world state, based on trial and error, to maximize a reward function. However, designing a reward function is a tedious task specially for complex environments as the size of the state-space of world states increases rapidly [33]. Moreover, this technique requires a long learning phase, which creates a serious bottleneck for real-world robotic applications [19]. On the other

hand, the *Supervisory* approaches employ preprogrammed coordination techniques using expert knowledge of the problem domain.

This work will investigate the *Supervisory* approaches of behavior coordination. The rest of the thesis will refer "*Supervisory* behavior coordination" simply as "behavior coordination". The major challenge in designing a behavior coordinator involves addressing following key problems.

1.2.1 Problem I: Behavior selection

The first problem is related to behavior selection mechanisms, which involves activating appropriate behaviors at a particular moment. The existing behavior selection mechanisms can be broadly categorized as either behavior arbitration [11, 21, 23, 34, 35, 36, 37, 38, 39] or command fusion [40, 41, 42, 43, 44, 45]. A behavior arbitration mechanism selects only one behavior given a particular environmental context and produces the best action, which can accomplish a given task. This approach is suitable for a system, which employs competitive behaviors for task execution [46]. However, competitive nature of behavior arbitration produces two major drawbacks, namely, instability [39] and starvation [37]. Instability arises when the control of the robot alternates between two behaviors and starvation occurs when a behavior does not gain control of the robot for a long time.

A command fusion technique activates all behaviors simultaneously and produces an emergent action by combining the actions of all behaviors. This approach is suitable for a system, which employs cooperative behaviors for task execution [19]. Command fusion of behaviors partially removes the instability and starvation problems since it considers actions of all behaviors simultaneously. However, a common problem arises in command fusion techniques, when competing behaviors issue conflicting control commands resulting in local minima or stagnant situations

of the robot [9, 46]. This problem is addressed by weighted decision-making techniques [47, 48, 49, 50, 51, 52, 53, 54, 55], where different sets of rules are used to weight the conflicting commands according to environmental contexts, thus, reducing the possibility of producing local minima.

The key research issue related to behavior selection problem is identified as follows.

- How behavior arbitration and command fusion are integrated in the same frame to facilitate the use of both competitive and cooperative behaviors for task execution.

1.2.2 Problem II: Knowledge representation

The second problem refers to the knowledge representation technique, which employs context rules to infer different world states from available sensory data. A behavior coordinator selects appropriate behaviors for a given world state. The available knowledge representation techniques include Finite State Automata (FSA) [35,36,39], Petri Net [16], decision-tree [13,24], and fuzzy-rule [48,49,50,51,52,53,54,55,56] based approaches. These approaches mainly differ from each other depending on the use of binary or multi-valued logic to construct the context rules. Binary logic uses bivalent reasoning to classify a world state. It performs well, when the sensory data is accurate. However, it increases the possibility of erroneous world sate detection in presence of noisy sensory data [56]. This leads to inappropriate behavior selection at a given world state. On the other hand, multi-valued logic employs fuzzy reasoning to classify a world state and therefore, reduces the possibility of erroneous world state detection, when the sensory perception is noisy. However, fuzzy-based approaches suffer the drawback of forming large rules-base for complex behavior-based systems. Tuning Membership Functions (MFs) is another cumbersome task of fuzzy-based approaches. Neuro-fuzzy and genetic-fuzzy systems [57, 58, 59] address this issue, where

Neural Network (NN) and Genetic Algorithm (GA) can be used to optimize rule-base construction and tuning of MFs.

The particular research issues related to knowledge representation problem are as follows.

- How world states are modeled using available sensory data.
- How sensory uncertainty is handled using fuzzy reasoning.

1.2.3 Problem III: Decision making and analysis

This problem is related to intelligent decision-making using the available knowledge representation of the world states. It involves addressing several issues, such as persistent behavior selection, reactive action generation, world-state prediction, and decision analysis. Persistence is an important aspect of the behavior coordination problem. It favors those behaviors, which contribute to the ongoing goal. Hence, a behavior coordinator must consider past and present states of the system so that the behavioral activities are controlled in a consistent manner while taking into account the present context of the world [13, 16, 22, 24, 34, 35, 36, 39, 42, 60, 61, 62, 63]. Preserving system memory allows one to remove conflicting behavior selection in successive decision cycles. A behavior coordinator must exhibit timely responses to environmental changes (which is known as reactivity). To achieve this goal, a coordination system employs computationally inexpensive methods for fast decision-making. Hence, hierarchical approaches, which include multilevel behavioral decomposition [18, 22, 23, 33, 36, 52, 60, 64, 65] and decision tree based techniques [13, 24], play an important role for fast behavior selection at a given context. Hierarchical approaches employ prior domain knowledge as well as current sensory data for fast and appropriate behavior selection. A behavior coordinator should possess the ability to predict probable world states as a consequence of an action execution. It enables

the system to avoid hazardous situations beforehand. As an example, the methods described in [38, 66] employ probabilistic inference to determine the expected utility of an action on the basis of probable world states. A behavior coordinator should provide some means of decision analysis that helps taking corrective actions if necessary. As an instance, the Discrete Event System (DES) based approach [35] is furnished with *observability* and *controllability* analysis that helps modeling inadequate sensory perceptions and occurrence of undesired world states. Thus, these analysis helps taking a relevant action, when perception is insufficient and unexpected changes occur in the robot's workspace.

The particular research issues related to this problem are outlined as follows.

- How persistent behavior selection is made.
- How reactive action is generated.
- How future world states are predicted.
- How decision is analyzed.

1.2.4 Problem IV: Modularization

This problem is related to the development of modularized behavior coordination architecture, which increases robustness of the system. In modular architecture [10, 11], addition of a new behavior does not effect the original behavior coordination architecture. It also infers that the malfunction of one part does not cause failure of the entire system. Modular approaches also increase scalability that enables modeling of a large behavior-based system. As an instance, modular fuzzy approaches (e.g, [65, 67]) that use separate fuzzy rule bases to determine the activity of each behavior is more robust than monolithic fuzzy approaches (e.g. [68]) that use a single fuzzy

rule base to control activities of all behaviors. Hence, addition of a new behavior directly affects the existing rule base of a monolithic approach.

The particular issue related to this problem is specified as follows.

- How modular architecture is developed to increase scalability and robustness of the system.

1.3 Motivation & objectives of the research

The main focus of this research is to devise a novel behavior coordination mechanism that addresses the issues related to the Problems I-IV described in Section 1.2. The present work employs DES and Fuzzy Logic (FL) based knowledge representation, which is capable of addressing most of the issues related to Problems I-IV.

1.3.1 Motivation

The DES-based techniques (e.g., [35,36]) formulate the behavior coordination problem as a sequence of events, where the events are constructed using the available sensory data and their occurrences lead to a new selection of behaviors with respect to the past selection of behaviors. This architecture provides the opportunity of multi-level behavioral decomposition, where the occurrence of an event results in a high level behavior, which is composed of another set of low level behavioral units. The DES-based techniques also provide means of decision analysis in terms of controllability and observability of the system [69]. Therefore, this architecture is characterized with the following properties:

- it employs behavior arbitration for behavior selection,
- it uses event-driven architecture for world-state modeling,

- it considers the past selection of behaviors to make the present selection consistent,
- it deploys multi-level behavioral decomposition for reactive decision-making, and
- it provides means of decision analysis using controllability and observability of the system.

However, the main disadvantage of the DES-based technique stems from the use of binary logic for event generation. The use of binary logic may produce inappropriate events in presence of noisy sensory data resulting in irrelevant behavior selection at a particular world state.

On the other hand, FL-based techniques (e.g., [65, 67]) employ fuzzy rules to determine the relevance (which is also known as activity) of a behavior at a given environmental context. The actions of the behaviors are weighed according to the behavioral activity, which is a fuzzy number in the range of 0 to 1. Thus the FL-based methods reduce the possibility of erroneous behavior selection as compared to the DES-based techniques, where the relevance of a behavior is either 0 or 1. The FL-based approach provides the opportunity of multi-level behavioral decomposition and modular decision-making using layered and independent fuzzy rule-base for the behaviors. Therefore, this architecture is characterized with the following properties:

- it employs command fusion for behavior selection,
- it uses fuzzy reasoning for world-state modeling and sensory uncertainty handling, and
- it deploys layered and independent fuzzy rule-base for multi-level behavioral decomposition and modularized behavior coordination.

Hence, the combination of DES and FL-based approach is able to address all of the issues related to the Problems I-IV (except the predictive decision-making, which requires probabilistic inference).

The present research aims to integrate the characteristics of DES and FL-based behavior coordination techniques in the same frame. This research goal is accomplished using the formalisms of Fuzzy Discrete Event System (FDES) [70,71,72,73,74], which facilitates the integration of event-driven architecture of DES and approximate decision-making of FL.

1.3.2 Objectives

In order to achieve the proposed research goal, the following objectives have been identified.

Objective I: A systematic analysis of the existing behavior coordination techniques and experimentations using the FL-based coordination method to specify the particular issues related to the behavior coordination Problems I-IV as mentioned in Section 1.2.

Objective II: Development of a novel behavior coordinator using FDES, which addresses the issues explored in Objective I.

Objective III: Analysis of the applicability of the proposed FDES based approach in the field of robotic applications, such as navigation, object manipulation, and visual attention.

1.4 Contributions

This thesis made following contributions in behavior-based robotics while fulfilling the three research objectives.

1. Contributions from Objective I:

   (a) A new knowledge-based classification is proposed for behavior arbitration and command fusion based behavior selection mechanisms.

   (b) A novel behavior-based approach of mobile robot navigation is proposed using FL-based coordination technique for motor schema [41] based behaviors. This method is presented as an alternative of the case-based navigation using motor schema [47]. It shows better performance in eliminating the shortcomings of the conventional motor schema based navigation.

2. Contributions from Objective II:

   This objective enables to devise a novel behavior-based robotic control approach using FDES. The proposed behavior coordinator

   (a) combines the characteristics of behavior arbitration and command fusion techniques to facilitate the coordination of competitive and cooperative behaviors,

   (b) uses event-driven architecture for world states modeling,

   (c) deploys FL to handle sensory uncertainty.

   (d) is capable of making persistent behavior selection using system memory,

   (e) provides the opportunity of multi-level behavioral decomposition for fast decision-making,

   (f) provides means of decision analysis using observability and controllability, and

   (g) employs modular architecture for behavior coordination.

3. Contributions from Objective III:

   This objective enables to implement three FDES-based robotic applications,

which include

(a) mobile robot navigation,

(b) object-pulling operation by mobile robot, and

(c) visual attention of mobile robots.

1.5 Organization of the thesis

Chapter 1 addresses the main problem areas of behavior coordination system for intelligent control of mobile robots. This chapter provides the list of research objectives and contributions of the proposed work.

Chapter 2 provides a detailed literature review on behavior coordination mechanisms and presents a new classification of the existing coordination approaches. This chapter ends by outlining the common issues of a behavior coordinator related to the problems identified in Chapter 1.

Chapter 3 performs an analysis where FL-based control architecture is used for mobile robot navigation. This analysis explains the advantages and disadvantages of the FL-based technique with respect to the common issues of a behavior coordinator outlined in Chapter 2.

Chapter 4 formulates the proposed FDES-based method. It explains the formalisms of FDES to develop the behavior-based control architecture. This chapter concludes with a justification that the proposed approach addresses the common requirements of a behavior coordinator as specified in Chapter 2.

Chapter 5 demonstrates a novel FDES-based architecture for mobile robot navigation. The navigation system employs both deliberative and reactive planning to produce goal-oriented and safe navigation in dynamic environments. The experimental results show that the robot successfully navigates through dynamic obstacles and

reaches the target position using the FDES-based behavior coordinator.

Chapter 6 describes a novel object-pulling operation with mobile robot to demonstrate the multi-level behavioral decomposition feature of the proposed architecture. The object-pulling task is divided into anchoring and navigation tasks, which are further decomposed into different low-level behaviors. The robot, first, anchors the object and then navigates to the target location. The experimental results show that the FDES-based approach is capable of providing reliable object-pulling operation.

Chapter 7 presents another FDES-based application that models the visual attention system of mobile robots according to the biased competitive hypothesis. It combines the bottom-up bias, which models the sensory feedback from the environment, and the top-down bias, which encounters the feedback from experience. The experimental results demonstrate that the proposed system is able to prevent abrupt changes in visual attention and produces smooth transitions between the motion commands, when visual attention changes from one object to another.

Chapter 8 summarizes the research work and provides suggestions for future work within this area of research.

Chapter 2

Behavior Coordination Mechanisms

2.1 Introduction

The behavior coordination problem involves selecting appropriate behaviors at a particular situation in order to execute relevant actions at that moment. In robotic applications, an action means motor movements, such as pan-tilt commands for a camera, steering commands for a mobile robot, and joint angle commands for a serial arm robot. Since the end-objective of behavior coordination problem is to select an appropriate action, the coordination problem is also known as action selection problem in the current literature. The logical classification of the existing coordination mechanisms was first suggested by Mackenzie *et al.* [75] as shown in Fig.2.1. This method categorizes coordination mechanisms into two major classes, state-based and continuous types. In state-based approaches, the behavior repertoire is clustered into groups of cooperative behaviors. The most appropriate group is selected and activated according to a given state of the environment. On the other hand, in con-

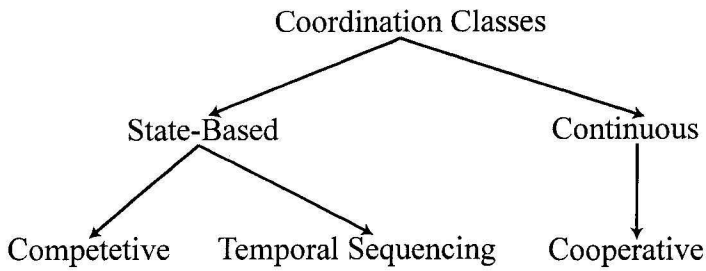


Figure 2.1: Coordination classes proposed in [75]

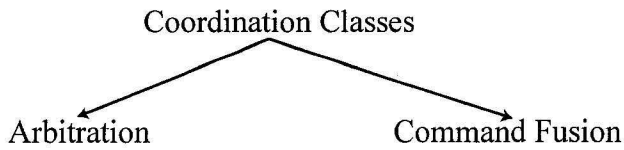


Figure 2.2: Coordination classes proposed in [56]

tinuous methods, all of the behaviors are assumed to be cooperative and are activated simultaneously irrespective of the environmental state. The state-based techniques are further decomposed into competitive and temporal sequencing categories. In the competitive approaches [11,34], the groups of behaviors are ranked with a predefined priority order or activation energies. The group having the highest priority or activation energy with respect to a given environmental context is selected to control the robot. The temporal sequencing techniques [35,36] employ a predefined rule base to select the most relevant group of behaviors using the present and past states of the environment. Saffiotti [56] proposes another classification scheme (see Fig.2.2), where the coordination mechanisms are divided into arbitration and command fusion types, which are similar to MacKenzie's state-based and continuous approaches, respectively. The action selected by arbitration mechanisms distinctly represents one behavior (or a group of behaviors) out of the behavior repertoire, which ensures that the robot always satisfies one goal of the system. However, loss of information is the major drawback of this approach, which stems from the fact that expected actions of

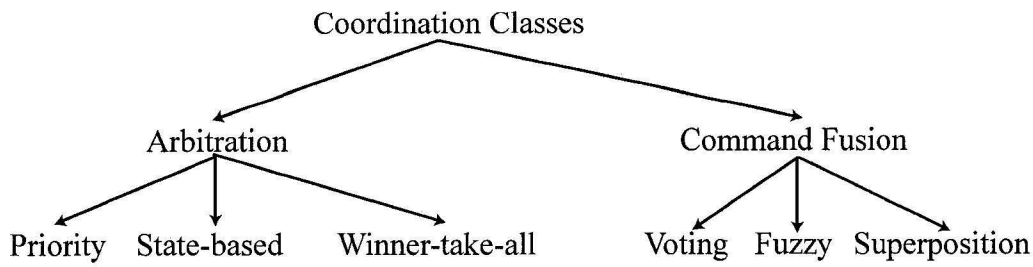


Figure 2.3: Coordination classes proposed in [33]

the losing behaviors are completely ignored [46]. As a consequence, if two conflicting behaviors are selected in successive decision cycles, the robot might exhibit oscillatory response. On the other hand, command fusion techniques fuse the individual actions of behaviors and produce an emergent action that may or may not represent a particular behavior (or a group of behaviors) out of the behavior repertoire. As a result, often the robot does not satisfy any particular goal of the system and in some cases it can be trapped in a stagnant position, called local minima [19]. Moreover, there exists a possibility of unnecessary information processing causing potential delay in action execution. Pirjanian [33] follows the same taxonomy as suggested by Saffiotti and proposes the classification scheme as shown in Fig.2.3. The priority based mechanisms [11, 12, 15, 17, 21, 76] select an action using predefined priorities. The state-based mechanisms [18, 22, 35, 36, 60] select a set of behaviors that is adequately relevant to the current situation. In the winner-take-all mechanisms [34, 37], individual behavior competes to take control of the robot by showing its preference over other behaviors and finally the winning behavior generates the motor command of the robot. In the voting based command fusion [33, 44, 45, 61, 64, 77], a behavior expresses its preferences for all possible actions. The preferences obtained from all behaviors are combined to determine the winning action to control the robot. In the fuzzy approaches [49, 50, 51, 52, 65, 67, 68], rule-based fuzzy behaviors produce preferences for all possible actions. The preferences are then aggregated using fuzzy

operators. Finally, a defuzzification technique is employed to determine the actuator signal. In the superposition-based techniques [42, 40, 62, 78], actions generated by different behaviors are linearly combined to produce the motor command of the robot.

2.2 The proposed classification

In this work, a similar taxonomy as suggested in [33] is followed, however, with more specific categorization with respect to knowledge representation and decision-making techniques of different methodologies. Fig.2.4 illustrates the proposed classification of behavior coordination mechanisms. The coordination classes indexed from 1 to 9 in Fig.2.4 are specified as arbitration categories, whereas coordination class 10 to 20 are designated as command fusion categories. The classification employs several terminologies, such as *priority*, *winnter-take-all*, *voting*, *superposition*, *with/without memory*, *single/multilevel*, *utility*, *context-rule*, *off-line learning*, *differential equation*, and *crisp/fuzzy logic* to specify different characteristics of the coordination mechanisms. The following is a brief description of these terminologies.

- The terms *priority*, *winnter-take-all*, *voting*, and *superposition* carry the same meanings as discussed in Section 2.1.

- ***With/without memory*:** When a coordination mechanism employs previously calculated activation levels of the behaviors (in addition to the currently calculated activation levels) [34, 35, 42, 44, 60, 62, 63], the method is characterized as *with memory*. If the system does not preserve previous activation states [11, 14, 18, 23, 33, 37, 38, 40, 52, 55, 66, 68, 77], it is characterized as *without memory*.

  Coordination systems with memory allows consistent decision-making and smooth

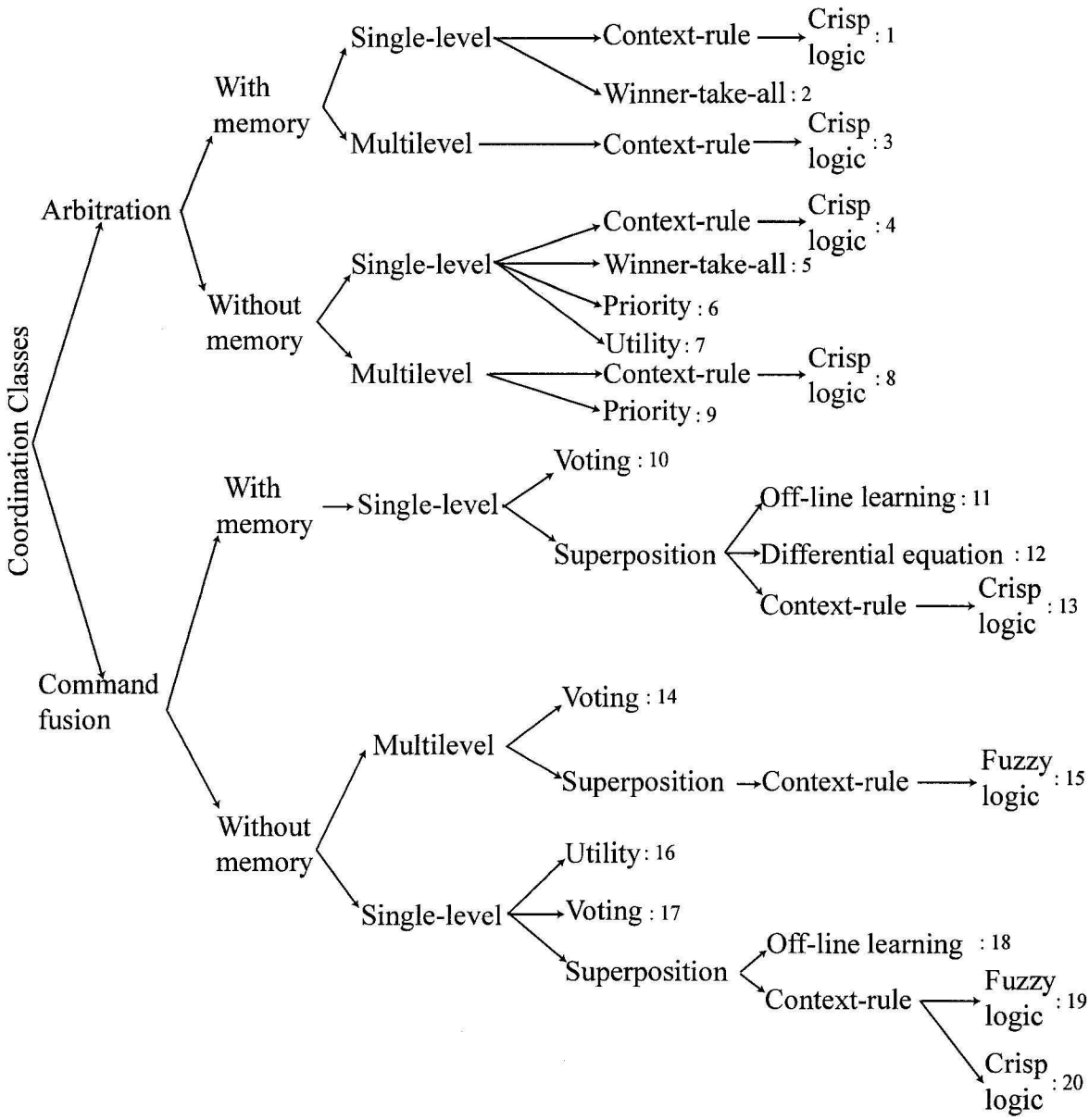


Figure 2.4: The proposed classification

control [10]. However, this causes an adverse effect of forming a complex rule-base as compared to *without memory* system.

- ***Single/multilevel:*** *Multilevel* approaches [18,22,23,64,65] use layered behavioral decomposition, which is constructed using prior knowledge. The higher level behaviors are abstract and are not associated with physical actions. The lowest level constitutes primitive behaviors associated with individual motor commands generated using sensory feedback. Behavior coordination using *multilevel* approach differs in arbitration and command fusion techniques. In arbitration methods [18, 22, 23], a higher level behavior is selected from each level and finally, a group of primitive behaviors are selected to produce actions at the lowest level. The actions of the selected lowest level behaviors are linearly combined or sequentially executed. This coordination approach is known as top-down type since the selection process is hierarchically downward. In command fusion techniques [64, 65], the coordination is bottom-up type. Here, behaviors at the lowest level are activated using varying weights. A higher level behavior is constructed using the weighted actions of its lower level behaviors. Finally, the top-most behavior generates the motion commands. *Single level* approaches [11, 13, 14, 34, 37, 38, 40, 42, 55, 61, 62, 63, 66, 77] do not utilize a layered behavioral decomposition; instead prior knowledge and current sensory feedback are directly used to select a subset of primitive behaviors.

  The main advantage of *single level* approach lies in the fact that it exploits inter-behavior dependencies of all behaviors simultaneously, which makes the decision-making more reliable [46]. However, this approach also results in potential delay in action selection and has negative effect on the reactivity of the system. On the contrary, *multilevel* approaches only consider a subset of behaviors at a time, which reduces the decision-complexity and helps taking timely

response in dynamic environments [9]. However, this approach may produce undesired motion commands since it does not consider inter-behavior dependencies of all behaviors simultaneously.

- ***Utility*:** In *utility* based approaches [38, 66], the robot's workspace is divided into a set of world states, where each world state is associated with a group of probabilistic measurements, called utility values. A utility value refers to the goodness of a world state with respect to an action, current sensory data, and the present goal of the robot. The action that maximizes the expected utility is selected to control the robot.

  The main challenge of this approach is in the formulation of an appropriate utility function, which integrates dynamic changes in the environment in order to produce an appropriate action to fulfill the current goal of the robot [46]. This approach becomes computationally expensive to accomplish a multi-objective task in a complex environment, where the number of world states and actions increase significantly.

- ***Context-rule*** and ***crisp/fuzzy logic*:** The *context-rule* based approaches [13, 14, 18, 22, 62, 65, 68] employ a predefined rule-base that processes the sensory feedback and prior knowledge of the system to select a relevant group of behaviors with respect to the current world state. The context rules are constructed using either *crisp logic* or *fuzzy logic*. *Crisp logic* uses bivalent reasoning where the truth value of a predicate is either 0 or 1; whereas *fuzzy logic* uses multi-valued reasoning where the truth value of a fuzzy predicate ranges from 0 to 1.

  Crisp logic based rule-base employs simple binary thresholding to form the predicates resulting in faster decision-making [13, 14, 18, 22, 62]. However, this

leads to wrong behavior selection when the sensory data is noisy and produces hard switching between conflicting behaviors resulting in oscillatory robot motions. Hence, *fuzzy logic* based rule-base [49, 50, 51, 65, 67, 68] becomes handy for showing robust performance against noisy sensory data and it minimizes the possibility of occurrence of hard switching between conflicting behaviors.

- ***Off-line learning*:** These techniques [55, 63] employ *off-line learning* approaches to train a Neural Network (NN) that determines the current world state using the sensory data and then selects an appropriate set of behaviors to control the robot.

  This approach also performs well against noisy sensory data. However, it takes lengthy learning period and requires adequate training data for desired performance [19].

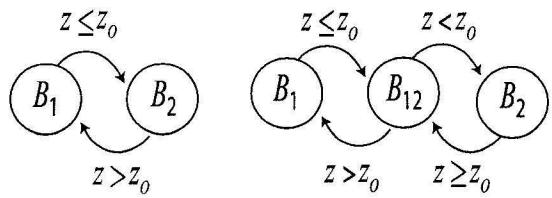
- ***Differential equation*:** These coordination mechanisms [42, 79] use *Differential equation* to determine the behavioral activity and provides the opportunity to preserve the previously calculated activity levels of the behaviors.

  This coordination class provides a systematic approach to model behavioral activity with differential equation. However, it performs well only in well defined situations and cannot be used in complex environments.

Table 2.1 lists the existing methodologies under different categories of behavior arbitration and command fusion mechanisms. The following section summarizes different aspects of individual methodologies.

Methodologies	Coordination class	Characteristics
Universal Plans '87 [13] Situated Control Rules '89 [16] Task Control Architecture '94 [24] Discrete Event System '94 [35] Hybrid Automata '00 [39]	1	arbitration\with memory single-level\context-rule\crisp logic
Activation Networks '90 [34]	2	arbitration\with memory single-level\winner-take-all
RAP '87 [22] Temporal Sequencing '94 [36] AASBA '02 [60]	3	arbitration\with memory multilevel\context-rule\crisp logic
Reactive Planning '87 [14]	4	arbitration\without-memory single-level\context-rule\crisp logic
Inter-behavior Bidding '94 [37]	5	arbitration\without-memory single-level\winner-take-all
Subsumption Architecture '86 [11] Reflexive Control '86 [21] Pengi '87 [12] GAPPS '88 [15] Supervenience Architecture '91 [17] SSS Architecture '92 [76]	6	arbitration\without-memory single-level\priority
Bayesian Decision Analysis '97 [38]	7	arbitration\without-memory single-level\utility
Conditional Sequencing '94 [18]	8	arbitration\without-memory multilevel\context-rule\crisp logic
Behavior Mediation '87 [23]	9	arbitration\without-memory multilevel\priority
Action Voting '95 [61]	10	command-fusion\with memory single-level\voting
Emotion-based '02 [63]	11	command-fusion\with memory single-level\superposition\off-line learning
Dynamical Systems '92 [42]	12	command-fusion\with memory single-level\superposition\differential equation
Multi-objective '01 [62]	13	command-fusion\with memory single-level\superposition\context-rule\crisp logic
Layered DAMN '93 [64] Decision-theoretic '98 [33]	14	command-fusion\without memory multilevel\voting
Fuzzy Activity '94 [65] Context & activity '04 [52]	15	command-fusion\without memory multilevel\superposition\context-rule\fuzzy logic
Utility Fusion '98 [66]	16	command-fusion\without memory single-level\utility
DAMN '89 [77] Fuzzy DAMN '95 [45] Action Map '97 [44]	17	command-fusion\without memory single-level\voting
Neuro-fuzzy '05 [55]	18	command-fusion\without memory single-level\superposition\off-line learning
Fuzzy Context '93 [68] Fuzzy Motives '97 [49] Modular Fuzzy '97 [67] Action & Activity '00 [50] Fuzzy Management '03 [51]	19	command-fusion\without memory single-level\superposition\context-rule\fuzzy logic
Potential Fields '86 [40] Motor Schema '87 [78]	20	command-fusion\without memory single-level\superposition\context-rule\crisp logic

Table 2.1: Classification of behavior coordination mechanisms



(a) (b)

Figure 2.5: A comparison between (a) conventional FSA, and (b) HA based approaches

2.3 Brief descriptions of the coordination classes

Coordination class 1: Examples of this category include Universal Plans [13], Situated Control Rules [16], Task Control Architecture [24], DES [35], and Hybrid Automata (HA) [39] based approaches. In Universal Plan [13] and Task Control Architecture [24], a plan is expressed in terms of a decision tree that organizes the possible world states. An action is selected considering the present and past world states. A similar approach is described in Situated Control Rules [16], where the world states are modeled using a Petri Net. The method described in [35] uses Finite State Automata (FSA) based formalisms of Discrete Event System to select an action at a given state. This technique also has the ability to perform a system analysis using observability and controllability. The method described in [39] employs HA to select a primitive behavior at a given world state. It differs from the conventional FSA based approaches, where a sliding behavior has been introduced in between the transition of two primitive behaviors. Fig.2.5 illustrates an example where the sliding behavior B_{12} is constructed using B_1 and B_2. Using conventional approach (see Fig.2.5(a)), the behavioral transition occurs on the basis of binary thresholding of sensory data z (z_o is the threshold). In HA-based approaches, B_{12} takes place in between B_1 and B_2. The sliding behavior B_{12} is constructed using weighted-average of B_1 and B_2. The author claims that the use of sliding behaviors reduces the possibility of hard

switches as observed in FSA based methods. The main challenge of this coordination class is to model a complete state-space model reflecting all possible world states. Furthermore, binary thresholding based state transition often causes selection of a wrong world state [56].

Coordination class 2: Activation Networks [34] is a typical example of this category. In this approach, each behavior is associated with an activation energy, which is continuously updated using the following factors.

- *Activation by the environment state:* Activation energy is spread from the environment to behaviors that match the current state.
- *Activation by the goal:* Each goal injects activation to behaviors so that their execution can fulfill the goals.
- *Inhibition by protected goals:* The activation level of a behavior is decreased if it can undo a goal that has already been achieved.
- *Activation of successor:* An executable behavior B spreads activation to behaviors with preconditions that will become true after activation of B.
- *Activation by predecessor:* A behavior B that is not executable spreads activation to other behaviors, which can make B executable.
- *Inhibition of conflictors:* Every behavior decreases the activation level of behaviors, called conflictors, that can make its preconditions false.

At each decision cycle, an executable behavior with the highest activation level is selected. The major challenge of this method is to design activation/inhibition functions that model the desired environmental dynamics and inter-behavior dependencies, which is a tedious task for a large behavior-based system.

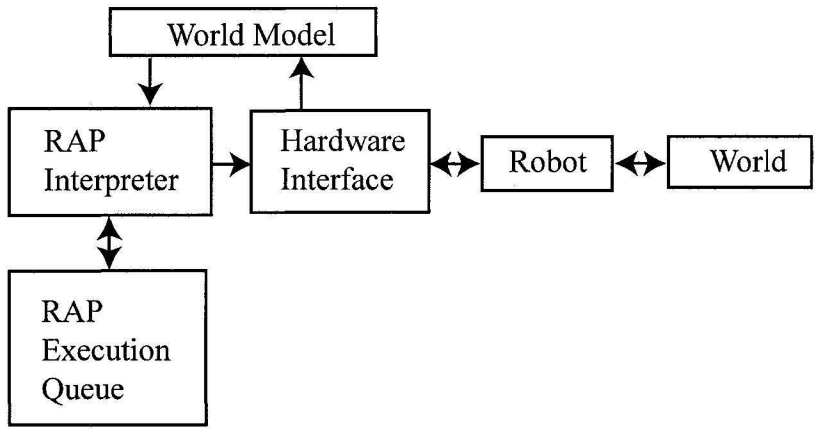


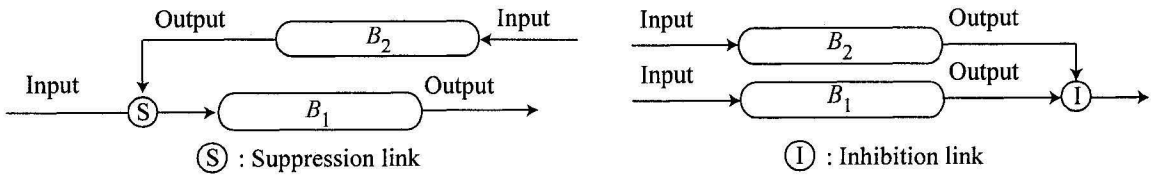
Figure 2.6: Behavior coordination in RAP [22, 80]

Coordination class 3: Examples of this coordination class include Reaction Action Package (RAP) [22, 80], Temporal Sequencing [36], and AASBA [60]. Fig.2.6 depicts the RAP execution environment. The heart of this system is the *RAP interpreter* that deploys context rules to select a high level behavior. The context rules are constructed using the updated world model and sensory feedback. The *RAP interpreter* also assigns various activation states to high level behaviors, such as *done*, *suspended*, and *waiting*. The *RAP execution queue* maintains a list of behaviors according to the activation states. A major shortcoming is observed in this approach, when a *waiting* behavior is never activated since its precondition is violated by another behavior [22]. In Temporal Sequencing approach [36], higher level behaviors are selected according to the state-transition structure of a FSA. Temporal Sequencing is often subjected to oscillatory responses due to the existence of binary logic in state-transition rules. AASBA [60] maintains emotion-oriented state variables such as *frustration*, *satisfaction*, etc. Hence, the context rules of AASBA selects a particular set of primitive behaviors depending on the emotional state of the robot. The major issue of this approach is in the development of an appropriate mechanism to update the state variables.

Coordination class 4: Reactive Planning [14] is an example of this coordination mechanism. In this method, available information is divided into separate categories, such as *Beliefs*, *Plans*, *Desires*, and *Intentions*, which are used to form a centralized rule-base. The coordinator employs this rule-base to select an appropriate action for task execution. The key issue of this method is in the rule-base design, which becomes difficult when the number of information-category increases for a complex task execution.

Coordination class 5: Inter-behavior Bidding [37] is an instance of this category. In Inter-behavior Bidding [37], behaviors bid among themselves to win the control of the robot. A bid is produced by each behavior to estimate how beneficial it would be for the behavior to gain control of the robot. The behavior with the highest bid wins the competition and generates the motor commands of the robot. The major drawback of this method is that it does not include an explicit model of inter-behavior dependencies. As a result, this approach may select conflicting actions in successive decision cycles leading to oscillatory motion commands [9].

Coordination class 6: Examples of this category includes Subsumption Architecture [11], Reflexive Control [21], Pengi [12], GAPPS [15], Supervenience Architecture [17], and SSS Architecture [76]. Subsumption Architecture [11] consists of a set of concurrent behaviors assigned with predefined priorities. A higher priority behavior can override the input and output of a lower priority behavior via suppression and inhibition links, as shown in Fig.2.7(a) and 2.7(b). However, as the number of behaviors goes up, designing appropriate suppression and inhibition links becomes increasingly complex to achieve a task [27, 81]. SSS Architecture [76] is similar to Subsumption Architecture with the exception that it uses both sensory data and a continuously updated world model as the behavioral inputs. In Reflexive Control [21], the behavior coordinator is termed as the *reflexive planner* that assesses progress to the given goal



(a) Suppression (b) Inhibition

Figure 2.7: Combining behaviors in Subsumption Architecture, adapted from [11]

and selects a subset of behaviors, called *activation set*, to accomplish the task. The behaviors are prioritized and the highest priority behavior of *activation set* is chosen to generate the actuator command. Pengi [12] is a simulated autonomous agent, where the procedural plan is replaced by a set of concurrent rules to select actions based on the current situation. It employs a priority based action selection to arbitrate among selected actions. GAPPS (Goals as Parallel Program Specifications) [15] is a programming language for specifying behaviors of an autonomous agent. The GAPPS compiler takes a declarative specification of the agent's goal as the input and generates a set of condition-action rules. The rules are ranked using a priority index and the action corresponding to the satisfied highest-ranked rule is selected to generate the actuator command. Behavior coordination in Supervenience Architecture [17] is similar to that of Subsumption Architecture [11]. However, it prohibits the use of inhibition or suppression link and uses a higher priority behavior to parameterize a lower priority behavior.

Coordination class 7: This coordination class employs probabilistic approaches to maximize the utility of an expected action and selects the behavior corresponding to the action having maximum utility. The mechanism described in [38] is an example of this approach (see Fig.2.8), which is based on Bayesian decision and utility theory. Here, z_{ij} denotes the estimated sensory data obtained from the sensor associated with behavior B_i and A_i is the expected action of B_i with respect to z_{ij}. The execution

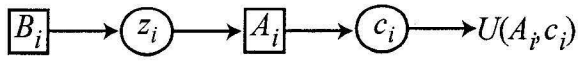


Figure 2.8: Decision process used in [38]

of action A_i leads to an environmental context (or state) c_i. The utility associated with A_i and c_i is calculated as $U(A_i, c_i)$ using Bayesian decision theory and then, the behavior corresponding to maximum utility is selected. The main problem of this approach is associated with the probabilistic models of the expected utility which is usually constructed experimentally through a number of experiments. Significant changes in the environment usually mean that the model should be regenerated.

Coordination class 8: Conditional Sequencing [18] is an example of this category. The overall plan is expressed as a set of instructions that are not sequentially ordered; instead, each instruction is invoked on the basis of the situation that unfolds during execution. Moreover, robustness is achieved by having a large number of contingency procedures which can recover from failure. However, designing and organizing a large number of contingency procedures for a complex behavior-based system becomes a tedious work.

Coordination class 9: Behavior Mediation [23] is an example of this approach (see Fig.2.9). Here, the top level behaviors are defined as *Level* 1, 2, and 3, where *Level* 3 is assigned the highest priority. The *Mediator*, first, finds the active behaviors with respect to current environment and then selects the highest priority active behavior to control the robot. A high level behavior is further decomposed into primitive behaviors (e.g., *follow walls*, *avoid obstacles*), which are mediated using a similar priority-based rules. However, the priority-based selection often causes *starvation* problem, where the same behavior is continuously selected ignoring the actions of other behaviors.

Coordination class 10: Action Voting [61] is an instance of this category. In

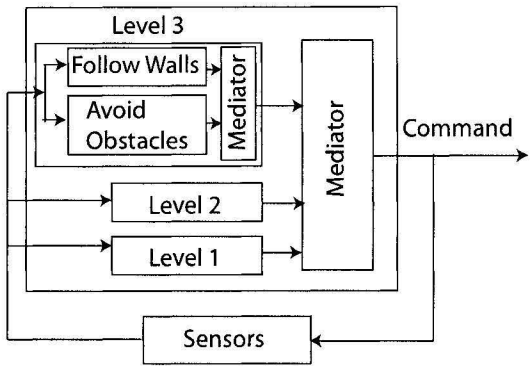


Figure 2.9: Behavior mediation, adapted from [23]

this approach, each behavior is implemented by a neural network, which maps the current sensory data into preferences (or votes) for the desired actions. Behaviors also compete by inhibiting conflicting actions which essentially corresponds to voting against undesired actions. The votes and inhibition values received from each behavior are summed for each action and the action having the highest value is selected. This method preserves the memory of the previous activation status of a behavior in terms of its adaptive inhibition functions. When a behavior is inactive for a long period of time, its inhibition function is amplified to allow the behavior to compete and win. If a behavior's suggested action leads another behavior to suffer, then the behavior's inhibition function is damped to allow loosing behaviors to win. The performance of this approach largely depends on adequate training of the neural networks as well as appropriate modeling of the inhibition functions that ensures conflict-free behavioral activation and oscillation-less motion of the robot.

Coordination class 11: Emotion-based approach [63] is an instance of this category. This method employs trained neural networks to determine the frustration levels of behaviors. Each neural network is constructed using following formula

$$h(k+1) = \tau_1 \sum_j \nu_j z_j + (1 + \tau_2 \sum_j \nu_j z_j - b)h(k)$$

where $h(k)$ represents the frustration level at time step k; $\{\tau\}$ and $\{\nu\}$ are the trained network coefficients; $\{z\}$ is the sensory data; and b is the threshold, which determines the patience against unpleasantness. The modulating factor $\beta = f(h(k+1))$, which is inversely proportional to the frustration level. Finally, β is deployed to weigh behavioral actions and the weighted-summation of the actions denotes the final motion command. The key issue associated with this approach is to adequately train the neural networks so that the motion commands are appropriately guided by the robot's emotion.

Coordination class 12: Dynamical Systems Approach [42, 79] is a typical example of this category. In this approach, each behavior is described by a differential equation. For example, *go-to-target* and *obstacle-avoidance* behaviors can be expressed using $\dot{\theta}_{\mathcal{T}} = f_1(\tilde{\theta})$ and $\dot{\theta}_{\mathcal{O}} = f_2(\tilde{\theta})$ respectively, where $\tilde{\theta}$ is the previous heading direction of the robot. The orientations of the target and obstacles are specified as $\theta_{\mathcal{T}}$ and $\theta_{\mathcal{O}}$. The combined behavior is obtained using their linear superposition as $\dot{\theta} = |\beta_{\mathcal{T}}|\dot{\theta}_{\mathcal{T}} + |\beta_{\mathcal{O}}|\dot{\theta}_{\mathcal{O}}$, where $\{\beta\}$ is the weighting factor. The solution θ of the combined differential equation gives the desired heading direction of the robot. The wighting factors are also determined using differential equations of the following form

$$\begin{aligned} \dot{\beta}_{\mathcal{T}} &= \tau_1 \times \beta_{\mathcal{T}} \times (1 - \beta_{\mathcal{T}}^2) - \nu_1 \times \beta_{\mathcal{O}}^2 \times \beta_{\mathcal{T}} \\ \dot{\beta}_{\mathcal{O}} &= \tau_2 \times \beta_{\mathcal{O}} \times (1 - \beta_{\mathcal{O}}^2) - \nu_2 \times \beta_{\mathcal{T}}^2 \times \beta_{\mathcal{O}} \end{aligned}$$

where $\{\tau\}$ denotes the preference for a behavior and $\{\nu\}$ indicates the inhibitory effects from other behaviors. One of the major challenging issues of this approach is to model inter-behavior dependencies in terms of $\{\nu\}$ that assists achieving desired goal of a large behavior-based system.

Coordination class 13: Multi-objective approach [62] presents an example of this category. Fig.2.10 shows the behavior coordination mechanism used in this method. This method employs two heuristically defined functions for behavior coordi-

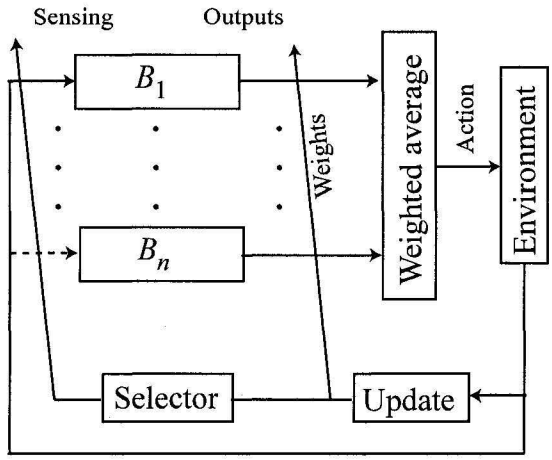


Figure 2.10: Behavior coordination in Multi-objective approach [62]

nation. The *selector* function selects a set of behaviors using the modulating weights generated by the *update* functions. The weights are produced while considering their past values and current sensory data. The final motion command is the weighted-average of the actions of active behaviors. The important aspect of this approach lies in defining the *selector* and *update* functions.

Coordination class 14: Layered-DAMN [64] and Decision-theoretic approach [33,54] are instances of this category. In Layered-DAMN [64], behaviors are arranged in multilevel. The action preferences of a lower level behaviors are constructed using the votes of the higher level behaviors. At the lowest level, the action with the highest vote is selected. In Decision-theoretic approach [33, 54], behaviors are, first, grouped into homogenous (or cooperative) behaviors and their action preferences are combined using a voting technique (see Fig.2.11). The combined action preferences are termed as heterogenous (competitive) behaviors. The action preferences of the heterogenous behaviors are weighed using subjective preferences in terms of weights. The actions that satisfy the preferences of all heterogenous behaviors to a certain degree are selected as the satisficing actions. Finally, another subjective knowledge (e.g., priority) is included to select a satisficing action in order to control the robot.

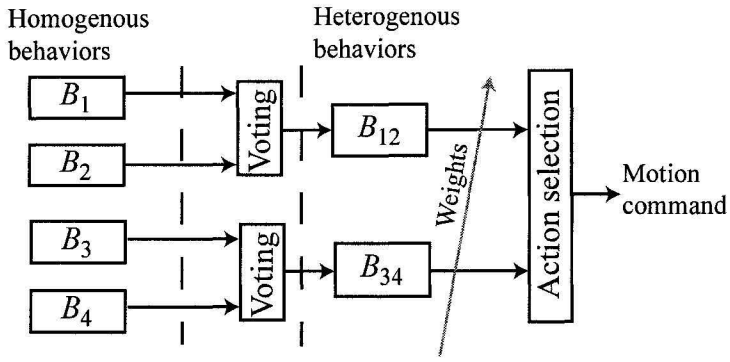
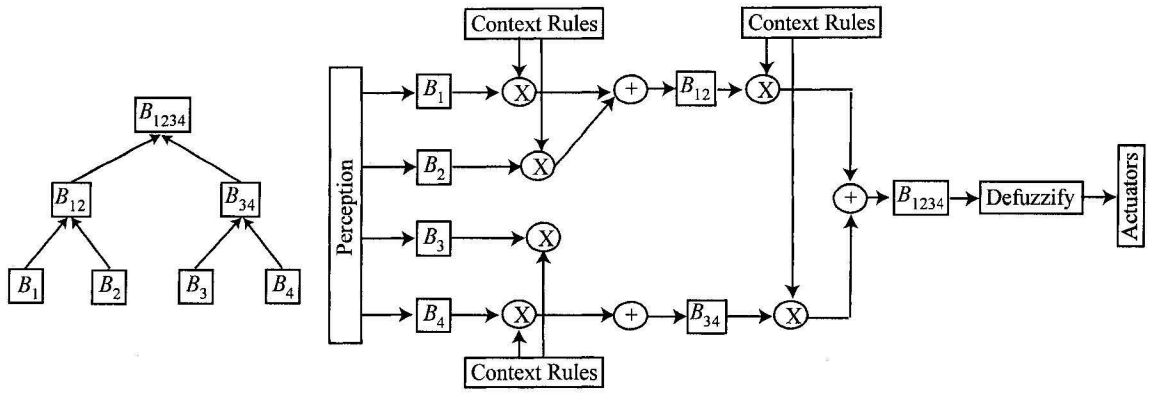


Figure 2.11: Behavior coordination using Decision-theoretic approach

One of the major issues associated with this method is in the generation of weights for selecting satisficing actions. The weights are produced to eliminate conflicts between behaviors at a given context of the environment.

Coordination class 15: The methods described in [52, 53, 65, 82] are examples of this category. In [53, 65, 82], behaviors are decomposed into multilevel (see Fig.2.12(a)), where the action preference (a fuzzy set over possible actions) of a higher level behavior is constructed using weighted aggregation of the action preferences of a subset of its lower level behaviors. For the construction of each higher level behavior, a separate set of fuzzy context rules are used to generate the weights for its associated lower level behaviors (see Fig.2.12(b)). The final control command of the robot is generated by defuzzifying the action preference associated with the top-most behavior. The main challenge of this method is in the formation of context rules that generate appropriate weights to control the behavioral activity. However, formation of context rules becomes a difficult task when a higher level behavior consists of a large number of lower level behaviors leading to significantly increased number of rules. In coordination technique reported in [52] also uses multilevel behavioral decomposition. However, this approach uses a central rule-base to select higher level behaviors, whereas the primitive behaviors associated with higher level behaviors are



(a) Hierarchical decomposition

(b) Behavior coordination

Figure 2.12: Hierarchical fuzzy behavior coordination in [65]

coordinated using fuzzy context rules as described in the previous method [65].

Coordination class 16: Utility Fusion [66] is an instance of this category. In this architecture , behaviors do not suggest any action; instead they define utility $U(c)$ for each possible context (or state) c in environment. Hence, the arbiter calculates the expected utility $U(A)$ for an action A that assists to reach the environmental state c. $U(A)$ is calculated as

$$U(A) = \sum_{c} U(c) \times P(c|A, z)$$

where $P(c|A, z)$ is the probability of reaching state c given action A is executed under sensory-observation z. The action with the highest expected utility is selected to generate the motion command. Performance of this method largely depends on the formation of utility function that adaptively defines the utility of different environmental states and ensure desired action selection to control the robot.

Coordination class 17: DAMN [43,77,83,84,85], Fuzzy DAMN [45], and Action Map [44] are the examples of this category. Fig.2.13 illustrates the behavior coordination technique in DAMN (Distributed Architecture for Mobile Navigation) [77], where each behavior votes for or against the possible set of actions of the robot. The

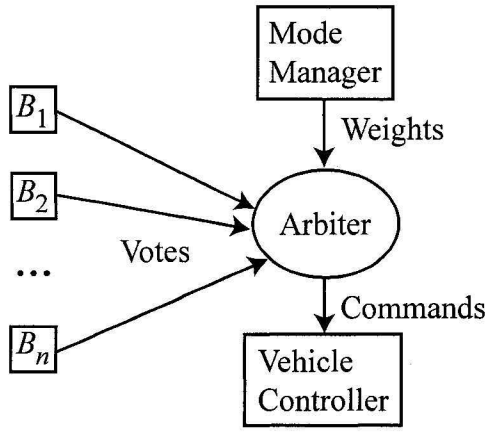


Figure 2.13: Behavior coordination in DAMN [77]

Arbiter combines the behavioral preferences using weighted summation of votes. The *Mode manager* performs an important role by generating appropriate weights that express context-dependent activity of the behaviors. Action Map [44] is similar to DAMN [77] except that action preferences of the behaviors are termed as actions maps. Fuzzy DAMN [45] is also similar to DAMN [77] except that the decision process is implemented using FL. Hence, behavioral votes are expressed using fuzzy sets over possible actions. The votes are combined using weighted aggregation of the fuzzy sets. Finally, the task of action selection is performed using a defuzzification technique, e.g., Centroid of Largest Area (CLA).

Coordination class 18: An example of this category is reported in [55], where a neuro-fuzzy based behavior coordinatior is used for mobile robot navigation (see Fig.2.14). In this approach, each behavior employs fuzzy rules to determine its expected action. The weighted summation of these actions is used as the final motion command. The weights are generated using a trained neural network. An error signal, which is obtained comparing the difference between the desired and online-generated robot's trajectory, is used as the input to the neural network. The key issue related to this approach is the lengthy training phase of the neural networks, which requires

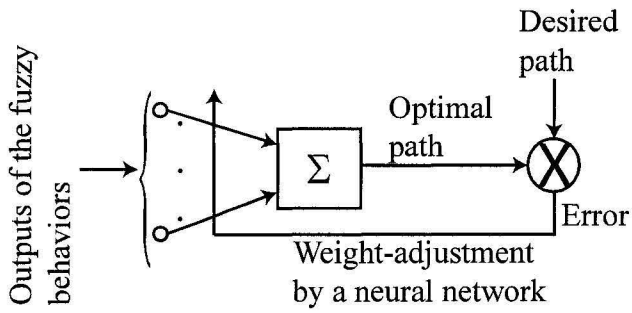


Figure 2.14: Neuro-fuzzy based behavior coordination in [55]

adequate training data for several environment configurations.

Coordination class 19: Examples of this category include Fuzzy Context [68, 48], Fuzzy Motive [49], Modular Fuzzy [67], Action and Activity [50], and Fuzzy Management [51]. In Fuzzy Context [68,48], each behavior generates its action preference using a fuzzy set over possible actions, which is further weighed using the modulating factors generated by a central fuzzy rule-base (see Fig.2.15). The aggregated fuzzy set is, then, defuzzified to generate the actuator command. Fuzzy Motive [49] is an

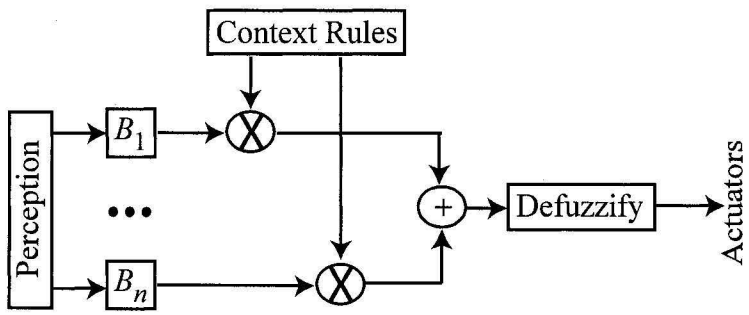


Figure 2.15: Context dependent fuzzy behavior blending, adapted from [68]

extension of the previous method, where the context rules are divided into different categories, such as *Motive*, *External Situation*, *Need*, and *Cognition*. *Motive* and *Need* select behaviors according to the current goals and needs of the robot. *External Situation* and *Cognition* select behaviors on the basis of external conditions in the

environment and updated topological map. The selection of behaviors is expressed in terms of positive and negative weights, which infer activation and inhibition respectively. The weights are aggregated using fuzzy max operator and are compared to predefined threshold values to activate individual behaviors. The final motion command is generated using the weighed action preferences of the activated behaviors as described in [68]. In Modular Fuzzy [67], and Action & Activity [50], a separate fuzzy rule-base is used for each behavior to generate its action and activity (in terms of weights). The final action is determined using weighted-average of the actions. In Fuzzy Management [51], fuzzy context rules are divided into three categories: *Motivation Condition*, *Activation Condition*, and *Internal Knowledge*. Behaviors are activated if internal states of the robot and environmental states are satisfied to a certain degree with respect to *Internal Knowledge*, and *Activation Condition*. The modulating weights are generated using *Motivation Condition*, which employs the knowledge of the current goal of the robot. Finally, the actuator command is produced using weighted superposition of the actions of activated behaviors.

Coordination class 20: Potential Field [40] and Motor Schema [41,78] fall into this category. Potential Field approach [40] presents a behavior-based navigational method, where *go-to-target* behavior is constructed using an attractive potential field and *avoid-obstacle* behavior is formed using a repulsive potential field. The attractive potential field ζ_{att} is defined such that a global minimum exists at the target configuration. On the other hand, the repulsive potential field ζ_{rep} constitutes a maximum at each obstacle configuration. Action selection in this terminology corresponds to the direction indicated by the force $\vec{V} = -\vec{\nabla}\zeta$, which is the negated gradient of the total potential $\zeta = \zeta_{att} + \zeta_{rep}$. The key issue related to this method is in the potential field definition, where there is a chance of creating a local minima, which can cause a trap situation of the robot. Hence, the potential fields are constructed

using heuristics that employ different potential functions at different environmental contexts. Motor Schema approach [41, 78] is an extension of the previous method, where the given mission of the robot is divided into a set of attractive forces and the unexpected environmental events are represented using repulsive forces. The force vectors are modulated using heuristically generated weights and their weighted vector summation determines the final motion command. The major challenge of this approach is in the determination of the modulating weights of the force vectors that can resolve the problem of behavioral conflicts and produces local minima-free control commands. Motor Schema approach also differs from Potential Field method in that the former only consider sensory data for force vector calculation whereas the later considers the entire workspace to generate the motion command.

2.4 Important aspects of a behavior coordinator

In the previous section, the coordination classes are described in a nutshell. It is believed that the methods described include relevant representatives of the particular categories. On the basis of the literature survey presented in Section 2.3, the important aspects of a behavior coordinator can be listed as bellow.

- A behavior coordinator must be capable of specifying competitive and cooperative behaviors for relevant action selection. Selection of a competitive behavior ensures goal-oriented action selection at a particular environmental context. However, often expected actions of the competing behaviors might be similar and their simultaneous cooperative execution provides an opportunity to satisfy the expected actions of different behaviors to a certain degree. Behavior arbitration techniques (presented in Table 2.1) provide suitable means for selecting a competitive behavior, whereas command fusion techniques (presented in Ta-

ble 2.1) report weighted-decision making with cooperative behaviors. Hence, combination of both the behavior arbitration and command fusion techniques are required for relevant action selection.

- A behavior coordinator must possess effective means to model world states so that it favors the appropriate behaviors at a given world state. Hence, Discrete Event System [35], Petri Net [16], decision-tree [13, 24], and fuzzy context-rule [56] based knowledge representations are eligible candidates for world states modeling.

- A behavior coordinator must possess adequate means to handle sensory uncertainty. To achieve this goal, FL-based techniques are proven better than crisp logic based approaches [86]. Fuzzy techniques use multi-valued logic, which is more tolerant to noisy sensory data and it prevents occurrences of abrupt changes to motion commands due to erroneous perceptions.

- Persistence is an important aspect of the behavior coordination problem. It favors those behaviors which contribute to the ongoing goal. Hence, a behavior coordinator must consider past and present states of the system so that the behavioral activities are controlled in a consistent manner while taking into account the present context of the world. Preserving system memory enables removing contradictory behavior selection in successive decision cycles, thus, providing oscillation-free action selection. The methods characterized as *with memory* (see Fig.2.4 and Table 2.1) employ this technique for action selection.

- A behavior coordinator must exhibit timely responses to environmental changes (which is known as reactivity). To achieve this goal, a coordination system employs computationally inexpensive methods for fast decision-making. Hence, hierarchical approaches, which include multilevel behavioral decomposition (e.g.,

Temporal Sequencing [36]) and decision tree based techniques (e.g., Task Control Architecture [24]), play an important role for fast behavior selection at a given context. Hierarchical approaches employ prior domain knowledge as well as current sensory data for fast and appropriate behavior selection at a given context.

- A behavior coordinator should possess the ability to predict the consequence of an action execution. It enables the system to avoid hazardous situations beforehand. Probabilistic methods are the best candidates to achieve this goal. As an example, utility based approaches [38, 66] employ probabilistic inference to determine the expected utility of an action on the basis of probable world states.

- A behavior coordinator should provide some means of decision analysis that helps taking corrective actions if necessary. As an instance, the DES based approach [35] is furnished with *observability* and *controllability* analysis that helps modeling inadequate sensory perceptions and occurrence of undesired world states. Thus, these analyses helps taking a relevant action when perception is insufficient and unexpected changes occur in the robot's workspace.

- Robustness is one of the important aspects of a behavior coordinator. It is achieved using modular architecture, where addition of a new behavior does not effect the original behavior coordination architecture. It also infers that the malfunction of one part does not cause failure of the entire system. Modular approaches also increase scalability that enables modeling of a large behavior based system. As an instance, modular fuzzy approaches (e.g, [65,67]) that use separate fuzzy rule bases for behavior activation is more robust than monolithic fuzzy approaches (e.g. [68]) with a single fuzzy rule base to control activities

of all behaviors. Hence, addition of a behavior directly affects the existing rule base of a monolithic approach.

Reliability of a behavior coordinator is another important issue in addition to the aspects mentioned above. However, it does not give emphasis to any particular property; instead refers to the overall quality of taking timely response in dynamic environments, handling noisy sensory data, and making corrective actions.

2.5 The proposed behavior coordinator

Upon completion of the literature survey, this work proposes that a behavior coordinator should

1. combine both behavior arbitration and command fusion techniques to facilitate the coordination of competitive and cooperative behaviors,
2. possess adequate means of world states modeling,
3. employ multi-valued logic to handle sensory uncertainties,
4. be capable of making persistent behavior selection,
5. provide the opportunity of hierarchical decision-making for reactive action generation,
6. be capable of predicting probable world states to handle environmental uncertainties beforehand.
7. provide satisfactory means of decision analysis, and
8. be modular to increase the robustness of the system.

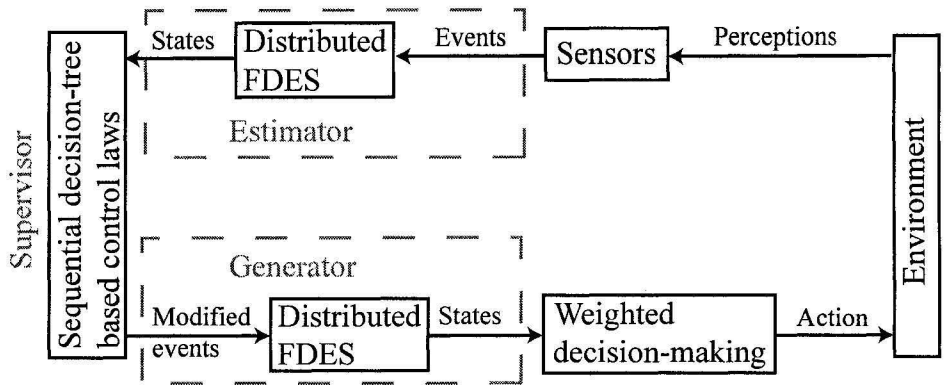


Figure 2.16: The proposed coordination system

The present work puts forward a novel behavior coordination technique that attempts to integrate the above mentioned characteristics of a behavior coordinator. However, property 6 will not be addressed in this work since the formalism required for predictive control requires probabilistic inferences, whereas the proposed work intends to exploit multi-valued logic based possibility theory. At the end of this thesis, further research directions will be provided to incorporate both possibility and probability theory in the same frame for the behavior coordination problem.

Fig.2.16 briefly illustrates the proposed framework. In this methodology, FDES is used for state modeling. The FDESs are employed in a distributed way, where separate FDESs are used to generate the activity of a behavior. The activity of a behavior is expressed using a fuzzy state vector, which is updated at each decision cycle using fuzzy event matrices. The proposed FDES-based approach exploits the supervisory control concept of DES, where the performance of a DES is controlled by modifying its events using predefined control rules. In the proposed method, behavioral activity is determined in two steps. First, activity states are estimated using the events constructed using sensory feedback. Second, a supervisor is engaged to assess the activity states using a predefined decision tree. The decision tree is formed using sequential hierarchy, where the behaviors are placed and ranked accord-

ing to their expected order of execution. The supervisor finds the most appropriate behavior considering the ranking and estimated activity states, and determine its cooperative behaviors. Finally, the events of the cooperative behaviors are modified to generate desired activity states. The motion command is determined using weighted actions of the behaviors, where the modulating weights are obtained by defuzzifying the activity state vectors. The FDES-based approach also provides tools for decision analysis using observability and controllability. Chapter 4 will justify the claim that the proposed method addresses the desired properties of a behavior coordinator.

2.6 Conclusion

In this chapter, a detail account of existing literature survey has been presented. This work extends the current taxonomy on behavior coordination problem on the basis of knowledge representation and summarized individual categories. The important aspects of a behavior coordinator is also outlined. Finally, a novel behavior coordination mechanism has been proposed to integrate the desired properties of a behavior coordinator. The following chapter will present an experiment using existing fuzzy context-dependent behavior blending to point out the relevance of the desired properties cited in this chapter. Next, in Chapter 4, the proposed FDES-based approach will be demonstrated.

Chapter 3

Mobile Robot Navigation Using Motor Schema and Fuzzy Context Dependent Behavior Modulation

3.1 Introduction

In this chapter, an experiment using two popular behavior-based approaches, motor schema [41] and fuzzy behavior modulation [48], is presented in the fields of mobile robot navigation. The experiment explores the merits and demerits of these approaches to establish the relevance of the proposed behavior-based technique. Motor schema [41,78,87] is a popular command fusion technique in mobile robot navigation, where the navigation task is decomposed into specific motor skills or schemas. Each schema is represented by a force vector and the summation of the force vectors is specified as the coordinated action to achieve a particular goal. In this approach, a goal is represented by an attractive force to the target, whereas obstacles are represented by repulsive force. The orientation of the resultant force vector provides

the expected heading direction of the robot. Koren *et al.* [88] have shown that this approach suffers from four significant problems as follows:

- Case I: Trap situations due to local minima

  A trap situation may occur when the robot runs into a dead end (e.g., inside a U-shaped obstacle). Here, the resultant force is zero due to equal magnitudes of the attractive and repulsive forces acting opposite to each other. As a result, the robot experiences stagnant situation and the change of resultant force due to noise causes the robot to oscillate back and forth. This phenomenon is known as trap situation at local minima.

- Case II: No passage between closely spaced obstacles

  When the robot attempts to pass in-between two closely spaced obstacles, the sum of all repulsive forces from obstacles may act opposite to the attractive force. This also could result local minima and the robot may be trapped.

- Case III: Oscillations in presence of obstacles

  Abrupt changes in the shape of obstacles may result in inconsistent directions of repulsive forces and this may lead to oscillatory response.

- Case IV: Oscillations in narrow passages

  When the robot travels in a narrow corridor, the net repulsive force acting on the robot in the transverse direction of travel may change its sign. This switching produces an oscillation in robot motion.

Koren *et al.* [88] employed separate modules to detect and overcome these trap situations. A case-based approach was also proposed by Ram *et al.* [47], where the environment is divided into a set of special cases and the force vectors were weighed according to a particular case to overcome the trap situations. However, the number of cases increases indefinitely with the environment complexity. Moreover, the cases

were detected using hard boundaries (thresholds) of different parameters, e.g., number of obstacles, distances to obstacles, distance to the target, resulting in erroneous case selection for noisy sensory perceptions.

FL provides better means to cope with noisy perceptions using soft boundaries of parameters [56]. Saffiotti *et al.* [48] proposed the concept of context dependent blending of fuzzy behaviors, where each behavior is implemented by a set of fuzzy rules. An additional set of centralized fuzzy rules, called meta rules (or context rules), are used to control the activity of individual fuzzy behaviors depending on the current sensory data and the overall goal of the robot. Saffiotti *et al.* claims that a complex controller is required in order to implement context dependent blending of behaviors and fuzzy meta-rule is recommended as the most promising candidate as it can reason out under partial and approximate knowledge of the environment and noisy sensing. Bonarini [51] extends the same concept of context dependent blending to fuzzy behavior management. Tunstel *et al.* [82] also proposed a similar concept of fuzzy behavior modulation where a set of fuzzy rules is used to weigh the output fuzzy sets of each behavior to control the behavioral activity. Abreu [50] proposed a variant of [48] and [82] where each behavior generates an action as well as its expected weight using separate fuzzy modules. Abreu [50] claims that the use of separate fuzzy modules reduces the design complexity by eliminating the requirement of centralized context rules. Vadakkepat *et al.* [52] combines the approaches described in [48], [82], and [50] and presents an application on a team of three soccer robots.

However, none of the existing fuzzy methods, mentioned above, provides solutions to overcome the navigational problems observed in the schema-based approach. This work aims to contribute in extending the concept of fuzzy context-dependent blending to overcome the aforementioned problems of motor schema. Hence, a novel approach to form a set of fuzzy meta rules is proposed for each motor schema. The meta

rules are then used to scale (or modulate) each force vector to overcome occurrences of local minima. The proposed method differs with [48] on formation of fuzzy meta rules. It uses only metric information of the map to form the fuzzy meta rules whereas Saffiotti *et al.* [48] deploys topological descriptions of the objects to form the meta rules. Furthermore, formation of strategic and reactive schemas (as behaviors), and determination of cooperative activity of strategic schemas with high priority reactive schemas are novel attempts as compared to all aforementioned fuzzy approaches.

In the proposed method, first, a set of motor schemas are formed as strategic and reactive schemas for goal oriented safe navigation. The fuzzy meta rules are then used to coordinate the schemas for successful robot navigation to the target. The fuzzy meta rules reason out the context dependent activity of each schema as well as cooperative activity between reactive and strategic schemas. The rest of the work is organized as follows: Section 3.2 defines the behavior modulation problem. Section 3.3 describes the construction of fuzzy meta rules. Section 3.4 demonstrates the real-time navigation results using the proposed approach and compares its performance with the conventional motor schema based method. Section 3.5 discusses different aspects of the experiment as compared to the proposed behavior coordination technique and finally, Section 3.6 draws the conclusion.

3.2 Problem definition

The goal of the navigation task is to integrate global path planning with local motion planning so that it optimizes the total traveled distance as well as ensures safe navigation through obstacles. Fig.3.1 shows the overall navigation architecture. The *Global* module preprocesses the 2-D world map of a given environment to generate a safe path through modeled obstacles. Fig.3.2(a) shows the experimental navigation environment generated using laser range data [89]. The physical dimension of the

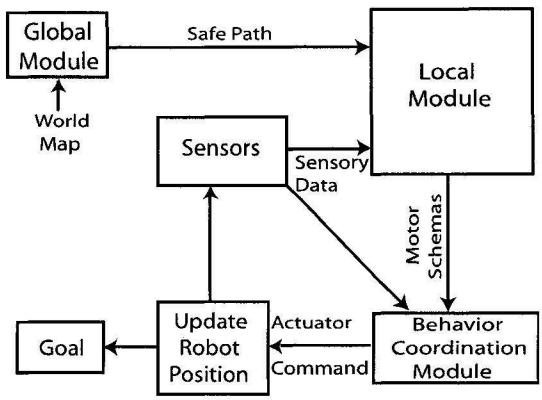


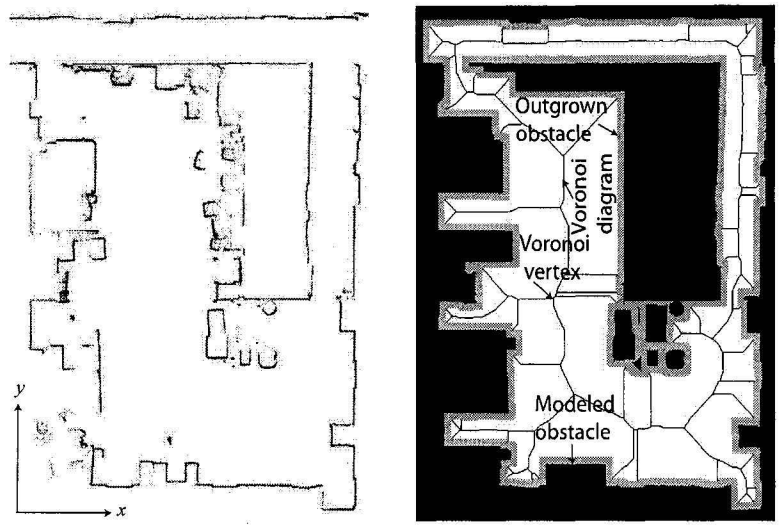
Figure 3.1: Overall navigation architecture

environment is 9 m $\times$ 12 m. In image plane, it is 450 pix $\times$ 600 pix, where each pixel is mapped to a physical dimension of 20 mm $\times$ 20 mm. For global path planning, the navigation is assumed in configuration space, i.e., obstacles are outgrown by an amount equal to the width of the robot (see Fig.3.2(b)). *Voronoi* diagram is used to trace a collision free path network. The *A-star* search algorithm [25] is then employed to identify an optimum navigational path between the initial and final desired positions of the robot [25]. The *Voronoi* vertices within the optimum path are considered as subgoals. The immediate subgoal (*Voronoi* vertex) in the map is updated at each time instance while discarding the past navigational points. The *Local* module, in Fig.3.1, creates four motor schemas (or force-based behaviors) using the safe path information and locally sensed obstacle data. The schemas are defined as follows.

- *Route-follow* ($\hat{\mathbf{V}}_{\mathcal{R}}$) schema is defined to follow the safe path. It is a unit vector directed to the nearest subgoal (towards the target) with respect to current robot's position (see Fig.3.3). Let (x_1, y_1) and (x_r, y_r) denote the coordinates of the nearest subgoal and robot, respectively. Then, ($\hat{\mathbf{V}}_{\mathcal{R}}$) is expressed as

$$\hat{\mathbf{V}}_{\mathcal{R}} = \frac{x_1 - x_r}{D(x_1, y_1)}\hat{x} + \frac{y_1 - y_r}{D(x_1, y_1)}\hat{y} \tag{3.1}$$

where $D(x_1, y_1)$ is the Euclidean distance between (x_1, y_1) and (x_r, y_r), and $\hat{x}$



(a) (b)

Figure 3.2: (a) Original map generated by laser range data, and (b) Processed map

and $\hat{y}$ are the unit vectors in x and y directions.

- *Go-to-target* ($\hat{\mathbf{V}}_{\mathcal{T}}$) schema is defined for path optimization. It is a unit vector directed to the second nearest subgoal towards the target with respect to the current robot's position (see Fig.3.3). Let (x_2, y_2) denotes the coordinate of the second nearest subgoal. Then, ($\hat{\mathbf{V}}_{\mathcal{T}}$) is calculated using the following equation

$$\hat{\mathbf{V}}_{\mathcal{T}} = \frac{x_2 - x_r}{D(x_2, y_2)}\hat{x} + \frac{y_2 - y_r}{D(x_2, y_2)}\hat{y} \tag{3.2}$$

where $D(x_2, y_2)$ is the Euclidean distance between (x_2, y_2) and (x_r, y_r).

- *Avoid-obstacle* ($\hat{\mathbf{V}}_{\mathcal{O}}$) schema is defined as a unit vector directed to the sum of repulsive forces generated by each obstacle point $(x_o, y_o) \in \mathcal{O}$, where $\mathcal{O}$ is the set of obstacle points detected in a decision cycle (or sampling period). Each repulsive force is inversely proportional to the square of the distance $D(x_o, y_o)$ between the obstacle point and the robot. Hence, the sum of the repulsive forces

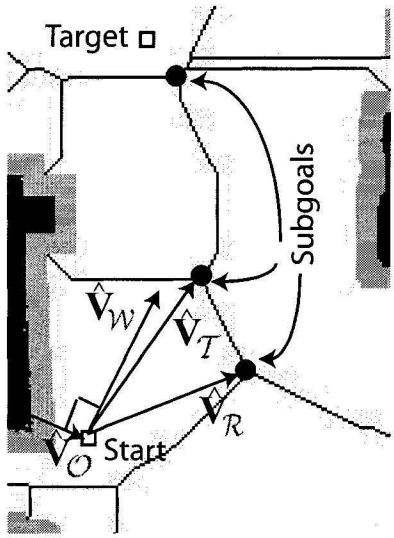


Figure 3.3: Motor schemas

$\mathbf{V}_{\mathcal{O}}$ is obtained using the following formula

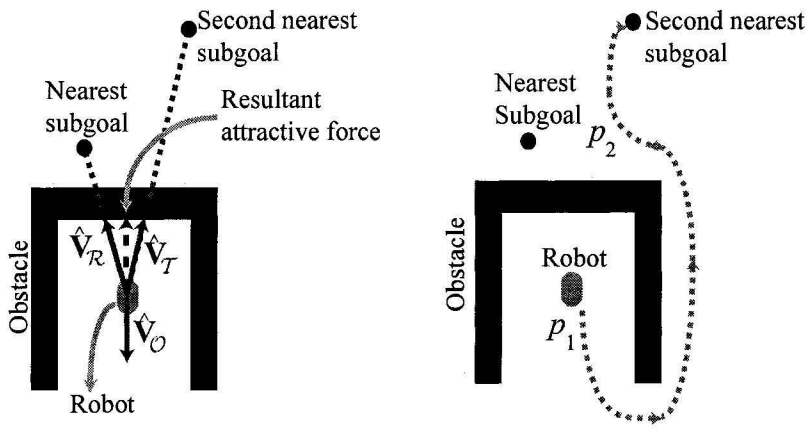
$$\mathbf{V}_{\mathcal{O}} = \sum_{(x_o,y_o)\in\mathcal{O}} \frac{1}{D^2(x_o,y_o)} \left(\frac{x_o - x_r}{D(x_o,y_o)}\hat{x} + \frac{y_o - y_r}{D(x_o,y_o)}\hat{y} \right) \tag{3.3}$$

and $\hat{\mathbf{V}}_{\mathcal{O}}$ is then defined as (see Fig.3.3)

$$\hat{\mathbf{V}}_{\mathcal{O}} = \frac{\mathbf{V}_{\mathcal{O}}}{|\mathbf{V}_{\mathcal{O}}|}. \tag{3.4}$$

- *Wall-follow* ($\hat{\mathbf{V}}_{\mathcal{W}}$) is defined in the direction normal to *avoid-obstacle*, i.e., $\hat{\mathbf{V}}_{\mathcal{W}} \perp \hat{\mathbf{V}}_{\mathcal{O}}$ and is directed towards the safe heading direction to the target (see Fig.3.3). Conventional motor schema based approaches use only attractive and repulsive forces for robot navigation, which leads to local minimum problems and creates oscillatory robot trajectories. Addition of *wall-follow* schema provides a better solution to this problem. As an instance, Fig.3.4(a) shows a local minimum problem in a U-shape obstacle, where the resultant attractive and repulsive forces are acting in opposite directions leading to a back and forth robot motion. Addition of *wall-follow* schema with a higher relative weight helps the

robot to escape from the U-shape obstacle and finally reaches the target (see Fig.3.4(b)).



(a) A trap situation

(b) Effect of the addition of *wall-follow* schema

Figure 3.4: Use of *avoid-obstacle* and *wall-follow* to escape from a U-shape obstacle

Following section will describe the online generation of relative weights for different schemas using the proposed fuzzy context dependent behavior modulation technique.

The *Route-follow* and *go-to-target* schemas are defined as strategic schemas. They incorporate the safe path information from the global map for providing motion command for navigation. The other two schemas, *avoid-obstacle* and *wall-follow* are defined as reactive schemas. They adapt for situations where the robot requires to make reactive decisions to avoid local minima, oscillations and collision-less navigation in case of dynamic obstacles during navigation.

The overall coordinated behavior $\mathbf{V}$ is generated by the *Behavior Coordination* module using the following formula

$$\mathbf{V} = \beta_{\mathcal{R}}\hat{\mathbf{V}}_{\mathcal{R}} + \beta_{\mathcal{T}}\hat{\mathbf{V}}_{\mathcal{T}} + \beta_{\mathcal{O}}\hat{\mathbf{V}}_{\mathcal{O}} + \beta_{\mathcal{W}}\hat{\mathbf{V}}_{\mathcal{W}} \tag{3.5}$$

where $\{\beta\}$ are the associated weights corresponding to individual schemas in (1). The objective of the *Behavior Coordination* module is to generate appropriate values for $\{\beta\}$ and a suitable adaptation mechanism using FL to satisfy each motor schema. The orientation of the coordinated behavior, $\theta = \angle\mathbf{V}$ is used to produce velocity commands for the robot. The navigation loop is continued until the goal is reached.

3.3 Context dependent behavior modulation of schemas

In this section, first, heuristic-based context dependent applicability of each schema is discussed to overcome the shortcomings of the conventional motor schema method (see Section 3.1). Next, the context dependent applicability of each schema is demonstrated to implement the proposed fuzzy context dependent behavior modulation mechanism.

3.3.1 Heuristic development for context dependent applicability

The context dependent applicability of different schemas are defined employing the heuristics developed using common experiences of a skilled human operator. While forming the heuristics, it is assumed that the reactive schemas are of higher priorities than the strategic schemas. The proposed heuristics are as follows:

- If obstacles are closely located in front of the robot, then increase the weight ($\beta_{\mathcal{O}}$) of *avoid-obstacle* schema.
- If obstacles are closely located on left or right side of the robot, then increase the weight ($\beta_{\mathcal{W}}$) of *wall-follow* schema.

- If *route-follow* and *go-to-target* is cooperative with *avoid-obstacle*, then increase the value of $\beta_{\mathcal{R}}$ and $\beta_{\mathcal{T}}$. (Two schemas are cooperative if the difference between their orientations is low).

- If the robot is at a safe distance from the obstacles and is away from the safe path, then increase the weight ($\beta_{\mathcal{R}}$) of *route-follow* schema. (The safe path is represented by the nearest subgoal)

- If the robot is at a safe distance from the obstacles and the target is near to the robot, then increase the weight ($\beta_{\mathcal{T}}$) of *go-to-target* schema. (The target is represented by the second nearest subgoal)

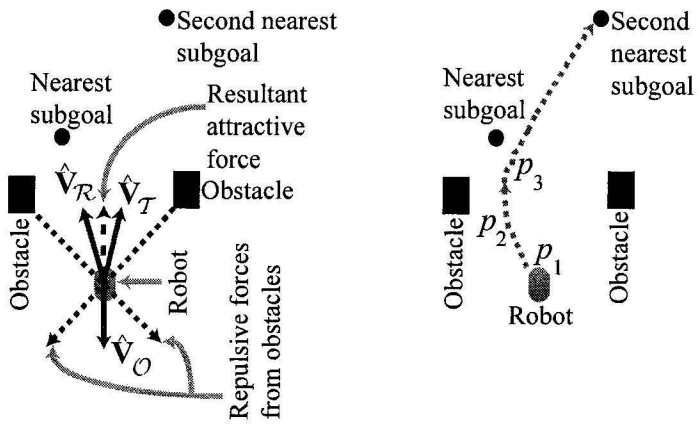
Note that the cooperativeness of strategic schemas is determined only with *avoid-obstacle* since it reduces wall-following tendency in presence of obstacles, when safe navigation is possible to the target. The following is an illustration of the proposed heuristics in removing the shortcomings of the conventional motor schema approach.

Case I: Trap situations due to local minima

Fig.3.4(a) shows an instance of a local minimum problem, where the robot is trapped at a local minimum in a U-shape obstacle. Using the aforementioned heuristics, the system increases weight $\beta_{\mathcal{O}}$ or $\beta_{\mathcal{W}}$, when the robot is inside the U-shape obstacle (at point p_1 in Fig.3.4(b)). When the robot stays outside the U-shape obstacle (at point p_2), the value of $\beta_{\mathcal{T}}$ is increased, which makes *go-to-target* cooperative with *avoid-obstacle* schema (i.e., both schemas point away from the obstacles). Moreover, when the distance to the second nearest subgoal is lower, it will infer higher values for $\beta_{\mathcal{T}}$. Consequently, the robot is expected to find a solution to avoid the local minima and reach the target safely (see 3.4(b)).

Case II: No passage between closely spaced obstacles

An instance of this situation is shown in Fig.3.5(a), where sum of the repulsive forces points to the opposite direction of the resultant attractive force. The same heuristics can be adopted to overcome this problem. First, the value of $\beta_{\mathcal{R}}$ is increased at point p_1 (in Fig.3.5(b)) since the robot is away from the obstacles and the nearest subgoal. At a close distance to the obstacles (at point p_2), $\beta_{\mathcal{W}}$ is assigned higher values than $\beta_{\mathcal{O}}$ since the obstacles are on left and right sides of the robot. This leads to successful navigation through the gap between the obstacles. Finally, when the robot reaches at point p_3, $\beta_{\mathcal{T}}$ is increased, which makes *go-to-target* cooperative with *avoid-obstacle* schema. Furthermore, lower values of the distance to the second nearest subgoal infer higher values of $\beta_{\mathcal{T}}$. As a result, the robot is expected to navigate between the obstacles and reach the target.



(a) (b)

Figure 3.5: (a) Local minima due to closely spaced obstacles, (b) Expected robot trajectory generated using the heuristics

Case III: Oscillations in presence of obstacles

The robot experiences oscillations in presence of obstacles when attractive and repulsive forces become stronger alternatively. Fig.3.6(a) demonstrates an example where the repulsive force suddenly increases in the presence of obstacles at point p_1. However, at point p_2, the robot starts moving toward the subgoals due to higher attractive forces. The alternative switching between attractive and repulsive forces causes oscillation. Application of the proposed heuristics can reduce the oscillation. At point p_1 in Fig.3.6(b), the robot is at a safe distance from the obstacles and away from the nearest subgoal, which leads to the increased value of $\beta_{\mathcal{R}}$. On the other hand, at point p_2, the value of $\beta_{\mathcal{O}}$ is increased since the obstacles are closely located in front of the robot. However, at point p_3, the value of $\beta_{\mathcal{W}}$ is increased as the obstacles are on the left side of the robot. Finally, at point p_4, *go-to-target* schema becomes cooperative with *avoid-obstacle* and the distance to the second nearest goal decreases, which leads to the increased values of $\beta_{\mathcal{T}}$. Consequently, the robot is expected to safely reach the target and reduce the frequent switching between attractive and repulsive forces.

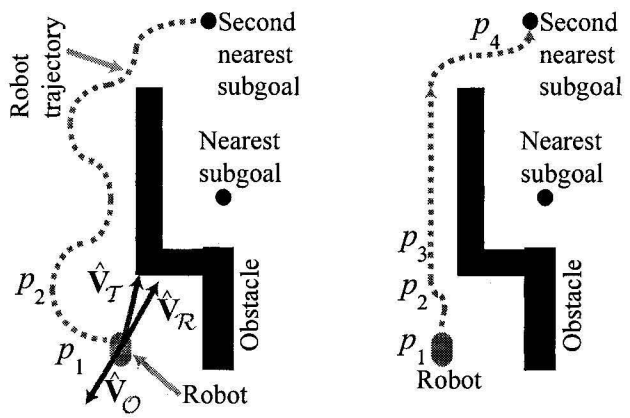


Figure 3.6: (a) Oscillation in the presence of obstacles, (b) Expected robot trajectory generated using the heuristics

Case IV: Oscillations in narrow passages

In a narrow passage, the robot experiences repulsive forces from opposite directions and a sudden change excites the robot into unstable oscillations (see Fig.3.7(a)). The proposed heuristics is able to remove the oscillation. At point p_1 in Fig.3.7(b), since the obstacles are closely located on left and right sides of the robot, the value of $\beta_{\mathcal{W}}$ is increased, which results in oscillation free navigation through the narrow passage. At point p_2, the value of $\beta_{\mathcal{T}}$ is increased as *go-to-target* schema becomes cooperative with *avoid-obstacle* and the distance to the second nearest goal decreases. As a result, the robot is expected to safely reache the target without oscillation.

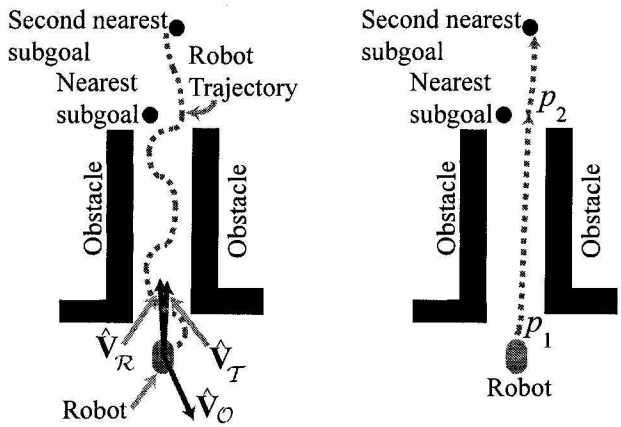


Figure 3.7: (a) Oscillation in narrow passages, (b) Expected robot trajectory generated using the heuristics

3.3.2 Fuzzy Context Dependent Blending of Schemas

The most challenging part of context-dependent blending of schemas is to implement the complex controller that generates the modulating factor using the knowledge of context dependent applicability of different schemas. With partial knowledge of the

environment and noisy sensory information, it is difficult to determine hard boundaries, when the distances to the subgoals and obstacles are low. Furthermore, an obstacle can be both in front of the robot as well as on left or right side with different degree of preferences. To cope with this subjective uncertainty, fuzzy rules are used to implement the context dependent blending of motor schemas employing soft boundaries and graded preferences of fuzzy sets.

For each motor schema, a max-min-centroid type Fuzzy Inference System (FIS) is employed to generate the weights. The i^{th} fuzzy rule of each FIS is formed as follows:

If $[P_i(Z)]$ Then $[Q_i(\beta)]$

where fuzzy predicates $P_i(\cdot)$ and $Q_i(\cdot)$ stand for input condition and output action, respectively. The set of sensory information used by each FIS is denoted as $Z = \{z_1, \ldots, z_J\}$ (J is the total number of sensory data). The input condition uses the sensory data to generate graded preference for the output action which reasons out the expected value of $\beta \in \{\beta_{\mathcal{R}}, \beta_{\mathcal{T}}, \beta_{\mathcal{O}}, \beta_{\mathcal{W}}\}$. Hence, $P_i(\cdot)$ and $Q_i(\cdot)$ are defined as

$$\begin{aligned} P_i(Z) &: (z_1 \text{ is } \mathcal{X}_{1n}) \text{ and } (z_2 \text{ is } \mathcal{X}_{2m}) \text{ and ... and} \\ &\quad (z_J \text{ is } \mathcal{X}_{Jl}) \\ Q_i(Z) &: \beta \text{ is } \mathcal{Y}_q \end{aligned}$$

where *and* denotes the min operation and $\mathcal{X}_{1n}, \mathcal{X}_{2m}, \ldots, \mathcal{X}_{Jl}$, and $\mathcal{Y}_q$ are the fuzzy sets defined over the expected ranges of $z_1, z_2, \ldots, z_J$, and β respectively. The fuzzy sets $\mathcal{X}_{1n}, \mathcal{X}_{2m}, \ldots, \mathcal{X}_{Jl}$, and $\mathcal{Y}_q$ are associated with the membership functions (MFs) $f_{1n}, f_{2m}, \ldots, f_{Jl}$, and f_q that map $z_1, z_2, \ldots, z_J$, and β to graded membership values $\mu \in [0, 1]$. The total numbers of MFs associated with $z_1, z_2, \ldots, z_J$ are $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_J$, i.e., $n = 1, \ldots, \mathcal{N}_1$, $m = 1, \ldots, \mathcal{N}_2$ and $l = 1, \ldots, \mathcal{N}_J$. Therefore, the total number fuzzy rule is $\mathcal{N}_1 \times \mathcal{N}_2 \times \ldots \times \mathcal{N}_J$. Higher the values of n, m, l, and q, higher the values of support of $f_{1n}, f_{2m}, \ldots, f_{Jl}$, and f_q. The index q is determined using $n, m, \ldots, l$.

The FIS uses $P_i(\cdot)$ and $Q(\cdot)$ to estimate β using (3.6).

$$\begin{aligned} f(\beta) &= \max_i \{\min\left(P_i(Z), Q_i(\beta)\right)\} \\ P_i(Z) &= \min\{f_{1n}(z_1), f_{2m}(z_2), \ldots, f_{Jl}(z_J)\} \\ \beta &= \frac{\int (\beta \times f(\beta))\, d\beta}{\int f(\beta) d\beta} \end{aligned} \tag{3.6}$$

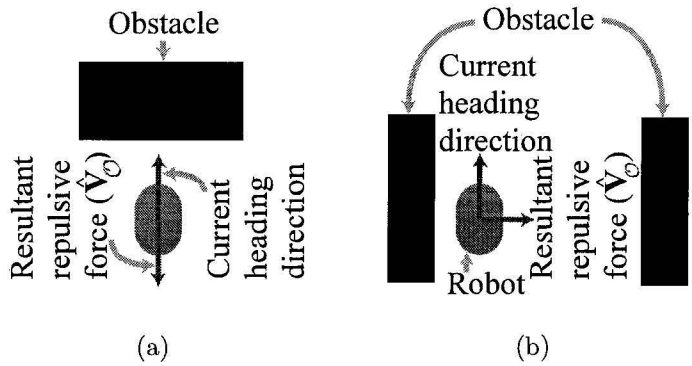
In the rest of this section, formation of fuzzy rules of different FISs will be illustrated. The fuzzy rules are constructed using the knowledge of the proposed heuristics in Section 3.3.1.

The FIS $FS_{\mathcal{O}}$

The FIS for *avoid-obstacle* is denoted as $FS_{\mathcal{O}}$, which takes two sensory inputs (i.e., $Z^{\mathcal{O}} = \{z_1^{\mathcal{O}}, z_2^{\mathcal{O}}\}$, and $J^{\mathcal{O}} = 2$) and generates the modulating weight $\beta_{\mathcal{O}}$ as the output. The sensory input $z_1^{\mathcal{O}} = \min_{(x_o, y_o) \in \mathcal{O}} \{D(x_o, y_o)\}$ is the closest distance to the obstacles, and $z_2^{\mathcal{O}} = \text{abs}(\angle \hat{\mathbf{V}}_{\mathcal{O}} - \tilde{\theta})$ is the absolute angle difference between the angle expected by *avoid-obstacle* ($\hat{\mathbf{V}}_{\mathcal{O}}$) and the current robot orientation $\tilde{\theta}$. Lower values of $z_1^{\mathcal{O}}$ should produces higher values of $\beta_{\mathcal{O}}$, which ensures higher importance of *avoid-obstacle* when obstacles are closely located to the robot. This implies that subscripts q and n of fuzzy sets $\mathcal{Y}_q^{\mathcal{O}}$ and $\mathcal{X}_n^{\mathcal{O}}$ should have monotonically decreasing relationship. The sensory data $z_2^{\mathcal{O}}$ infers whether the obstacles are in front of robot or on left or right side. In case of obstacles present in front of the robot, the value of $z_2^{\mathcal{O}}$ is maximum (180^o), i.e., the repulsive force is acting to the opposite direction of the current robot heading (see Fig.3.8(a)). Therefore, higher values of $z_2^{\mathcal{O}}$ infer higher values of $\beta_{\mathcal{O}}$. This implies that the subscripts q and m of fuzzy sets $\mathcal{Y}_q^{\mathcal{O}}$ and $\mathcal{X}_{2m}^{\mathcal{O}}$ should have monotonically increasing relationship. Hence, the i^{th} fuzzy rule of $FS_{\mathcal{O}}$ is defined as

$$\begin{aligned} &\text{IF } \left[(z_1^{\mathcal{O}} \text{ is } \mathcal{X}_{1n}^{\mathcal{O}}) \text{ and } (z_2^{\mathcal{O}} \text{ is } \mathcal{X}_{2m}^{\mathcal{O}})\right] \\ &\text{THEN } \left[\beta_{\mathcal{O}} \text{ is } \mathcal{Y}_q^{\mathcal{O}}\right] \end{aligned} \tag{3.7}$$

where $q = [\mathcal{N}_1^{\mathcal{O}} - (n-1)] + (m-1)$ and $i = \mathcal{N}_2^{\mathcal{O}}(n-1) + m$. The total number of rules of $FS_{\mathcal{O}}$ is $\mathcal{N}_1^{\mathcal{O}} \times \mathcal{N}_2^{\mathcal{O}}$. The weight $\beta_{\mathcal{O}}$ is calculated using (3.6).



(a) (b)

Figure 3.8: (a) Condition for maximum $\beta_{\mathcal{O}}$, (b) Condition for maximum $\beta_{\mathcal{W}}$

The FIS $FS_{\mathcal{W}}$

The FIS for *wall-follow* is denoted as $FS_{\mathcal{W}}$. It takes the same sensory inputs as *avoid-obstacle*, i.e., $Z^{\mathcal{W}} = \{z_1^{\mathcal{W}}, z_2^{\mathcal{W}}\}$, $z_1^{\mathcal{W}} = z_1^{\mathcal{O}}$, $z_2^{\mathcal{W}} = z_2^{\mathcal{O}}$, and $J^{\mathcal{W}} = J^{\mathcal{O}}$. It generates the modulating weight $\beta_{\mathcal{W}}$ as the output. Hence, lower values of $z_1^{\mathcal{W}}$ infer higher values of $\beta_{\mathcal{W}}$, which implies that the subscripts q and n of fuzzy sets $\mathcal{Y}_q^{\mathcal{W}}$ and $\mathcal{X}_n^{\mathcal{W}}$ should have monotonically decreasing relationship. Input $z_2^{\mathcal{W}} = 90^o$ indicates that obstacles are present either on left or right side of the robot (see Fig.3.8(b)). Therefore, between 0^o and 90^o, the subscripts q and m of fuzzy sets $\mathcal{Y}_q^{\mathcal{W}}$ and $\mathcal{X}_{2m}^{\mathcal{W}}$ should have monotonically increasing relationship, whereas between 90^o and 180^o, q and m should have monotonically decreasing relationship. Therefore, the i^{th} rule of $FS_{\mathcal{W}}$ is formed as follows.

$$\begin{aligned} &\text{IF } [(z_1^{\mathcal{W}} \text{ is } \mathcal{X}_{1n}^{\mathcal{W}}) \text{ and } (z_2^{\mathcal{W}} \text{ is } \mathcal{X}_{2m}^{\mathcal{W}})] \\ &\text{THEN } [\beta_{\mathcal{W}} \text{ is } \mathcal{Y}_q^{\mathcal{W}}] \end{aligned} \tag{3.8}$$

Here, q is defined as

$$q = \begin{cases} [\mathcal{N}_1^{\mathcal{W}} - (n-1)] + 2(m-1) & \text{if } m \leq \frac{\mathcal{N}_2^{\mathcal{W}}}{2} \\ [\mathcal{N}_1^{\mathcal{W}} - (n-1)] + 2(\mathcal{N}_2^{\mathcal{W}} - m) & \text{if } m > \frac{\mathcal{N}_2^{\mathcal{W}}}{2} \end{cases}$$

and $i = \mathcal{N}_2^{\mathcal{W}}(n-1) + m$. It is assumed that $f_{2m}^{\mathcal{W}}$ is centered on 90^o for $m = \frac{\mathcal{N}_2^{\mathcal{W}}}{2}$. The total number of rules of $FS_{\mathcal{W}}$ is $\mathcal{N}_1^{\mathcal{W}} \times \mathcal{N}_2^{\mathcal{W}}$. The weight $\beta_{\mathcal{W}}$ is estimated using (3.6).

The FIS $FS_{\mathcal{R}}$

The FIS for *route-follow* is denoted as $FS_{\mathcal{R}}$. It takes three sensory inputs, i.e., $Z^{\mathcal{R}} = \{z_1^{\mathcal{R}}, z_2^{\mathcal{R}}, z_3^{\mathcal{R}}\}$, and $J^{\mathcal{R}} = 3$. The sensory input $z_1^{\mathcal{R}} = D(x_1, y_1)$ is the distance to the nearest subgoal, $z_2^{\mathcal{R}} = z_1^{\mathcal{O}}$ is the distance to the closest obstacle, and $z_2^{\mathcal{R}} = \text{abs}(\angle\hat{\mathbf{V}}_{\mathcal{R}} - \angle\hat{\mathbf{V}}_{\mathcal{O}})$ is the absolute angle difference between the angle expected by *route-follow* ($\hat{\mathbf{V}}_{\mathcal{R}}$) and the angle expected by *avoid-obstacle* ($\hat{\mathbf{V}}_{\mathcal{O}}$). Here, higher values of $z_1^{\mathcal{R}}$ and $z_2^{\mathcal{R}}$ infer higher values of $\beta_{\mathcal{R}}$, which implies that the subscripts n and m of fuzzy sets $\mathcal{X}_n^{\mathcal{R}}$ and $\mathcal{X}_{2m}^{\mathcal{R}}$ should have monotonically increasing relationship with the subscript q of fuzzy set $\mathcal{Y}_q^{\mathcal{R}}$. On the other hand, higher values of $z_3^{\mathcal{R}}$ indicates lower cooperativeness with *avoid-obstacle*, which in turn infers lower values of $\beta_{\mathcal{R}}$. Consequently, the subscript l of fuzzy set $\mathcal{X}_{3m}^{\mathcal{R}}$ should have monotonically decreasing relationship with q. Hence, the i^{th} fuzzy rule of $FS_{\mathcal{R}}$ is formed as

$$\begin{aligned} &\text{IF } [(z_1^{\mathcal{R}} \text{ is } \mathcal{X}_{1n}^{\mathcal{R}}) \text{ and } (z_2^{\mathcal{R}} \text{ is } \mathcal{X}_{2m}^{\mathcal{R}}) \text{ and } (z_3^{\mathcal{R}} \text{ is } \mathcal{X}_{3l}^{\mathcal{R}})] \\ &\text{THEN } [\beta_{\mathcal{R}} \text{ is } \mathcal{Y}_q^{\mathcal{R}}] \end{aligned} \tag{3.9}$$

where $q = (n-1) + (m-1) + [\mathcal{N}_3^{\mathcal{R}} - (l-1)]$ and $i = \mathcal{N}_2^{\mathcal{R}}\mathcal{N}_3^{\mathcal{R}}(n-1) + \mathcal{N}_3^{\mathcal{R}}(m-1) + l$. The total number of rules in $FS_{\mathcal{R}}$ is $\mathcal{N}_1^{\mathcal{R}} \times \mathcal{N}_2^{\mathcal{R}} \times \mathcal{N}_3^{\mathcal{R}}$. The weight $\beta_{\mathcal{R}}$ is estimated using (3.6).

The FIS $FS_{\mathcal{T}}$

The FIS for *go-to-target* is denoted as $FS_{\mathcal{T}}$. It takes three sensory inputs, i.e., $Z^{\mathcal{T}} = \{z_1^{\mathcal{T}}, z_2^{\mathcal{T}}, z_3^{\mathcal{T}}\}$ and $J^{\mathcal{T}} = 3$. The sensory input $z_1^{\mathcal{T}} = D(x_2, y_2)$ is the distance to the second nearest subgoal, $z_2^{\mathcal{T}} = z_1^{\mathcal{O}}$ is the distance to the closest obstacle, and $z_3^{\mathcal{T}} = \text{abs}(\angle\hat{\mathbf{V}}_{\mathcal{T}} - \angle\hat{\mathbf{V}}_{\mathcal{O}})$ is the absolute angle difference between the angle expected by *go-to-target* ($\hat{\mathbf{V}}_{\mathcal{T}}$) and the angle expected by *avoid-obstacle* ($\hat{\mathbf{V}}_{\mathcal{O}}$). Here, higher values of $z_2^{\mathcal{T}}$ infers higher values of $\beta_{\mathcal{T}}$ which implies a monotonically increasing relationship between the subscripts m and q of fuzzy sets $\mathcal{X}_{2m}^{\mathcal{T}}$ and $\mathcal{Y}_q^{\mathcal{T}}$. On the other hand, lower values of $z_1^{\mathcal{T}}$ and $z_3^{\mathcal{T}}$ infer higher values of $\beta_{\mathcal{T}}$. This implies that the subscripts n and l of fuzzy sets $\mathcal{X}_n^{\mathcal{T}}$ and $\mathcal{X}_{3m}^{\mathcal{T}}$ should have a monotonically decreasing relationship with q. Hence, the i^{th} fuzzy rule of $FS_{\mathcal{T}}$ is defined as

$$\begin{aligned} &\text{IF } \left[(z_1^{\mathcal{T}} \text{ is } \mathcal{X}_{1n}^{\mathcal{T}}) \text{ and } (z_2^{\mathcal{T}} \text{ is } \mathcal{X}_{2m}^{\mathcal{T}}) \text{ and } (z_3^{\mathcal{T}} \text{ is } \mathcal{X}_{3l}^{\mathcal{T}})\right] \\ &\text{THEN } \left[\beta_{\mathcal{T}} \text{ is } \mathcal{Y}_q^{\mathcal{T}}\right]. \end{aligned} \tag{3.10}$$

Here q is defined as $q = (\mathcal{N}_1^{\mathcal{T}} - n) + m + (\mathcal{N}_3^{\mathcal{T}} - l)$ and $i = \mathcal{N}_2^{\mathcal{T}}\mathcal{N}_3^{\mathcal{T}}(n-1) + \mathcal{N}_3^{\mathcal{T}}(m-1) + l$. The total number of rules in $FS_{\mathcal{T}}$ is $\mathcal{N}_1^{\mathcal{T}} \times \mathcal{N}_2^{\mathcal{T}} \times \mathcal{N}_3^{\mathcal{T}}$. The weight $\beta_{\mathcal{T}}$ is estimated using (3.6).

3.3.3 Behavior coordination module

Fig.3.9 demonstrates the *Behavior Coordination* module of Fig.3.1 using fuzzy context-dependent blending of motor schema. Using locally sensed information, the motor schemas are modulated by the weights generated by the FISs. The summation of the modulated schemas denotes the coordinated behavior. Orientation θ of the coordinated schema (i.e., $\theta = \angle\mathbf{V}$) is used to produce the velocity commands for the robobt.

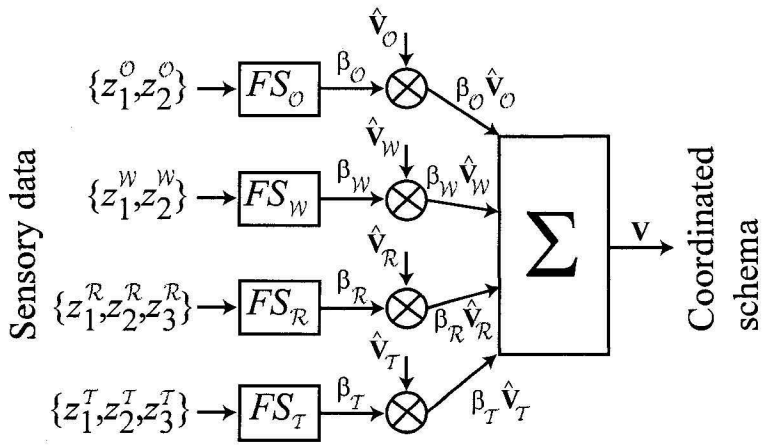


Figure 3.9: Behavior coordination module

3.4 Experiments

3.4.1 Defining membership functions

The experiments presented in this work use $\mathcal{N}_1^{\mathcal{O}} = \mathcal{N}_1^{\mathcal{W}} = \mathcal{N}_1^{\mathcal{R}} = \mathcal{N}_1^{\mathcal{T}} = 3$, $\mathcal{N}_2^{\mathcal{O}} = \mathcal{N}_2^{\mathcal{W}} = \mathcal{N}_2^{\mathcal{R}} = \mathcal{N}_2^{\mathcal{T}} = 3$, and $\mathcal{N}_3^{\mathcal{R}} = \mathcal{N}_3^{\mathcal{T}} = 3$. This results in 9 fuzzy rules for $FS_{\mathcal{O}}$ and $FS_{\mathcal{W}}$ using (3.7) and (3.8), and 27 fuzzy rules for $FS_{\mathcal{R}}$ and $FS_{\mathcal{T}}$ using (3.9) and (3.10). Fig.3.10(a) and 3.10(b) show the MFs used for the sensory input $Z^{\mathcal{O}}$, which is same as $Z^{\mathcal{W}}$. Fig.3.10(c) and 3.10(d) show the MFs used for the sensory inputs $z_1^{\mathcal{R}}$ and $z_3^{\mathcal{R}}$ in $FS_{\mathcal{R}}$. The MFs of $z_2^{\mathcal{R}}$ are same as of $z_1^{\mathcal{O}}$. For the sensory inputs $z_2^{\mathcal{T}}$ and $z_3^{\mathcal{T}}$, the MFs are similar to those of $z_2^{\mathcal{R}}$ and $z_3^{\mathcal{R}}$. Fig.3.10(e) shows the MFs used for $z_1^{\mathcal{T}}$. The ranges of all MFs are defined experimentally. For the given values of $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$, the number of MFs for $\beta_{\mathcal{O}}$ and $\beta_{\mathcal{W}}$ is 5 whereas it is 7 for $\beta_{\mathcal{R}}$ and $\beta_{\mathcal{T}}$. Fig.3.11(a) and 3.11(b) show the MFs for the modulating weight $\beta_{\mathcal{O}}$ and $\beta_{\mathcal{R}}$, respectively. The MFs for $\beta_{\mathcal{W}}$ is same as shown in Fig.3.11(a) and the MFs for $\beta_{\mathcal{T}}$ is same as shown in Fig.3.11(b). Table 3.1 shows the fuzzy rules of $FS_{\mathcal{O}}$ and $FS_{\mathcal{W}}$. Table 3.2 describes the fuzzy rules of $FS_{\mathcal{R}}$ and $FS_{\mathcal{T}}$.

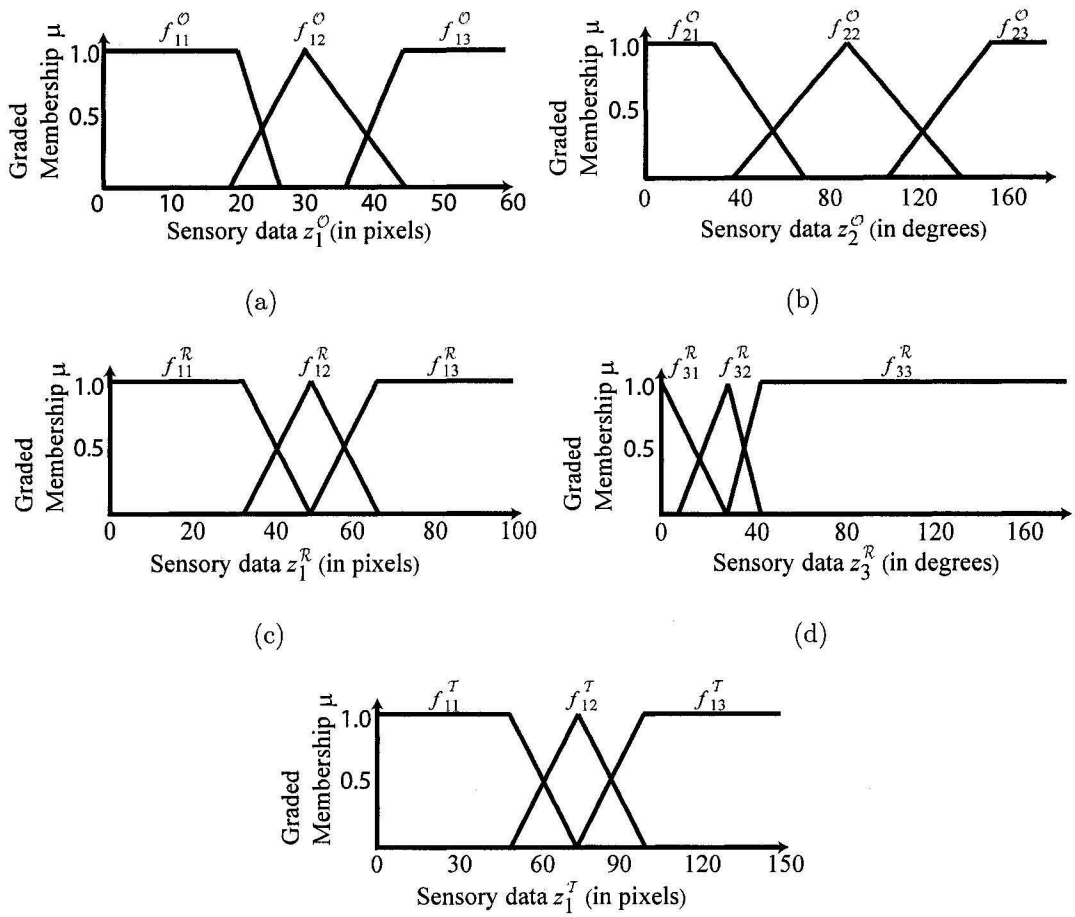


Figure 3.10: Membership functions for the sensory data: (a) $z_1^{\mathcal{O}}$, (b) $z_2^{\mathcal{O}}$, (c) $z_1^{\mathcal{R}}$, (d) $z_3^{\mathcal{R}}$, and (e) $z_1^{\mathcal{T}}$

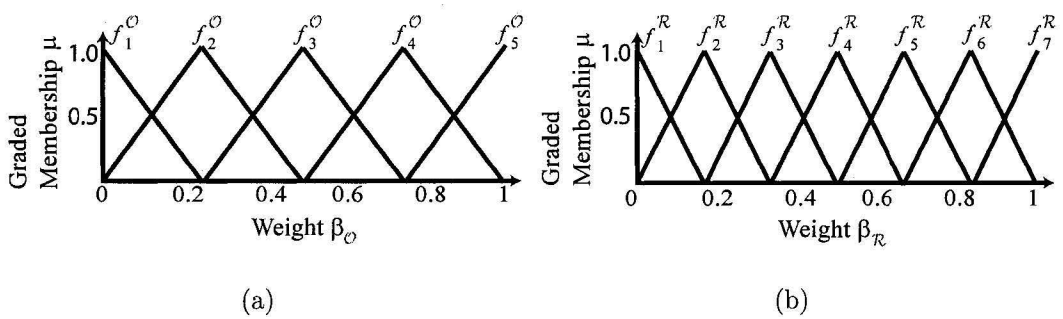


Figure 3.11: Membership functions for (a) $\beta_{\mathcal{O}}$, and (b) $\beta_{\mathcal{R}}$

$z_1^{\mathcal{O}}$ \ $z_2^{\mathcal{O}}$	$\mathcal{X}_{21}^{\mathcal{O}}$	$\mathcal{X}_{22}^{\mathcal{O}}$	$\mathcal{X}_{23}^{\mathcal{O}}$
$\mathcal{X}_{11}^{\mathcal{O}}$	$\mathcal{Y}_3^{\mathcal{O}}$	$\mathcal{Y}_4^{\mathcal{O}}$	$\mathcal{Y}_5^{\mathcal{O}}$
$\mathcal{X}_{12}^{\mathcal{O}}$	$\mathcal{Y}_2^{\mathcal{O}}$	$\mathcal{Y}_3^{\mathcal{O}}$	$\mathcal{Y}_4^{\mathcal{O}}$
$\mathcal{X}_{13}^{\mathcal{O}}$	$\mathcal{Y}_1^{\mathcal{O}}$	$\mathcal{Y}_2^{\mathcal{O}}$	$\mathcal{Y}_3^{\mathcal{O}}$

(a) Rules generated with (3.7)

$z_1^{\mathcal{W}}$ \ $z_2^{\mathcal{W}}$	$\mathcal{X}_{21}^{\mathcal{W}}$	$\mathcal{X}_{22}^{\mathcal{W}}$	$\mathcal{X}_{23}^{\mathcal{W}}$
$\mathcal{X}_{11}^{\mathcal{W}}$	$\mathcal{Y}_3^{\mathcal{W}}$	$\mathcal{Y}_5^{\mathcal{W}}$	$\mathcal{Y}_3^{\mathcal{W}}$
$\mathcal{X}_{12}^{\mathcal{W}}$	$\mathcal{Y}_2^{\mathcal{W}}$	$\mathcal{Y}_4^{\mathcal{W}}$	$\mathcal{Y}_2^{\mathcal{W}}$
$\mathcal{X}_{13}^{\mathcal{W}}$	$\mathcal{Y}_1^{\mathcal{W}}$	$\mathcal{Y}_3^{\mathcal{W}}$	$\mathcal{Y}_1^{\mathcal{W}}$

(b) Rules generated with (3.8)

Table 3.1: Rule-base for (a) $\beta_{\mathcal{O}}$ in $FS_{\mathcal{O}}$, and (b) $\beta_{\mathcal{W}}$ in $FS_{\mathcal{W}}$

$z_2^{\mathcal{R}}, z_3^{\mathcal{R}}$ \ $z_1^{\mathcal{R}}$	$\mathcal{X}_{11}^{\mathcal{R}}$	$\mathcal{X}_{12}^{\mathcal{R}}$	$\mathcal{X}_{13}^{\mathcal{R}}$
$\mathcal{X}_{21}^{\mathcal{R}}, \mathcal{X}_{31}^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$
$\mathcal{X}_{21}^{\mathcal{R}}, \mathcal{X}_{32}^{\mathcal{R}}$	$\mathcal{Y}_2^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$
$\mathcal{X}_{21}^{\mathcal{R}}, \mathcal{X}_{33}^{\mathcal{R}}$	$\mathcal{Y}_1^{\mathcal{R}}$	$\mathcal{Y}_2^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$
$\mathcal{X}_{22}^{\mathcal{R}}, \mathcal{X}_{31}^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$	$\mathcal{Y}_6^{\mathcal{R}}$
$\mathcal{X}_{22}^{\mathcal{R}}, \mathcal{X}_{32}^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$
$\mathcal{X}_{22}^{\mathcal{R}}, \mathcal{X}_{33}^{\mathcal{R}}$	$\mathcal{Y}_2^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$
$\mathcal{X}_{23}^{\mathcal{R}}, \mathcal{X}_{31}^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$	$\mathcal{Y}_6^{\mathcal{R}}$	$\mathcal{Y}_7^{\mathcal{R}}$
$\mathcal{X}_{23}^{\mathcal{R}}, \mathcal{X}_{32}^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$	$\mathcal{Y}_6^{\mathcal{R}}$
$\mathcal{X}_{23}^{\mathcal{R}}, \mathcal{X}_{33}^{\mathcal{R}}$	$\mathcal{Y}_3^{\mathcal{R}}$	$\mathcal{Y}_4^{\mathcal{R}}$	$\mathcal{Y}_5^{\mathcal{R}}$

(a) Rules generated with (3.9)

$z_2^{\mathcal{T}}, z_3^{\mathcal{T}}$ \ $z_1^{\mathcal{T}}$	$\mathcal{X}_{11}^{\mathcal{T}}$	$\mathcal{X}_{12}^{\mathcal{T}}$	$\mathcal{X}_{13}^{\mathcal{T}}$
$\mathcal{X}_{21}^{\mathcal{T}}, \mathcal{X}_{31}^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$
$\mathcal{X}_{21}^{\mathcal{T}}, \mathcal{X}_{32}^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$	$\mathcal{Y}_2^{\mathcal{T}}$
$\mathcal{X}_{21}^{\mathcal{T}}, \mathcal{X}_{33}^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$	$\mathcal{Y}_2^{\mathcal{T}}$	$\mathcal{Y}_1^{\mathcal{T}}$
$\mathcal{X}_{22}^{\mathcal{T}}, \mathcal{X}_{31}^{\mathcal{T}}$	$\mathcal{Y}_6^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$
$\mathcal{X}_{22}^{\mathcal{T}}, \mathcal{X}_{32}^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$
$\mathcal{X}_{22}^{\mathcal{T}}, \mathcal{X}_{33}^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$	$\mathcal{Y}_2^{\mathcal{T}}$
$\mathcal{X}_{23}^{\mathcal{T}}, \mathcal{X}_{31}^{\mathcal{T}}$	$\mathcal{Y}_7^{\mathcal{T}}$	$\mathcal{Y}_6^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$
$\mathcal{X}_{23}^{\mathcal{T}}, \mathcal{X}_{32}^{\mathcal{T}}$	$\mathcal{Y}_6^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$
$\mathcal{X}_{23}^{\mathcal{T}}, \mathcal{X}_{33}^{\mathcal{T}}$	$\mathcal{Y}_5^{\mathcal{T}}$	$\mathcal{Y}_4^{\mathcal{T}}$	$\mathcal{Y}_3^{\mathcal{T}}$

(b) Rules generated with (3.10)

Table 3.2: Rule-base for (a) $\beta_{\mathcal{R}}$ in $FS_{\mathcal{R}}$, and (b) $\beta_{\mathcal{T}}$ in $FS_{\mathcal{T}}$

3.4.2 Methods of comparison

Three methods will be employed to compare the navigation results:

- **Method 1** is the conventional motor schema that uses attractive forces representing the targets and repulsive forces representing the obstacles. Each force is inversely proportional to square of the distance to the target or an obstacle from the robot's position. Hence, the weights are defined as: $\beta_{\mathcal{O}} = |\mathbf{V}_{\mathcal{O}}|$, $\beta_{\mathcal{W}} = 0$, $\beta_{\mathcal{R}} = \frac{1}{D^2(x_1,y_1)}$, and $\beta_{\mathcal{T}} = \frac{1}{D^2(x_2,y_2)}$.

- **Method 2** is the unmodulated vector summation where all modulating weights are set to 1, i.e., $\beta_{\mathcal{O}} = \beta_{\mathcal{W}} = \beta_{\mathcal{R}} = \beta_{\mathcal{T}} = 1$.

- **Method 3** is the proposed fuzzy context dependent modulation of motor schemas.

Experiments are performed using a Pioneer 3-AT mobile robot [90] equipped with sonar sensors. The range measurements obtained from the sonar sensors are employed for obstacle avoidance. At each decision cycle, the robot is controlled by sending a rotational velocity command (Ω) and a translational velocity command (v). The rotational velocity command Ω is proportional to θ, i.e., $\Omega = \kappa\theta$ deg/s, where $\theta = \angle\mathbf{V}$ deg. κ is set to 1 for the examples presented in this work. The translational velocity command v is adjusted proportional to the rotational velocity $(180 - \text{abs}(\Omega))$ and is measured in mm/s. For unmodulated coordination, the sampling time period (i.e., the length of a decision cycle) is set at $t_p = 50$ ms.

Using the navigation environment shown in Fig.3.2, four different scenarios have been created.

Case I: A U-shape obstacle is placed on the way to the target (Fig.3.12(a)).

Case II: Two closely spaced obstacles are placed to obstruct the way to the target (Fig.3.13(a)).

Case III: The obstacles are placed in a way so that the robot experiences a sudden change in the obstacles' shape (Fig.3.14(a)).

Case IV: A narrow passage is created on the way to the target (Fig.3.15(a)).

For the navigation examples, only odometry is used for localization of the robot. In each experiment the robot path is traced and the results are shown in Fig.3.12-3.16.

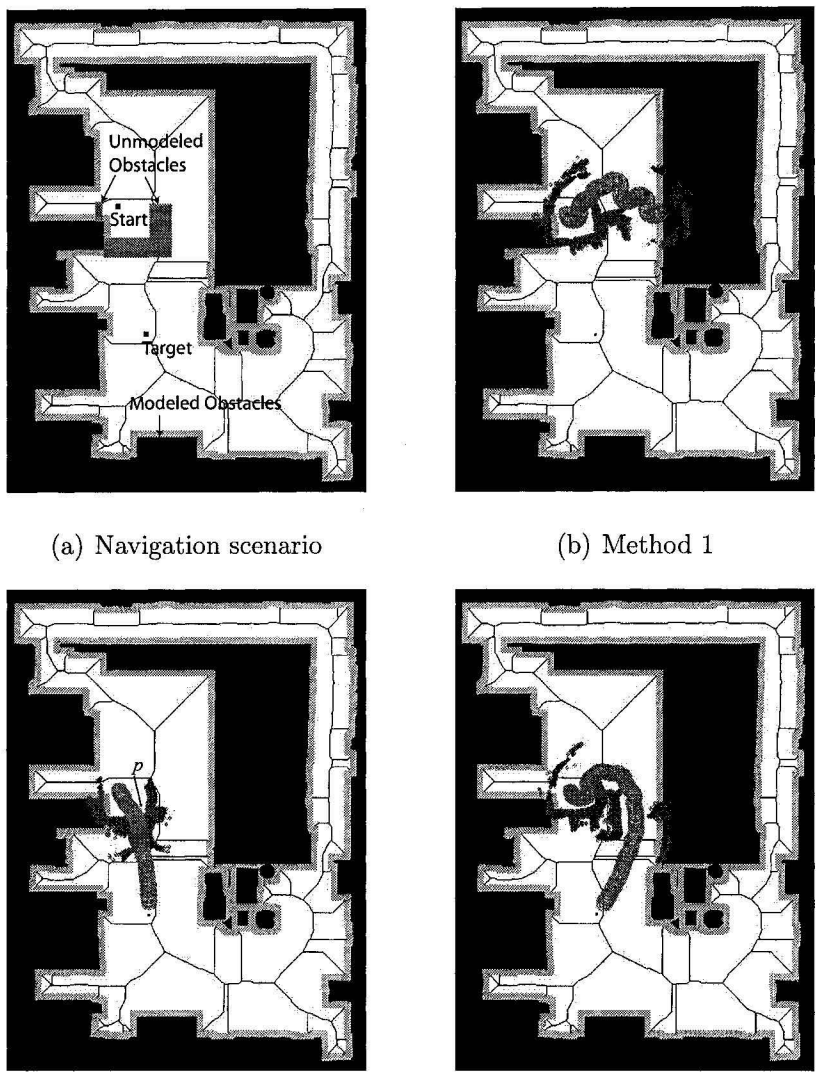
For the four cases the video clips corresponding to robot navigation under the conventional motor schema method (CaseIschema.mov, CaseIIschema.mov, CaseIIIschema.mov, and CaseIIIschema.mov) and the proposed fuzzy context-dependent blending of schemas (CaseIfuzzy.mov, CaseIIfuzzy.mov, CaseIIIfuzzy.mov, and CaseIIIfuzzy.mov) are provided on a CD labelled as 'Multimedia'.

3.4.3 Navigation results

The performance analysis reveals that fuzzy context dependent blending of motor schemas provides the most superior performance in all of the four cases and in particular the system shows robustness against dynamic changes in the environment.

Method 1

Both in Cases I and II, the robot is trapped in a local minimum and fails to reach the target using method 1. In Cases III and IV, the robot exhibits oscillatory trajectories for irregular obstacle-shape and narrow passage, respectively. Since this method uses only attractive and repulsive forces as motor schemas, it causes unstable velocity commands. As a result, the robot undergoes nonsmooth trajectory, higher traveled distance, and higher navigation time as compared to method 3.



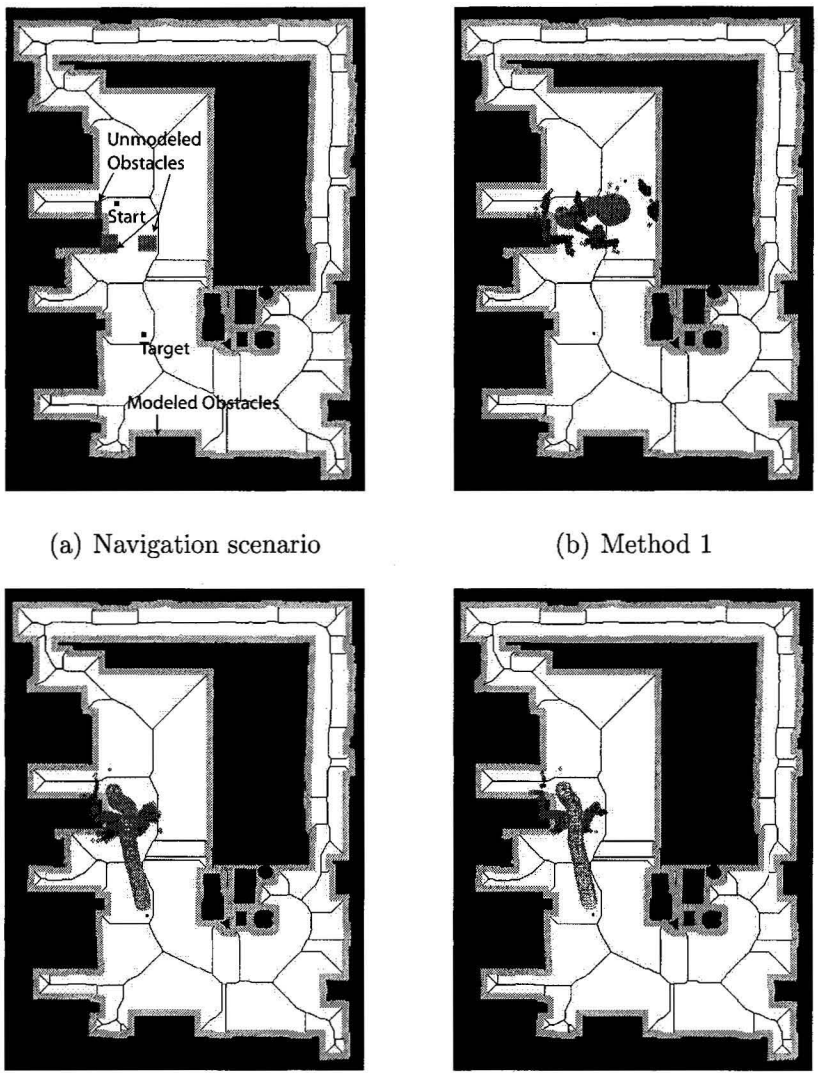
(a) Navigation scenario

(b) Method 1

(c) Method 2

(d) Method 3

Figure 3.12: Navigation results in Case I



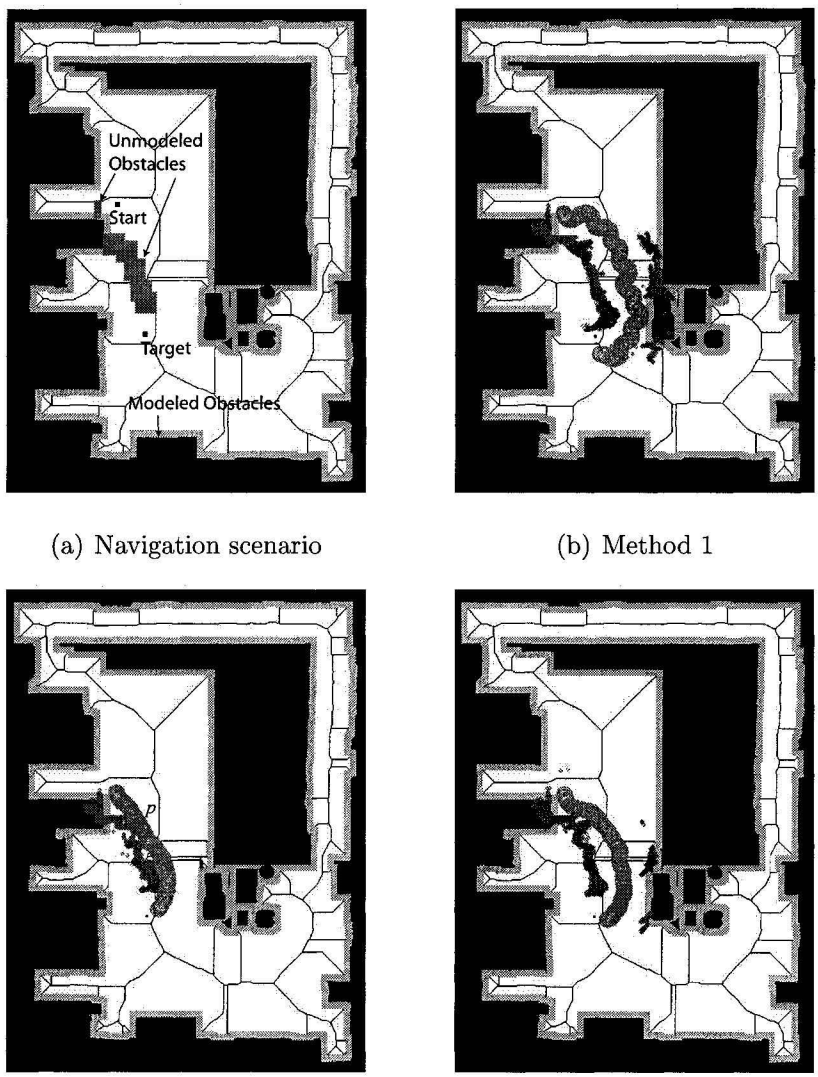
(a) Navigation scenario

(b) Method 1

(c) Method 2

(d) Method 3

Figure 3.13: Navigation results in Case II



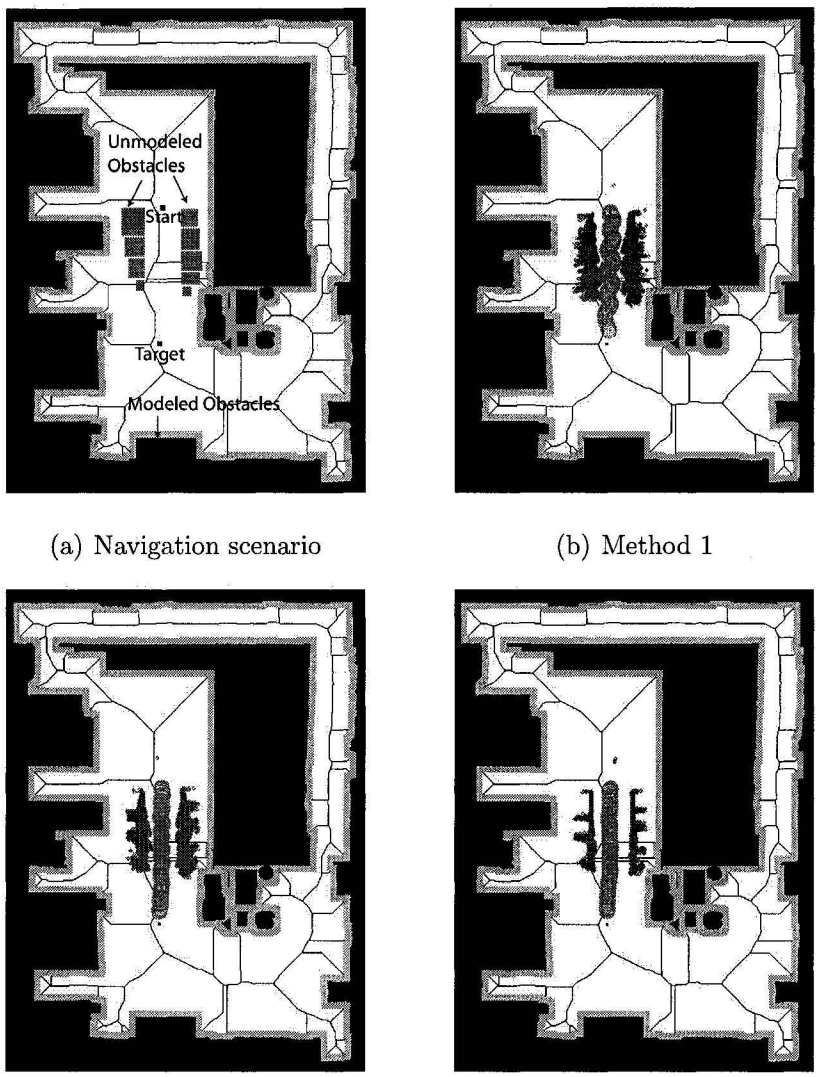
(a) Navigation scenario

(b) Method 1

(c) Method 2

(d) Method 3

Figure 3.14: Navigation results in Case III



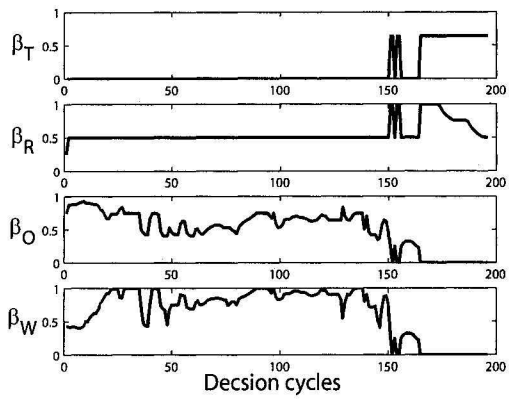
(a) Navigation scenario

(b) Method 1

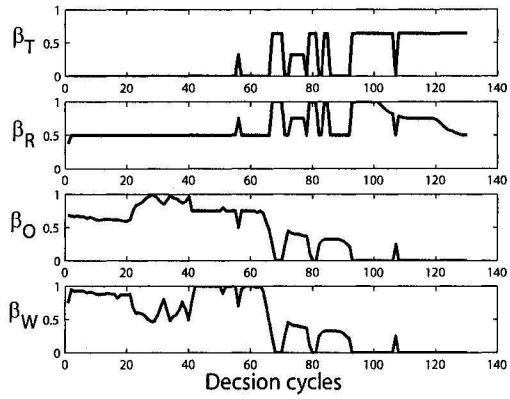
(c) Method 2

(d) Method 3

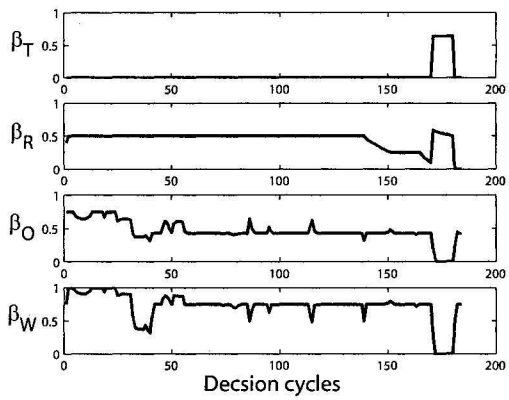
Figure 3.15: Navigation results in Case IV



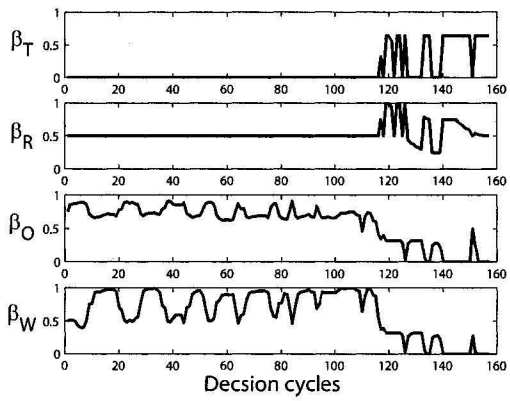
(a) Case I



(b) Case II



(c) Case III



(d) Case IV

Figure 3.16: Weights generated by the fuzzy coordinator in different cases

Method 2

In this case, the schemas are weighed equally and has no flexibility to suppress *go-to-target* and *route-follow* schemas in order to prioritize *avoid-obstacle* and *wall-follow* schemas for safe navigation. As a result the system experiences the highest number of collisions. In Case I (Fig.3.12(c)) and III (Fig.3.14(c)), the robot collided with the obstacle at point p. However, this method produces the shortest traveled distance and navigation time, and the smoothest trajectory as compared to other methods.

Method 3

The robot successfully completed the navigational tasks and showed the overall best performance in terms of oscillation-free trajectory and collision-less navigation (see Fig.3.12(d), 3.13(d), 3.14(d), and 3.15(d)). The effects of environmental dynamics on navigational performance are minimal as compared to other coordinators. This method produces collision-free navigation with reasonable traveled distance and navigation time. Fig.3.16 describes the modulating weights generated by the fuzzy coordinator for the motor schemas. The modulating weights demonstrate that the motor schemas are dynamically weighed based on the locally sensed information and is capable of avoiding local minima created by U-shape obstacles and closely spaced obstacles (see Fig.3.12(d) and 3.13(d)). Furthermore, online balancing of *avoid-obstacle* and *wall-follow* schema reduces oscillations in the generated trajectory of the robot (see Fig.3.14(d) and 3.15(d)).

3.5 Different aspects of the FL-based coordinator

The FL-based methodology used in the experiment falls into coordination class 19 as described in Section 2.2. In other words, this technique combines the weighted

actions of different behaviors to produce the final motion command of the robot. The weights are generated using memory-less fuzzy context rules. Moreover, this approach employs single level behavioral decomposition to coordinate the behaviors.

Different aspects of the FL-based behavior coordinator with respect to the properties of a behavior coordinator as discussed in Section 2.4 are summarized as follows.

- The FL-based coordinator does not include any explicit means (e.g., priority ranking of behaviors for predefined execution order) to incorporate arbitration mechanism. While forming fuzzy rules, reactive behaviors are given more importance than strategic behaviors for safe navigation. The activity of a strategic behavior is determined using its cooperativeness with the reactive behaviors. As a result, this approach implicitly combines the features of an arbitration technique using fuzzy context rules. However, the major disadvantage observed in this approach is that determination of the cooperativeness of a behavior becomes difficult when the number of behaviors increases. This also requires producing large number of rules. A possible solution of this problem can be solved in two steps. First, each behavior will produce its expected activity. Then, a separate decision-layer will find the cooperative behaviors and their modulating weights using an arbitration technique.

- The FL-based coordinator employs fuzzy context-rule based knowledge representation for world-state modeling. The world states are better represented using FL since it can mimic human reasoning to handle sensory uncertainty. However, the major challenge involves in this approach is to design optimum number of rules for world-state modeling.

- The FL-based coordinator does not incorporate previous information of the system. The process of decision-making process entirely depends on the current sensory data. Therefore, it does not fulfill the requirement of persistency. The

incorporation of previous state information requires including higher number of input variables in the fuzzy rules. This leads to the formations of complex fuzzy predicates and higher number of rules. However, the requirement of previous state information does not effect significantly the robot's performance when the environment is static as used in the present experiment.

- The FL-based coordinator does not incorporate any means to predict a likely event in the dynamic environment. However, predictive decision-making has less effect on the robot's performance when the probability of occurrence of a dynamic event is low.

- The FL-based coordinator uses separate fuzzy modules to determine the modulating weights for each behavior. Such a modular approach increases the robustness of the system. However, this method violates one requirement of the modular approach, namely scalability. Hence, addition of a new behavior will change the existing fuzzy modules since the construction of the rules exploits the inter-behavior dependencies.

- This method does not employ hierarchical approach, which includes multi-level behavioral decomposition and decision-tree based behavior coordination. As a result, it inhibits further opportunity to reduce the computational complexity. Hierarchical approaches partition the behaviors into different groups according to their execution orders and exploit the inter-behavior dependencies of each group to form the fuzzy rules. As a consequence, the possibilities of formations of complex fuzzy predicates and large number of fuzzy rules are reduced.

- The FL-based coordinator does not provide any measure to infer the quality of decision-making. For example, this approach does not provide any analysis to infer the quality of the current decision with respect to the sensory uncertainty,

or the quality of avoiding undesired world states of the environment.

3.6 Conclusion

This chapter presents a novel behavior-based architecture for mobile robot navigation, which employs fuzzy context dependent behavior modulation for motor schema based behaviors. The proposed approach contributes in eliminating the most common problems associated with motor schema based navigation. The FL-based approach provides the opportunity of world-state modeling using multi-valued logic and has the ability to mimic human reasoning for sensory uncertainty handling. The experiments further reveal that fuzzy reasoning outperforms the conventional motor schema and unmodulated vector summation based navigation methods. However, this approach does not provide adequate flexibility to incorporate features of behavior arbitration techniques. Moreover, the construction of fuzzy rules becomes difficult if the previous system information is required to be preserved for a large number of behaviors. It also lacks hierarchical approach of decision-making and suiatble means of decision analysis. The following chapter will present a novel behavior coordination technique that attempts to address the above mentioned aspects of a behavior coordinator using FDES.

Chapter 4

Behavior-based Robot Control Using Fuzzy Discrete Event System

4.1 Introduction

This chapter demonstrates a novel a behavior coordination technique that addresses several properties of a behavior coordinator as outlined in Section 2.4. The properties are revisited as follows.

1. A behavior coordinator combine both behavior arbitration and command fusion techniques to facilitates the coordination of competitive and cooperative behaviors,
2. It should possess adequate means of world states modeling,
3. It should employ multi-valued logic to handle sensory uncertainties,
4. It should be capable of making persistent behavior selection,
5. It should provide the opportunity of hierarchical decision-making for reactive action generation,

6. It should be capable of predicting probable world states to handle environmental uncertainties beforehand.

7. It should provide satisfactory means of decision analysis, and

8. It should be modular to increase the robustness of the system.

The present work attempts to integrate the above mentioned characteristics of a behavior coordinator. However, property 6 has not been addressed in this work since the formalism required for predictive control requires probabilistic inferences, whereas the proposed work is intended to exploit multi-valued logic based possibility theory. At the end of this thesis, further research directions will be provided to incorporate both possibility and probability theory in the same frame for the behavior coordination problem.

The proposed approach employs Fuzzy Discrete Event System (FDES) [70,71,72, 73,74] to formulate a behavior coordinator. It combines the state-based formalism of DES with the deterministic vagueness of fuzzy decision-making. DES is an effective tool to analyze complex systems that are difficult to model with differential equations but can be described by sequence of events [69]. A sequence of events can record the changes in the state of a system. When DES is employed for behavior-based robotic control, the state of a system is updated using the events constructed from sensory data. The DES theory also provides the formalism to analyze system characteristic. However, DES may lead to erroneous states if sensory information is inaccurate. The sensory uncertainty reflected into system states can be better represented by FL. With partially known (or incomplete) information, the states can be described using different grades of membership. The formalism of FDES proposed in [70, 71, 72, 73, 74] facilitates the integration of state-based analysis of DES along with the approximate reasoning of FL. FDES is implemented using a fuzzy automaton, where the states and the events are fuzzily defined. The possibility of occurrence of an event

varies according to the imprecision and uncertainty in sensory information, human observation, judgement and interpretation. An application of FDES in the field of biomedicine is presented in [72], where the change of a person's health status, say from *excellent*, to another, say *good*, is shown imprecise, since it depends on human observation and judgement. This approach has generalized the crisp observability into fuzzy observability, which is capable of describing the extent to which the output of a system contains sufficient information to make a right decision. Some optimal control problems are also discussed in [70]. The traditional supervisory control problems of FDES are discussed in [73, 74], where controllability issues of a fuzzy language are demonstrated in detail.

The rest of this chapter will describe the extension of FDES in the context of behavior-based robotics. Section 4.2 states the modulation technique for behavior-based robotic control. Sections 4.3 and 4.4 describe the FDES formalism and behavior modulation mechanism. The parameter tuning procedure is illustrated in Section 4.5. Section 4.6 presents a discussion on different aspects of the proposed methodology. Finally, Section 4.7 draws the conclusion.

4.2 Problem statement

This section provides a formalism of the FDES-based behavioral model and briefly describes the purpose of the proposed behavior modulation technique.

Definition 1. A behavior is a triplet

$$B_i = (A_i, Z_i, F_i)\,,$$

where:

B: the identification label of a behavior (e.g., *go-to-target*)

i: the execution priority order of a behavior. It is predefined and ranges from $1, \ldots, M$ (M has the highest priority and 1 has the lowest priority).

A_i: the expected action to be accomplished by a behavior (e.g., velocity command)

Z_i: the set of sensory information z_{ij}, $j = 1, \ldots, J_i$ (J_i is the total number of sensory information associated with B_i).

F_i: a FDES that determines the state-based activity of B_i to modulate the action A_i.

The state-based prediction of F_i is performed using a set of fuzzy event matrices (Σ_i) and are generated using another set of FDES, F_{ij} as shown in Fig. 4.1. Each F_{ij} is used to produce state-based prediction using single sensory information and generates the activity of the i^{th} behavior based on the j^{th} sensory information z_{ij}. Hence, F_i combines state-based predictions made by different sensory information to estimate the activity of behavior B_i. Fig. 4.1 shows the construction of F_i. Throughout this work, the current state of a FDES is denoted as s, whereas the expected state is specified as $\acute{s}$. The expected activity state $\acute{s}_{ij}$ is generated by F_{ij} while taking a set

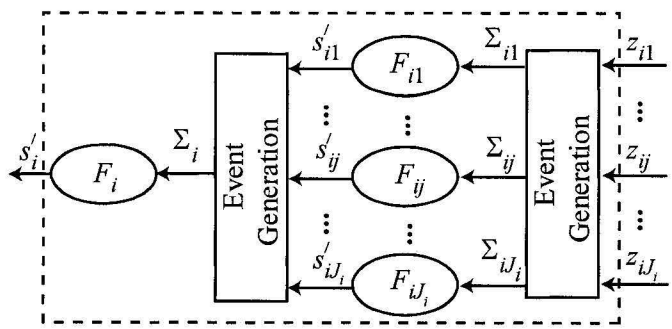


Figure 4.1: Construction of F_i

of fuzzy event matrices Σ_{ij}. These event matrices are generated using the sensory information z_{ij}. The set of event matrices Σ_i is formed using $\acute{s}_{ij}$, which is further used by F_i to generate the combined activity state $\acute{s}_i$.

The task of the proposed behavior modulator is to coordinate actions of different behaviors as

$$A = \sum_{i=1}^{M} \beta_i \times A_i \tag{4.1}$$

where the modulating factor β_i is generated using $\acute{s}_i$.

4.3 Modeling of FDES

This section describes the general formulation of FDES and modeling of F_i and F_{ij}.

4.3.1 General formulation of FDES

FDES is implemented using a fuzzy automaton [70] and is modified for the present work as follows.

Definition 2. A FDES is implemented using a fuzzy automaton, which is a four-touple

$$G = (S, \Sigma, \xi, s_o),$$

where:

S: a set of fuzzy state vectors s,

Σ: a set of fuzzy event matrices α^e,

ξ: a state transition function from $S \times \Sigma$ to S, and

s_o: a initial fuzzy state vector.

The set of fuzzy state vectors is defined as

$$S = \left\{ s | s = [\mu_n]_{1 \times N}, \mu_n \in [0, 1] \right\}. \tag{4.2}$$

Here, N is the number of states in the system and μ_n is the degree of possibility of being in the n^{th} state. The set of fuzzy event matrices is defined as

$$\Sigma = \{\, \alpha^e \,|\alpha^e = [\mu^e_{n\acute{n}}]_{N\times N}\,, \mu^e_{n\acute{n}} \in [0,1]\,, e = 1,\ldots,E\}. \tag{4.3}$$

Here, E denotes the number of events in a FDES and $\mu^e_{n\acute{n}}$ indicates the state transition possibility from the n^{th} state to the $\acute{n}^{th}$ state when the e^{th} event α^e occurs. The state transition function, $\xi : S \times \Sigma \rightarrow S$, generates the next state vector $\acute{s}^e$ with respect to the current state vector s upon occurrence of an event α^e. Hence,

$$\xi(s,\alpha^e) = \acute{s}^e = s \circ \alpha^e = [\acute{\mu}^e_n]_{1\times N} \;\;,\;\; \acute{\mu}^e_n \in [0,1] \tag{4.4}$$

where $(\circ)$ is the max-product operator. The overall next state vector $\acute{s}$ is determined using (4.5).

$$\acute{s} = \left[\max_e \{\acute{\mu}^e_n\}\right]_{1\times N} = [\acute{\mu}_n]_{1\times N} \;\;,\;\; \acute{\mu}_n \in [0,1] \tag{4.5}$$

Example 1. Fig.4.2 shows a FDES with $N = E = 3$. The state-label $1|0.4$ indicates that the current possibility of being in state 1 is 0.4. Therefore, the initial state vector is $s_o = [\;0.4 \quad 0.7 \quad 0.2\;]$. The event-label $\alpha^1|0.9$ denotes that the possibility of occurrence of event α^1 is 0.9. The event matrices are formed as shown bellow.

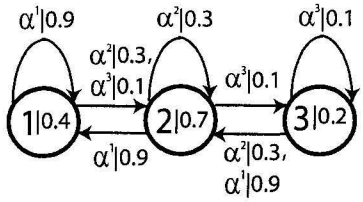


Figure 4.2: FDES for *Example* 1

$$\alpha^1 = \begin{bmatrix} 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \\ 0 & 0.9 & 0 \end{bmatrix},\; \alpha^2 = \begin{bmatrix} 0 & 0.3 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0.3 & 0 \end{bmatrix}, \text{and } \alpha^3 = \begin{bmatrix} 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \\ 0 & 0 & 0.1 \end{bmatrix}$$

The next state vector $\acute{s}^1$ is determined as follows.

$$\begin{aligned}
\acute{s}^1 &= [\ 0.4 \quad 0.7 \quad 0.2\ ] \circ \begin{bmatrix} 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \\ 0 & 0.9 & 0 \end{bmatrix} \\
&= [\max\{0.36, 0.63, 0\} \ \max\{0, 0, 0.18\} \ \max\{0, 0, 0\}] \\
&= [\ 0.63 \quad 0.18 \quad 0\ ]
\end{aligned}$$

Similarly,

$$\acute{s}^2 = [\ 0 \quad 0.21 \quad 0\ ] \text{ and } \acute{s}^3 = [\ 0 \quad 0.04 \quad 0.07\ ]$$

The overall next state vector $\acute{s}$ is calculated using the following formula,

$$\acute{s} = [\max\{0.63, 0, 0\} \ \max\{0.18, 0.21, 0.04\} \ \max\{0, 0, 0.07\}] = [\ 0.63 \quad 0.21 \quad 0.07\ ].$$

■

State-based observability

Each event α^e in a FDES is associated with a user defined degree of observability (certainty) $d_e \in [0, 1]$ and a degree of unobservability (uncertainty) $d_e^c \in [0, 1]$ where $d_e + d_e^c = 1$. They can usually be determined using experimental data. The parameters d_e and d_e^c can be interpreted as certainty and uncertainty, respectively, associated with the sensory data used in constructing α^e. The major distinction between an observable and unobservable event is the uncertainty in their occurrence. The occurrence of an observable event is certain, whereas the occurrence of an unobservable event is uncertain due to the sensor failure. In the worst case, an unobservable event can occur N times and loops back to a previously visited state without being detected. The unobservable form of an event is denoted as

$$\acute{\alpha}^e = I\tilde{\cup}(\alpha^e)_1\tilde{\cup}(\alpha^e)_2\tilde{\cup}\cdots\tilde{\cup}(\alpha^e)_N \tag{4.6}$$

where $\tilde{\cup}$ is a fuzzy OR operator (maximal) and I is the identity matrix. The matrix $(\alpha^e)_N$ is calculated as

$$(\alpha^e)_N = \alpha^e \otimes \alpha^e \cdots \alpha^e \otimes \alpha^e \quad (N \text{ times}). \tag{4.7}$$

Here, $\otimes$ is the max-min fuzzy composition. If observability is incorporated, ξ is redefined with (4.8).

$$\xi(s, \alpha^e, d_e, d_e^c) = \acute{s}^e = s \circ (d_e.\alpha^e \tilde{\cup} d_e^c.\acute{\alpha}^e) \tag{4.8}$$

The product $d_e.\alpha^e$ stands for the observable part of the event α^e, whereas $d_e^c.\acute{\alpha}^e$ is its unobservable part.

The measure of observability of a FDES described in [70] is interpreted to define state-based observability in order to provide state-based decision making for a physical agent like mobile robot. Hence, state-based decision-making increases reactivity to a robot system. The state-based observability (O) for a FDES is calculated using (4.9)

$$O = 1 - \frac{\sum_{e=1}^{E} \left(\acute{s}^e \circ L \circ (\acute{s}^e)^{\mathrm{T}}\right)}{\sum_{e=1}^{E} \left(\acute{s}^e \circ (\acute{s}^e)^{\mathrm{T}}\right)} \tag{4.9}$$

where T is used for transpose and matrix $L = [l_{n\acute{n}}]_{N \times N}$ is the inconsistency matrix defined by the user. The degree of inconsistency between the n^{th} and $\acute{n}^{th}$ states in a FDES is denoted as $l_{n\acute{n}}$. If $O = 1$, a FDES is completely observable for next state and consistent decisions can be made based on the observations.

Example 2. For the FDES presented in *Example* 1, the unobservable form of event

α^1 is determined as follows.

$$\acute{\alpha}^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tilde{\cup} \begin{bmatrix} 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \\ 0 & 0.9 & 0 \end{bmatrix} \tilde{\cup} \begin{bmatrix} 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \end{bmatrix} \tilde{\cup} \begin{bmatrix} 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \\ 0.9 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0.9 & 1 & 0 \\ 0.9 & 0.9 & 1 \end{bmatrix}$$

Similarly, $\acute{\alpha}^2$ and $\acute{\alpha}^2$ are calculated as

$$\acute{\alpha}^2 = \begin{bmatrix} 1 & 0.3 & 0 \\ 0 & 1 & 0 \\ 0 & 0.3 & 1 \end{bmatrix} \text{ and } \acute{\alpha}^3 = \begin{bmatrix} 1 & 0.1 & 0.1 \\ 0 & 1 & 0.1 \\ 0 & 0 & 1 \end{bmatrix}.$$

For $d_e = 0.8$ and $d_e^c = 0.2$, $e = 1, 2, 3$,

$$\acute{s}^1 = [\ 0.4 \quad 0.7 \quad 0.2\ ] \circ \left(\begin{bmatrix} 0.72 & 0 & 0 \\ 0.72 & 0 & 0 \\ 0 & 0.72 & 0 \end{bmatrix} \tilde{\cup} \begin{bmatrix} 0.2 & 0 & 0 \\ 0.18 & 0.2 & 0 \\ 0.18 & 0.18 & 0.2 \end{bmatrix} \right)$$

$$= [\ 0.4 \quad 0.7 \quad 0.2\ ] \circ \begin{bmatrix} 0.72 & 0 & 0 \\ 0.72 & 0.2 & 0 \\ 0.18 & 0.72 & 0.2 \end{bmatrix}$$

$$= [\ 0.504 \quad 0.144 \quad 0.04\ ].$$

Similarly,

$$\acute{s}^2 = [\ 0.08 \quad 0.168 \quad 0.12\ ] \text{ and } \acute{s}^3 = [\ 0.08 \quad 0.14 \quad 0.056\ ]$$

The overall next state is $\acute{s} = [\ 0.504 \quad 0.168 \quad 0.12\ ]$. If

$$L = \begin{bmatrix} 0 & 0.35 & 0.85 \\ 0.35 & 0 & 0.35 \\ 0.85 & 0.35 & 0 \end{bmatrix}$$

then the measure of observability is calculated as

$$O = 1 - \frac{0.0254016 + 0.00816 + 0.00392}{0.254 + 0.0282 + 0.0196}. = 0.8758$$

■

State-based controllability

Controllability of a FDES refers to the achievement of desired state transitions using appropriate set of events. Supervisory control techniques employ a high level decision maker (e.g., the *Arbiter* in the proposed method described in Section 4.4) that generates the appropriate set of events to achieve the desired state transitions. Hence, the measure of controllability quantifies the achievement of desired states at each state-transition. For a FDES, if the current sate is s and the next state is $\acute{s}$, then the state-based controllability is measured as

$$C = 1 - s \circ W \circ (\acute{s})^T \tag{4.10}$$

where the user defined matrix $W = [w_{n\acute{n}}]_{N \times N}$ shows the undesired state transition between the n^{th} and $\acute{n}^{th}$ states in a FDES. The matrix component $w_{n\acute{n}} = 1$ indicates undesired transition between the n^{th} and $\acute{n}^{th}$ states, whereas $w_{n\acute{n}} = 0$ indicates a desired one. If $C = 1$, the next state transition is completely controllable and desired decisions can be made.

Example 3. As a continuation of *Example* 1 and 2, this example calculates the measure of state-based controllability for the FDES shown in Fig.4.2. For

$$s = [\ 0.4 \quad 0.7 \quad 0.2\ ],\ \acute{s} = [\ 0.504 \quad 0.168 \quad 0.12\ ],\ \text{and}\ W = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

the measure of state-based controllability is

$$C = 1 - [\ 0.4 \quad 0.7 \quad 0.2\ ] \circ \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \circ \begin{bmatrix} 0.504 \\ 0.168 \\ 0.12 \end{bmatrix} = 0.8992.$$

■

4.3.2 Modeling of FDES F_{ij}

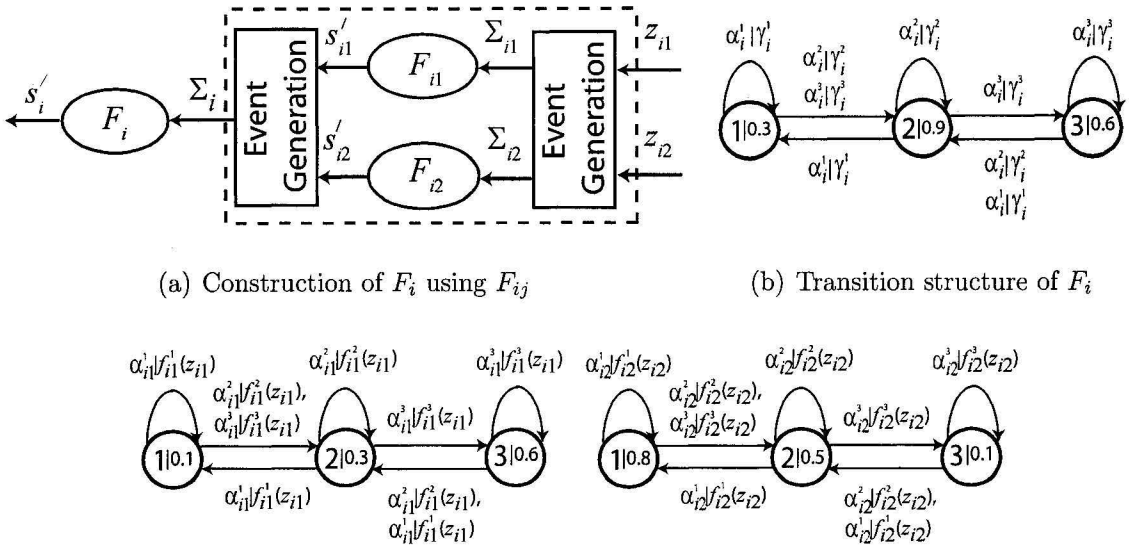
The general formulation described in Section 4.3.1 is now modified to define the FDES F_{ij} that manipulates the j^{th} sensory information of the i^{th} behavior. State 1 and state N are specified as the lowest and highest activity state, respectively. Here, α_{ij}^e is determined using

$$\begin{aligned} \alpha_{ij}^e &= [\mu_{n\acute{n}}^e]_{N\times N} \qquad (4.11) \\ \mu_{n\acute{n}}^e &= \begin{cases} f_{ij}^e(z_{ij}) & \text{if transition exists when} \\ & \alpha_{ij}^e \text{ occurs} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where $f_{ij}^e(\cdot)$ represents a fuzzy MF that maps sensory information z_{ij} to a membership value over $[0, 1]$.

4.3.3 Modeling of FDES F_i

FDES F_i is also defined using the formulation described in Section 4.3.1. State 1 of F_i refers to the lowest activity state and state N stands for the highest activity state. In F_i, the number of events is equal to the number of states, i.e., $E = N$. Here, event



(a) Construction of F_i using F_{ij} (b) Transition structure of F_i

(c) Transition structure of F_{i1} (d) Transition structure of F_{i2}

Figure 4.3: Constructions of the FDESs in *Example* 4

α_i^e is determined using

$$\begin{aligned} \alpha_i^e &= [\mu_{n\acute{n}}^e]_{N\times N} \\ \mu_{n\acute{n}}^e &= \begin{cases} \gamma_i^e & \text{if transition exists when} \\ & \alpha_i^e \text{ occurs} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4.12}$$

where γ_i^e is calculated as

$$\begin{aligned} \bar{s} &= \underset{j}{\text{mean}}\{\acute{s}_{ij}\} = [\bar{\mu}_{\acute{n}}]_{1\times N}, \quad j = 1, \cdots, J_i \\ \gamma_i^e &= \bar{\mu}_{\acute{n}}, \quad \acute{n} = e. \end{aligned} \tag{4.13}$$

Here, $\bar{s}$ is obtained by taking element wise average of the output state vectors $\acute{s}_{ij}$ of FDES F_{ij}.

Example 4. Fig.4.3(a) shows an example of F_i, which is formed using F_{i1} and F_{i2} (i.e., $J_i = 2$). Fig.4.3(b)-4.3(d) depict the transition structures of the FDESs. The event-label $\alpha_i^e|\gamma_i^e$ denotes that the possibility of occurrence of event α_i^e in FDES F_i

is γ_i^e. Similarly, the event-label $\alpha_{ij}^e | f_{ij}^e(z_{ij})$ denotes the possibility of occurrence of event α_{ij}^e in FDES F_{ij} is $f_{ij}^e(z_{ij})$. Let $f_{i1}^1(z_{i1}) = 0.3$, $f_{i1}^2(z_{i1}) = 0.5$, $f_{i1}^3(z_{i1}) = 0.7$, $f_{i2}^1(z_{i2}) = 0.1$, $f_{i2}^2(z_{i2}) = 0.9$, and $f_{i2}^3(z_{i2}) = 0.2$. Events $\alpha_{ij}^e \in \Sigma_{ij}$ are calculated using (4.11) and the output states $\acute{s}_{ij}$ are determined as $\acute{s}_{i1}$=[0.09 0.3 0.42] and $\acute{s}_{i2}$=[0.08 0.72 0.1]. Next, the possibilities of occurrences of the events $\alpha_i^e \in \Sigma_i$ are calculated as $\gamma_i^1 = \text{mean}\{0.09, 0.08\} = 0.085$, $\gamma_i^2 = \text{mean}\{0.3, 0.72\} = 0.51$, and $\gamma_i^3 = \text{mean}\{0.42, 0.1\} = 0.26$, which are used to form α_i^e using (4.12). Finally, FDES F_i updates its state vector as $\acute{s}_i$=[0.0255 0.1960 0.0780].

■

4.4 Behavior modulation mechanism

Fig.4.4 shows the proposed FDES-based supervisory behavior modulation technique. The construction of the supervisor leads to the following definition.

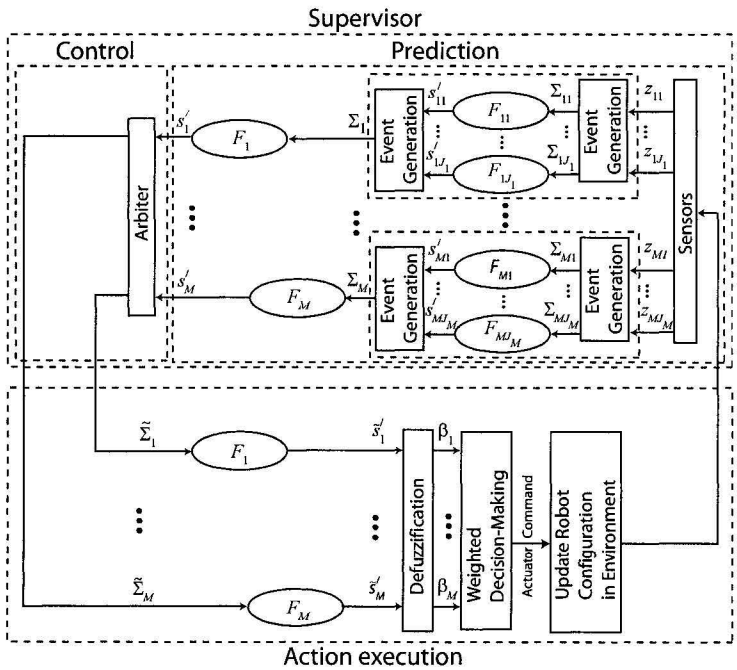


Figure 4.4: FDES-based behavior modulation

Definition 3. A supervisor is a function SUP that maps the sensory feedback into the desired set of events for FDES F_i, $i = 1, \ldots, M$ (M is the total number of F_i associated with each behavior).

$$\begin{aligned} SUP &: Z_1 \times \ldots \times Z_i \times \ldots \times Z_M \rightarrow \tilde{\Sigma}_1 \times \ldots \times \tilde{\Sigma}_i \times \ldots \times \tilde{\Sigma}_M \\ Z_i &= \{z_{ij} | j = 1, \ldots, J_i\} \end{aligned}$$

Here, Z_i is the set of sensory data used in the FDESs and $\tilde{\Sigma}_i$ is the desired set of events produced by the supervisor for FDES F_i.

The proposed behavior modulation technique constitutes three stages: *Prediction*, *Control*, and *Action execution*. The *Prediction* stage employs desired models of the FDESs to estimate the expected behavioral states $\acute{s}_i$ using sensory feedback Z_i. In the *Control* stage, predefined control laws are used by the *Arbiter* to produce appropriate set of events $\tilde{\Sigma}_i$ for FDES F_i using $\acute{s}_i$. The *Action execution* stage uses $\tilde{\Sigma}_i$ to recalculate the behavioral states $\acute{\acute{s}}_i$, which are further used to generate the modulating factors β_i.

The formation of the control laws depends on the type of the behavioral arrangement. The existing behavioral arrangements in a robotic application can be classified into three categories.

Type I: All the behaviors are cooperative, i.e., they have similar goals and assist each other to accomplish their tasks.

Type II: All the behaviors are non-cooperative, i.e., they have different goals and do not assist each other to accomplish their tasks.

Type III: Some of the behaviors are cooperative and the rest are non-cooperative.

Types I and II lead to the following theorems.

Theorem 1. Optimal control is possible for a set of cooperative behaviors using

$\tilde{\Sigma}_i = \Sigma_i$ that produces the predicted behavioral states $\acute{\tilde{s}}_i = \acute{s}_i$.

Proof: The *Prediction* stage employs the desired models of the FDESs, where the FDES F_i produces the expected behavioral state $\acute{s}_i$ using the current state s_i and events $\alpha_i^e \in \Sigma_i$. In the *Action execution* stage, the FDES F_i recalculates the output state vector $\acute{\tilde{s}}_i$ using the same current state vector s_i and the events $\tilde{\alpha}_i^e \in \tilde{\Sigma}_i$. But $\tilde{\Sigma}_i = \Sigma_i$, which implies that $\acute{\tilde{s}}_i = \acute{s}_i$. This completes the proof.

■

Theorem 2. A *Pareto optimal* control is possible for a set of non-cooperative behaviors using $\tilde{\Sigma}_i = \Sigma_i$ only for the i^{th} FDES F_i.

Proof: For a set of non-cooperative behaviors, generation of a desired state of one behavior accompanied with the deteriorated (or undesired) states of other behaviors, which is the requirement of a *Pareto optimal* control. Therefore, the assignment $\tilde{\Sigma}_i = \Sigma_i$ results in the desired behavioral state for FDES F_i. Whereas generation of undesired states of other behaviors requires $\tilde{\Sigma}_k \neq \Sigma_k$, $k \neq i$. This completes the proof.

■

Type III is a combination of Types I and II. Here, the behaviors are, first, categorized into separate groups and then a criterion is setup to determine the desired group of cooperative behaviors, for which *Theorem* 1 is applied. The event sets of other behaviors are determined using heuristics.

The *Arbiter* uses the following assumptions to generate the event sets $\tilde{\Sigma}_i$.

***Assumption* 1:** The behavioral arrangement is of Type III.

***Assumption* 2:** Each FDES F_i has the same number of states N.

***Assumption* 3:** For all FDESs, state 1 stands for the lowest behavioral activity state, whereas the n^{th} state denotes the highest behavioral activity state.

***Assumption* 4:** For all FDESs, if an event α^e occurs N times, the resulted fuzzy state vector has the highest possibility of being in the n^{th} state, where $n = e$. In other words, the final state caused by the event $(\alpha^e)_N$ is the n^{th} state, where $n = e$.

The *Arbiter* performs the following steps to accomplish the event modification process.

Step 1. While comparing the fuzzy membership values of the state vectors, the *Arbiter* first determines the best possible state for each behavior. Let assume the states are labeled as $state_1, state_2, ..., state_N$. Note, the N^{th} state has the highest activity. The behaviors are grouped according to the highest degree of membership value of being in a given state. As an example, if the highest membership of the state vector $\acute{s}_i$ is in $state_n$, the behavior B_i will be grouped as of being in the n^{th} activity state. Hence,

$$[state_n] = \left\{ B_i | \acute{s}_i = [\acute{\mu}_{\acute{n}}]_{1\times N}\,,\ n = \arg\max_{\acute{n}} \{\acute{\mu}_{\acute{n}}\} \right\}$$

where $[state_n]$ is the set of behaviors having the highest memberships in $state_n$.

Step 2. Find the group of behaviors (from non-empty groups) $[state_{\hat{n}}]$ having the highest activity state, say $\hat{n}$. In other words, $[state_{\hat{n}}]$ is selected if

$$\hat{n} = \arg\min_{n} \{N - n\} \quad \text{and} \quad [state_{\hat{n}}] \neq \{\phi\}.$$

Step 3. Within this group select the highest priority behavior, say $\hat{B}$.

Step 4. The cooperative behaviors of $\hat{B}$ are chosen using heuristic knowledge. To find the cooperative behaviors, an equivalence relation $\equiv_{co}$ is defined so that their actions do not conflict with the actions proposed by $\hat{B}$. The set of cooperative behaviors are determined as

$$\left[\hat{B}\right] = \left\{ B_i | B_i \equiv_{co} \hat{B} \right\}.$$

As an example, in force vector modulation, behaviors having directions within δ^o of the direction of $\hat{B}$ can be considered as cooperative behaviors. Therefore, $\equiv_{co}$ is defined as

$$B_i \equiv_{co} \hat{B} \text{ if } \text{abs}(\angle A_i - \angle \hat{A}) \leq \delta^o$$

where $\hat{A}$ represents the expected action of $\hat{B}$. The threshold value δ can be defined as a scalar value or a matrix $\delta = [\delta_{ij}]_{M \times M}$. When δ represents a scalar value, the same threshold is used for all comparisons, whereas in the matrix form, a different threshold is used for each comparison. For example, the matrix component δ_{ij} indicates that the i^{th} behavior is selected as $\hat{B}$ and the j^{th} behavior is cooperative with it if the difference between their expected heading directions remains within δ^o_{ij}.

Step 5. Once the cooperative behaviors are chosen, the *Arbiter* redefines the set of event matrices as $\tilde{\Sigma}_i$ based on the activity states of the behaviors $(\acute{s}_i)$. The modified event matrices in turn alter the state prediction $(\acute{\tilde{s}}_i)$ of each F_i. The *Arbiter* assigns $\tilde{\alpha}^e_i = \alpha^e_i$ for $B_i \in [\hat{B}]$ according to *Theorem* 1. This ensures generation of desired activities for cooperative behaviors. For non-cooperative behaviors $B_i \notin [\hat{B}]$, event $\tilde{\alpha}^e_i$ is modified to cause a transition to the lowest activity, i.e., $state_1$. Hence, event $\tilde{\alpha}^e_i = [\tilde{\mu}^e_{n\acute{n}}]_{N \times N}$ is determined for $e = 1$ as

$$\tilde{\mu}^e_{n\acute{n}} = \begin{cases} \max\limits_{\acute{n}} \{\acute{\mu}_{\acute{n}}\} & \text{if transition exists when} \\ & \tilde{\alpha}^e_i \text{ occurs} \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

and for $e \neq 1$,

$$\tilde{\mu}^e_{n\acute{n}} = \begin{cases} \min\limits_{\acute{n}} \{\acute{\mu}_{\acute{n}}\} & \text{if transition exists when} \\ & \tilde{\alpha}^e_i \text{ occurs} \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

where $\acute{s}_i = [\acute{\mu}_{\acute{n}}]_{1\times N}$.

The F_i determines the final activity state vector $\acute{\tilde{s}}_i = [\acute{\tilde{\mu}}_n]_{1\times N}$ using event matrices $\tilde{\alpha}_i^e$. The activity state $\acute{\tilde{s}}_i$ is then defuzzified using (4.16) to modulate the expected actions A_i of B_i.

$$\beta_i = \frac{\sum_{n=1}^{N} \left(\acute{\tilde{\mu}}_n \times \frac{(n-1)}{N-1} \right)}{\sum_{n=1}^{N} \acute{\tilde{\mu}}_n} \tag{4.16}$$

Finally, actions of different behaviors are combined using (4.1).

Remark 1. DES-based coordinator is a special form of FDES-based coordinator, which requires two modifications:

- The membership functions are changed to support either 1 or 0 membership.
- Event matrix $\alpha_i^e = [\mu_{n\acute{n}}^e]_{N\times N}$ is changed so that the state transition possibility $\mu_{n\acute{n}}^e$ can be either 0 or 1. Here, $\mu_{n\acute{n}}^e$ is calculated as:

$$\mu_{n\acute{n}}^e = \begin{cases} 1 & \text{if transition exists when} \\ & \alpha_i^e \text{ occurs and } e = \arg\max_{\acute{e}}(\gamma_i^{\acute{e}}) \\ 0 & \text{otherwise} \end{cases} \tag{4.17}$$

  where γ_i^e is calculated using (4.13).

Remark 2. Behavior arbitration is defined as a special form of DES-based coordinator where the following modifications are accomplished:

- Assign $[\hat{B}] = \{\hat{B}\}$, i.e., no cooperative behaviors are allowed with the highest priority behavior in the most desirable state.
- To make $\hat{B}$ the only active behavior, set $\beta_i = 0$ for $B_i \neq \hat{B}$.

Example 5. A system comprises of three behaviors (i.e., $i = 1, 2, 3$) and each FDES has 3 states and 3 events (i.e., $N = E = 3$). Let the updated state vectors of FDESs F_i are $\acute{s}_1$ =[0.12 0.35 0.42], and $\acute{s}_2$=[0.1 0.2 0.6], and $\acute{s}_3$=[0.25 0.72 0.1]. The behavioral actions are expressed as $A_i = |A_i|\angle\theta_i$, which indicates that the action A_i is directed to θ_i and has magnitude $|A_i|$. The actions are set to $A_1 = 1\angle 25°$, $A_2 = 1\angle 65°$, and $A_3 = 1\angle 60°$ (which are unit vectors to the expected heading directions). The equivalence relation is defined as

$$B_i \equiv_{co} \hat{B} \text{ if } \text{abs}(\angle A_i - \angle \hat{A}) \leq \delta^o$$

where $\hat{A}$ represents the expected action of the highest priority behavior $\hat{B}$ in the maximum activity state and $\delta = 5°$. Hence, the control algorithm is described as follows.

Step 1. Categorize behaviors as $[state_1]=\phi$, $[state_2]=\{B_3\}$, and $[state_3]=\{B_1, B_2\}$.

Step 2. Find behaviors in the maximum activity state as $[state_3]=\{B_1, B_2\}$.

Step 3. Find the highest priority behavior as $\hat{B}=B_2$.

Step 4. Find cooperative behaviors as $[\hat{B}]=\{B_2, B_3\}$.

Step 5. For $B_i \in [\hat{B}]$, event $\tilde{\alpha}_i^e$ is set equal to α_i^e and for $B_i \notin [\hat{B}]$, if $e = 1$, $\tilde{\alpha}_i^e = [\tilde{\mu}_{n\acute{n}}^e]_{3\times 3}$ is determined using

$$\tilde{\mu}_{n\acute{n}}^e = \begin{cases} \max\{0.12, 0.35, 0.42\} = 0.42 & \text{if transition exists when} \\ & \tilde{\alpha}_i^e \text{ occurs} \\ 0 & \text{otherwise} \end{cases}$$

and for $e \neq 1$,

$$\tilde{\mu}_{n\acute{n}}^e = \begin{cases} \min\{0.12, 0.35, 0.42\} = 0.12 & \text{if transition exists when} \\ & \tilde{\alpha}_i^e \text{ occurs} \\ 0 & \text{otherwise.} \end{cases}$$

Let the updated state vectors of FDES F_i are

$$\begin{aligned} \acute{s}_1 &= [\ 0.44 \quad 0.35 \quad 0.12\ ], \\ \acute{s}_2 &= [\ 0.2 \quad 0.32 \quad 0.9\ ], \text{ and} \\ \acute{s}_3 &= [\ 0.02 \quad 0.76 \quad 0.19\ ]. \end{aligned}$$

Using (4.16), the modulating factors are calculated as β_1=0.324, β_2=0.746, and β_3=0.588. As an instance, β_1 is determined as

$$\beta_1 = \frac{0.44 \times \frac{0}{2} + 0.35 \times \frac{1}{2} + 0.12 \times \frac{2}{2}}{0.44 + 0.35 + 0.12} = 0.324.$$

Next, the modulated action is obtained using

$$\begin{aligned} A &= 0.324\angle 25^\circ + 0.746\angle 65^\circ + 0.588\angle 60^\circ \\ &= 1.6\angle 55.665^\circ. \end{aligned}$$

Hence, the coordinated action A is directed to 55.665° and its magnitude is 1.6.

■

4.4.1 Computational complexity

The computational complexity of the modulation process is governed by the process of event and state generation in a FDES, which is on the order of EN^2 or $\bigcirc(EN^2)$. Furthermore, the number of states N and the number of events E are usually equal in a FDES. Therefore, when $E = N$, the computational complexity of a FDES will be $\bigcirc(N^3)$. The overall complexity of the decision process depends on the number of FDES F_i, which is M and the number of FDES F_{ij}, which is $J = \sum_{i=1}^{M} J_i$. Hence, the overall complexity is $\bigcirc((2M + J)N^3)$ (M is counted twice since FDES F_i is used twice in the decision process). Therefore, to reduce the number of elementary operations in a large behavior-based system, it requires selection of fewer number of sensory information as well as FDESs with small number of states.

4.4.2 Multilevel behavioral decomposition

The proposed method incorporates multilevel behavioral decomposition using a decision tree, where the given task is divided into sequentially ordered high level (or abstract) behaviors. A high level behavior does not generate any motion command, instead it selects a lower-level abstract behavior or a group of primitive behaviors at the lowest level. The primitive behaviors are implemented and coordinated with the FDES-based methodology described in the previous sections.

Example 6. Fig.4.5 shows a multilevel behavioral decomposition where the *navigation-and-pulling* task given to a robot is divided into *anchoring* and *navigate-to-goal* behaviors that are to be executed sequentially (refer to Chapter 6). The robot, first, anchors itself with an object and then pulls it to a goal position. The primitive behav-

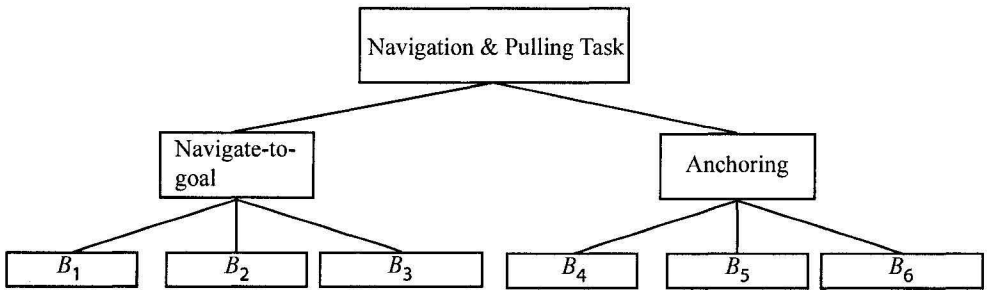


Figure 4.5: An example of multilevel behavioral decomposition

iors B_i associated with *navigate-to-goal* ($i = 1, \ldots, 3$) and *anchoring* ($i = 4, \ldots, 6$) are coordinated with separate FDES-based behavior modulators.

■

The use of separate modulators helps reducing the overall computational complexity. This is illustrated using the following example.

Example 7. Let assume the *navigation-and-pulling* task uses single level behavioral decomposition as shown in Fig.4.6 and employs a single behavior coordinator. If each FDES uses three sensory data (i.e., $J_i = 3$) and their transition structures employ

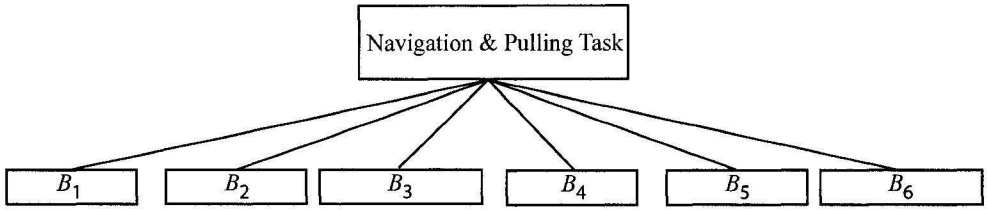


Figure 4.6: An example of multilevel behavioral decomposition

three states (i.e., $N = 3$), the number of elementary operations required at each decision cycle is $((2 \times 6 + 18) \times 3^3) = 810$. On the other hand, the use of separate modulators requires $((2 \times 3 + 9) \times 3^3) = 405$ elementary operations at each decision cycle.

■

Therefore, multilevel behavioral decomposition provides the capability of fast decision-making by reducing the computational complexity of the overall decision process.

4.5 Parameter tuning

The FDES-based behavior modulator requires assignment of following parameters.

- Priority ranking of the behaviors (i),
- Degree of observability d_o^e of each event that reflects the reliability of the sensory information associated with the event,
- Matrix L that describes the inconsistency between the actions taken in different activity states,
- Matrix W that describes the undesired transitions between the activity states, and

- Ranges of MFs.

The rest of this section describes the proposed parameter tuning methods.

4.5.1 Priority ranking i

The following general guidelines are used to assign priority ranking of the behaviors:

- Assign the highest priority to a behavior that models dynamic changes in the environment, e.g., *avoid-obstacle* behavior in mobile robot navigation.
- Next, assign the priority ranking according to the execution order of behaviors.

4.5.2 The degree of observability d_o^e

The degree of observability is a probabilistic measure that describes the reliability of a sensory data used in an event. It is a ratio of the number of valid sensor readings to the total number of sensor readings taken in an experiment. For example, $d_o^e = 0.8$ indicates that the reliability of the sensory data used in the e^{th} event is found 80% in the experiments. If an event uses an information that consists of more than one sensory data, the product of the reliability of each sensory data is assigned to d_o^e.

4.5.3 Matrix L

Matrix $L = [l_{n\acute{n}}]_{N\times N}$ defines the inconsistency between the actions taken in different activity states. The inconsistency can be approximated by taking the difference between the estimated modulating factors of two different states. A state vector having possibility 1 of being in the n^{th} state modulates an action using the factor $\left(\frac{n-1}{N-1}\right)$. Therefore, the action inconsistency between the n^{th} and $\acute{n}^{th}$ states can be approximated as $\left(\frac{n-1}{N-1} - \frac{\acute{n}-1}{N-1}\right)$. However, MFs generating state transition possibilities are

generally kept overlapping. This indicates that the activity states are also overlapping. An overlap between two states infers similarity between their actions. Hence, the following formula is proposed as a guideline to define $l_{n\acute{n}}$

$$l_{n\acute{n}} = l_{\acute{n}n} = \text{abs}\left(\frac{n-1}{N-1} - \frac{\acute{n}-1}{N-1}\right) \times (1 - \lambda_{n\acute{n}}) \tag{4.18}$$

where $\lambda_{n\acute{n}}$ is the amount of overlap between the n^{th} and $\acute{n}^{th}$ states measured in terms of the intersected membership value of their corresponding MFs. When two states are not adjacent and the corresponding MFs do not intersect, their actions are related in terms of the intermediate states. Hence, $\lambda_{n\acute{n}}$ is determined using

$$\lambda_{n\acute{n}} = \lambda_{n\acute{n}} \times \cdots \times \lambda_{(\acute{n}+1)\acute{n}} \tag{4.19}$$

where $(n - \acute{n}) > 1$. The estimated value of $l_{n\acute{n}}$ can further be modified according to the expert knowledge.

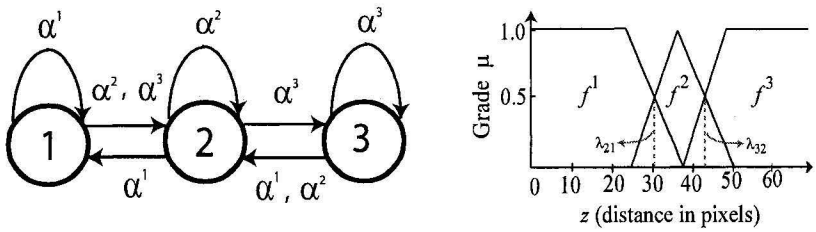
The inconsistency matrix L_{ij} of FDES F_{ij} is determined using (4.18) and the inconsistency matrix L_i of FDES F_i is determined by averaging entries of each L_{ij} with the same indices.

4.5.4 Matrix W

Matrix $W = [w_{n\acute{n}}]_{N\times N}$ denotes the undesired transitions between the activity states. The expected state transitions between the n^{th} and $\acute{n}^{th}$ states are denoted using $w_{n\acute{n}} = 0$, whereas the undesired transitions are specified using $w_{n\acute{n}} = 1$.

Example 8. A FDES is implemented using the transition structure shown in Fig.4.7(a) and employs the MFs depicted in Fig.4.7(b). Here, $N = 3$, $\lambda_{32} = \lambda_{21} = 0.5$, and $\lambda_{31} = 0.5 \times 0.5 = 0.25$. Matrix L is determined with (4.18) and (4.19) as

$$L = \begin{bmatrix} 0 & 0.25 & 0.75 \\ 0.25 & 0 & 0.25 \\ 0.75 & 0.25 & 0 \end{bmatrix}.$$



(a) Transition structure (b) Membership functions

Figure 4.7: Construction of the FDES in *Example* 8

According to the transition structure shown in Fig.4.7(a), direct transitions between state 1 and 3 are inhibited. Hence, matrix W is determined as

$$W = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

■

4.5.5 Ranges of MFs

Tuning parameters of MFs is a common issue in FL-based applications. The tuning procedure mainly depends on the expert knowledge. The following guidelines are proposed to set the ranges of the MFs:

- The input range of the sensory information associated with a FDES is divided equally to define the MFs when the corresponding states are equally possible. When the possibility of being in a particular state is expected to be increased, the range of the corresponding MF is widened.

- The peak values of the MFs are chosen so that the effects of the states become prominent at those point.

- An overlap between two MFs is determined according to the intended consistency of actions taken in the corresponding states.

4.6 Different aspects of the FDES-based method

In Section 4.1, it was proclaimed that the proposed FDES-based methodology integrates several important properties of a behavior coordinator. The following discussion justifies this claim.

1. The proposed method combines features of both behavior arbitration and command fusion techniques to facilitates the coordination of competitive and cooperative behaviors. The inclusion of explicit priority of behaviors denotes behavior arbitration as discussed in the priority-based coordination classes (see Fig.2.4 and Table 2.1), whereas incorporation of behavior modulation indicates command fusion as described in the superposition-based coordination classes (see Fig.2.4 and Table 2.1). Competitive behaviors are prioritized according to their predefined ranking and expected activity at a particular moment. Cooperative behaviors of the highest priority behavior are determined according to their action similarity. Thus, both the competitive and cooperative behaviors are selected on the basis of current context and their actions are weighed using the modulating factors produced by the FDES-based technique. The present method employs separate layers to generate the activity of a behavior and to determine its cooperative behaviors. This reduces the overall computational complexity of the decision process unlike the FL-based approaches discussed in Section 3.5.

2. This method provides adequate means for world states modeling with DES and FL. The state transition structure of the proposed approach implements context

rules using state vectors and event matrices that models the current sensory information. The FDES-based method outperforms the conventional DES-based technique since fuzzily defined state vectors and event matrices exhibit robust performance in case of noisy perceptions. In a finite state automaton, the number of the state vectors is fixed since the possibility of being in a state is either 0 or 1; whereas in a fuzzy automaton, the number of the state vectors is infinite since the possibility of being in a state can take any value in the range [0,1]. Therefore, FDES adds additional flexibility to model the world states using the same state transition structure as DES.

3. This method provides satisfactory means of decision analysis using state-based observability and controllability. Observability models the decision-vagueness based on the uncertainty associated with the sensory data and controllability models the decision-risk based on the dynamic changes within the observed environment. Thus the present method provides the opportunity to analyze the vagueness and risk of decision made depending on the current context and takes appropriate actions to cope with sensory uncertainty and environmental dynamics. In the following chapter, it will be demonstrated that the measures of observability and controllability can be used for velocity and sampling-frequency modulation for safe mobile robot navigation.

4. This method possesses the capability of persistent decision-making, which contributes to the current goals. It employs both the past state information and the present environmental context to generate the activity of a behavior. In the proposed FDES-based approach, the system memory is preserved with fuzzy state vectors and the current environmental context is modeled with fuzzy event matrices. The state vector is updated using both the past state vector and event matrices. Thus the present approach reduces the possibility of activation of

contradictory behaviors in the successive decision cycles. However, the computational complexity of the overall decision process largely depends on the number of states N (or the dimension of a state vector), which imposes a constraint to maintain the system memory (see Section 4.4.1).

5. This method employs FDES in a distributed manner, where separate FDESs are used to model the activity of a behavior. The FDESs associated with a behavior are independent of each other. Thus addition of a new behavior does not require further modifications of the existing FDESs. Hence, the FDES-based activity generation is modular. In Section 4.4.1, it has been shown that the computational complexity of the proposed method is tractable, i.e., it does not increase exponentially with the increase in the number of behaviors. Therefore, the FDES-based approach is also scalable to a large behavior-based system.

6. This method incorporates hierarchical decision-making by adopting multilevel behavioral decomposition. It employs sequential decision tree to implement the hierarchical selection of behaviors. In Section 4.4.2, it has been shown that multilevel behavioral decomposition reduces the overall computational complexity of the decision process. As a result, it helps taking timely response in case of dynamic changes in the observed environment.

7. This method provides the opportunity to incorporate multi-valued logic, which helps implementing deterministic vagueness using human reasoning. As a result this approach is more flexible to model the sensory uncertainty and environmental dynamics as compared to the traditional binary logic. In the following chapter, it will be demonstrated that the FL-based multi-valued logic reduces the possibility of wrong behavior selection as compared to the binary logic and generates less-oscillatory motion commands of the robot.

4.7 Conclusion

In this chapter, a novel behavior modulation technique has been described using FDES. The theoretical development is presented along with mathematical examples. It has been demonstrated that the proposed method integrates several important properties of a behavior coordinator. It combines behavior arbitration and command fusion. It provides adequate means of world states modeling and decision analysis. It accomplishes persistent behavior selection. In addition, it is modular, hierarchical, and multi-valued logic based. However, the proposed method requires tuning several parameters, which is the main challenge in designing this system. In the following chapters, the performance of the FDES-based system will be justified using real world mobile robot applications.

Chapter 5

Mobile Robot Navigation Using Fuzzy Discrete Event System

5.1 Introduction

Autonomous navigation includes both global and local path planning for goal-oriented safe maneuvering in dynamic environments. Global techniques [91] require a complete model of the robot's environment in order to allow off-line computation of the complete trajectory from the starting point to the target point. However, global approaches are not appropriate for dynamically varying environment. On the other hand, local methods [92, 93, 94, 95, 96, 97, 98, 99, 100, 101] require only a fraction of the world model during motion planning. Local approaches emphasize more on real-time obstacle avoidance than creating optimal solutions. Integration of global and local methods allows to react to changes in the environment without suspending the predefined course plan [102, 103, 104, 105]. The method described in [102] defines a collision-free path and is called elastic band. To improve the shape of the path artificial forces are applied and the elastic band continues to deform in response to any

changes in the environment. However, [106] reports that elastic band approach may result in a local minima, where the robot can be trapped within U-shape obstacles. In addition, the algorithm is not suitable for moving obstacles [106]. In [103], a cell decomposed method is used to generate a collision-free path between the initial and goal position, and a potential field collision-avoidance method is used at execution time. This method suggests replanning of the path if an local minima occurs. The method described in [104] uses both global and local planners. The global planner employs a global description of the free space given by the network of minimum potential valleys and it selects a candidate path that is likely to be collision-free. The local planner then modifies the candidate path to avoid collisions and locally optimizes the path length for achieving smooth motion. The algorithm fails to respond properly for complicated shape of obstacles [104] and is designed only for static environments. The algorithm described in [105] uses Electrostatic Potential Fields (EPF) as a global planner and two layer FL system for sensor fusion and real-time obstacle avoidance. The EPF planner is reinvoked every time the environment map is updated. As a result, this algorithm reacts slowly in the presence of unknown moving obstacles.

This chapter presents a novel behavior-based architecture to combine the global and local path planning for mobile robot navigation. It employs *Voronoi* diagram [107] to define a global path, which remains unchanged during the course of navigation. Local motion planning is accomplished using motor schema-based behaviors [78, 41], which are updated online using the knowledge of predefined course plan and locally sensed sensory data. The proposed FDES-based behavior coordinator is employed to autonomously select appropriate behaviors to produce collision-free goal oriented navigation. The method does not assume specific shapes of obstacles and is not restricted to static environments. FDES-based selection of behaviors also eliminates generation of conflicting motion commands and produces local minima free

navigation in presence of U-shape obstacles. The proposed method is also capable of detecting temporal changes in the environment using the measurement of FDES-based controllability, which is used to modulate the robot's velocity in case of moving obstacles.

The rest of the chapter is organized as follows. Section 5.2 describes the navigation architecture. Sections 5.3 and 5.4 present the FDES-based behavior coordinator and its non-fuzzy forms. Sections 5.5-5.9 demonstrate the experimental results and different aspects of the proposed navigation architecture. Finally, Section 5.10 draws the conclusion.

5.2 Navigation architecture

The overall navigation architecture is similar to the behavior-based system shown in Fig.3.1 of Chapter 3. In this architecture, the *Global* module generates a safe path using *Voronoi* diagram. The *Voronoi* vertices between the robot and the target are denoted as subgoals (see Fig.3.2). Once the robot reaches closer to a subgoal point, the subgoal is discarded and the available closest *Voronoi* vertex is redefined as the next subgoal. The *Local* module takes the sensory inputs as well as the safe path information and forms the following motor schemas (or vector behaviors).

- *Go-to-target* (V_1) is used for path optimization, which is a unit vector directed to the second nearest subgoal with respect to the current robot's position (see Fig.5.1).

- *Route-follow* (V_2) is used to follow the safe path, which is a unit vector directed to the nearest subgoal with respect to the current robot's position (see Fig.5.1).

- *Avoid-obstacle* (V_3) is a unit vector that is normal to the direction of the resultant repulsive force obtained from the position vectors of obstacles and it is

directed towards the current orientation of the robot. This behavior employs wall-following approach to avoid collisions. (see Fig.5.1).

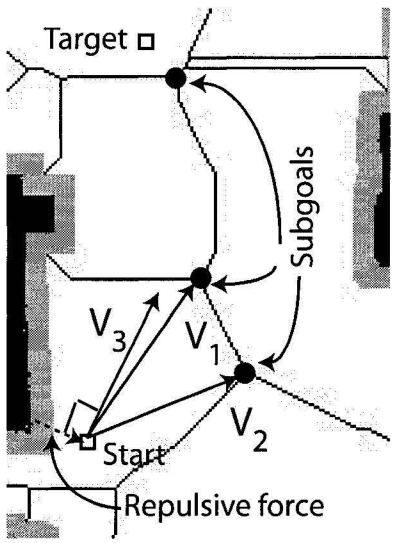


Figure 5.1: Motor schemas

The *Behavior coordination* module generates the modulating weights β_1, β_2, and β_3 and coordinates the behaviors using

$$\mathrm{V} = \beta_1.\mathrm{V}_1 + \beta_2.\mathrm{V}_2 + \beta_3.\mathrm{V}_3 \tag{5.1}$$

where V is the coordinated behavior and $\theta = \angle \mathrm{V}$ is the commanded heading direction.

For performance comparison, four different behavior coordinators are employed:

1. Unmodulated vector summation, i.e., $\beta_i = 1$, $\forall i$
2. FDES-based coordinator,
3. DES-based coordinator, and
4. Behavior arbitration.

The following sections describe formation of different behavior coordinators that generate the modulatory weights β_i.

5.3 FDES coordinator

The FDES-based approach redefines the *go-to-target*, *route-follow* and *avoid-obstacle* behaviors as B_1, B_2, and B_3 respectively. It employs a single level behavioral decomposition for the given navigation task as shown in Fig.5.2. The *go-to-target* behavior

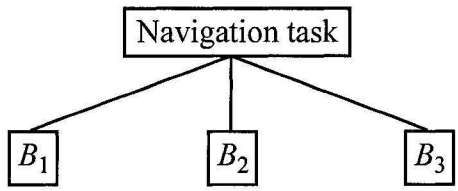


Figure 5.2: Single level decomposition of the navigation task

is given the lowest priority and is defined as

$$B_1 = (A_1, Z_1, F_1)\,.$$

Here, $A_1 = \mathrm{V}_1$ and $Z_1 = \{z_{1j} | j = 1, 2, 3\}$, where

z_{11} = distance to the closest obstacle with respect to current robot's position. Higher values of z_{11} infer higher activity of B_1.

$z_{12} = (D_2 - D_1)$, where D_1 and D_2 are the distances from the current robot's position to the nearest and the second nearest subgoal, respectively. Higher values of z_{12} infer lower activity of B_1.

$z_{13} = \mathrm{abs}(\angle \mathrm{V}_3 - \angle \mathrm{V}_1)$. Higher values of z_{13} infer lower activity of B_1.

F_1 is composed of F_{1j}, $j = 1, 2, 3$ where each FDES has the transition structure shown in Fig.5.3. State 1, 2, and 3 denote *low*, *medium* and *high* activity, respec-

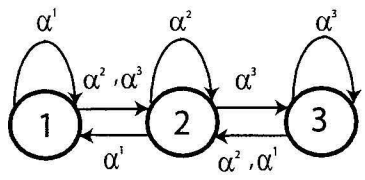


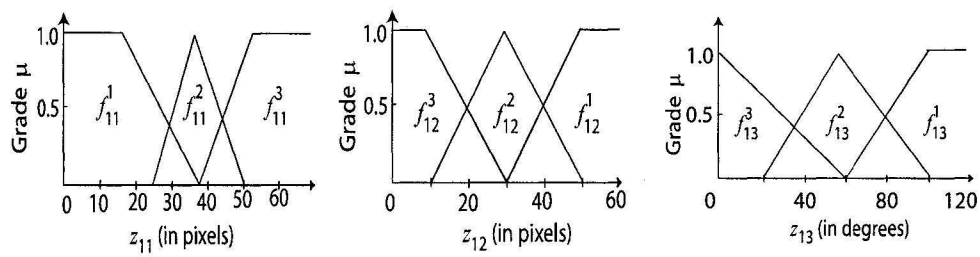
Figure 5.3: Transition structure of FDESs

tively. Event α_{1j}^{e} is determined using (4.11) as below:

$$\alpha_{1j}^{1} = \begin{bmatrix} f_{1j}^{1}(z_{1j}) & 0 & 0 \\ f_{1j}^{1}(z_{1j}) & 0 & 0 \\ 0 & f_{1j}^{1}(z_{1j}) & 0 \end{bmatrix}, \quad \alpha_{1j}^{2} = \begin{bmatrix} 0 & f_{1j}^{2}(z_{1j}) & 0 \\ 0 & f_{1j}^{2}(z_{1j}) & 0 \\ 0 & f_{1j}^{2}(z_{1j}) & 0 \end{bmatrix}, \text{ and}$$

$$\alpha_{1j}^{3} = \begin{bmatrix} 0 & f_{1j}^{3}(z_{1j}) & 0 \\ 0 & 0 & f_{1j}^{3}(z_{1j}) \\ 0 & 0 & f_{1j}^{3}(z_{1j}) \end{bmatrix}$$

where membership functions f_{11}^{e}, f_{12}^{e}, and f_{13}^{e} are defined as shown in Fig.5.4. For



(a) (b) (c)

Figure 5.4: Membership functions

state-based observability and controllability analysis, $L_{ij} = L$ and $W_1 = W_{1j} = W$ are used where L and W are defined as

$$L = \begin{bmatrix} 0 & 0.4 & 0.8 \\ 0.4 & 0 & 0.4 \\ 0.8 & 0.4 & 0 \end{bmatrix}, \quad W = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

The matrices L and W are determined using the procedure described in Section 4.5. Event α_1^e is calculated using (4.12) as

$$\alpha_1^1 = \begin{bmatrix} \gamma_1^1 & 0 & 0 \\ \gamma_1^1 & 0 & 0 \\ 0 & \gamma_1^1 & 0 \end{bmatrix}, \alpha_1^2 = \begin{bmatrix} 0 & \gamma_1^2 & 0 \\ 0 & \gamma_1^2 & 0 \\ 0 & \gamma_1^2 & 0 \end{bmatrix}, \text{ and } \alpha_1^3 = \begin{bmatrix} 0 & \gamma_1^3 & 0 \\ 0 & 0 & \gamma_1^3 \\ 0 & 0 & \gamma_1^3 \end{bmatrix}.$$

Here γ_1^e is determined using (4.13).

The next behavior *Route-follow* is given a higher priority than *go-to-target* and is defined as

$$B_2 = (A_2, Z_2, F_2).$$

Here, $A_2 = \mathrm{V}_2$ and $Z_2 = \{z_{2j} | j = 1, 2, 3\}$, where $z_{21} = z_{11}$, $z_{22} = z_{12}$, and $z_{23} = \mathrm{abs}(\angle \mathrm{V}_3 - \angle \mathrm{V}_2)$. Higher values of z_{21} and z_{22} infer higher activity of B_2, whereas lower values of z_{23} infer higher activity of B_2. The FDES F_2 constitutes F_{2j}, $j = 1, 2, 3$. Hence, F_2 and F_{2j} have the same transition structures as shown in Fig.5.3. Event α_{2j}^e is determined using (4.11) with $f_{21}^e = f_{11}^e$, $f_{22}^e = f_{12}^{(3-e+1)}$, and $f_{23}^e = f_{13}^e$. $L_{2j} = L$ and $W_2 = W_{2j} = W$ are used for observability and controllability analysis. Event α_2^e is calculated using (4.12).

Finally, *avoid-obstacle* behavior is given the highest priority and is defined as

$$B_3 = (A_3, Z_3, F_3).$$

Here, $A_3 = \mathrm{V}_3$ and $Z_3 = \{z_{3j} | j = 1, 2, 3\}$, where $z_{31} = z_{11}$, $z_{32} = z_{13}$, and $z_{33} = z_{23}$. Lower values of z_{31} infer higher activity of B_3, whereas higher values of z_{32} and z_{33} infer higher activity of B_3. The FDES F_3 constitutes F_{3j} having the same transition structures as shown in Fig.5.3. Event α_{3j}^e is determined using (4.11) with $f_{31}^e = f_{11}^{(3-e+1)}$, and both $f_{32}^e, f_{33}^e = f_{13}^{(3-e+1)}$. $L_{3j} = L$ and $W_3 = W_{3j} = W$ are used for observability and controllability analysis. Event α_3^e is calculated using (4.12).

Initial state vectors of all FDESs in B_i are set to [0.1 0.9 0.1]. The grade of membership of *medium* activity is assigned the highest value, which enables a quick

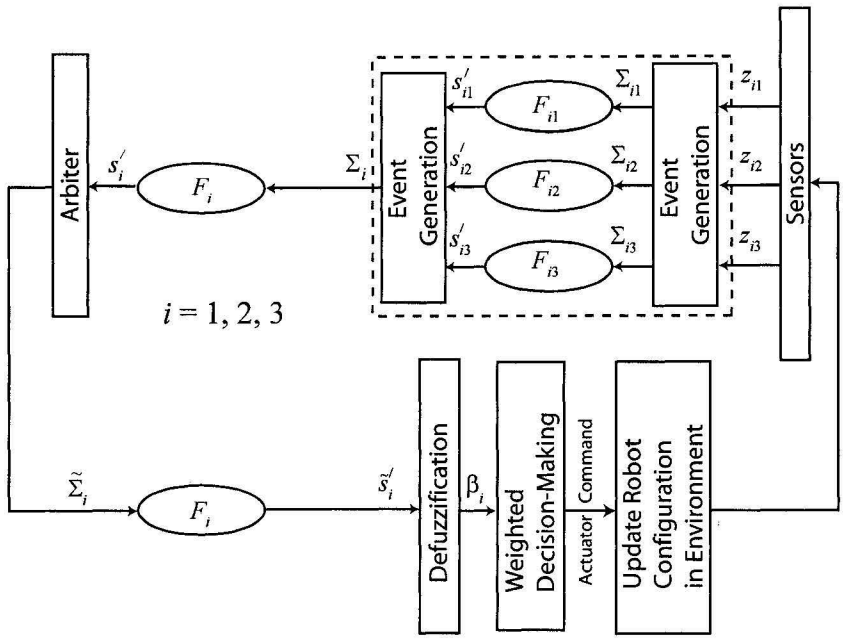


Figure 5.5: Behavior modulation for the navigation task

state-transition from *medium* to *low* or *high* activity depending on the sensory data. Hence, assignment of the highest value to *medium* activity reduces the time delay in reaching the expected activity state. For all F_{ij}, $d_e = 0.8$ and $d_e^c = 0.2$ are used. These value are determined experimentally. It indicates that the sensory information used in an event is 80% reliable. Fig.5.5 shows the behavior modulation technique, which is implemented as described in Section 4.4. The *Arbiter* defines the equivalence relation as

$$B_i \equiv_{co} \hat{B} \text{ if } \text{abs}(\angle A_i - \angle \hat{A}) \leq \delta^o$$

where $\hat{A}$ represents the expected action of the highest priority behavior $\hat{B}$ in maximum activity state (see Section 4.4). The threshold δ is used as a scalar quantity and is set to 15^o.

5.4 DES coordinator and behavior arbitration

DES coordinator is obtained by modifying FDES-based coordinator as mentioned in *Remark 1* (see Section 4.4). Hence, the membership functions shown in Fig.5.4 are changed so $f_{ij}^{e}(z_{ij}) = 1$ if $e = \arg\max_{\acute{e}}\{f_{ij}^{\acute{e}}(z_{ij})\}$, $\acute{e} = 1, 2, 3$; otherwise, $f_{ij}^{e}(z_{ij}) = 0$. Behavior arbitration is obtained using *Remark 2* (see Section 4.4).

5.5 Navigation results

An Active Media Pioneer 3-AT robot quipped with a sonar ring is used in the experiments. The sonar sensors are employed to obtain range measurements to avoid obstacles. At each decision cycle, the robot is controlled by sending a rotational velocity command (Ω) and a translational velocity command (v). For unmodulated coordination, the rotational velocity command Ω is proportional to θ, i.e., $\Omega = \kappa\theta$ deg/s, where $\theta = \angle \mathrm{V}$ deg. κ is set to 1 for the examples presented in this work. The translational velocity command v is adjusted proportional to rotational velocity ($180 - \mathrm{abs}(\Omega)$) and v is measured in mm/s. For unmodulated coordination, the sampling time period (i.e., the length of a decision cycle) is set at $t_p = 50$ ms.

For the FDES-based coordinator and its non-fuzzy forms, the velocity commands are weighed by the average observability (O_a) and controllability (C_a) of F_i, computed using (4.9) and (4.10), respectively. Hence, the velocity commands are modified as

follows.

$$
\begin{aligned}
O_a &= \frac{1}{3}\sum_{i=1}^{3} O_i \\
C_a &= \frac{1}{3}\sum_{i=1}^{3} C_i \\
\Omega &= O_a \times C_a \times \theta \ \text{deg/s} \\
v &= O_a \times C_a \times (180 - \text{abs}(\Omega)) \ \text{mm/s} \\
t_p &= O_a \times C_a \times 50 \ \text{ms}
\end{aligned}
$$

The average observability O_a gives an index of average vagueness of the decision made and the average controllability C_a gives an index of average change of undesired behavioral activity (which is caused by the dynamic changes in the environment). The bottom-line idea is that the robot should move slowly but take samples faster in case of vague decision and dynamic changes in the environment.

The following performance measures are defined in identifying robustness of different coordinators proposed in this work:

1. Average distance to the nearest obstacle D_{obs} is defined as

$$D_{obs} = \frac{1}{K}\sum_{k=1}^{K} D_{obs}(k) \ \text{mm}$$

   where K is the total number of decision cycles and $D_{obs}(k)$ is the distance to the nearest obstacle in the k^{th} decision cycle. Higher values of D_{obs} indicate safer navigation.

2. Total traveled distance D_{tra} (in mm) is expected to be minimum to optimize the traveled distance.

3. Total navigation time t_{nav} (in ms) is expected to be minimum for fast navigation.

4. Total number of collisions COL should be zero for safe navigation.

5. Average rate of change of velocity Δv is defined as

$$\Delta v = \frac{1}{K-1}\sum_{k=2}^{K}\frac{|\Omega(k)-\Omega(k-1)|}{t_p(k-1)} \text{ deg/s}^2$$

where $\Omega(k)$ is the rotational velocity at the k^{th} decision cycle and $t_p(k-1)$ is the length of the $(k-1)^{th}$ decision cycle. Here, only Ω is considered since v varies linearly with Ω and for $v_{t,max} = 180$ mm/s, Δv is numerically the same for both velocities. Lower values of Δv indicate consistent velocity of the robot.

6. Average radius of curvature R_{cur} is defined as

$$\begin{aligned}
R_{cur} &= \frac{1}{K-2}\sum_{k=3}^{K} R_{cur}(k) \text{ mm} \\
R_{cur}(k) &= \frac{\left[(\Delta x_r(k))^2 + (\Delta y_r(k))^2\right]^{\frac{3}{2}}}{|\Delta x_r(k)\Delta^2 y_r(k) - \Delta y_r(k)\Delta^2 x_r(k)|} \\
\Delta x_r(k) &= x_r(k) - x_r(k-1) \\
\Delta y_r(k) &= y_r(k) - y_r(k-1) \\
\Delta^2 x_r(k) &= \Delta x_r(k) - \Delta x_r(k-1) \\
\Delta^2 y_r(k) &= \Delta y_r(k) - \Delta y_r(k-1)
\end{aligned}$$

where $(x_r(k), y_r(k))$ is the robot coordinate with respect to the world map in the k^{th} decision cycle. Higher values of R_{curr} indicate smoother trajectory of navigation.

Using the navigation environment shown in Fig.3.2 three different scenarios have been created.

Case I: A box representing an obstacle is placed on the safe path (Fig.5.6(a)).

Case II: A U-shaped obstacle is placed at the initial position of the robot and a second obstacle is positioned on the way to the target (Fig.5.9(a)).

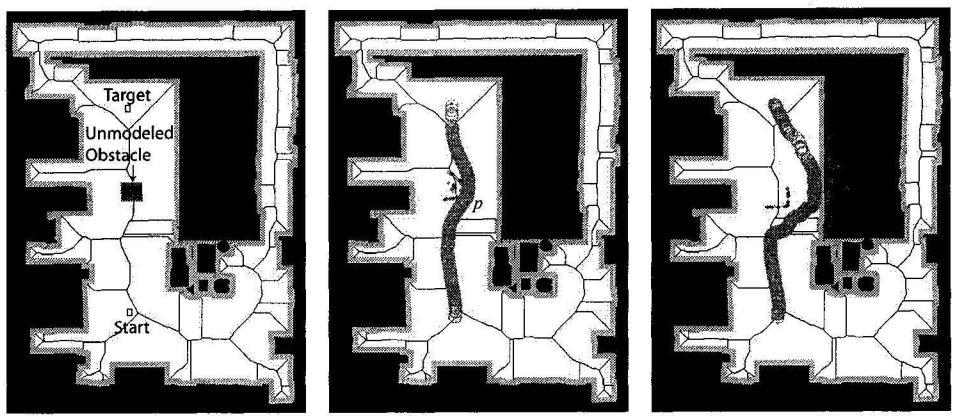
Case III: The obstacles are placed in such a way that the robot is forced to pass through a narrow passage (Fig.5.11(a)).

During navigation the robot updates its current position (i.e., localizes itself) using the gyro-corrected odometry data, which is further adjusted using an experimentally developed uncertainty model of the roobt's odometry (see Section 5.8). In each experiment the robot path is traced and the results are depicted in Fig.5.6-5.12. The performance measures evaluated for each case are summarized in Table 5.1. For the three cases the video clips corresponding to robot navigation using FDES driven coordinator are also provided on a CD attached with this thesis (fdesCaseI.wmv, fdesCaseII.wmv, and fdesCaseIII.wmv).

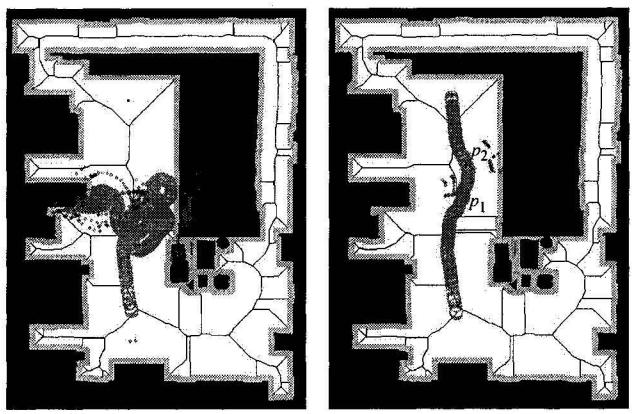
5.6 Results comparison

5.6.1 Unmodulated coordinator

In this case the vectors are weighed equally ($\beta_i = 1$) and the method has no flexibility to suppress *go-to-target* (V_1) and *route-follow* (V_2) behaviors to prioritize *avoid-obstacle* (V_3) behavior. As a result, the system has experienced maximum number of collisions in all three cases. In Case I (Fig.5.6(b)), the robot has collided with the box at point p, and in Cases II and III (Fig.5.9(b) and Fig.5.11(b)), the robot has encountered two collisions at points p_1 and p_2. Consequently, this method produces poor performance in terms of D_{obs}. In addition, with increased environmental complexity this method undergoes higher values of Δv (i.e., inconsistent velocity) and R_{cur} (i.e., irregular robot trajectory). Despite the collisions, the robot can travel faster using this approach with the shortest traveled distance and navigation time (see Table 5.1). In other words, in absence of any obstacles or with no changes in environment, this method is easier to implement and faster in performance.

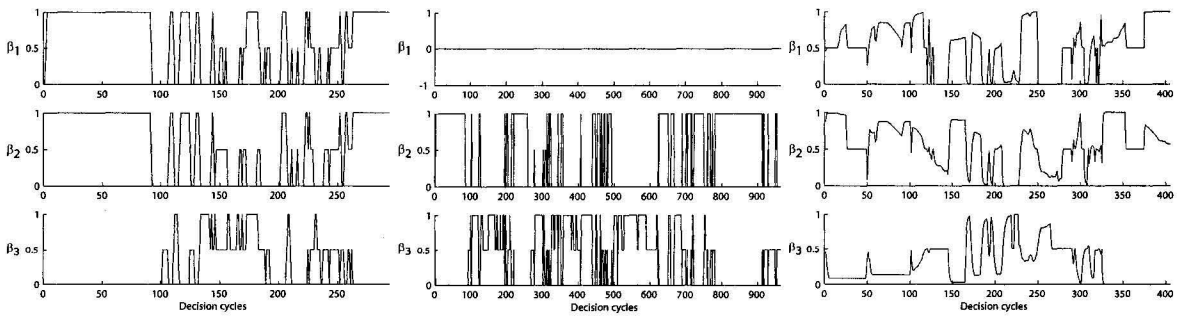


(a) Navigation scenario (b) Unmodulated coordinator (c) DES coordinator



(d) Behavior arbitration (e) FDES coordinator

Figure 5.6: Robot trajectories in case I



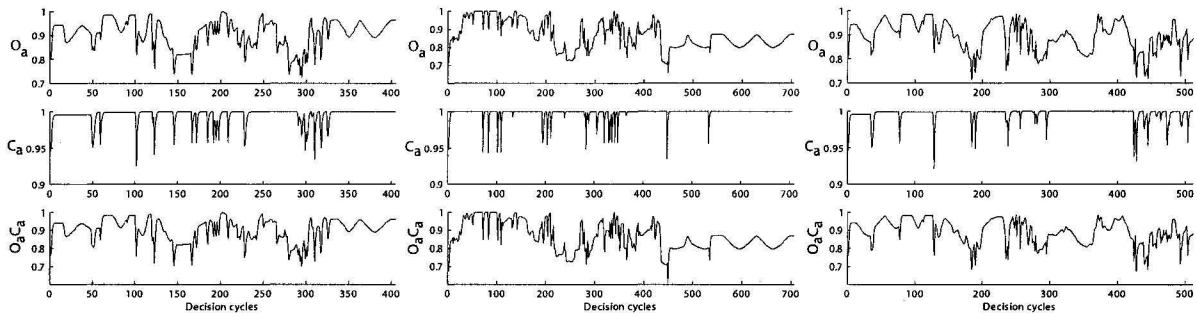
(a) DES coordinator (b) Behavior arbitration (c) FDES coordinator

Figure 5.7: Weights generated by different coordinators in case I

Case I						
Coordinator type	D_{obs} (mm)	D_{tra} (mm)	t_{nav} (ms)	COL	Δv (deg/s^2)	R_{cur} (mm)
Unmodulated	2388	8204	43953	1	34	9×10^{11}
DES	2599	9320	52865	0	78	7×10^{11}
Arbitration	Fail	Fail	Fail	Fail	Fail	Fail
FDES	2017	8437	81930	0	45	9×10^{11}
Case II						
Unmodulated	878	8519	68530	2	122	3×10^{11}
DES	1277	26347	153441	0	167	1×10^{11}
Arbitration	Fail	Fail	Fail	Fail	Fail	Fail
FDES	1554	10591	124601	0	0 97	5×10^{12}
Case III						
Unmodulated	2671	8730	49617	2	119	1×10^{13}
DES	Fail	Fail	Fail	Fail	Fail	Fail
Arbitration	Fail	Fail	Fail	Fail	Fail	Fail
FDES	2948	9382	86598	0	77	4×10^{12}

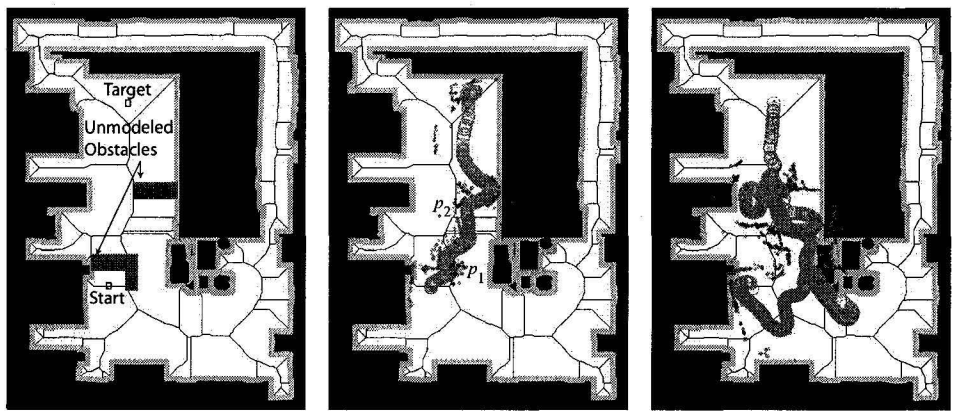
'Fail': failed to reach the target

Table 5.1: Performance measures

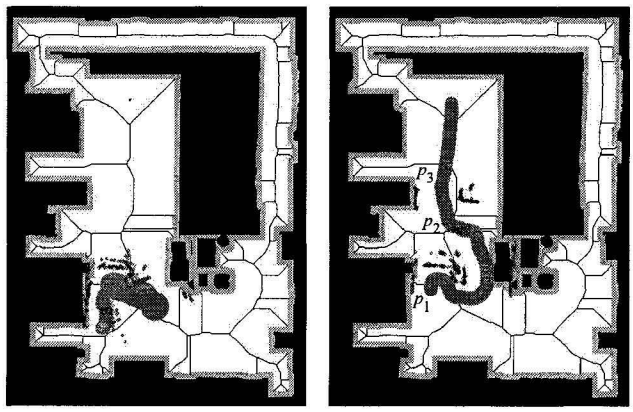


(a) Case I (b) Case II (c) Case III

Figure 5.8: Average observability and controllability

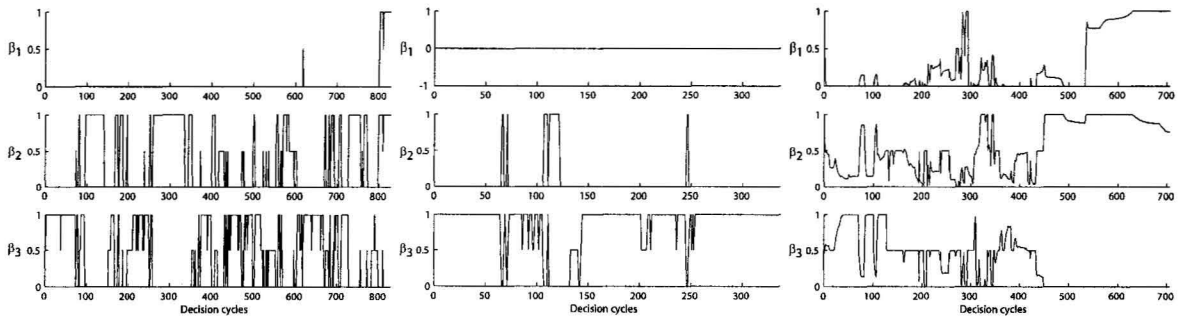


(a) Navigation scenario (b) Unmodulated coordinator (c) DES coordinator



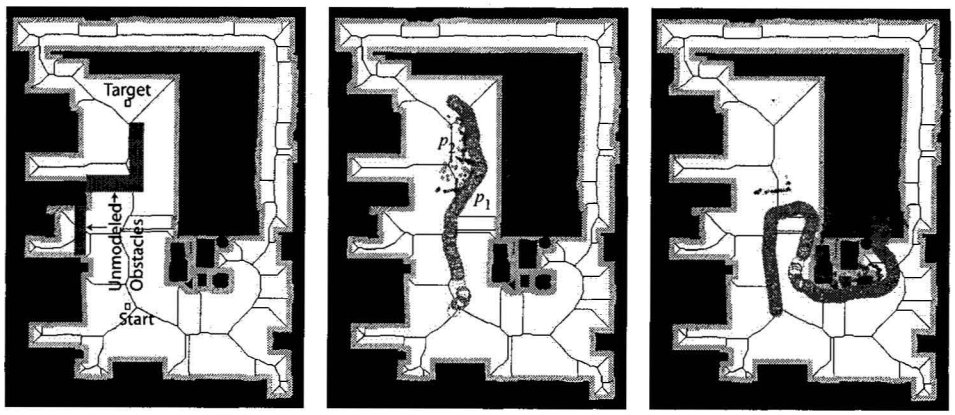
(d) Behavior arbitration (e) FDES coordinator

Figure 5.9: Robot trajectories in case II

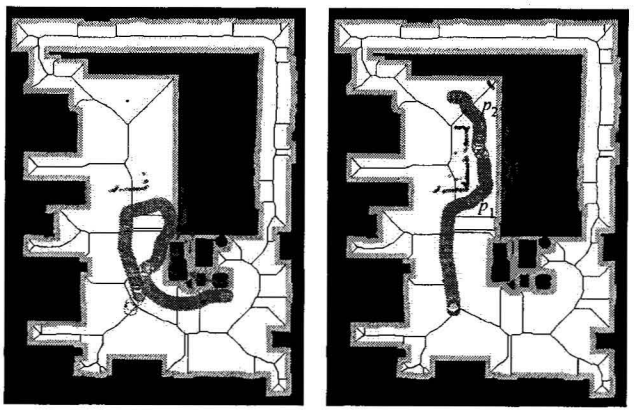


(a) DES coordinator (b) Behavior arbitration (c) FDES coordinator

Figure 5.10: Weights generated by different coordinators in case II

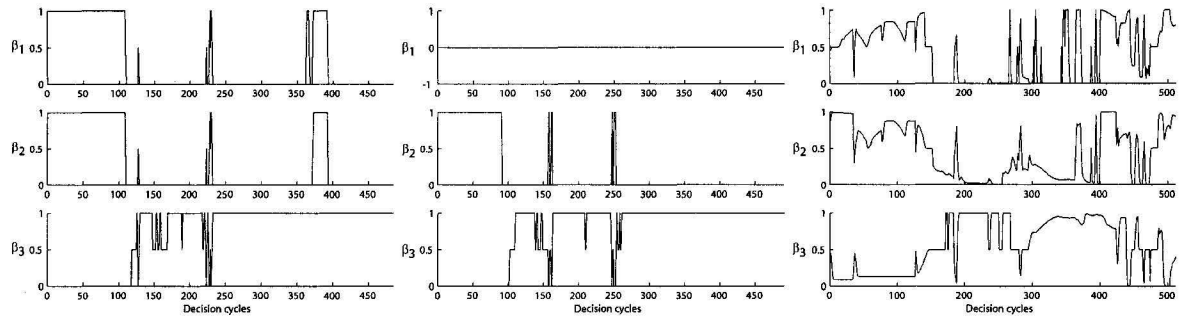


(a) Navigation scenario (b) Unmodulated coordinator (c) DES coordinator



(d) Behavior arbitration (e) FDES coordinator

Figure 5.11: Robot trajectories in case III



(a) DES coordinator (b) Behavior arbitration (c) FDES coordinator

Figure 5.12: Weights generated by different coordinators in case III

5.6.2 DES coordinator

Table 5.1 demonstrates that all the performance measures associated with the DES-based coordinator degrades compared to the FDES-based coordinator as the environmental complexity increases in Cases II and III. This indicates inconsistent navigational performance of the DES-based coordinator in complex environments. Moreover, in Case III, the system has failed to complete the navigational task (Fig.5.11(c)). In this coordination technique, the activity states are mutually exclusive, i.e., the activity of a behavior is either *low*, *medium* or *high*. Hence, the modulating factor can take only three possible values, i.e., $\beta_i \in \{0, 0.5, 1\}$ (see Fig.5.7(a), 5.10(a), and 5.12(a)). The use of discrete states (or equivalently hard boundaries in membership functions) results in frequent switching between behaviors and produces a larger value of Δv. According to the transition structure shown in Fig.5.3, this approach always produces $O_a = 1$ and $C_a = 1$. This indicates that the system has no velocity modulation.

5.6.3 Behavior arbitration

This technique has failed to accomplish any navigational task in the experiments (see Fig.5.6(d), 5.9(d), and 5.11(d)). This is because, once the robot finds an obstacle within *low* distance, the control of the robot is completely given to *obstacle-avoidance* behavior (see Fig.5.7(b), 5.10(b), and 5.12(b) where β_3 is dominating). As a result, the robot starts wall-following without considering the expected actions of *go-to-target* and *route-follow* behaviors, and this may cause a failure to reach the target.

5.6.4 FDES coordinator

The robot has successfully completed the navigational tasks using this method and has demonstrated consistent performances against environmental complexities (see Fig.5.6(e), 5.9(e), and 5.11(e)). The effect of environmental dynamics on performance measures are minimal as compared to other coordinators (see Table 5.1). In this case, the velocities and sampling frequency are modulated by O_a and C_a. Therefore, the robot has the ability to slow down in case of changing environments in order to assist safe navigation. On the other hand, this can result in slower navigation. However, the modulation through O_a and C_a helps to increase the sampling frequency. As an example, in Case I (Fig.5.6(e)), the product O_aC_a becomes considerably lower around the 150^{th} and 300^{th} decision cycles (see Fig.5.8(a)). This refers to the cases where the robot is either approaching or leaving the unmodeled box placed within the navigational path. In Case II, the robot has detected three significant changes in the environment, while using the product O_aC_a at the 1^{st}, 250^{th}, and 450^{th} decision cycles (see Fig.5.8(b)). The changes are corresponding to points p_1, p_2, and p_3 in Fig.5.9(e). In Case III, the robot has detected two significant changes in the environment where the values of O_aC_a at the 180^{th} and 280^{th} decision cycles (see Fig.5.8(c)) correspond to the points p_1 and p_2, respectively in Fig.5.11(e). This particular feature is established

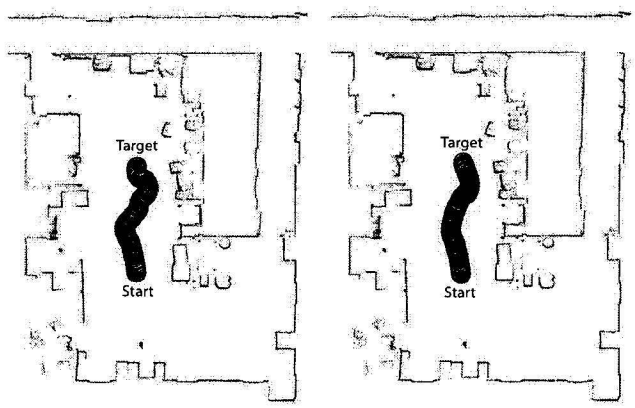
through the state-based observability and controllability. This is the key advantage of using FDES systems over general FL-based navigational systems. Figures shown in 5.7(c), 5.10(c), and 5.12(c) describe modulating weights β_i. Abrupt changes of β_i are automatically restricted by the transition structure of F_i and this feature enables to produce consistent velocity and smooth trajectory.

5.7 Reliability

Reliability includes reactivity, error recovery, and uncertainty handling. Reactivity provides robustness against unpredictable environmental changes. Table 5.1 reveals that the FDES-based approach provides consistent values of the performance measures compared to other coordinators when the environmental complexity increases (e.g., in Cases II and III). As an instance, only the FDES-based approach provides 100% successful navigation without collisions in the presence of unpredictable obstacles.

Error recovery includes continuous monitoring of the behavioral performance and taking corrective actions if necessary. The proposed FDES-based approach accommodates monitoring of behavioral performance in terms of state-based observability and controllability measures. They describe the uncertainty in sensory information and environmental dynamics. In order to provide corrective actions, velocity and sampling-frequency modulations are used to produce slower speed and faster sampling in changing environments. Velocity modulation prevents sharp turning of the robot in presence of dynamic obstacles. On the other hand, higher sampling rate enables faster perceptions. As an instance, a simple experiment has been performed where an obstacle periodically comes closer and moves away from the robot. Without velocity and frequency modulation, the robot has experienced oscillation near the obstacle (see Fig.5.13(a)). However, application of velocity and frequency mod-

ulation can slow down the robot's movement near obstacles to provide enough time for analyzing the updated sensory information. This enables to avoid occurrence of oscillations (see Fig.5.13(b)).



(a) Without modulation (b) With modulation

Figure 5.13: Effect of velocity and frequency modulation

Uncertainty handling employs predictions using approximate reasoning to process faulty sensory information. This capability is accomplished in the proposed FDES-based approach using the graded membership (or soft boundary) of FL. For example, in Section 5.6.2, it has been shown that the use of hard boundaries for event generation in the DES-based approach outcomes frequent behavioral switching (e.g., see Fig.5.10(a)), which is reduced in the FDES-based approach (see Fig.5.7(c)).

5.8 Robot localization

This section briefly discusses the effect of online localization for the FDES-based robot navigation. A large-scale map is used for the experiment having a physical dimension 35.1 m $\times$ 23.76 m and 1755 pix $\times$ 1188 pix in image plane (see Fig.5.14). The existing robotic applications use SLAM techniques (e.g., Kalman filter based methods [108, 109], particle filer based methods [110, 111], or a soft computing based

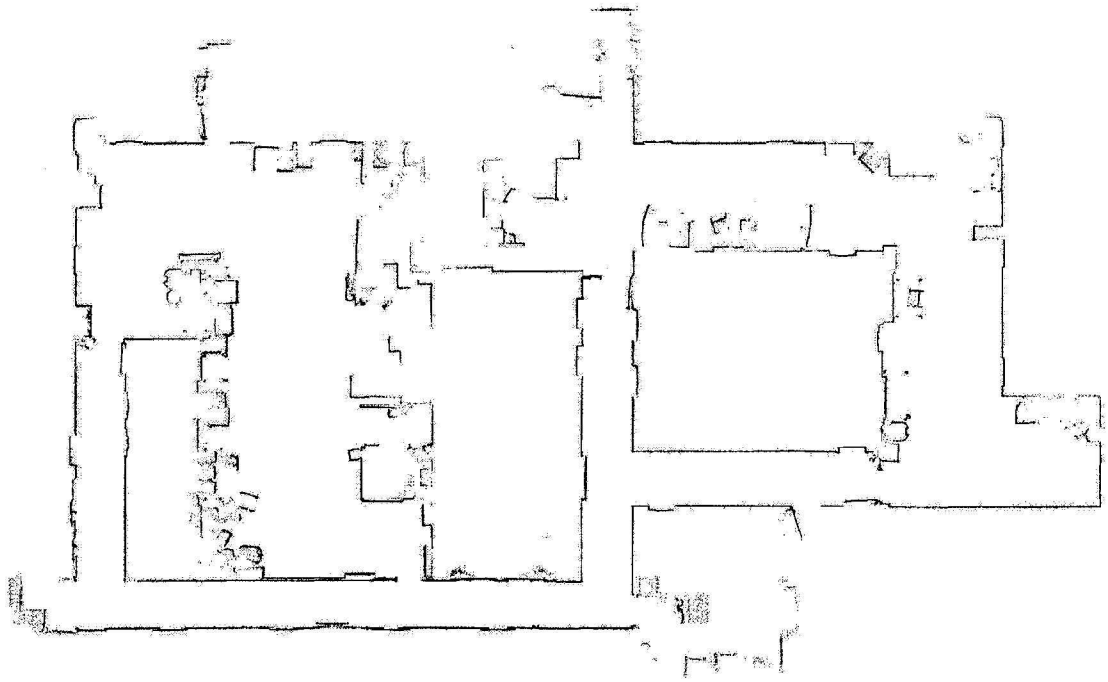


Figure 5.14: Large-scale environment for localization testing

technique [112]) for robot localization, where the robot determines its current position while minimizing uncertainties in the sensory information and then updates the map using the sensory data to accommodate any environmental changes. An online localization technique using Genetic Algorithm (GA) based scan matching (adapted from [112]) with respect to a given map has been implemented for the experiment. In the experiment, the localization module is invoked when the robot changes its subgoal or the traveled distance exceeds 2 m with respect to the previously localized position. The threshold distance 2 m is chosen based on experimental observations. Fig.5.15 shows the navigation result using online robot localization, where the localization module is invoked at the positions denoted as $p_1, \ldots, p_7$. At each invocation of the

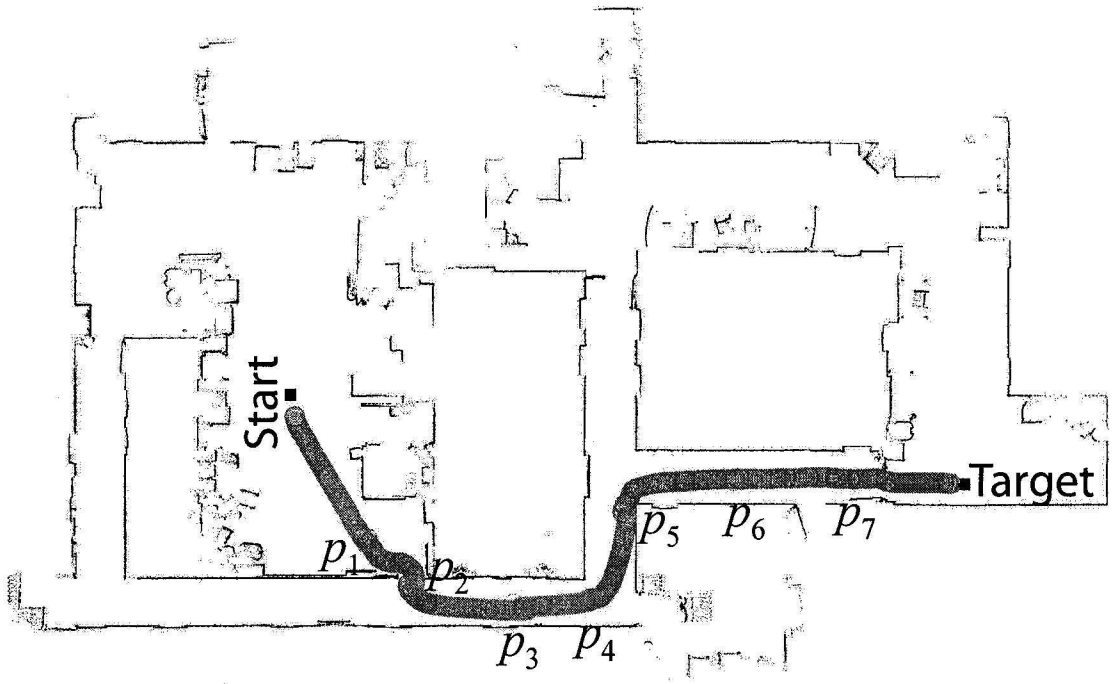


Figure 5.15: Navigation with online localization

localization module, the errors are estimated as follows.

$$\begin{aligned} xE &= 0.5 \times x\acute{E} + 0.5 \times (x_{odo} - x_{loc}) \\ yE &= 0.5 \times y\acute{E} + 0.5 \times (y_{odo} - y_{loc}) \\ \theta E &= 0.5 \times \theta\acute{E} + 0.5 \times (\theta_{odo} - \theta_{loc}) \end{aligned}$$

where xE and yE are the translational errors in x and y directions, and θE denotes the orientational error of the robot. $[x\acute{E}, y\acute{E}, \theta\acute{E}]^{\mathrm{T}}$ is the previously estimated odometry error, $[x_{odo}, y_{odo}, \theta_{odo}]^{\mathrm{T}}$ is the current robot's position estimated using odometry, and $[x_{loc}, y_{loc}, \theta_{loc}]^{\mathrm{T}}$ is the current robot's position estimated using the localization module. Between two successive invocations of the localization module, the robot configuration

in each decision cycle is corrected using

$$\begin{aligned} \hat{x}_{odo} &= x_{odo} - xE \\ \hat{y}_{odo} &= y_{odo} - yE \\ \hat{\theta}_{odo} &= \theta_{odo} - \theta E \end{aligned}$$

where $[\hat{x}_{odo}, \hat{y}_{odo}, \hat{\theta}_{odo}]^{\mathrm{T}}$ is the corrected robot's position. Fig.5.16 shows the navigation result without using localization module. Because of the accumulated localization errors, the robot fails to detect the subgoal at position p. The robot starts oscillating at that point and collides with the wall. The navigation result shown in Fig.5.16 infers

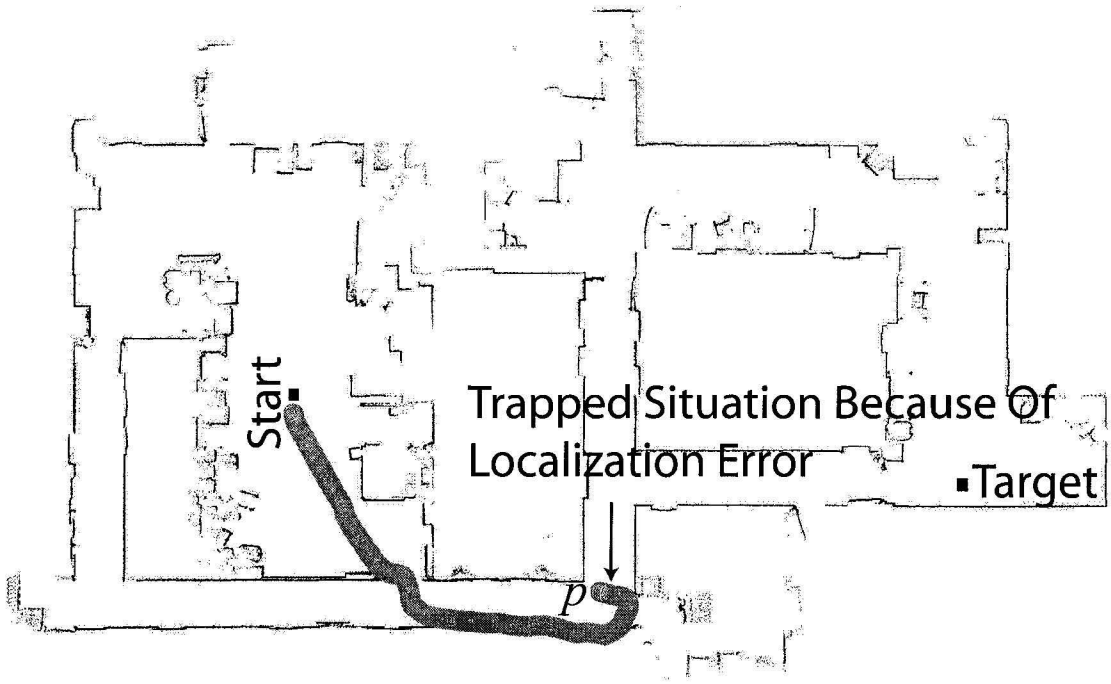


Figure 5.16: Navigation without online localization

that accumulated odometry error increases uncertainty in subgoal detection leading to wrong action selection for *go-to-target* and *route-follow* behaviors. This causes oscillatory velocity commands of the robot. In the navigation experiments presented in this work, this problem has been partially compensated by increasing the radius

of uncertainty of a subgoal. To model the radius of uncertainty ($\sqrt{xE^2 + yE^2}$), 25 observations are taken for different goal positions, which are further approximated using least-square method (see Fig.5.17). While navigating, the robot automatically

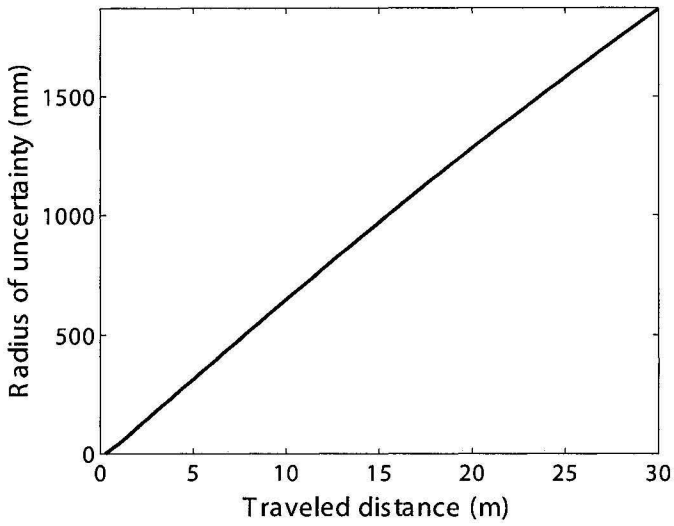


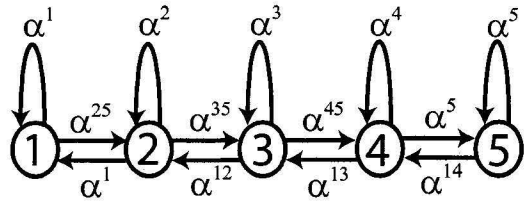
Figure 5.17: The experimentally developed radius of uncertainty

changes the radius of uncertainty of a subgoal with respect to its traveled distance using the approximated model. A subgoal is changed when the robot reaches within its radius of uncertainty. Although this approach provides satisfactory results for small-scale navigation environments, it does not provide adequate means of robot localization in large-scale environments, where an online robot localization technique must be incorporated.

5.9 Issues related to transition structure

The proposed approach provides the opportunity to control the response time of the expected action of a behavior using appropriate state transition structures of a FDES. The higher number of states provides more flexible control. For example, Fig.5.18 shows two transition structures, which consist of five states. The state diagram shown

in Fig.5.18(a) introduces maximum amount of delay to activate (or deactivate) an action of a behavior, where only the successive states are accessible from each other. In the state diagram depicted in Fig.5.18(b), higher states are accessible from each

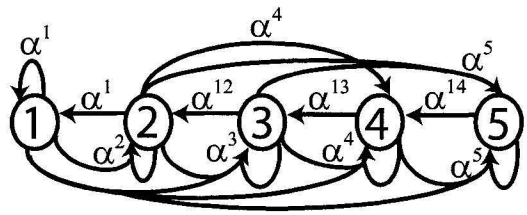


$\alpha^{25}=\{\alpha^2,\alpha^3,\alpha^4,\alpha^5\}$ $\alpha^{14}=\{\alpha^1,\alpha^2,\alpha^3,\alpha^4\}$

$\alpha^{35}=\{\alpha^3,\alpha^4,\alpha^5\}$ $\alpha^{13}=\{\alpha^1,\alpha^2,\alpha^3\}$

$\alpha^{45}=\{\alpha^4,\alpha^5\}$ $\alpha^{12}=\{\alpha^1,\alpha^2\}$

(a)



$\alpha^{14}=\{\alpha^1,\alpha^2,\alpha^3,\alpha^4\}$

$\alpha^{13}=\{\alpha^1,\alpha^2,\alpha^3\}$

$\alpha^{12}=\{\alpha^1,\alpha^2\}$

(b)

Figure 5.18: Controlling behavior activation using state diagrams

of the lower states. This indicates the fastest activation of an action of a behavior. On the other hand, lower states are only accessible from its immediate higher state. This indicates smooth deactivation of a behavior and its associated action. Thus, the proposed method provides the flexibility for controlling the activation (or deactivation) time of a behavior and its associated action depending on the problem at hand. As an illustration, a security robot uses a subgoal-based predefined path to navigate

a building floor. A change in subgoal exhibits a possibility of sudden change in the robot's heading direction. Hence, it requires slower activation of the path-following behavior to generate a smooth trajectory. The robot also needs to track an intruder. This requires quick responses to the intruder's movement. To cope with this situation, the robot can employ the state diagram shown in Fig.5.18(a) for the path following behavior, whereas the state diagram shown in Fig.5.18(b) can be employed for the intruder-tracking behavior.

Although higher number of states provides flexibility in behavior activation, it imposes a constraint on the overall computational complexity. In Section 4.4.1, it has been shown that time complexity of the decision process is $\bigcirc((2M + J)N^3)$, where N is the number of states in a FDES. As a result, there exists a trade-off between the activation time of a behavior and the time complexity of the decision process.

5.10 Conclusion

This chapter presents an application of FDES for mobile robot navigation to validate the performance of the proposed method. Navigation results are shown for four behavior coordinators. It has been observed that the performance measures of the FDES-based system are unaffected even under changing or complex environments. The FDES-based system produces collision free navigation in the experiments. Unmodulated vector summation is prone to collision in the presence of unknown obstacles. The DES-based coordinator and behavior arbitration produce oscillation in behavior selection. Furthermore, both of them suffer the starvation problem due to selection of the same behavior for several decision cycles. The hard boundaries selected for generating event matrices in the DES-based coordinator result in oscillations. The state-based observability and controllability phenomenons in FDES make it possible to produce modulated velocity and sampling frequency. This allows the

system to produce safe navigation. Lower values of observability and controllability result in lower velocity and higher sampling frequency.

The following chapter will demonstrate another application of FDES to control a box-pulling robot. This application will illustrate the multilevel behavior decomposition feature of the proposed method.

Chapter 6

Behavior-based Control of a Box-pulling Mobile Robot Using Fuzzy Discrete Event System

6.1 Introduction

Object manipulation is a common application of robotics. Moving an object from one place to another is the simplest kind of object manipulation, which is reported by many researchers in the form of an object-pushing operation by mobile robots. Chen *et al.* [113] consider the situation where a robot pushes objects off its path. Takagi *et al.* [114] proposed an algorithm for a mobile robot to push a box using the knowledge in the form of *If-then* rule-bases. The rules were constructed using hard boundaries (boolean membership) over input variables, which causes an uneven control surface leading to an abrupt change in robot motion. Okawa and Yokoyama [115] applied the classical PID feedback control theory for pushing operation, while Okawa and Aoki [116] used fuzzy control. Okawa and Aoki claimed that fuzzy control shows bet-

ter performance as compared to boolean membership based approaches [114]. Their method replaces the boolean membership with graded membership to obtain a continuous control surface over the input variables. Verma *et al.* [117] described an approach using off-board, environment embedded sensor network to help the pusher robot in detecting and pushing objects. However, this method imposes a restriction that a large environment requires a large number of environment-embedded sensors, which further leads to the problem of efficient sensor selection and sensor fusion. Emery and Balch [118] proposed a behavior-based approach where the pushing task is comprised of several sub-skills called behaviors. The behaviors include: *scan* for target object detection, *go-to-target* for reaching the final goal position, *swirl-obstacles* for avoiding obstacles, *dock* for aligning the robot with the object with respect to the goal position, and *push* for pushing the object to the goal location. This approach employed perceptual sequencing method [36] for appropriate behavior activation. Emery and Balch claimed that the classical path-and-push planning approaches (e.g., [119, 120]) are computationally expensive and have addressed only static environments. However, the method described in [118] suffers from the drawback of being trapped in a potential well or box canyon as it does not incorporate global motion planning that uses prior knowledge of the environment. The perceptual sequencing method used in [118] employs finite state machine for behavior activation. Here, a group of behaviors is activated when an appropriate event is generated using the locally sensed sensory data. The event generation process uses binary thresholding of sensory data, which may generate inappropriate event in presence of noisy and uncertain sensory data. The activated behaviors are combined using predefined weighed vector summation, where the weights are predefined. The fixed weights may result in inappropriate robot heading for operation in dynamic environments.

The main challenges in object-pushing operation by a single robot is in the align-

ment and application of the pushing force in such a way that the object will move without making any rotation. Any misalignment of the external force against the center of mass causes to generate a rotational moment leading to an unexpected rotation of the object. This problem is partially eliminated using multi-robot object-pushing approaches [121], where a group of robots work together to push an object to a goal position. Although multi-robot environment improves the pushing task, it imposes the requirement of efficient multi-robot coordination, which in turn raises the issue of information sharing among the robots for path-and-push planning. Dynamic environment is another constraint for path-and-push planning in multi-robot environment [121].

This work focuses on single robot object manipulation. A variant of the object-pushing operation is proposed, namely object-pulling operation where the pushing operation is replaced with the pulling operation. Fetching an object using pulling operation is faster than pushing operation since the task of correct alignment of the external force is no longer required in pulling operation. However, this advantage comes at the price of the assumption that both the object and the robot are equipped with a suitable anchor system enabling the pulling task. This work also attempts to combine global motion planning with local motion planning for the integration of prior knowledge of the environment and the locally sensed information. In the present work, Fuzzy Discrete Event System (FDES) is used for motion-and-pulling operation. It combines fuzzy logic based robust sensory handling [116] and state machine based behavior sequencing [118]. The proposed FDES-based technique is more tolerant to noisy sensory data compare to [118]. Moreover, unlike the perceptual sequencing method used in [118], the proposed method provides an opportunity to generate online weights for the weighed vector summation of the activated behaviors. Thus the proposed method provides the flexibility of context-dependent coordination of

activated behaviors.

Section 6.2 outlines the behavior-based path-and-pulling problem and Section 6.3 describes the proposed FDES-based behavior coordination mechanism for the object manipulation task. Section 6.5 shows the experimental results that compares the proposed approach and the conventional perceptual sequencing method described in Section 6.4. Finally, Section 6.6 draws the conclusion.

6.2 Behavior-based path-and-pulling planning

In this architecture, the robot, first, anchors itself with the object and then performs subgoal-based navigation to reach the goal location (refer to Section 3.2). The overall planning architecture is similar to the behavior-based navigation architecture shown in Fig.3.1. In this architecture, the *Global* module generates a safe path and defines the subgoals between the initial position of the robot to the goal location (see Fig.3.2). The *Local* module takes the sensory inputs as well as the safe path information to form a set of motor schemas (or vector behaviors). The *Behavior coordination* module assesses the current environment using sensory data and selects appropriate motor schemas that assist in performing the navigation and pulling task. The motor schemas are described as follows:

- *Go-to-goal* (V_1) is a unit vector directed to the nearest subgoal with respect to the current robot position. Once the anchoring process is accomplished, this schema is activated to guide the robot to the goal location.

- *Anchor-object* (V_2) is directed to the anchor attached with the object. However, this schema becomes active only when the robot reaches very close to the anchor. After that it will remain active for a predefined duration to accomplish the anchoring process. In the present work, the anchoring process is demonstrated

using a fastening tape consisting of a strip of nylon with a surface of minute hooks (attached with the object) that fasten to a corresponding strip (attached with the robot) with a surface of uncut pile.

- *Go-to-anchor* (V_3) is a unit vector directed to the anchor (a black fastening tape) attached with the object. The anchor is detected using the pan-tilt camera. This schema is selected to reach the anchor attached to the object.

- *Swirl-object* (V_4) is a unit vector perpendicular to V_5, which is directed towards the anchor-system attached to the object. If the anchor-system is not visible, V_4 will be directed towards the current robot orientation. This schema is used to find the anchor attached to the object.

- *Go-to-object* (V_5) is a unit vector directed to the centroid of the target object with respect to the current robot position. A box rapped in red paper is used as the target object and is detected using a Bumblebee camera mounted on a pan-tilt unit. The image-centroid of the box is used to form V_5. This schema is selected to reach the object. It is assumed that at the start of navigation, the object is always within the field-of-view of the camera. If the object is occluded with dynamic obstacles, the robot uses the previously calculated V_5 until the object reappears in its field of view.

- *Swirl-obstacle* (V_6) is a unit vector perpendicular to the resultant repulsive force generated by obstacles. V_6 is directed towards the target object. If the target object is anchored, it is directed towards the nearest subgoal. This schema is activated in presence of obstacles to avoid collisions.

6.3 FDES-based planning of path-and-pulling

The FDES-based behavior coordinator is implemented using a multilevel behavioral decomposition as shown in Fig.6.1. The given *navigation and pulling* task is divided

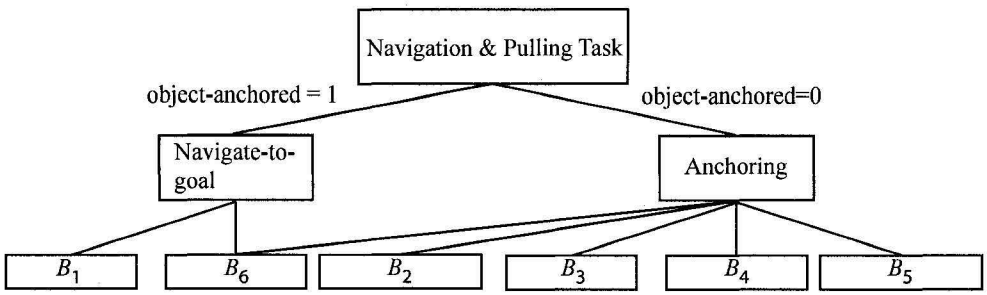


Figure 6.1: Multilevel behavioral decomposition for the box-pulling task

into two abstract behaviors: *navigate-to-goal* and *anchoring*. Activations of these behaviors are explicitly distinguished by a discrete event $object - anchored$, which is either 0 or 1. The *anchoring* behavior remains active until the object is anchored with the robot (i.e., $object - anchored = 1$). The *navigate-to-goal* behavior is then activated to reach the target location. The primitive behaviors $B_1, \ldots, B_6$ are used to implement the abstract behaviors. Behavior B_6 used for obstacle avoidance is common in both of the high level behaviors since each of them requires dynamic obstacle avoidance to successfully accomplish their goals.

The behaviors $B_i, i = 1, \ldots, 6$ are defined using the formalisms presented in Chapter 5. Each FDES used in the definition of a behavior has the same transition structure as shown in Fig.6.2. State 1, 2, and 3 denote *low*, *medium* and *high* activity, respectively. The events α_{ij}^e and α_i^e of each FDES are determined using (4.11) and (4.12).

Go-to-goal has the lowest priority order, i.e. 1 and is defined as

$$B_1 = (A_1, Z_1, F_1)\,.$$

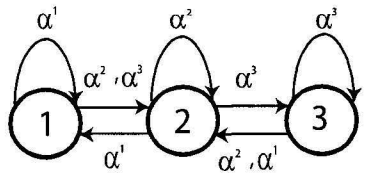
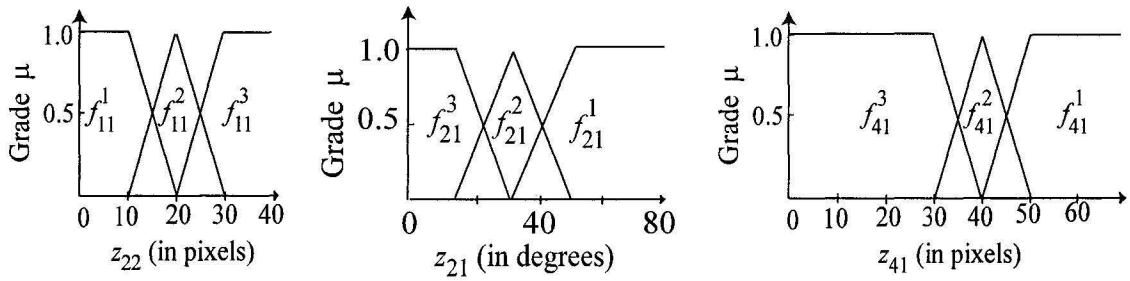


Figure 6.2: Transition structure of FDESs

Here, $A_1 = \mathrm{V}_1$ and $Z_1 = \{z_{1j} | j = 1\}$, where z_{11}=distance to the nearest subgoal. This behavior is weighed higher when the target-subgoal is away (i.e., z_{11} is high). The FDES F_1 constitutes F_{11} that uses MFs f_{11}^e (see Fig.6.3(a)) for event generation.



(a) MFs for z_{11} (b) MFs for z_{21} (c) MFs for z_{41}

Figure 6.3: Membership functions

Anchor-object has the priority order 2 and is defined as

$$B_2 = (A_2, Z_2, F_2)\,.$$

Here, $A_2 = \mathrm{V}_2$ and $Z_2 = \{z_{2j} | j = 1, 2\}$, where $z_{21} = \mathrm{abs}(\angle \mathrm{V}_3 - \angle \mathrm{V}_5)$ and z_{22} = distance to the anchor with respect to current robot position. Both the vision system (Bublebee camera mounted on a pan-tilt unit) and the range sensor (sonar) are used to determine z_{22}. First, the orientation of the anchor is determined using the vision system and then the available range data is correlated with the orientation to obtain z_{22}. *Anchor-object* becomes significant when the robot is very close and aligned with

the anchor attached to the object (i.e., z_{21} and z_{22} are low). This behavior occurs only one time, and is activated when it reaches the highest activity. The corresponding decision cycle is extended until the robot anchors itself with the object. This behavior maintains the status of the anchoring process. It assigns $object - anchored = 1$ after executing its motion commands. The FDES F_3 constitutes F_{3j} that uses MFs f^e_{21} as shown in Fig.6.3(b) and $f^e_{22} = f^{(3-e+1)}_{11}$.

Go-to-anchor has the priority order 3 and is defined as

$$B_3 = (A_3, Z_3, F_3)\,.$$

Here, $A_3 = \mathrm{V}_3$ and $Z_3 = \{z_{3j}|j = 1\}$, where $z_{31} = z_{21}$. *Go-to-anchor* is assigned higher activity when the robot is aligned with the anchor attached to the object (i.e., z_{31} is low). The FDES F_3 constitutes F_{3j} that uses MFs $f^e_{31} = f^e_{21}$.

Swirl-object has the priority order 4 and is defined as

$$B_4 = (A_4, Z_4, F_4)\,.$$

Here, $A_4 = \mathrm{V}_4$ and $Z_4 = \{z_{4j}|j = 1, 2\}$, where z_{41}= distance to the object with respect to current robot position and $z_{42} = z_{21}$. The input z_{41} is determined using the similar approach employed for z_{22}. *Swirl-object* is given higher importance when the target object is close to the robot (i.e., z_{41} is low) and is assigned lower activity when the robot is aligned with the anchor attached to the object (i.e., z_{42} is low, which indicates that *go-to-object* and *go-to-anchor* coincide with each other). The FDES F_4 constitutes F_{4j} that uses MFs f^e_{41} as shown in Fig.6.3(c) and $f^e_{42} = f^{(3-e+1)}_{31}$.

Go-to-object has the priority order 5 and is defined as

$$B_5 = (A_5, Z_5, F_5)\,.$$

Here, $A_5 = \mathrm{V}_5$ and $Z_5 = \{z_{5j}|j = 1\}$, where $z_{51} = z_{41}$. Higher values of z_{51} infer higher activity of B_5. The FDES F_5 constitutes F_{5j} that uses MFs $f^e_{51} = f^{(3-e+1)}_{41}$.

Swirl-obstacle is given the highest priority 6 and is defined as

$$B_6 = (A_6, Z_6, F_6)\,.$$

Here, $A_6 = \mathrm{V}_6$ and $Z_6 = \{z_{6j} | j = 1, 2\}$, where z_{61} = distance to the nearest obstacle. Input z_{62} differs depending on the anchoring status of the robot. If $object - anchored = 1$, $z_{62} = \mathrm{abs}(\angle \mathrm{V}_1 - (\angle \mathrm{V}_6 + 90^o))$; otherwise $z_{62} = \mathrm{abs}(\angle \mathrm{V}_5 - (\angle \mathrm{V}_6 + 90^o))$. The angle $(\angle \mathrm{V}_6 + 90^o)$ denotes the direction of the resultant repulsive force generated by obstacles. *Swirl-obstacle* is given higher importance when obstacles are closely located to the robot (i.e., z_{61} is low) and the direction of the repulsive force does not coincide with that of *go-to-goal* or *go-to-object* (i.e., z_{62} is high). F_6 is composed of F_{6j} that uses MFs $f^e_{61} = f^e_{41}$ and $f^e_{62} = f^{(3-e+1)}_{31}$.

Initial state vectors of all FDESs in B_i are set to [0.1 0.9 0.1]. The grade of membership of *medium* activity is assigned a higher value that reduces initial delay to reach the expected activity state. The FDES-based coordinator is described in Fig.6.4. where the values of i and j are determined depending on the high level

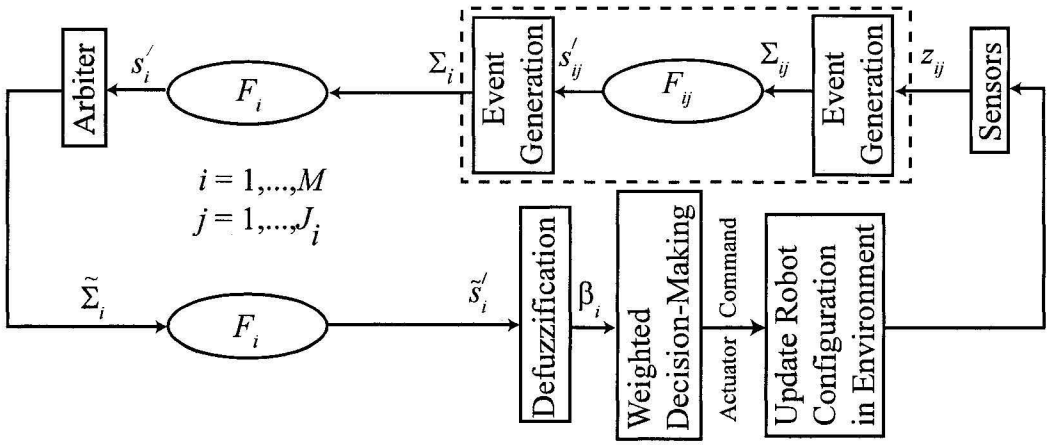


Figure 6.4: The FDES-based behavior coordinator

tasks. Table 6.1(a) and 6.1(b) shows the specific values of i and j for *anchoring* and *navigate-to-goal* tasks, respectively.

Table 6.1: The values of i and j for different high level tasks

(a)

Anchoring					
i	2	3	4	5	6
j	1, 2	1	1, 2	1	1, 2

(b)

Navigate-to-goal		
i	1	6
j	1	1, 2

The equivalence relation used in the *Arbiter* is defined as follows

$$B_i \equiv_{co} \hat{B} \text{ if } \text{abs}(\angle A_i - \angle \hat{A}) \leq \delta^o$$

where $\hat{A}$ represents the expected action of the highest priority behavior $\hat{B}$ in maximum activity state. The present work uses $\delta = 15^o$.

6.4 Path-and-Pulling planning using Perceptual Sequencing

Fig.6.5 describes the behavior coordination method using perceptual sequencing approach, which uses a finite state automaton to select an appropriate behavior. Each state is specified as a behavior and a state transition occurs using the corresponding event that is generated with the sensory data. Here, the events are generated using binary thresholding of the sensory data and are defined as follows.

$$\begin{aligned}
\alpha_{aobs} &: (z_{61} \leq 30 \text{ pix}) \text{ and } (z_{62} \geq 15^\circ) \\
\alpha_{gobj} &: z_{51} \geq 30 \text{ pix} \\
\alpha_{sobj} &: (z_{41} < 30 \text{ pix}) \text{ and } (z_{42} \geq 15^\circ) \\
\alpha_{ganc} &: z_{31} < 15^\circ \\
\alpha_{anch} &: (z_{21} < 15^\circ) \text{ and } (z_{22} \leq 10 \text{ pix}) \\
\alpha_{goal} &: z_{11} \leq 10 \text{ pix}
\end{aligned}$$

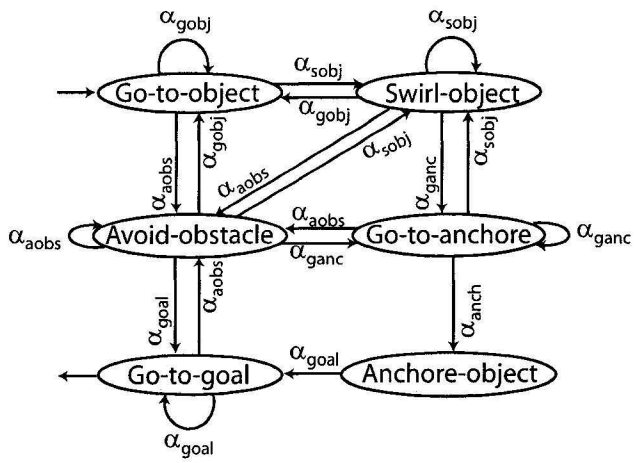


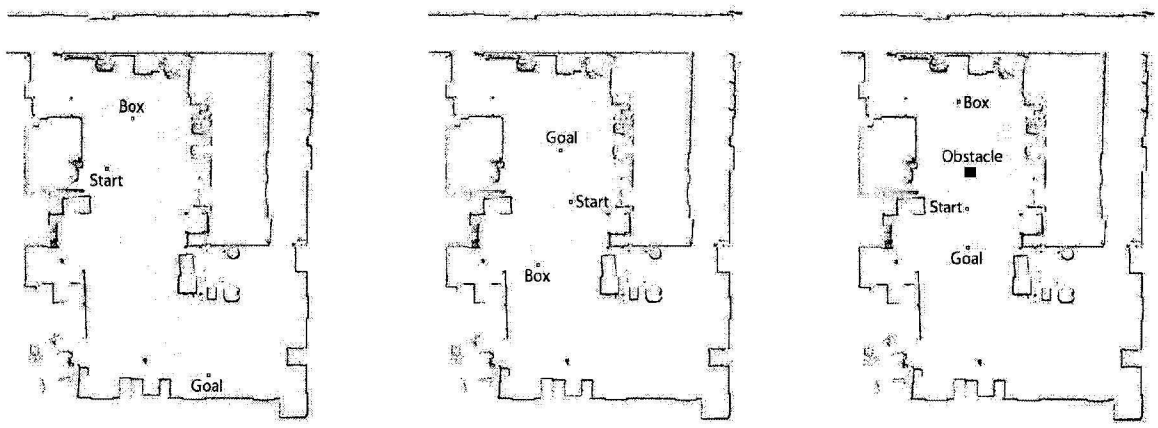
Figure 6.5: Behavior coordination using perceptual sequencing

The threshold values are chosen as the peak values of MFs of z_{ij}, which ensures individual behavioral goal achievement.

6.5 Experimental results

For the experiments, an Active Media Pioneer 3-AT robot is used and sonar sensor readings are taken to define range measurements for obstacle detection. At each decision cycle, the robot is controlled by sending a rotational velocity command (Ω) and a translational velocity command (v). The rotational velocity command Ω is proportional to $\angle A$, i.e., $\Omega = \kappa \times \angle A$ deg/s, where $\angle A$ denotes the expected heading direction of the coordinated action. κ is set to 1 for the examples presented in this work. The translational velocity command v is adjusted proportional to the rotational velocity (180 – abs(Ω)) and is measured in mm/s. The sampling time period (i.e., the length of a decision cycle) T_p is changed according to the anchoring status of the robot. The anchoring process requires image-based object recognition (using Bumblebee stereo camera) and to accommodate this process a higher value of T_p (250 ms) is used. When *Anchor-object* behavior gains the highest activity, T_p is set to

3000 ms to complete the anchoring process. However, T_p is reset to 50 ms once the anchoring process has been completed. As the environment used in the experiment is relatively small, the robot uses gyro-corrected odometry for localization. However, for large-scale environment position uncertainty increases that leads to improper subgoal detection and unreliable goal reaching navigation. Hence, online robot localization methods (e.g., [110]) should be incorporated for reliable navigation. The present work ignores the kinematic problem of object-pulling by assuming that the weight of the object is negligible as compared to the robot. A small cardboard box weighed about 0.5 kg is used as the object (The robot is weighed about 32 kg). The box is wrapped in red paper to reduce the computational complexity associated with image-based object-recognition process.



(a) Case I (b) Case II (c) Case III

Figure 6.6: Task environments

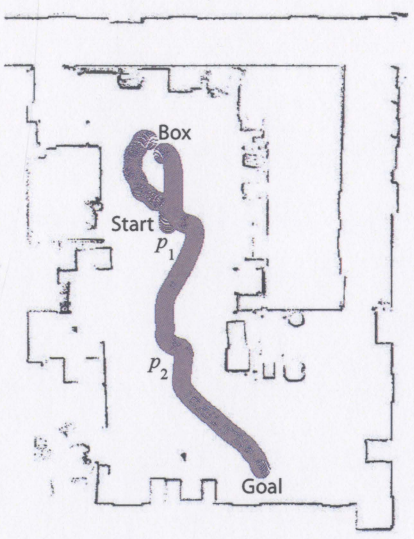
The experiments are performed in the following task environments.

Case I: Fig.6.6(a) shows the scenario for this case. It depicts the initial position of the robot (*Start*), object position (*Box*), and the goal location (*Goal*). In this case, dynamic moving obstacles are placed while performing the *navigate-to-goal* task between the *Box* and *Goal* positions.

Case II: The task environment for this case is shown in Fig.6.6(b). In this case, dynamic moving obstacles are introduced in both *anchoring* and *navigate-to-goal* tasks. Hence, one obstacle is placed between the *Start* and *Box* positions, and another obstacle is placed between the *Box* and *Goal* positions.

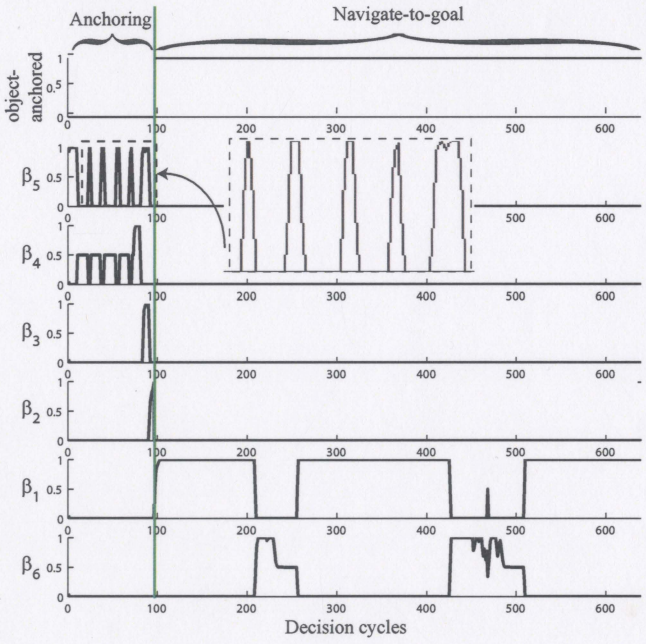
Case III: Fig.6.6(c) describes this case, which includes a static obstacle between the *Start* and *Box* positions. This environment is made static so that the robot experiences the same situation in comparing the results of the proposed method and perceptual sequencing technique.

Figures 6.7-6.10 show the experimental results. In Case I (Fig.6.7(a)), the box is placed in the opposite direction to the goal position. The robot first anchors itself with the box and then starts goal reaching navigation. Two moving objects are introduced at p_1 and p_2. The robot successfully avoids the dynamic obstacles and reaches the goal. Fig.6.7(b) shows the navigation scenario at p_1 in Case I when the robot avoids a dynamic obstacle. Fig.6.7(c) depicts the online-generated modulating factors used in Case I. The first 100 decision cycles represent the execution of *anchoring* task (i.e., $object - anchored = 0$), where the robot reaches and anchors itself with the box. The rest of the decision cycles represent the execution of *navigate-to-goal* task (i.e., $object - anchored = 1$), where the robot navigates to the goal location. The magnified picture of the variation of β_5 reveals an important characteristic of FDES-based behavior modulation. Here, the abrupt variations are restricted using state-based transition that leads to gradual variation of modulating factors. Thus, unlike perceptual sequencing method [118] the proposed approach reduces the possibility of complete control-transfer to an inappropriate behavior. As observed in Fig.6.7(c), the control of the robot switches between *go-to-object* and *swirl-object* because of faulty perception which is compensated by activating both behaviors with different degrees of activity. The variations in the values of β_6 also reveal that the robot faces the



(a) Trajectory

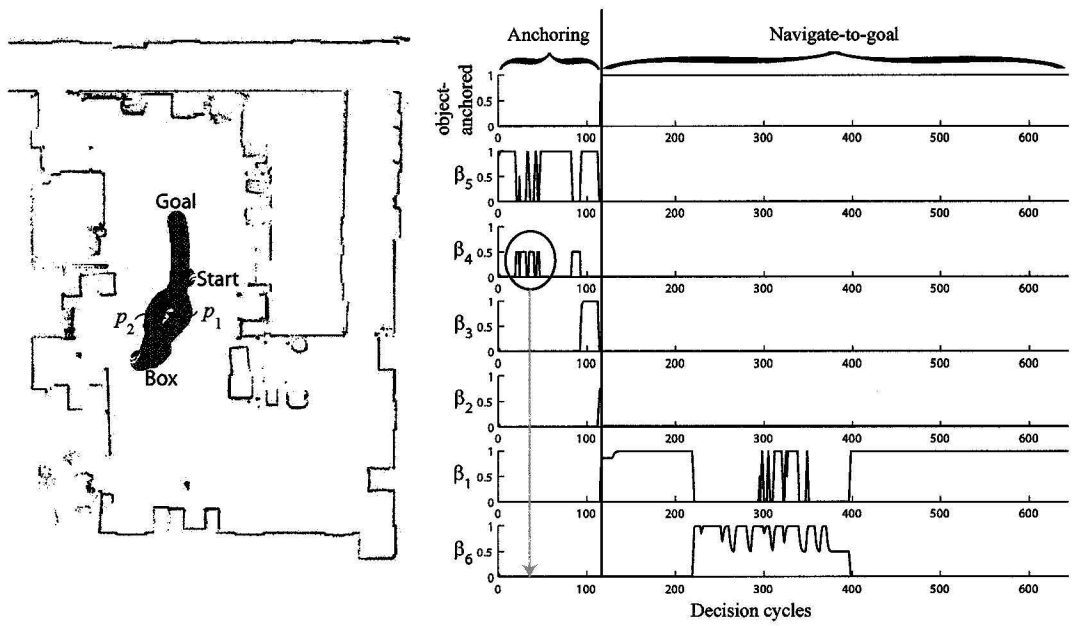
(b) The navigation scenario at p_1



(c) Modulating factors

Figure 6.7: Results in Case I

dynamic obstacles around the 250^{th} and 450^{th} decision cycles, which correspond to the points p_1 and p_2 in Fig.6.7(a).



(a) Trajectory

(b) Modulating factors

Figure 6.8: Results in Case II

In Case II, moving obstacles are introduced before and after anchoring the box (see point p_1 and p_2 in Fig.6.8(a)). The robot successfully avoids the dynamic obstacles and executes the given task. Fig.6.8(b) depicts the online-generated modulating factors, where the first 115 decision cycles represent *anchoring* task (i.e., $object - anchored = 0$) and the rest of the decision cycles represent *navigate-to-goal* task (i.e., $object - anchored = 1$). The variations in the values of β_4 and β_6 reveal a limitation of the current implementation. Since dynamic obstacles are introduced before and after the 115^{th} decision cycle, the values of β_6 should be high both in *anchoring* and *navigate-to-goal* tasks. However, it is observed that the modulating weight β_6 for *swirl-obstacle* behavior is prominent only in *navigate-to-goal* task around the 300^{th} decision cycle corresponding to the location p_2 in Fig.6.8(a). Whereas the

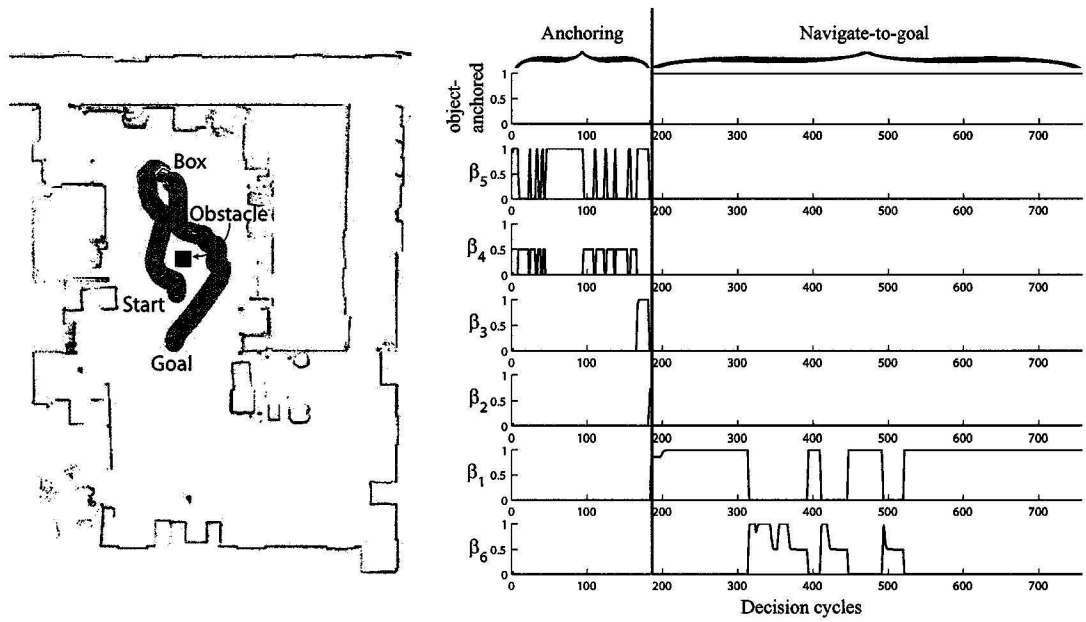
modulating factor β_4 for *swirl-object* behavior becomes high around the 30^{th} decision cycle corresponding to the location p_1 in Fig.6.8(a). This shortcoming stems from the erroneous classification of the target object and the obstacles. The range data corresponding to the target object is identified by correlating the orientation obtained from the vision data. Hence, a dynamic obstacle placed in between the robot and the object leads to erroneous classification of the range data, which causes producing high values of *swirl-object* instead of *swirl-obstacle*. However, this behavioral activation does not affect the robot's performance since the robot recalculates the orientation of the object using the updated vision data. The possibility of erroneous classification can be reduced by incorporating stereo-vision based range calculation of the target object.

Case III is performed to compare the reliability of the proposed method and perceptual sequencing method. A stationary environment has been used so that the robot experiences the same environment for both approaches. A stationary obstacle is placed at p_1 (see Fig.6.9(a) and 6.10(a)). Table 6.2 summarizes the results of 10 trials for each method. The performance evaluation reveals that the proposed approach provides 90% successful anchoring and collision free navigation, whereas the perceptual sequencing results in 60% successful anchoring and 4 collisions. Fig.6.9(a) and

	FDES-based coordination	Perceptual sequencing
Successful anchoring	90%	60%
No. of collisions	0	4

Table 6.2: Performance of different methods

6.10(a) show results of the first trial. The robot successfully performed the anchoring task using both the approaches. With perceptual sequencing method the robot collided with the obstacle at p_1. This happened due to faulty range measurements us-



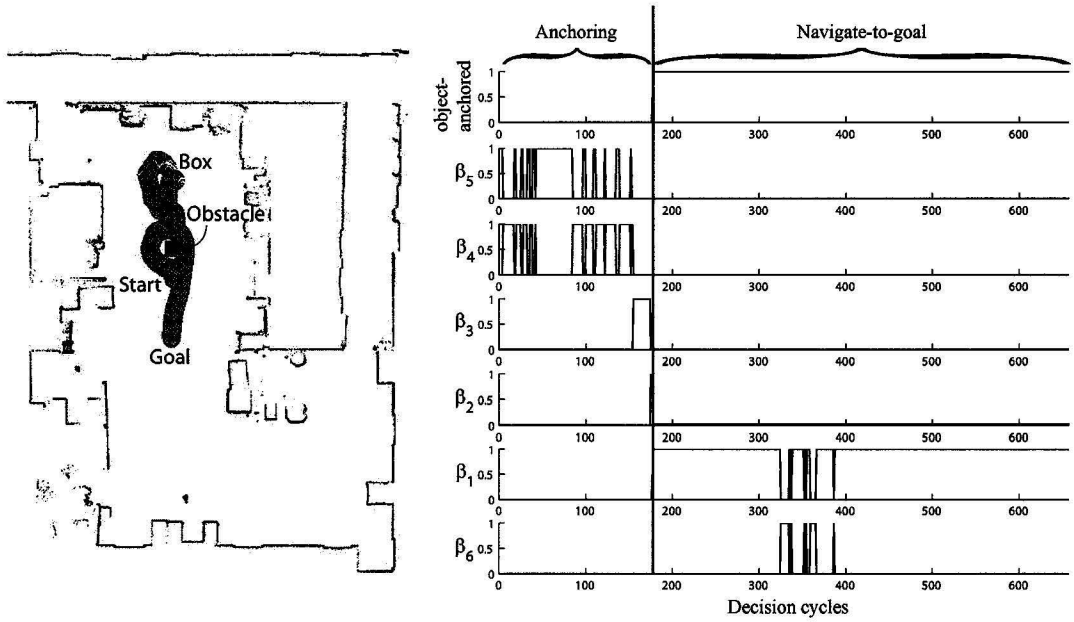
(a) Trajectory

(b) Modulating factors

Figure 6.9: Results produced by the FDES-based modulator in Case III

ing sonar sensors. Fig.6.9(b) and 6.10(b) show the modulating factors describing the behavioral activation in these experiments. In the FDES-based approach, the *anchoring* task is extended up to the 184^{th} decision cycle and in the perceptual sequencing method, it is extended up to the 176^{th} decision cycle. Fig.6.10(b) illustrates that the i^{th} behavior is selected by defining $\beta_i = 1$, $\beta_k = 0$, $k \neq i$. It also demonstrates that the application of binary thresholding causes frequent switching between the behaviors, which may lead to inappropriate action execution at a particular moment.

Two video clips (anchor.wmv and navigate.wmv) are provided on a CD showing the *anchoring* and *navigate-to-goal* tasks executed using the proposed FDES-based approach.



(a) Trajectory

(b) Modulating factors

Figure 6.10: Results produced by the sequencing method in Case III

6.6 Conclusion

In this chapter, the multilevel behavioral decomposition feature of the proposed FDES-based approach is demonstrated using a behavior-based object-pulling task of a mobile robot. The given task is decomposed into *anchoring* and *navigate-to-goal* subtasks (or high level abstract behaviors), which are further subdivided into low level FDES-based primitive behaviors. The goal of the *anchoring* task is to reach the target object and anchor it with the robot. The robot executes *navigate-to-goal* task to deliver the object to a given location. The high level tasks are controlled by separate FDES-based behavior modulators. The proposed method has been implemented using a physical robot where a box is used as the object. The current implementation does not consider the kinematic problem associated with pulling heavy objects. The performance of the proposed approach is compared with an existing behavior

modulation technique, called perceptual sequencing method. The experimental results demonstrate that the proposed approach is more reliable than the perceptual sequencing-based pulling task. The FDES-based approach produces 90% successful anchoring and collision-free safe navigation. Whereas the perceptual sequencing method results in 60% successful anchoring and 4 collisions with dynamic obstacles.

The following chapter will present another FDES-based real-world robotic application in the field of attention modeling of humanoid robots.

Chapter 7

Biased Competitive Model of Visual Attention Using Fuzzy Discrete Event System

7.1 Introduction

This chapter demonstrates an application of the FDES-based architecture in devising a visual attention system for cognitive robots while mimicking some of the major aspects of human visual attention process. The FDES-based method has the capacity to approximately analysis sensor perception, which is perfectly suitable to model the sensitivity of human visual system to different visual stimuli. At the same time, the proposed approach provides supervisory control technique to model the role of human mind in attentional selection. The extension of the FDES-based architecture for modeling an artificial visual attention system demonstrates the power of this proposed architecture in solving diversified real-world problems. This work mainly focuses on the propositions of a widely accepted neurodynamic theory of visual attention namely,

biased competition hypothesis, to develop a FDES-based model for attention selection.

7.1.1 Visual attention

Biased competitive hypothesis [122] is a widely accepted theory of visual attention. It integrates the psychophysical and neurophysiological evidences as well as the findings from behavioral data in a single theory of primates' visual attention. As a result, this theory is extensively accepted by the current researchers [123, 124, 125]. In this theory, multiple stimuli within the visual field excite cells in different areas of the visual cortex according to their 'unique strength'. Neurons are more sensitive to a stimulus in the foveal region than that on the peripheral region. Unique strength of an stimulus is defined by its primitive features [126], e.g. color, shape, orientation, contrast, etc. They are commonly known as bottom-up stimulus and contribute in constituting the saliency map in most computational models of visual attention [127] [128]. The bottom-up stimulus-activated neurons in areas of visual cortex interact with each other in a mutually suppressive manner. The top-down modulating feedback generated from regions outside of visual cortex can bias this competition in favor of a behaviorally relevant stimuli. This behavioral relevance might be in terms of object features e.g. color, shape, etc. or specific location in the visual field [129]. Accordingly, top-down bias can be classified into two categories, namely 1) bias in favor of object features and 2) bias in favor of spatial location. The former is termed as object bias and the latter is known as spatial selection (selection based on some cue to the location of target information) [130]. The object bias can be delivered either from long-term memory [129] or from working memory (also known as attention templet [131], short-term memory). When top-down bias is delivered from Long-term memory (LTM), it usually favors novelty [129]. When the top-down bias is delivered from working memory (WM), the attentional template specifies a

set of required property of the target stimulus e.g. color, shape, orientation, texture, etc. and the process of attentional selection essentially becomes a visual search for a single or conjunction of feature(s) [132, 133]. The stimulus in the visual field that is a good match with the attentional template receives strong top-down competitive bias, while the poor match stimuli receive weak feedback. In case of spatial selection behavioral relevance is established solely in terms of spatial location of a stimulus. A prior knowledge about the target's spatial location is stored in the working memory and accordingly a top-down bias is delivered in favor of the stimuli at that specific location.

This work aims to functionally model the visual attention system on the basis of the *biased competitive hypothesis* for robotic applications. The proposed computational model employs FDES-based coordination system to integrate the top-down and bottom-up biases to control the pan-tilt motion of a camera mounted on a robot. The FDES-based system is able to prevent abrupt changes in focus of attention and produces smooth transition in motion commands when visual attention changes between the objects.

The rest of the chapter is organized as follows. Section 7.2 describes the redefinitions of symbols required to extend the concept of a behavior into an object in a visual field. Section 7.3 outlines the proposed computational model of visual attention. Section 7.4 shows the experimental results and finally, Section 7.5 draws the conclusion.

7.2 Redefinition of symbols

The following is a list of symbols redefined to extend the concept of behavior into an object in the visual field of a robot.

B_i: the i^th object in the current visual field. The index i also denotes the priority

order of attention, which is determined depending on the distance of the object from the center of the visual field. The lower distance indicates the higher priority, i.e., higher values of i. The values of i range from 1 to M, where M is the total number of objects in the current visual field. This priority assignment models the phenomena that neurons are more sensitive to a stimulus in the foveal region than that on the peripheral region.

A_i: the set of expected actions $\{A_{i,pan}, A_{i,tilt}\}$ required to control the pan-tilt motion of a camera to focus the i^th object at the center of current visual field.

Z_i: the set of features $z_{ij}, j = 1, \ldots, J_i$ associated with the stimulus of the i^th object (J_i is the total number of features). In the present work, it is assumed that each stimulus in the current visual field has the same number of features, i.e., $J_1 = \ldots = J_i = \ldots = J_M$. Examples of the features include color, hue, saturation, and texture.

F_i: a FDES that determines the overall strength of the stimulus corresponding to the i^th object. F_i consists of FDES F_{ij}, which evaluates the contribution (i.e., bottom-up saliency) of the j^th feature z_{ij} of the i^th object. F_i combines the feature contributions generated by each F_{ij} to produce the overall strength of the stimulus.

The task of the proposed attention model is to determine the overall action $A = \{A_{i,pan}, A_{i,tilt}\}$ using (7.1) and (7.2).

$$A_{pan} = \sum_{i=1}^{M} \beta_i \times A_{i,pan} \tag{7.1}$$

$$A_{tilt} = \sum_{i=1}^{M} \beta_i \times A_{i,tilt} \tag{7.2}$$

Here, $\theta_{pan} = \angle A_{pan}$ and $\theta_{tilt} = \angle A_{tilt}$ are used as the pan and tilt commands to control the camera motion of the robot. In rest of this chapter, the terms 'visual

field' and 'frame' will be used interchangeably.

7.3 The proposed computational model using FDES

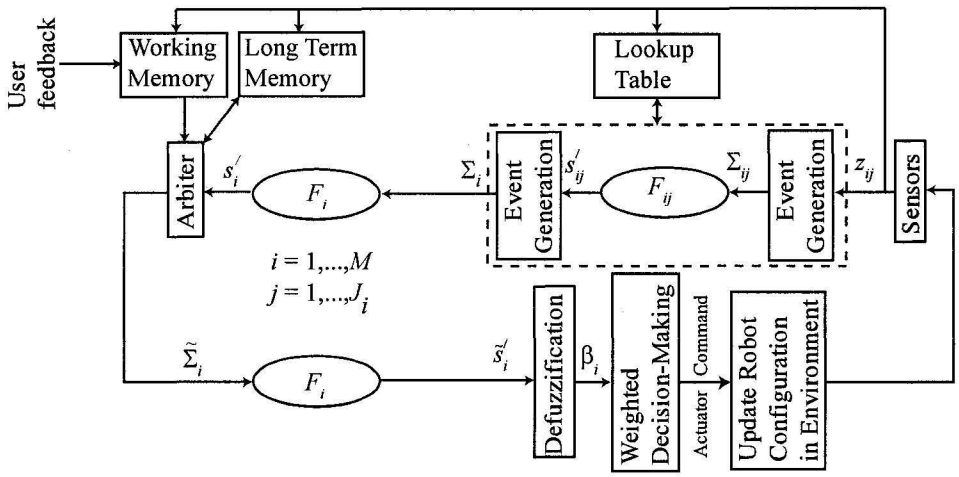


Figure 7.1: The proposed computational model

Fig.7.1 shows the proposed computational model using FDES. The sensory data z_{ij} quantifies the the j^{th} feature into a value, which is used as the input to F_{ij}. The outputs ($\acute{s}_{ij}$) of each F_{ij} are combined by F_i in order to determine the overall strength ($\acute{s}_i$) of the i^{th} stimulus. Note that the proposed method does not model the inhibitory effect between the neighboring objects in order to reduce the computational complexity. The FDES-based architecture also provides the following useful properties:

- It provides an opportunity to implement the rule-based human reasoning using the transition structure of a FDES. As a result, it enables non-linear mapping of the feature values (z_{ij}) into feature-contribution ($\acute{s}_{ij}$). The state-transition structure of each F_{ij} can be different from each other, thus, reflecting varying importance of each feature.

- This architecture uses fuzzy logic to estimate the contribution of each feature.

Hence, it offers the flexibility of modeling deterministic vagueness of human reasoning. Appropriate tuning of the MFs largely mimics the non-linear nature of human decision-making.

- The generation-process of each feature-contribution is state-dependent. This indicates that the effect of a feature corresponding to a stimulus from an object depends on its previous strength found in the last frame (visual field). As a result, the FDES-based system prevents abrupt change in the focus of attention visual attention and produces smooth motor commands for robotic motion.

The F_i and F_{ij} are constructed dynamically at each frame. Consequently, it requires an object tracking mechanism between two successive visual fields so that F_i and F_{ij} are initialized with the corresponding output states of its matched FDESs in the previous frame. To achieve this goal, a lookup table is maintained containing the feature vectors $[z_{ij}(k-1)]_{j=1,\ldots,J_i}$ and the output states $\acute{s}_i(k-1)$ and $\acute{s}_{ij}(k-1)$, $j = 1, \ldots, J_i$ estimated in the previous visual field, say the $(k-1)^{th}$ frame. In the current visual field, say the k^{th} frame, the best match of an object is determined by calculating the Euclidean distance between its feature vector and those stored in the lookup table. The FDES components to the matched objects are initialized using the states shown in the lookup table and FDES components to other unmatched objects are initialized using default states.

According to the *biased competitive hypothesis* the stimuli are competitive in nature. This characteristic is included in the proposed system using an arbitration mechanism called the *Arbiter* (see Fig.7.1) as described in Chapter 4. However, this hypothesis also states that the competition is influenced by the top-down bias, which includes novelty/importance bias and feature/location bias. The novelty/importance bias is generated from the LTM of the system and feature/location bias is generated from the WM or user feedback. The *Arbiter* only considers those stimuli, which

are favored by the top-down bias for the arbitration process. Furthermore, the *Arbiter* assigns higher priority to the WM by overriding the selection obtained from the LTM. In absence of the top-down bias the arbitration process is guided only by the bottom-up saliency.

In the proposed architecture, the LTM is implemented to provide the novelty bias. The LTM keeps records of the newly attended objects by learning their feature vectors. At each decision cycle, the feature vectors corresponding to the objects present in the current visual field are compared with the LTM to generate the novelty bias. When an object is selected by the *Arbiter*, the LTM updates its feature representation. The LTM is implemented by an Adaptive Resonance Theory 2 (ART2) neural network [134]. The number of clusters in ART2 is increased dynamically to incorporate the representation of the newly learned objects. In robotic applications, the feature vector of a particular object may differ in successive decision cycles due to continuous camera motion. To make a generalized representation of an object the ART2 requires to learn the same object for few decision cycles. According to the state-transition structure of a FDES, a stimulus gains its highest strength (in the worst case) if it is attended continuously for N number of decision cycles (N is the total number of states in a FDES). Hence, the NN continues learning a new object for N number of decision cycles to increase robustness of the learned feature vector.

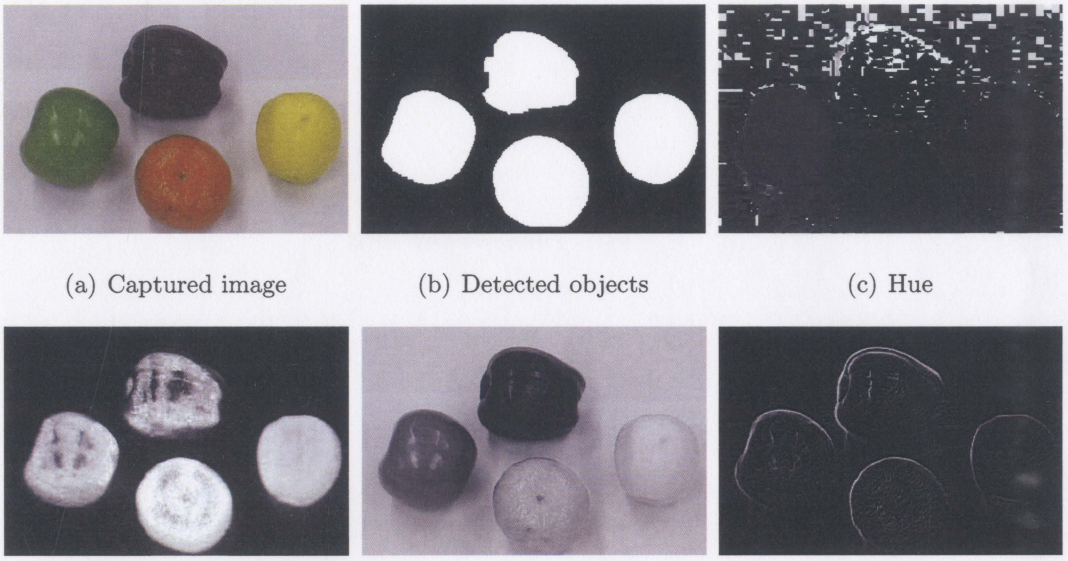
The WM generates feature/location bias depending on the user feedback. A feature bias is generated when the given feature specification matches with one of the object in the current visual field. The best match is determined by calculating the Euclidean distance between the user defined feature vector and those present in the current visual field. The objects similar to the user defined specification are considered by the *Arbiter* for the arbitration process. The location bias is generated in a similar manner. In case of location bias, the user defined spatial location is com-

pared to the locations of the existing object in the current visual field and the objects close to the given location (in terms of Euclidean distance) are considered for the arbitration process.

After combining the top-down and bottom-up bias the *Arbiter* generates the set of modified events matrices ($\tilde{\Sigma}_i$), which is used by F_i to recalculate the strength ($\acute{\tilde{s}}_i$) for the i^{th} stimulus. The stimulus-strength $\acute{\tilde{s}}_i$ is expressed in the form of a fuzzy state vector. Consequently, it is defuzzified to produce a scalar weight value β_i, which is used for weighted-decision-based action generation. Action A_i associated with the i^{th} object consists of two unit vectors $A_{i,pan}$ and $A_{i,tilt}$. These unit vectors are directed to the expected pan and tilt angles required to position the object at the center of the visual field. The overall action $A = \{A_{pan}, A_{tilt}\}$ is generated using (7.1) and (7.2). Here, $\theta_{pan} = \angle A_{pan}$ and $\theta_{tilt} = \angle A_{tilt}$ are used as the pan and tilt commands to control the camera motion of the robot.

The proposed FDES-based architecture provides additional means of system analysis, which measures the decision vagueness and sudden changes in focus of attention in terms of state-based observability and controllability of the FDESs associated with the attended object. The robot employs slower pan-tilt motion in presence of vague decision and sudden changes in focus of attention to ensure smooth camera motion. If B_i is selected as $\hat{B}$ (i.e., the most appropriate object for the robot's attention) by the *Arbiter* (see Section 4.4 in Chapter 4), the pan and tilt commands are modified using (7.3) and (7.4).

$$
\begin{aligned}
O_a &= \frac{1}{J_i}\sum_{j=1}^{J_i} O_{ij} \\
C_a &= \frac{1}{J_i}\sum_{j=1}^{J_i} C_{ij} \\
\theta_{pan} &= O_a \times C_a \times \angle A_{pan} \ \text{deg} \qquad (7.3) \\
\theta_{tilt} &= O_a \times C_a \times \angle A_{tilt} \ \text{deg} \qquad (7.4)
\end{aligned}
$$



(a) Captured image (b) Detected objects (c) Hue

(d) Saturation (e) Intensity (f) Texture (edge)

Figure 7.2: Object detection and feature extraction

Here O_a and C_a are the average measures of state-based observability (O_{ij}) and controllability (C_{ij}) of the FDES F_{ij}.

7.4 Experiments

7.4.1 Object detection and feature extraction

The current experimental setup uses colored objects on white background to allow faster object detection and feature extraction for the demonstrations. Fig.7.2 describes an example, where the captured image is shown in Fig.7.2(a). The white-background is subtracted using color-based thresholding in RGB (Red-Green-Blue) plane. The connected image segments are specified as the objects and are shown in Fig.7.2(b).

This work uses the HSI (Hue-Saturation-Intensity) representation of image for the

feature extraction process. Hue varies from 0 to 1 and the corresponding colors vary from red, through yellow, green, cyan, blue, and magenta. Saturation varies from 0 to 1 and the corresponding colors (hues) vary from unsaturated (shades of gray) to fully saturated (no white component). Intensity, or brightness, varies from 0 to 1 and the corresponding colors become increasingly brighter.

The following features are used to represent the i^{th} object in the current visual field.

- Object-saturation (z_{i1}) denotes the average saturation value of the object.
- Object-intensity (z_{i2}) presents the average intensity value of the object.
- Object-hue (z_{i3}) indicates the average hue value of the object.
- Intensity-contrast (z_{i4}) determines the average intensity contrast between the background and the object.
- Object-texture (z_{i5}) evaluates the average strength of the intensity gradient to represent the texture of the object.
- Object-area (z_{i6}) calculates the normalized number of pixels belonging to the object with respect to a reference object-area (10000 pixels).

The feature values $z_{i1}, \ldots, z_{i5}$ vary from 0 to 1 and z_{i6} ranges in the limit [0,1). Figures 7.2(c)-7.2(f) describe the HSI representation and the texture (edge) of the captured image shown in Fig.7.2(a). Table 7.1 outlines the feature vectors corresponding to the objects detected in Fig.7.2(b).

7.4.2 FDES modeling

The visual field of the robot includes M number of objects as candidate to receive attention (see Fig.7.3) The i^{th} object is represented using $B_i = (A_i, Z_i, F_i)$. The unit

	z_{i1}	z_{i2}	z_{i3}	z_{i4}	z_{i5}	z_{i6}
Red apple	0.6583	0.1394	0.1335	0.5361	0.0761	0.6307
Green apple	0.7419	0.3517	0.2101	0.3238	0.1174	0.6806
Golden apple	0.8107	0.7	0.1396	0.0244	0.0642	0.629
Orange	0.9102	0.602	0.0788	0.0736	0.1597	0.7233

Table 7.1: Feature vectors

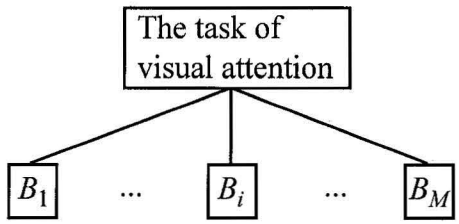


Figure 7.3: Visual attention system

vectors associated with the action $A_i = \{A_{i,pan}, A_{i,tilt}\}$ are determined as follows

$$\angle A_{i,pan} = \frac{FV}{WFV} \times D_{i,pan} \text{ deg}$$
$$\angle A_{i,tilt} = \frac{FV}{HFV} \times D_{i,tilt} \text{ deg.}$$

FV denotes the field of view of the camera, which is 63^o for the Bumblebee camera used in the experiment. WFV and HFV stand for the width (320 pix) and height (240 pix) of an image captured by the camera. $D_{i,pan}$ and $D_{i,tilt}$ denote the distances from the center of the visual field to the centroid of i^{th} object along x-axis and y-axis, respectively (see Fig.7.4).

The sensory data (or feature values) $Z_i = \{z_{ij} | j = 1, \ldots, J_i\}$, $J_i = 6$, where z_{ij} is described in Section 7.4.1.

FDES F_i is composed of F_{ij}, $j = 1, \ldots, J_i$. Fig.7.5 shows the state-transition diagrams of the FDESs, which model the feature contributions and overall strength of the stimulus. The FDESs consist of 5 states (i.e., $N = 5$) and 5 events (i.e., $E = 5$).

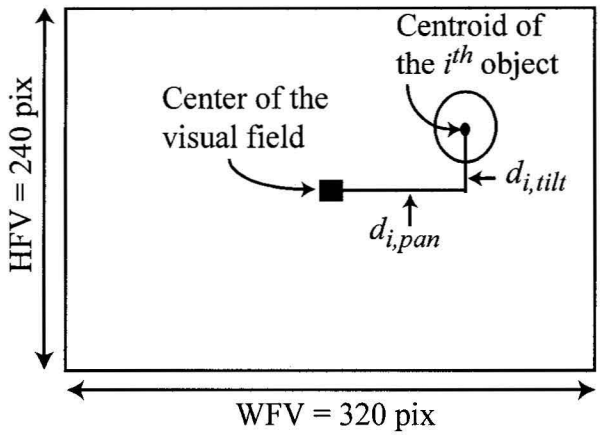
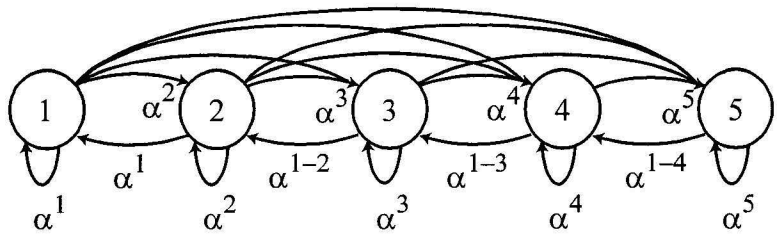


Figure 7.4: Field of view

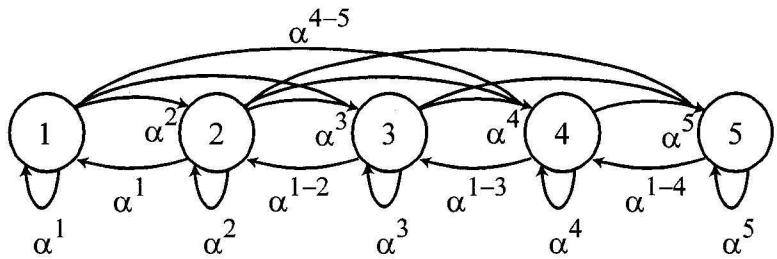
Here $state_1, \ldots, state_5$ indicate *lower*, *low*, *medium*, *high*, and *higher* strengths, respectively. Fig.7.5(a) shows the transition-structure of F_{i1}. It evaluates the effect of z_{i1} using the event matrices formed using transitions rules. Events α^e_{i1}, $e = 1, \ldots, E$ are constructed using (4.11) as follows:

$$\alpha^1_{i1} = \begin{bmatrix} f^1_{i1}(z_{i1}) & 0 & 0 & 0 & 0 \\ f^1_{i1}(z_{i1}) & 0 & 0 & 0 & 0 \\ 0 & f^1_{i1}(z_{i1}) & 0 & 0 & 0 \\ 0 & 0 & f^1_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & 0 & f^1_{i1}(z_{i1}) & 0 \end{bmatrix},$$

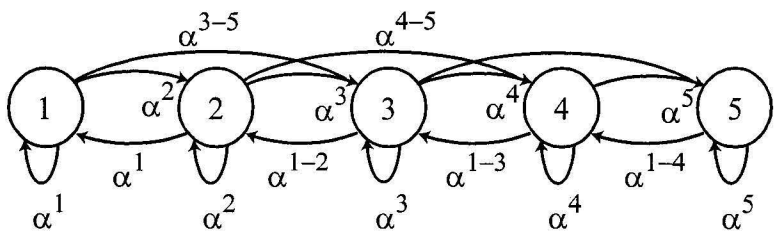
$$\alpha^2_{i1} = \begin{bmatrix} 0 & f^2_{i1}(z_{i1}) & 0 & 0 & 0 \\ 0 & f^2_{i1}(z_{i1}) & 0 & 0 & 0 \\ 0 & f^2_{i1}(z_{i1}) & 0 & 0 & 0 \\ 0 & 0 & f^2_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & 0 & f^2_{i1}(z_{i1}) & 0 \end{bmatrix}, \quad \alpha^3_{i1} = \begin{bmatrix} 0 & 0 & f^3_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & f^3_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & f^3_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & f^3_{i1}(z_{i1}) & 0 & 0 \\ 0 & 0 & 0 & f^3_{i1}(z_{i1}) & 0 \end{bmatrix},$$



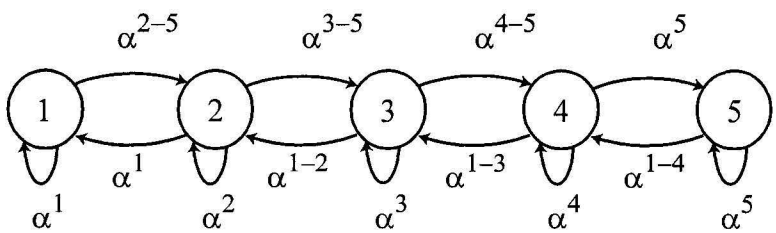
(a) F_{i1} and F_{i2}



(b) F_{i3}



(c) F_{i4}



(d) F_{i5}, F_{i6}, and F_i

$\alpha^{1-2}=\{\alpha^1, \alpha^2\}$ $\quad$ $\alpha^{4-5}=\{\alpha^4, \alpha^5\}$

$\alpha^{1-3}=\{\alpha^{1-2}, \alpha^3\}$ $\quad$ $\alpha^{3-5}=\{\alpha^3, \alpha^{4-5}\}$

$\alpha^{1-4}=\{\alpha^{1-3}, \alpha^4\}$ $\quad$ $\alpha^{2-5}=\{\alpha^2, \alpha^{3-5}\}$

Figure 7.5: State-transition diagrams

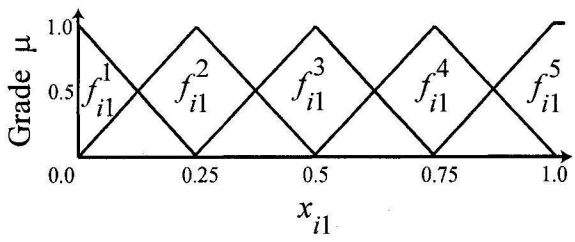


Figure 7.6: Membership functions

$$\alpha_{i1}^4 = \begin{bmatrix} 0 & 0 & 0 & f_{i1}^4(z_{i1}) & 0 \\ 0 & 0 & 0 & f_{i1}^4(z_{i1}) & 0 \\ 0 & 0 & 0 & f_{i1}^4(z_{i1}) & 0 \\ 0 & 0 & 0 & f_{i1}^4(z_{i1}) & 0 \\ 0 & 0 & 0 & f_{i1}^4(z_{i1}) & 0 \end{bmatrix}, \text{ and } \alpha_{i1}^5 = \begin{bmatrix} 0 & 0 & 0 & 0 & f_{i1}^5(z_{i1}) \\ 0 & 0 & 0 & 0 & f_{i1}^5(z_{i1}) \\ 0 & 0 & 0 & 0 & f_{i1}^5(z_{i1}) \\ 0 & 0 & 0 & 0 & f_{i1}^5(z_{i1}) \\ 0 & 0 & 0 & 0 & f_{i1}^5(z_{i1}) \end{bmatrix}.$$

The MF f_{i1}^e associated with the event α_{i1}^e is shown in Fig.7.6. It is evident from the state-diagram and MFs shown in Fig.7.5(a) and 7.6 that the feature-contribution corresponding to object-saturation is proportional to z_{i1}. This feature is given the highest priority by allowing direct transitions from lower states to higher states. However, lower states are only reachable from immediate higher states. This prevents abrupt distraction in focus of attention in case of noisy perception. For state-based observability and controllability analysis, L_{i1} and W_{i1} are defined as

$$L_{i1} = \begin{bmatrix} 0 & 0.125 & 0.375 & 0.65625 & 0.9375 \\ 0.125 & 0 & 0.125 & 0.375 & 0.65625 \\ 0.375 & 0.125 & 0 & 0.125 & 0.375 \\ 0.65625 & 0.375 & 0.125 & 0 & 0.125 \\ 0.9375 & 0.65625 & 0.375 & 0.125 & 0 \end{bmatrix} \text{ and } W_{i1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

The matrix L_{i1} is determined using (4.18) and W_{i1} is defined according to the unreachable state-information.

F_{i2} is similar to F_{i1}, where event α_{i2}^e is determined using the feature value z_{i2}.

The MF $f_{i2}^{e} = f_{i1}^{e}$, matrix $L_{i2} = L_{i1}$ and $W_{i2} = W_{i1}$. The feature-contribution corresponding to object-intensity is proportional to the feature value z_{i2}.

F_{i3} has the transition structure shown in Fig.7.5(b). The events of F_{i3} are determined as follows:

$$\alpha_{i3}^{1} = \begin{bmatrix} f_{i3}^{1}(z_{i3}) & 0 & 0 & 0 & 0 \\ f_{i3}^{1}(z_{i3}) & 0 & 0 & 0 & 0 \\ 0 & f_{i3}^{1}(z_{i3}) & 0 & 0 & 0 \\ 0 & 0 & f_{i3}^{1}(z_{i3}) & 0 & 0 \\ 0 & 0 & 0 & f_{i3}^{1}(z_{i3}) & 0 \end{bmatrix},$$

$$\alpha_{i3}^{2} = \begin{bmatrix} 0 & f_{i3}^{2}(z_{i3}) & 0 & 0 & 0 \\ 0 & f_{i3}^{2}(z_{i3}) & 0 & 0 & 0 \\ 0 & f_{i3}^{2}(z_{i3}) & 0 & 0 & 0 \\ 0 & 0 & f_{i3}^{2}(z_{i3}) & 0 & 0 \\ 0 & 0 & 0 & f_{i3}^{2}(z_{i3}) & 0 \end{bmatrix}, \quad \alpha_{i3}^{3} = \begin{bmatrix} 0 & 0 & f_{i3}^{3}(z_{i3}) & 0 & 0 \\ 0 & 0 & f_{i3}^{3}(z_{i3}) & 0 & 0 \\ 0 & 0 & f_{i3}^{3}(z_{i3}) & 0 & 0 \\ 0 & 0 & f_{i3}^{3}(z_{i3}) & 0 & 0 \\ 0 & 0 & 0 & f_{i3}^{3}(z_{i3}) & 0 \end{bmatrix},$$

$$\alpha_{i3}^{4} = \begin{bmatrix} 0 & 0 & 0 & f_{i3}^{4}(z_{i3}) & 0 \\ 0 & 0 & 0 & f_{i3}^{4}(z_{i3}) & 0 \\ 0 & 0 & 0 & f_{i3}^{4}(z_{i3}) & 0 \\ 0 & 0 & 0 & f_{i3}^{4}(z_{i3}) & 0 \\ 0 & 0 & 0 & f_{i3}^{4}(z_{i3}) & 0 \end{bmatrix}, \text{ and } \alpha_{i3}^{5} = \begin{bmatrix} 0 & 0 & 0 & f_{i3}^{5}(z_{i3}) & 0 \\ 0 & 0 & 0 & 0 & f_{i3}^{5}(z_{i3}) \\ 0 & 0 & 0 & 0 & f_{i3}^{5}(z_{i3}) \\ 0 & 0 & 0 & 0 & f_{i3}^{5}(z_{i3}) \\ 0 & 0 & 0 & 0 & f_{i3}^{5}(z_{i3}) \end{bmatrix}.$$

The MF $f_{i3}^{(5-e+1)} = f_{i1}^{e}$, which indicates that the feature-contribution corresponding to object-hue is inversely proportional to the feature value z_{i3}. This feature is given less importance than saturation and intensity by imposing the constraint that a lower state can only reach three successive higher states. For observability and

controllability analysis, $L_{i3} = L_{i1}$ is used and W_{i3} is defined as

$$W_{i3} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

F_{i4} has the transition structure shown in Fig.7.5(c). The events of F_{i4} are determined as follows:

$$\alpha^1_{i4} = \begin{bmatrix} f^1_{i4}(z_{i4}) & 0 & 0 & 0 & 0 \\ f^1_{i4}(z_{i4}) & 0 & 0 & 0 & 0 \\ 0 & f^1_{i4}(z_{i4}) & 0 & 0 & 0 \\ 0 & 0 & f^1_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & 0 & f^1_{i4}(z_{i4}) & 0 \end{bmatrix},$$

$$\alpha^2_{i4} = \begin{bmatrix} 0 & f^2_{i4}(z_{i4}) & 0 & 0 & 0 \\ 0 & f^2_{i4}(z_{i4}) & 0 & 0 & 0 \\ 0 & f^2_{i4}(z_{i4}) & 0 & 0 & 0 \\ 0 & 0 & f^2_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & 0 & f^2_{i4}(z_{i4}) & 0 \end{bmatrix}, \quad \alpha^3_{i4} = \begin{bmatrix} 0 & 0 & f^3_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & f^3_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & f^3_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & f^3_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & 0 & f^3_{i4}(z_{i4}) & 0 \end{bmatrix},$$

$$\alpha^4_{i4} = \begin{bmatrix} 0 & 0 & f^4_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & 0 & f^4_{i4}(z_{i4}) & 0 \\ 0 & 0 & 0 & f^4_{i4}(z_{i4}) & 0 \\ 0 & 0 & 0 & f^4_{i4}(z_{i4}) & 0 \\ 0 & 0 & 0 & f^4_{i4}(z_{i4}) & 0 \end{bmatrix}, \text{ and } \alpha^5_{i4} = \begin{bmatrix} 0 & 0 & f^5_{i4}(z_{i4}) & 0 & 0 \\ 0 & 0 & 0 & f^5_{i4}(z_{i4}) & 0 \\ 0 & 0 & 0 & 0 & f^5_{i4}(z_{i4}) \\ 0 & 0 & 0 & 0 & f^5_{i4}(z_{i4}) \\ 0 & 0 & 0 & 0 & f^5_{i4}(z_{i4}) \end{bmatrix}.$$

The MF $f^e_{i4} = f^e_{i1}$, which indicates that the feature-contribution corresponding to intensity contrast between the object and the background is proportional to the feature value z_{i4}. This feature is given less importance than the previous features by impos-

ing the constraint that a lower state can only reach two successive higher states. For observability and controllability analysis, $L_{i4} = L_{i1}$ is used and W_{i4} is defined as

$$W_{i4} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

F_{i5} has the transition structure shown in Fig.7.5(d). The events of F_{i5} are determined as follows:

$$\alpha^1_{i5} = \begin{bmatrix} f^1_{i5}(z_{i5}) & 0 & 0 & 0 & 0 \\ f^1_{i5}(z_{i5}) & 0 & 0 & 0 & 0 \\ 0 & f^1_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & 0 & f^1_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & 0 & f^1_{i5}(z_{i5}) & 0 \end{bmatrix},$$

$$\alpha^2_{i5} = \begin{bmatrix} 0 & f^2_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & f^2_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & f^2_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & 0 & f^2_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & 0 & f^2_{i5}(z_{i5}) & 0 \end{bmatrix}, \quad \alpha^3_{i5} = \begin{bmatrix} 0 & f^3_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & 0 & f^3_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & f^3_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & f^3_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & 0 & f^3_{i5}(z_{i5}) & 0 \end{bmatrix},$$

$$\alpha^4_{i5} = \begin{bmatrix} 0 & f^4_{i5}(z_{i5}) & 0 & 0 & 0 \\ 0 & 0 & f^4_{i5}(z_{i5}) & 0 & 0 \\ 0 & 0 & 0 & f^4_{i5}(z_{i5}) & 0 \\ 0 & 0 & 0 & f^4_{i5}(z_{i5}) & 0 \\ 0 & 0 & 0 & f^4_{i5}(z_{i5}) & 0 \end{bmatrix}, \text{ and}$$

$$\alpha_{i5}^{5} = \begin{bmatrix} 0 & f_{i5}^{5}(z_{i5}) & 0 & 0 & 0 \\ 0 & 0 & f_{i5}^{5}(z_{i5}) & 0 & 0 \\ 0 & 0 & 0 & f_{i5}^{5}(z_{i5}) & 0 \\ 0 & 0 & 0 & 0 & f_{i5}^{5}(z_{i5}) \\ 0 & 0 & 0 & 0 & f_{i5}^{5}(z_{i5}) \end{bmatrix}.$$

The MF $f_{i5}^{e} = f_{i1}^{e}$, which infers that the feature-contribution corresponding to object-texture is proportional to the feature value z_{i5}. This feature is given the lowest importance by imposing the constraint that a lower state can only reach the successive higher state. For observability and controllability analysis, $L_{i5} = L_{i1}$ is used and W_{i5} is defined as

$$W_{i5} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

F_{i6} is similar to F_{i5}, where event α_{i6}^{e} is determined using the feature value z_{i6}. The MF $f_{i6}^{e} = f_{i1}^{e}$, matrix $L_{i6} = L_{i1}$ and $W_{i6} = W_{i5}$. The feature-contribution corresponding to object-area is proportional to the feature value z_{i6}.

The overall FDES F_i has the same state-transition structure as F_{i5} and F_{i6}. It prevents abrupt changes in focus of attention by using the restrictive transition structure described in Fig.7.5(d). Event α_i^e is calculated using (4.12) as

$$\alpha_i^1 = \begin{bmatrix} \gamma_i^1 & 0 & 0 & 0 & 0 \\ \gamma_i^1 & 0 & 0 & 0 & 0 \\ 0 & \gamma_i^1 & 0 & 0 & 0 \\ 0 & 0 & \gamma_i^1 & 0 & 0 \\ 0 & 0 & 0 & \gamma_i^1 & 0 \end{bmatrix}, \alpha_i^2 = \begin{bmatrix} 0 & \gamma_i^2 & 0 & 0 & 0 \\ 0 & \gamma_i^2 & 0 & 0 & 0 \\ 0 & \gamma_i^2 & 0 & 0 & 0 \\ 0 & 0 & \gamma_i^2 & 0 & 0 \\ 0 & 0 & 0 & \gamma_i^2 & 0 \end{bmatrix}, \alpha_i^3 = \begin{bmatrix} 0 & \gamma_i^3 & 0 & 0 & 0 \\ 0 & 0 & \gamma_i^3 & 0 & 0 \\ 0 & 0 & \gamma_i^3 & 0 & 0 \\ 0 & 0 & \gamma_i^3 & 0 & 0 \\ 0 & 0 & 0 & \gamma_i^3 & 0 \end{bmatrix},$$

$$\alpha_i^4 = \begin{bmatrix} 0 & \gamma_i^4 & 0 & 0 & 0 \\ 0 & 0 & \gamma_i^4 & 0 & 0 \\ 0 & 0 & 0 & \gamma_i^4 & 0 \\ 0 & 0 & 0 & \gamma_i^4 & 0 \\ 0 & 0 & 0 & \gamma_i^4 & 0 \end{bmatrix}, \text{ and } \alpha_i^5 = \begin{bmatrix} 0 & \gamma_i^5 & 0 & 0 & 0 \\ 0 & 0 & \gamma_i^5 & 0 & 0 \\ 0 & 0 & 0 & \gamma_i^5 & 0 \\ 0 & 0 & 0 & 0 & \gamma_i^5 \\ 0 & 0 & 0 & 0 & \gamma_i^5 \end{bmatrix}$$

where γ_i^e is determined using (4.13). For observability and controllability analysis, $L_i = L_{i1}$ and $W_i = W_{i5}$ are used.

The FDESs corresponding to a new object are initialized with a default state vector [0.9 0.1 0.1 0.1 0.1 0.1], which infers *lower* stimulus-strength. In the arbitration process δ is set to 0, which infers that the winner object is given the highest importance while directing the attention.

7.4.3 Results

This section presents three experiments to explore different aspects of the proposed approach.

Case I: In this experiment, only one object is placed within the visual field of the robot. The robot learns the feature vector of the object and controls the pan-tilt motion of the camera to focus the object at the center of its visual field.

Case II: In this experiment, two objects are placed in the robot's visual field. The robot employs bottom-up saliency to focus the first object and then uses novelty bias to focus the second object.

Case III: In this experiment, three objects are sequentially placed in the robot's visual field. The robot focuses the objects using both the bottom-up saliency and novelty bias. Furthermore, location bias is also introduced to focus one of the objects using the learned feature vector of the object.

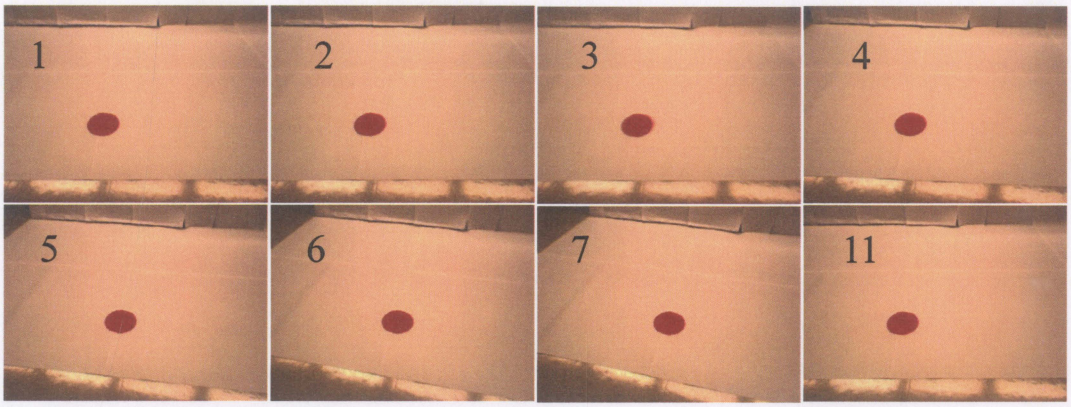


Figure 7.7: Image sequence in experiment 1

An ActivMedia Pioneer 3AT robot is used in the experiments. The robot is equipped with a pan-tilt unit, which controls the motion of a Bumblebee camera. The motion commands are defined using (7.3) and (7.4).

Fig.7.7 depicts an image sequence that describes the process of visual attention in experiment 1. The first image shows the experimental setup with respect to the home position of the camera. A circular red cardboard is placed on the white background. Images 2-7 in Fig.7.7 describe the gradual shift in camera position to place the red object at the center of visual field. The object is learned $2N$ (i.e., 10) times and then the camera is reset to its home position. Fig.7.8 demonstrates the associated parameter variations to control the camera motion. The weight of the selected object is shown in Fig.7.8(a), which describes the gradual increase in the importance of red object. According to the state diagram (see Fig.7.5), it requires at least N (i.e, 5) successive decision cycles to attain the maximum value of the weight. This is evident from Fig.7.8(a), where the value of weight becomes saturated after the 6^{th} decision cycle. The average state-based observability O_a and controllability C_a are shown in Fig.7.8(b), where the variations infer that the initial visual attention is made on the basis of vague decision (i.e., lower values of O_a) and the bottom-up bias tries to make

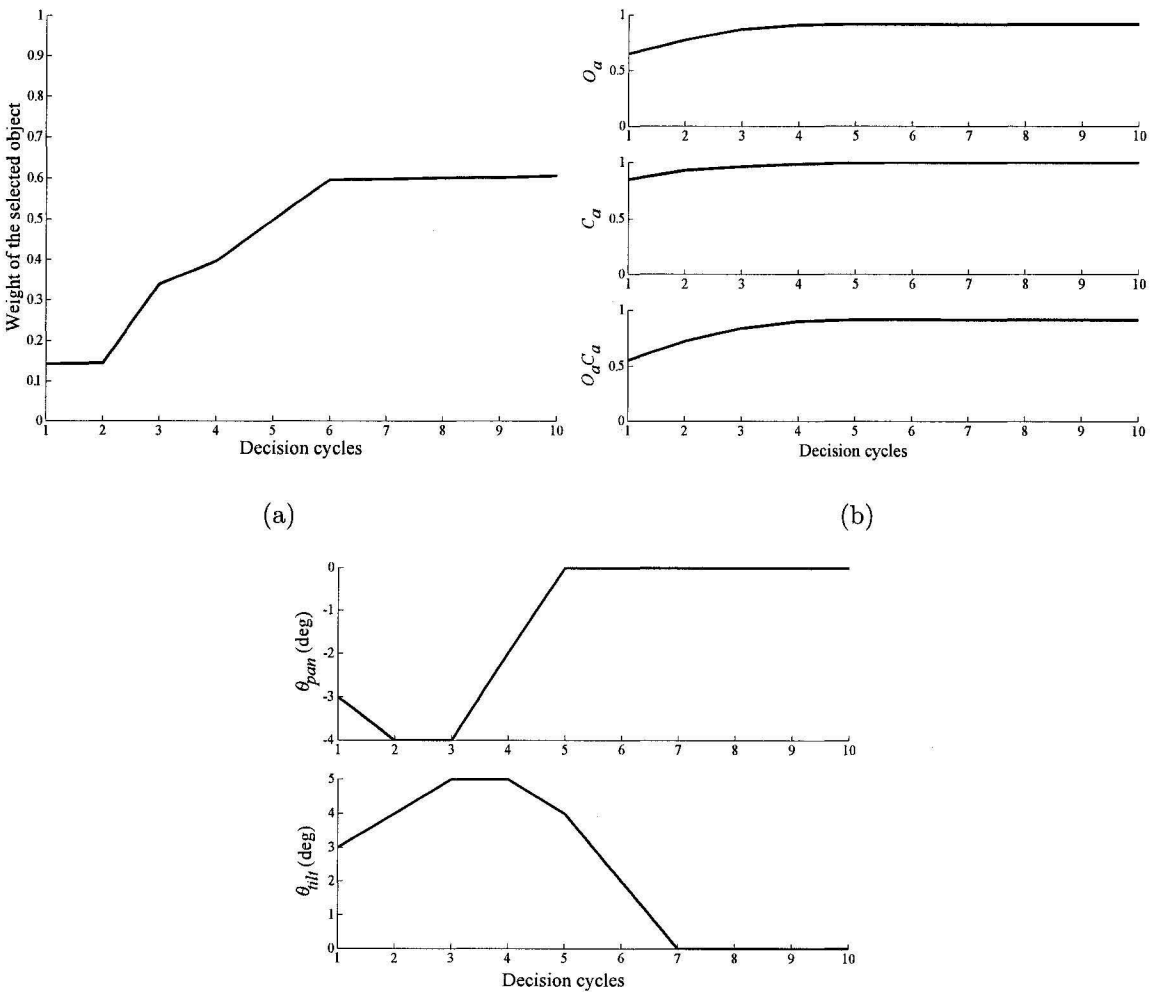
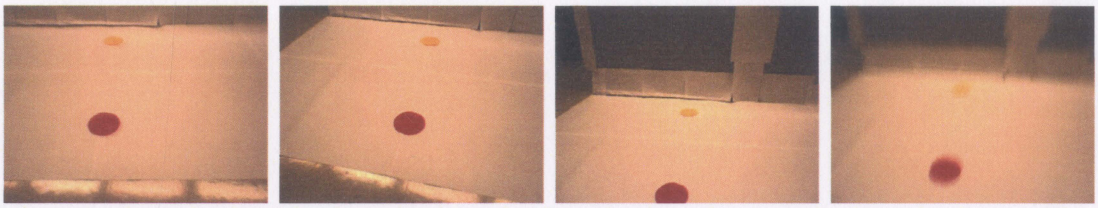


Figure 7.8: Parameter variations in experiment 1

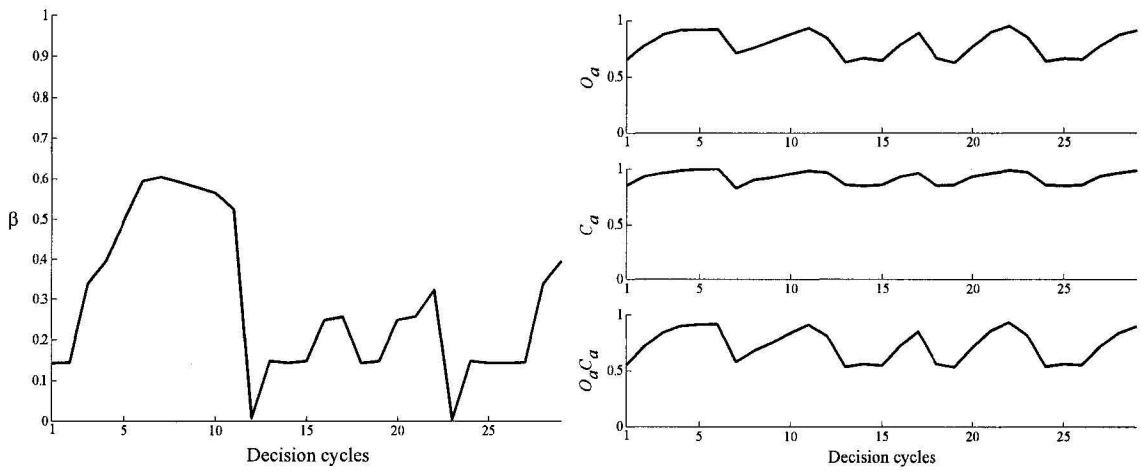


(a) (b) (c) (d)

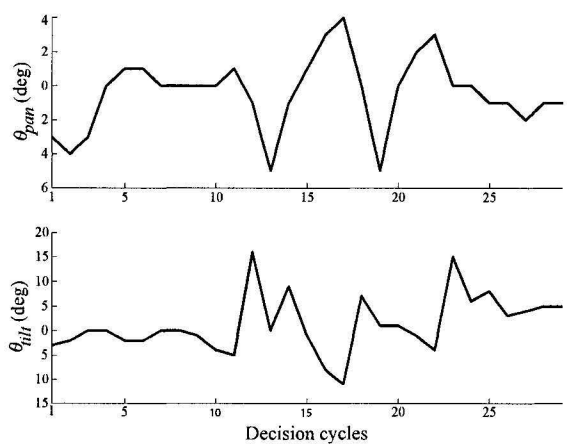
Figure 7.9: Objects in experiment 2

abrupt changes in focus of attention (i.e., lower values of C_a). These effects are offset by modulating the pan and tilt commands with O_aC_a, which produce slower camera motion in presence of vague decision and abrupt change in focus of attention. The pan and tilt commands converge to 0 when the attended object is positioned at the center of the visual field.

Fig.7.9(a) depicts the setup for experiment 2, where two objects, colored red and yellow, are placed on the white background. According to the bottom-up bias the red object is first focused and is positioned at the center of visual field (see Fig.7.9(b)). The novelty bias is automatically activated after the 10^{th} decision cycle and the yellow object is attended by the robot as shown in Fig.7.9(c). However, Fig.7.10(a) shows that the weight of the selected object start decreasing before the 20^{th} decision cycle. The attention to the yellow object is distracted because of noisy perception as shown in Fig.7.9(d). Two new patterns are formed corresponding to the red and yellow objects. Consequently, the attention is temporarily governed by the newly formed perception of red object from the 17^{th} to 19^{th} decision cycles. The yellow object is focused again at the 20^{th} decision cycle and the robot continues attending it until the total number of attended decision cycles (including the previous ones) is 10 for the yellow object. Fig.7.10(b) and 7.10(c) show the corresponding measures of average observability and controllability, and the modulated pan-tilt commands. The modulated pan-tilt commands produce slower camera motion when the visual

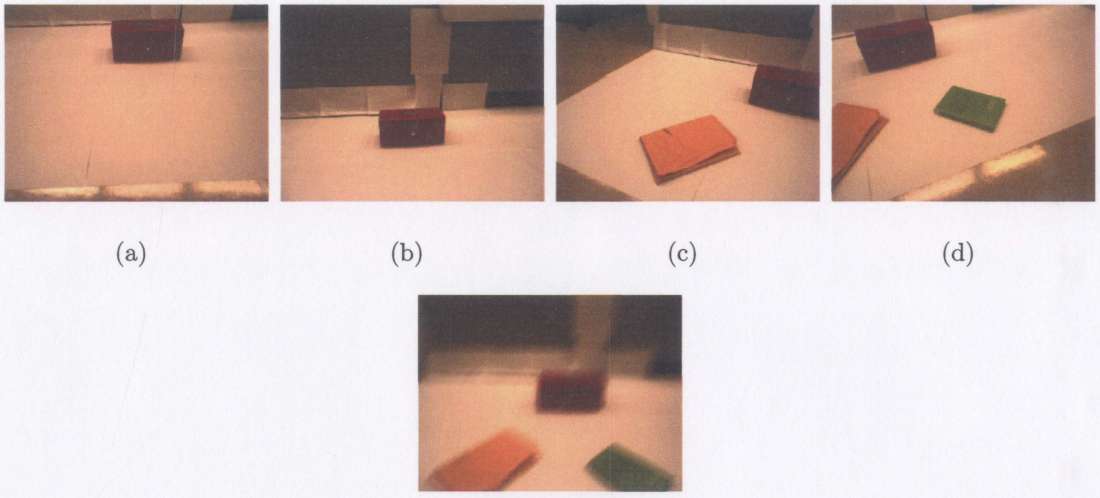


(a) (b)



(c)

Figure 7.10: Parameter variations in experiment 2



(a) (b) (c) (d)

(e)

Figure 7.11: Objects in experiment 3

attention is changed from one object to another.

In experiment 3, the robot learns three objects one after another. Fig.7.11(a) shows the initial setup of the experiment, where a red box is placed alone on the white background. The weight of the selected object shown in Fig.7.12(a) describes that the box is learned around the 20^{th} decision cycle. Fig.7.11(b) shows that the focused red box is positioned at the center of visual attention. A yellow cardboard is then placed on the left side of the box, which is focused (see Fig.7.11(c)) and learned because of the novelty bias around the 45^{th} decision cycle. The novelty bias is again activated when a green book is placed on the right side of the red box. The green book is focused (see Fig.7.11(d)) and learned around the 65^{th} decision cycle. However, Fig.7.12(a) indicates that the value of the selected object is oscillatory when the green object is focused. This infers that the visual attention to the green object has been distracted due to noisy sensing as shown in Fig.7.11(e). This experiment also employs WM-bias for guided search in the visual field. The first WM-bias is applied in the form of location bias in favor of the green object around the 75^{th} decision

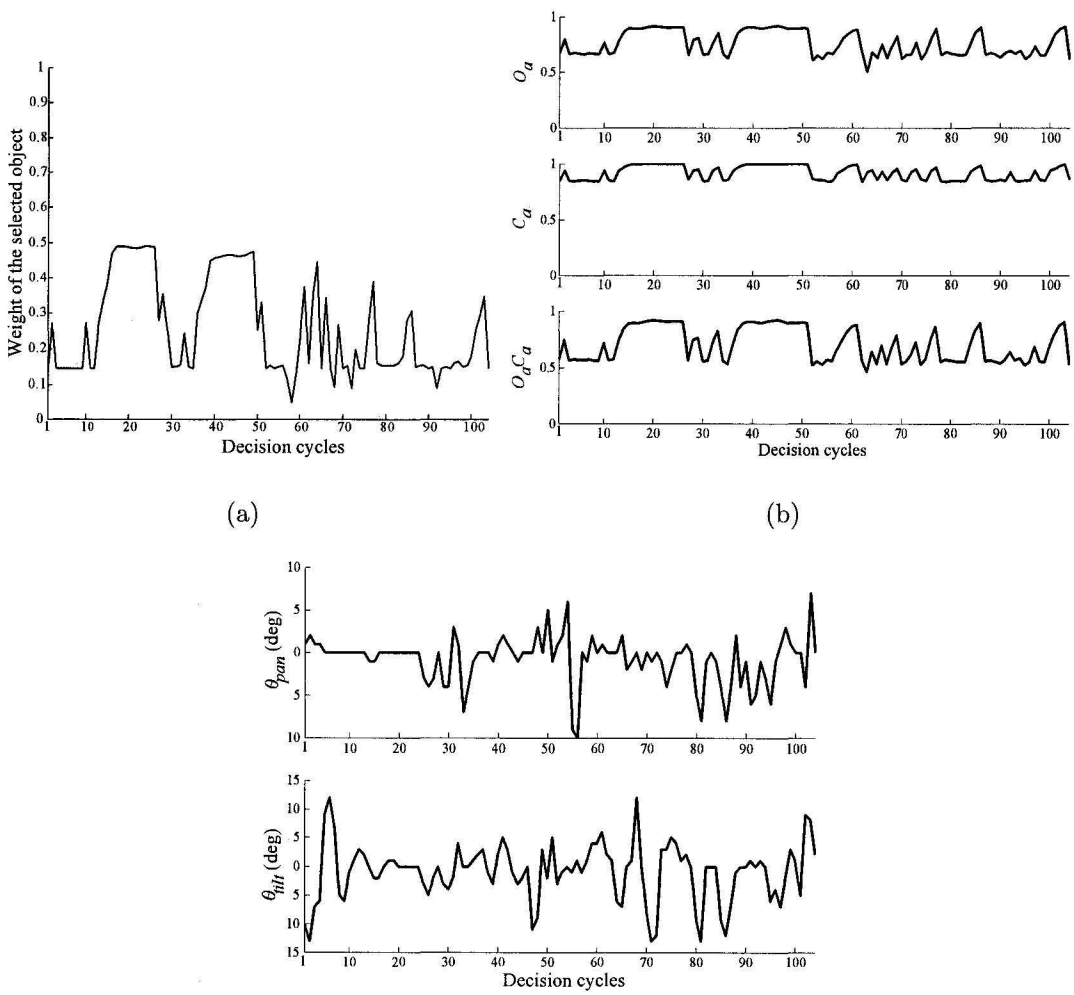


Figure 7.12: Parameter variations in experiment 3

cycle, which is represented by a peak in the value of weight around this decision cycle in Fig.7.12(a). The WM-bias is also applied in the form of feature-bias, where the yellow cardboard is first focused around the 85^{th} decision cycle and then, the red box is attended around the 100^{th} decision cycle in favor of the feature-bias. These feature biases are represented by two peak values of weight corresponding to the 85^{th} and 100^{th} decision cycles in Fig.7.12(a). The measures of average observability and controllability are shown in Fig.7.12(b) and the modulated pan-tilt commands are shown in Fig.7.12(c).

The experimental results reveal that the proposed architecture successfully combines the bottom-up and top-down biases for visual attention of the robot. The weight of the selected object is controlled using the FDES-based state structure which prevents abrupt changes in robotic visual attention. This method also employs system characteristics, such as observability and controllability, to produce desired motion commands to control the camera position, which focuses the attended object at the center of the visual attention. However, it is observed that the visual attention is distracted when the process of object detection and feature extraction is erroneous due to noisy perceptions as shown in Fig.7.9(d) and 7.11(e). To improve this situation, robust image processing techniques are required to filter out noisy perceptions. Moreover, appropriate parameter tuning of the ART2 prevents creating redundant clusters in presence of the noisy sensing.

7.5 Conclusion

This chapter presents a novel application of the proposed FDES-based architecture to model visual attention of a robot. The method attempts to implement the *Biased competitive hypothesis*, where the bottom-up bias is combined with the top-down bias for attention modeling. The bottom-up bias represents saliency of the objects' fea-

tures, whereas the top-down bias represents the relevance of an object with current behavioral state of the robot. In the proposed implementation of *biased competitive hypothesis*, FDES is used to model the sensory feedback from the environment to generate the object-saliency. The *Arbiter* combines the top-down bias with the object-saliency and selects an appropriate object for visual attention. The FDES-based architecture provides the opportunity to implement heuristic-based reasoning using state-transition structure of a FDES. It also employs fuzzy logic to incorporate deterministic vagueness of human reasoning. Moreover, state-based generation of the object saliency prevents abrupt change in focus of attention, which helps generating consistent motion commands for the camera. This method uses FDES-based observability and controllability to measure the decision vagueness and sudden change in the visual attention. These measurements are used to modulate the pan-tile commands of the camera to produce slower speed when the attention is suddenly distracted. Three real-world experiments are presented to validate different aspects of the proposed system.

Chapter 8

Conclusions & Future Perspectives

The main focus of this thesis is to investigate on behavior coordination mechanisms and to outline a novel coordination technique using the key features of the existing methods. This thesis sets three objectives to fulfill the proposed research goal. The rest of this chapter will summarize the research issues required to be addressed in order to fulfill these objectives.

8.1 Research summary based on Objective I

Objective 1 raises two research issues, first, investigation of the current literature to outline the key properties of a behavior coordinator and second, experimentation on existing methodologies to determine their shortcomings. The first research issue enables to outline following properties of a behavior coordinator, (a) both behavior arbitration and command fusion techniques should be combined in order to coordinate competitive and cooperative behaviors, (b) the coordinator should possess adequate means of modeling the current state of the world, (c) sensory uncertainties should be modeled using multi-valued logic, (d) persistent behavior selection should be performed, (e) the coordinator should provide the opportunity to accommodate

hierarchical decision-making for reactive action generation, (f) it should possess predictive decision-making capability for handling future environmental uncertainties, (g) it should provide satisfactory means of decision analysis, and finally (h) it should be modular to achieve robustness with a larger number of behaviors. This research issue also enables to propose a new classification of the existing behavior coordination techniques using the characteristics of knowledge representation and decision-making methods.

The second research issue includes experimentation, where different combinations of the existing methodologies are employed to inquire about the key aspects of a behavior coordinator. This experimentation leads to a novel attempt, where a FL-based controller is used in association with motor schema based behaviors for mobile robot navigation. The FL-based method uses fuzzy meta rules to generate the online weights for the motor schemas. It uses human reasoning to model the deterministic uncertainty of the noisy perception and reduces the possibility of inappropriate weight generation for the motor schemas. The experimental results demonstrate that fuzzy logic based approach overcomes the disadvantages of the traditional schema-based approaches, namely trap situations due to local minima, no passage between closely spaced obstacles, oscillations in the presence of obstacles and narrow passages. However, further investigation also reveals that 1) the FL-based method imposes a scalability problem when the system consists of both competitive and cooperative behaviors, 2) it does not provide any measure for decision analysis, and 3) it imposes a scalability problem when previous state information is incorporated for persistent decision-making.

8.2 Research summary based on Objective II

Objective 2 is the main focus of this thesis. It leads to the development of a novel behavior coordination architecture using Fuzzy Discrete Event System (FDES). This architecture addresses the shortcomings of the FL-based technique using the complimentary properties of Discrete Event System (DES). As a result, the combination of FL and DES provides the opportunity to integrate several key features of the existing behavior coordination techniques. The proposed method employs fuzzily defined events and state transition diagram to determine the importance (activity) of a behavior at a given environmental context. It uses supervisory approach to combine priority-based behavior ranking with context dependent behavior activity to integrate behavior arbitration and command fusion. The state transition structure of the proposed approach implements context rules for world state modeling and uses fuzzy state vectors and event matrices to model the current sensory information. This method provides suitable means of decision analysis using state-based observability and controllability, where observability denotes decision-vagueness based on the uncertainty associated with the sensory data and controllability indicates the decision-risk based on the dynamic changes within the observed environment. In this approach, system memory is preserved using fuzzy state vectors, which help making persistent decisions by preventing abrupt changes in behavior activity. The method employs FDES in a distributed manner, where separate FDESs are used to model the activity of a behavior. As a result, addition of a new behavior does not require further modifications of the existing FDESs. Hence, the FDES-based activity generation is modular. The proposed method also provides the opportunity of hierarchical decision-making by adopting multilevel behavioral decomposition, which reduces the overall computational complexity of the decision process and helps taking timely response against the dynamic changes in the observed environment. However, the proposed system

does not address predictive decision-making, which requires probabilistic inferences; whereas the FDES-based approach is intended to exploit multi-valued logic based possibility theory.

8.3 Research summary based on Objective III

The third objective validates the proposed architecture using real-world robotic experiments. A novel FDES-based architecture is presented for mobile robot navigation. The navigation system employs strategic behaviors to incorporate the effects of deliberative planning and uses reactive behaviors for dynamic obstacle avoidance. The experimental results show that the performance measures of the FDES-based system are unaffected even under changing or complex environments. The FDES-based system is able to produce oscillation-free and collision-less navigation in the experiments. The state-based observability and controllability phenomenons in FDES make it possible to produce modulated velocity and sampling frequency. Velocity modulation generates slower speed to prevent sharp turning of the robot in presence of dynamic obstacles and frequency modulation generates higher sampling rate for faster perceptions. Therefore, application of velocity and frequency modulation can slow down the robot's movement near obstacles, which provides enough time to analyze the updated sensory information and avoids occurrence of oscillations. A novel object-pulling operation with mobile robot is described to demonstrate the multi-level behavioral decomposition feature of the proposed architecture. The object-pulling task is divided into anchoring and navigation tasks, which are further decomposed into low-level behaviors. The robot, first, anchors the object and then navigates to the target location. The experimental results demonstrate that the FDES-based approach provides reliable execution of anchoring task and produces collision-free navigation to the target location as compared to a DES-based method. Finally, an FDES-based application

is presented to model the humanoid visual attention system according to the biased competitive hypothesis. The FDES-based architecture provides several characteristics to effectively combine the bottom-up bias (object-saliency) with the top-down influence (object bias generated from experience and user feedback) for robotic visual attention. It provides an opportunity to implement rule-based human reasoning using the transition structure of a FDES. This architecture uses fuzzy logic to estimate the deterministic vagueness of a feature contribution. The FDES-based system employs temporary memory of previous visual attention to prevent abrupt changes in visual attention and produces smooth motor commands for robotic motion. This method also employs observability and controllability to measure the decision vagueness and sudden change in the visual attention. The measurements are used to control the pan-tilt motion of the camera to produces slower speed when the attention is suddenly distracted.

8.4 Contributions

To summarize, this thesis made following contributions in behavior-based robotics while fulfilling the three research objectives.

1. Contributions from Objective I:

   (a) A new knowledge-based classification is proposed for behavior arbitration and command fusion based behavior selection mechanisms.

   (b) A novel behavior-based approach of mobile robot navigation is proposed using FL-based coordination technique for motor schema [41] based behaviors.

2. Contributions from Objective II:

This research objective enables to devise a novel behavior-based robotic control approach using FDES. The proposed behavior coordinator

(a) combines the characteristics of behavior arbitration and command fusion techniques to facilitate the coordination of competitive and cooperative behaviors,

(b) uses event-driven architecture for world states modeling,

(c) deploys FL to handle sensory uncertainty.

(d) is capable of making persistent behavior selection using system memory,

(e) provides the opportunity of multi-level behavioral decomposition for fast decision-making,

(f) provides means of decision analysis using observability and controllability, and

(g) employs modular architecture for behavior coordination.

3. Contributions from Objective III:

   This research objective enables to implement three robotic applications using the proposed FDES-based architecture. The applications include

   (a) mobile robot navigation,

   (b) object-pulling operation by mobile robots, and

   (c) visual attention of mobile robots.

This thesis leads to the following technical papers that report the contributions of the proposed work.

1. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Behavior Modulation Technique in Mobile Robotics Using Fuzzy Discrete Event System , in *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 903-916, 2006.

2. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Distributed Fuzzy Discrete Event System for Robotic Sensory Information Processing ," in the Special Issue of *Expert System on Advances in Intelligent Information Processing*, vol. 23, no. 5, pp. 273-289.

3. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "An Application of Fuzzy Discrete Event System to Control a Box-pulling Robot ," Under review in *IEEE Transactions on Robotics.*

4. Rajibul Huq, Momotaz Begum, George K. I. Mann, and Raymond G. Gosine, "A Visual Attention Model for Cognitive Robots Using Fuzzy Discrete Event System ," Under review in *IEEE Transactions on Robotics.*

5. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Mobile Robot Navigation Using Motor Schema and Fuzzy Context Dependent Behavior Modulation ," Under review in *Applied Soft Computing (ELSEVIER).*

6. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Behavior based robot control: A survey towards supervisory behavior coordination," Under review in Robotics and Autonomous Systems.

7. Rajibul Huq, Momotaz Begum, George K. I. Mann, and Raymond G. Gosine, "Biased Competitive Model of Humanoid Visual Attention Using Fuzzy Discrete Event System ," in the Proceedings of *IEEE International Conference on Robotics and Biomimetics*, 2006.

8. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "A Behavior-based control of a Box-pulling robot using Fuzzy Discrete Event System ," in the Proceedings of *IEEE International Conference on Intelligent Robots and Systems*, 2006.

9. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Behavior-based Robot Control Using Fuzzy Discrete Event System ," in the Proceedings of *IEEE International Conference on Fuzzy Systems*, 2006, pp. 1146–1153.

10. Rajibul Huq, George K. I. Mann, and Raymond G. Gosine, "Fuzzy Discrete Event System Based Behavior Modulation in Mobile Robotics ," in the Proceedings of *IEEE International Conference on Intelligent Robots and Systems*, 2005, pp. 2241–2246.

11. Rajibul Huq, George Mann, and Raymond Gosine, "Integrated Motion Planning for Indoor Mobile Robots Using Motor Schema and Adaptive Fuzzy Behavioral Modulation ," in the Proceedings of *IEEE International Conference on Intelligent Robots and Systems*, 2004, pp. 2985–2990.

12. Rajibul Huq, Raymond Gosine, and George Mann, "An Integrated Approach for Multilane Robot Motion Planning Using Motor Schema and Fuzzy Logic ," in the Proceedings of *IEEE International Conference on Systems, Man, and Cybernetics*, 2004, pp. 5301–5306.

8.5 Future research directions

This work leads to several potential research areas. The following is a brief discussion on these research directions.

- The proposed method lacks the opportunity of online parameter adaption to attain robust performance in dynamic environments. Consequently, it requires developing online parameter tuning methods to incorporate dynamic changes in the robot's workspace. As an example, the MFs used in a FDES can be tuned dynamically with NN-based or GA-based techniques [58, 59] to assist the

current goal of the robot.

- The proposed FDES-based system only models the deterministic uncertainty with possibility theory to update the activity state of a behavior. However, robotic applications also require modeling of non-deterministic uncertainty with probability theory to control the activity states of a behavior using predictive decision-making. This requirement directs to the future research on combining FDES with Probabilistic Discrete Event Systems (PDES) [135] to integrate both the possibility and probability theory in the same frame.

- DES has two major knowledge representation techniques: Finite State Automata (FSA) and Petri Net [69]. The present work exploits FSA-based DES combined with fuzzy logic for behavior coordination. Consequently, it remains open to investigate on fuzzy Petri Net [136] for behavior-based robotic control.

- The FDES-based approach is extendable for multiagent coordination. In a multiagent system, a task is assigned to a team of robots, where each robot accomplishes a subtask of the given goal. Hence, a robot is similar to a behavior and the multiagent coordination problem is identical to the behavior coordination problem [137]. As a result, the present research provides an opportunity to investigate on multiagent coordination using FDES.

Bibliography

[1] N. Nilsson, "Shakey the robot," Menlo Park, CA, Tech. Rep. SRI Tech. Nore 323, 1984.

[2] H. Durrant-Whyte, "Integration, coordination, and control of multi-sensor robot systems," Ph.D. dissertation, Univ. of Pennsylvania, Philadelphia, PA, 1986.

[3] A. Elfes, "A sonar-based mapping and navigation system," in *Proc. IEEE International Conference on Robotics and Automation*, San Francisco, CA, 1986, pp. 1151–1156.

[4] M. Turk, D. Morgenthaler, K. Gremban, and M. Marra, "VITS-A vision system for autonomous land vehicle navigation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, no. 3, pp. 342–361, May 1988.

[5] H. Moravec, "The standford cart and the cmu rover," I. Cox and G. Wilfong, Eds. Springer-Verlag, 1990.

[6] D. Pomerlean, *Neural Network Perception for Mobile Robot Guidance.* Boston, MA: Kluwer Academic Publishing, 1993.

[7] C. Thorpe, Ed., *Vision and Navigation: The Carnegi Mellon Navlab.* New York: Kluwer, 1993.

[8] T. Jochem, D. Pomerlean, D. Kumar, and J. Armstrong, "PANS: A portable navigation platform," in *Proc. IEEE Symposium on Intelligent Vehicles*, Detroit, Michigan, Sept. 1995, pp. 107–112.

[9] J. K. Rosenblatt and J. Handler, "Architectures for mobile robot control," *Advances in Computers*, vol. 48, pp. 315–353, 1999.

[10] P. Maes, "Behavior-based artificial intelligence," in *Proc. Second Conference on Simulated and Adaptive Behavior.*

[11] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Automat.*, vol. 2, no. 1, pp. 14–23, Mar. 1986.

[12] P. Agre and D. Chapman, "Pengi: An implemetation of a theory of activity," in *Proc. 6th National Conference on Artificial Intelligence (AAAI-87)*, Seattle, Wash, July 1987, pp. 268–272.

[13] M. Schoppers, "Universal plans for reactive robots in unpredictable environments," in *Proc. 10th International Joint Conference on Artificial Intelligence*, Milan, Italy, 1987, pp. 1039–1046.

[14] M. P. Georgeff and A. L. Lansky, "Reactive reasoning and planning," in *Proc. AAAI-87 6th National Conference on Artificial Intelligence*, 1987, pp. 677–682.

[15] L. P. Kaelbling, "Goals as parallel program specifications," in *Proc. 7th National Conference on Artificial Intelligence (AAAI-88)*, St. Paul, Minn, 1988, pp. 60–65.

[16] M. Drummond, "Situated control rules," in *Proc. First International Conference on Principle of Knowledge Representation and Reasoning*, 1989, pp. 103–113.

[17] L. Spector and J. Hendler, "The supervenience architecture," in *Proc. AAAI Fall Symposium on Sensory Aspects of Robotic Intelligence*, Nov. 1991, pp. 93–100.

[18] E. Gat and G. Dorais, "Robot navigation by conditional sequencing," in *Proc. IEEE International Conference on Robotics and Automation*, 1994, pp. 1293–1299.

[19] F. Hoffmann, "An overview on soft computing in behavior based robotics," in *Proc. International Fuzzy Systems Association World Congress*, Istanbul, Turkey, 2003, pp. 554–551.

[20] P. Maes, "Situated agents can have goals," *Robotics and Autonomous Systems*, vol. 6, pp. 49–70, 1990.

[21] D. W. Payton, "An architecture for reflexive autonomous vehicle control," in *Proc. IEEE Int. Conf. Robot. Automat.*, San Francisco, CA, Apr. 1986, pp. 1838–1845.

[22] J. R. Firby, "An investigation into reactive planning in complex domains," in *Proc. 6th National Conference on Artificial Intelligence (AAAI-87)*, Seattle, Wash, July 1987, pp. 202–206.

[23] L. P. Kaelbling, "An architecture for intelligent reactive systems," in *Reasoning About Actions and Plans*, M. P. Georgeff and A. L. Lansky, Eds. Morgan Kaufmann, 1987, pp. 395–410.

[24] R. G. Simmons, "Structured control for autonomous robots," *IEEE J. Robot. Automat.*, vol. 10, no. 1, pp. 34–43, Feb. 1994.

[25] R. R. Murphy, *Intrduction To AI Robotics.* Cambridge, Massachusetts, London, England: A Bradford Book, The MIT Press, 2000.

[26] P. Maes, "Modeling adaptive autonomous agents," in *Artificial Life Journal*, C.Langton, Ed., vol. 1, no. 1.

[27] P. Maes and R. A. Brooks, "Learning to coordinate behaviors," in *Proc. AAAI*, Boston, MA, Aug. 1990, pp. 796–802.

[28] L. J. Lin, "Reinforcement learning for robots using neural networks," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 1993.

[29] J. R. Pimental, D. Gachet, L. Moreno, and L. A. Salichs, "Learning to coordinate behaviors for real-time path planning of autonomous systems," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 1993, pp. 541–546.

[30] A. Bonarini, "Learning to coordinate fuzzy behaviors for autonomous agents," in *Proc. European Conference of Fuzzy Information Technology*, Aachen, Germany, Oct. 1994, pp. 475–479.

[31] M. Humphrys, "Action selection methods using reinforcement learning," Ph.D. dissertation, Computer Laboratory, University of Cambridge, 1997.

[32] C. Ye, N. H. C. Yung, and D. Wang, "A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance," *IEEE Trans. Syst., Man, Cybern. B*, vol. 33, no. 1, pp. 17–27, 2003.

[33] P. Pirjanian, "Multiple objective action selection and behavior fusion using voting," Ph.D. dissertation, Aalborg Univ., Aalborg, Denmark, Aug. 1998. [Online]. Available: http://www.robotics.usc.edu/~paolo/publications

[34] P. Maes, "Situated agents can have goals," *Robotics and Autonomous Systems*, vol. 6, pp. 49–70, 1990.

[35] J. Košecká and R. Bajcsy, "Discrete event systems for autonomous mobile agents," *Robotics and Autonomous Systems*, vol. 12, pp. 187–198, 1994.

[36] R. Arkin and D. MacKenzie, "Temporal coordination of perceptual algorithms for mobile robot navigation," *IEEE J. Robot. Automat.*, vol. 10, no. 3, pp. 276–286, June 1994.

[37] M. K. Sahota, "Action selection in robotics in dynamic environments through inter-behavior bidding," in *Proc. Third International Conference on Simulation of Adaptive Behavior*, Brighton, England, 1994, pp. 138–142.

[38] S. Kristensen, "Sensor planning with bayesian decision theory," *Robotics and Autonomous Systems*, vol. 19, pp. 273–286, Mar. 1997.

[39] M. Egerstedt, "Behavior based robotics using hybrid automata," in *Proc. Third International Workshop on Hybrid Systems: Computation and Control*, 2000, pp. 103–116.

[40] O. Khatib, "Real-time obstacle avoidance for robot manipulator and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[41] R. C. Arkin, "Motor schema-based navigation for a mobile robot," *International Journal of Robotics Research*, vol. 8, no. 4, pp. 92–112, 1989.

[42] G. Schöner, M. Dose, and C. Engels, "Dynamics of behavior: Theory and application for autonomous robot architectures," *Robotics and Autonomous Systems*, vol. 16, no. 2, pp. 213–246, 1995.

[43] J. K. Rosenblatt, "The distributed architecture for mobile navigation," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, pp. 339–360, 1997.

[44] J. Reikki and J. Roning, "Reactive task execution by combining action maps," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Sept. 1997, pp. 224–230.

[45] J. Yen and N. Pfluger, "A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 6, pp. 971–978, June 1995.

[46] P. Pirjanian, "Behavior coordination mechanisms - state-of-the-art," University of Southern California, Tech. Rep. IRIS-99-375, Oct. 1999.

[47] A. Ram, R. C. Arkin, *et al.*, "Cased-based reactive navigation: A cased-based method for on-line selection and adaption of reactive control parameters in autonomous robotic systems," Georgia Institute of Technology, Atlanta, Georgia, USA, Tech. Rep. GIT-CC-92/57, 1992.

[48] A. Saffiotti, K. Konolige, and E. Ruspini, "A multivalued-logic approach to integrating planning and control," *Articial Intelligence*, vol. 76, pp. 481–526, 1995.

[49] F. Michaud, "Selecting behaviors using fuzzy logic," in *Proc. IEEE 6th Int. Conf. Fuzzy Syst.*, July 1997, pp. 585–592.

[50] A. Abreu and L. Correia, "Behavior based decision control in autonomous vehicles: A fuzzy approach using khepera," in *Proc. IEEE 9th Int. Conf. Fuzzy Syst.*, May 2000, pp. 140–145.

[51] A. Bonarini, G. Invernizzi, T. H. Labella, and M. Matteucci, "An architecture to coordinate fuzzy behaviors to control an autonomous robot," *Fuzzy Sets and Systems*, vol. 134, no. 1, pp. 101–115, 2003.

[52] P. Vadakkepat, O. C. Miin, X. Peng, and T. H. Lee, "Fuzzy behavior-based control of mobile robots," *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 4, pp. 559–565, Aug. 2004.

[53] E. Tunstel, "Coordination of distributed fuzzy behaviors in mobile robot control," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 1995, pp. 4009–4014.

[54] P. Pirjanian and M. Mataric, "A decision-theoretic approach to fuzzy behavior coordination," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat.*, Nov. 1999.

[55] S. Khatoon and S. Khatoon, "Behavior coordination of autonomous mobile robot navigation by neuro-fuzzy sytem," in *Proc. the IEEE 31st Annual Northeast Bioengineering Conf.*, Apr. 2005, pp. 56–60.

[56] A. Safotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Computing*, vol. 1, no. 4, pp. 180–197, 1997.

[57] E. Tunstel and M. Jamshidi, "On genetic programming of fuzzy rule based systems for intelligent control," *International Journal of Intelligent Automation & Soft Computing*, vol. 2, no. 2, pp. 273–284, 1996.

[58] J. Cao, X. Liao, , and E. Hall, "Reactive navigation for autonomous guided vehicle using the neuro-fuzzy techniques," in *Intelligent Robots and Computer Vision XVIII: Algorithms, Techniques, and Active Vision*, D. P. Casasent, Ed., vol. 3837. Boston, MA, USA: The International Society for Optical Engineering (SPIE), 1999, pp. 108–117.

[59] N. Noguchi, J. F. Reid, Q. Zhang, and L. F. Tian, "Vision intelligence for precision farming using fuzzy logic optimized genetic algorithm and artificial neural

network," in *An ASAE Meeting Presentation UILU-ENG-98-7020.* Paper No. 983034, 1998.

[60] M. Scheutz, "Affective action selection and behavior arbitration for autonomous robots," in *Proc. International Conference on Artificial Intelligence*, 2002, pp. 334–340.

[61] G. Hoff and G. Bekey, "An architecture for behavior coordination learning," in *Proc. IEEE Int. Conf. Neural Networks*, Nov. 1995, pp. 2375–2380.

[62] N. Kubota, H. Masuta, and K. Taniguchi, "Behavioral dimension of multiobjective behavior coordination for a mobile robot," in *Proc. the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, July/Aug. 2001, pp. 374–379.

[63] D. D. Tsankova, "Emotionally influenced coordination of behaviors for autonomous mobile robots," in *Proc. the IEEE International International Symposium on Intelligent Systems*, Sept. 2002, pp. 92–97.

[64] T. Tyrell, "The use of hierarchies for action selection," in *From animals to animats 2: Proc. the Second International Conference on Simulation of Adaptive Behaviors*, J.-A. Meyer, H. L. Roitblat, and S. W. Wilson, Eds., 1993, pp. 138–147.

[65] E. Tunstel and M. Jamshidi, "Fuzzy logic and behavior coordination control strategy for autonomous mobile robot mapping," in *Proc. IEEE 4th Int. Conf. Fuzzy Syst.*, June 1994, pp. 514–517.

[66] J. K. Rosenblatt, "Optimal selection of uncertain actions by maximizing expected utility," *Autonomous Robots*, vol. 9, pp. 17–25, 2000.

[67] S. G. Goodridge, M. G. Kay, and R. C. Luo, "Multi-layered fuzzy behavior fusion for reactive control of an autonomous mobile robot," in *Proc. the IEEE International Conference on Fuzzy Systems*, July 1997, pp. 579–584.

[68] A. Saffiotti, E. Ruspiniand, and K. Konolige, "Blending reactivity and goal-directedness in a fuzzy controller," in *Proc. IEEE 2nd Int. Conf. Fuzzy Syst.*, 1993, pp. 134–139.

[69] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems.* Norwell, MA: Kluwer, 1999.

[70] F. Lin and H. Ying, "Modeling and control of fuzzy discrete event systems," *IEEE Trans. Syst., Man, Cybern. B*, vol. 32, no. 4, pp. 408–415, 2002.

[71] F. Lin, H. Ying, X. Luan, R. D. MacArthur, J. A. Cohn, D. C. Barth-Jones, and L. R. Crane, "Control of fuzzy discrete event systems and its applications to clinical treatment planning," in *Proc. 43rd IEEE Conf. Decis. Contr.*, Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 519–524.

[72] X. Luan, H. Ying, F. Lin, R. D. MacArthur, J. A. Cohn, D. C. Barth-Jones, H. Ye, and L. R. Crane, "A fuzzy discrete event system for hiv/aids treatment," in *Proc. 14th IEEE Int. Conf. Fuzzy Syst.*, May 2005, pp. 167–172.

[73] Y. Cao and M. Ying, "Supervisory control of fuzzy discrete event system," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 2, pp. 366–371, 2005.

[74] D. W. Qiu, "Supervisory control of fuzzy discrete event systems: A formal approach," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 1, pp. 72–88, 2005.

[75] D. MacKenzie, R. Arkin, and J. Cameron., "Multiagent mission specification and execution," *Autonomous Robots*, vol. 4, no. 1, pp. 29–52, 1997.

[76] J. H. Connel, "SSS: A hybrid architecture applied to robot navigation," in *Proc. the IEEE International Conference on Robotics and Automation*, Nice, France, May 1992, pp. 2719–2724.

[77] J. K. Rosenblatt and D. W. Payton, "A fine-grained alternative to the subsumption architecture for mobile robot control," in *Proc. IEEE Int. Joint Conf. Neural Networks*, vol. 2, Washington DC, June 1989, pp. 317–323.

[78] R. C. Arkin, "Motor schema-based navigation for a mobile robot: An approach to programming by behavior," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1987, pp. 264–271.

[79] E. W. Large, H. I. Christensen, and R. Bajcsy, "Scaling the dynamic approach to path planning and control: Competition among behavioral constraints," *International Journal of Robotics Research*, vol. 18, pp. 37–58, Jan. 1997.

[80] J. R. Firby, "Task networks for controlling continuous processes," in *Proc. 2nd International Conference on Artificial Intelligence Planning Systems*, 1994, pp. 49–54.

[81] J. K. Tsotsos, "Behaviorist intelligence and the scaling problem," *Artificial Intelligence*, vol. 75, no. 2, pp. 135–160, June 1995.

[82] E. Tunstel, T. Lippincott, and M. Jamshidi, "Behavior hierarchy for autonomous mobile robots: Fuzzy-behavior modulation and evolution," *International Journal of Intelligent Automation and Soft Computing*, vol. 3, no. 1, pp. 37–49, 1997.

[83] D. W. Payton, J. K. Rosenblatt, and D. M. Keirsey, "Plan guided reaction," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 6, pp. 1370–1382, 1990.

[84] J. K. Rosenblatt, "DAMN: A distributed architecture for mobile navigation," in *Proc. AAAI Spring Symposium on Lessons Learned from Implemented Software Architecturs for Physical Agents.* Standford, CA: AAAI Press, Mar. 1995, pp. 167–178.

[85] J. K. Rosenblatt and C. E. Thorpe, "Combining multiple goals in a behavior-based architecture," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Pittsburgh, PA, Aug. 1995, pp. 136–141.

[86] A. Saffiotti, "Fuzzy logic in autonomous robotics: Behavior coordination," in *Proc. IEEE 6th Int. Conf. Fuzzy Syst.*, 1997, pp. 573–578.

[87] M. A. Arbib, "Schema theory," in *The Encyclopedia of Artificial Intelligenc*, S. Shapiro, Ed. Wiley-Interscience, 1992, pp. 1427–1443.

[88] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, Sacramento, California, Apr. 1991, pp. 1398–1404.

[89] *Laser Measurement System*, 8 008 970/01-2002, SICK.

[90] *Pioneer* 3^{TM} *Operations Manual*, ActivMedia Robotics, 2004.

[91] J.-C. Latombe, *Robot Motion Planning.* Kluwer Academic Publisher, 1991.

[92] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 1989, pp. 1179–1187.

[93] ——, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE J. Robot. Automat.*, vol. 7, no. 3, pp. 278–288, 1991.

[94] J. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1991, pp. 790–796.

[95] R. Simmons, "The curvature-velocity method (CVM) for local obstacle avoidance," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1996, pp. 3375–3382.

[96] D. Fox, W. Burgrad, and S. Thrun, "Controlling synchro-drive robots with the dynamic window approach to collision avoidance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 1996, pp. 1280–1287.

[97] I. Ulrich and J. Borenstein, "VFH+: Reliable obstacle avoidance for fast mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1998, pp. 1572–1577.

[98] ——, "VFH*: Local obstacle avoidance with look-ahead verification," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2000, pp. 2505–2511.

[99] F. Zhang, D. Tan, and Z. Wu, "Obstacle avoidance for mobile robots based on relative coordinates," in *Proc. IEEE Int. Conf. Robot. Intell. Syst. Signal Processing*, 2003, pp. 616–621.

[100] ——, "Multiple obstacles avoidance for mobile robot in unstructured environments," in *Proc. IEEE Int. Conf. Robot. Automat. Mechatronics*, 2004, pp. 141–146.

[101] D. Vikenmark and J. Minguez, "Reactive obstacle avoidance for mobile robots that operate in confined 3d workspaces," in *Proc. IEEE Int. Conf. Mediterranean Electrotechnical*, 2006, pp. 1246–1251.

[102] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1993, pp. 802–807.

[103] W. Choi, D. J. Zhu, and J.-C. Latombe, "Contingency-tolerant motion planning and control," in *Proc. IEEE/RSJ Int. Workshop Intell. Robot. Syst.*, 1989, pp. 78–85.

[104] Y. K. Hwang and N. Ahuja, "A potential field approach to path planning," *IEEE J. Robot. Automat.*, vol. 8, no. 1, pp. 23–32, 1992.

[105] N. C. Tsourveloudis, K. P. Valavanis, and T. Hebert, "Autonomous vehicle navigation utilizing electrostatic potential fields and fuzzy logic," *IEEE J. Robot. Automat.*, vol. 17, no. 4, pp. 490–497, 2001.

[106] J. Minguez and L. Montano, "Nearness diagram navigation (ND): A new real time collision avoidance approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2000, pp. 2094–2100.

[107] H. Choset and J. Burdick, "Sensor based motion planning: The hierarchical generalized voronoi graph," Ph.D. dissertation, California Institute of Technology, 1996.

[108] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous Robot Vehicles*, vol. 8, pp. 167–193, 1990.

[109] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *IEEE J. Robot. Automat.*, vol. 17, no. 3, pp. 242–257, 2001.

[110] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research (JAIR)*, vol. 11, pp. 391–427, Nov. 1999.

[111] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *Journal of Robotics Research,*, vol. 20, no. 5, pp. 335–363, 2001.

[112] M. Begum, G. Mann, and R. Gosine, "Concurrent mapping and localization for mobile robot using soft computing techniques," in *Proc. IEEE/RSJ 9th Int.*

Conf. Intelligent Robots and Systems, Edmonton, Alberta, Canada, Aug. 2005, pp. 1059–1064.

[113] P. C. Chen and Y. K. Hwang, "Practical path planning among movable obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 1, Apr. 1991, pp. 444–449.

[114] S. Takagi and Y. Okawa, "Rule-based control of a mobile robot for the push-a-box operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3, Nov. 1991, pp. 1338–1343.

[115] Y. Okawa and K. Yokoyama, "Control of a mobile robot for the push-a-box operation," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 1, May 1992, pp. 761–766.

[116] Y. Okawa and J. Aoki, "Fuzzy control of a mobile robot for the push-a-box operation," in *Proc. the 4th IEEE Int. Conf. Tools with Artificial Intelligence*, Nov. 1992, pp. 172–179.

[117] A. Verma, J. Boyoon, and G. Sukhatme, "Robot box-pushing with environment-embedded sensors," in *Proc. IEEE Int. Symposium on Computational Intelligence in Robotics and Automation*, July 2001, pp. 212–217.

[118] R. Emery and T. Balch, "Behavior-based control of a non-holonomic robot in pushing tasks," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 3, May 2001, pp. 2381–2388.

[119] P. K. Agarwal, J.-C. Latombe, R. Motwani, and P. Raghavan, "Nonholonomic path planning for pushing a disk among obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 4, Apr. 1997, pp. 3124–3129.

[120] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability and planning," *Int. J. Robotics Research*, vol. 15, no. 6, pp. 533–556, Dec. 1996.

[121] S. Yamada and J. Saito, "Adaptive action selection without explicit communication for multirobot box-pushing," *IEEE Trans. Syst., Man, Cybern. C*, vol. 31, no. 3, pp. 398–404, 2001.

[122] R. Desimone, "Visual attention mediated by biased competition in extrastriate visual cortex," *Philosophical transactions of the Royal Society of London, Series B, Biological sciences*, vol. 353, p. 1245 1255, 1998.

[123] G. Deco, *Biased competitive mechanisms for visual attention in a multimodular neurodyanmical system.* LNAI 2036: Springer-Verlag Berlin, 2001, pp. 114–126.

[124] G. Deco and E. T. Rolls, "A neurodynamical cortical model of visual attention and invariant object recognition," *Vision Research*, vol. 44, p. 621 642, 2004.

[125] S. Kastner and L. G. Ungerleider, "The neural basis of biased competition in human visual cortex," *Vision Research*, vol. 40, p. 1489 1506, 2000.

[126] A. M. Treisman and G. Gelade, "A feature intigration theory of attention," *Cognitive Psychology*, vol. 12, p. 97 136, 1980.

[127] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature Reviews: Neuroscience*, vol. 2, p. 194 203, 2001.

[128] ——, "A saliency based search mechanism for overt and covert shift of visual attention," *Vision Research*, vol. 40, p. 1489 1506, 2000.

[129] R. Desimone and J. Duncan, "Neural mechanisms of selective visual attention," *Annual Reviews of Neurscience*, vol. 18, p. 193 222, 1995.

[130] M. I. Posner, C. R. R. Snyder, and B. J. Davidson, "Attention and the detection of signals," *Journal of Experimental Psychology: General*, vol. 109, pp. 160–174, 1980.

[131] J. Duncan, "Selective attention and the organization of visual information," *Journal of Experimental Psychology*, vol. 113, pp. 501– 513, 1984.

[132] J. M. Wolfe, "Visual search in continuous naturalistic stimuli," *Vision Research*, vol. 34, p. 1187 1195, 1994.

[133] J. H. Reynolds, L. Chelazzi, and R. Desimone, "Competitive mechanisms subserve attention in macaque areas V2 and V4," *Journal of Neuroscience*, vol. 19, pp. 1736– 1753, 1999.

[134] G. Carpenter and S. Grossberg, "ART2: Self-organization of stable category recognition codes for analog input patterns," *Applied Optics*, vol. 26, no. 23.

[135] Y. Li, F. Lin, and Z. H. Lin, "Supervisory control of probabilistic discrete-event systems with recovery," *IEEE Trans. Automat. Contr.*, vol. 44, no. 10, pp. 1971–1975, 1999.

[136] H. H. Ammar and L. Yu, "Fuzzy marking Petri Nets: Concepts and definition," in *Proc. IEEE Int. Symp. Intell. contr.*, Aug. 1995, pp. 291–297.

[137] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 2, May 2002, pp. 1217 – 1222.