Adaptive Co-operative Mobile Robots

A thesis

submitted to the

Cardiff University

in candidature for the degree of

Doctor of Philosophy

by

Medhat Hussein Ahmed Awadalla

Intelligent Systems Laboratory

Manufacturing Engineering centre

Cardiff University

United Kingdom

2005

UMI Number: U584751

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI U584751
Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.
Microform Edition © ProQuest LLC.
All rights reserved. This work is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Synopsis

This work proposes a biologically inspired collective behaviour for a team of co-operating robots. Collective behaviour is achieved by controlling the local interactions among a set of identical mobile robots, each robot performing a set of simple behaviours in order to realise group goals. A modification of the subsumption architecture is proposed for implementing control of individual robots. This architecture is adopted because it is computationally inexpensive and potentially suitable for low-level reactive and reflexive behaviours.

In this scenario, the individual behaviours of the robots have different aims, which may cause conflict. To address this issue, a fuzzy logic-based approach for multiple behaviour coordination within each robot is proposed.

The work also focuses on the development of intelligent multi-agent robot teams capable of acting autonomously and of collaborating in a dynamic environment to achieve team objectives. A knowledge-based software architecture is proposed that enables these robots to select co-operative behaviours and to adapt their performance during the specified time of the mission. These abilities are important because of uncertainties in the environmental conditions and because of possible functional failures in some team members. Improvement in team performance is achieved by updating the control of the robots based on knowledge acquired on-line. This architecture is implemented in a simulated team of mobile robots performing a proof-of-concept collaborative task. The results show a significant improvement in overall group performance and the robot team is able to achieve adaptive cooperative control despite dynamic changes in the environment and variation in the capabilities of the team members. Finally, a task involving real mobile robots is undertaken to demonstrate a practical, though simplified, implementation of the proposed collective behaviour.

Dedication

This dissertation is dedicated to my mother, my wife, my son Mohannad and my family who gave me my aspirations during this work.

Acknowledgements

I would like to thank my supervisor Prof. D. T. Pham for his excellent supervision, continuous encouragement and support. He was a brilliant supervisor.

I also want to thank all my colleagues at the Intelligent System Laboratory, especially Shankir, Jerome, Mark, Slava, Jan, Dayal, Fahmy, Sameh, Afshin, who were very good to me and very helpful whenever I needed them.

Contents

List of Figures

Chapter 2:

Chapter 3:

Chapter 4:

Chapter 5:

Chapter 6:

ABBREVIATIONS

A/D	Analogue to digital converter
ASMs	Action selection mechanisms
ASP	Action selection problem
BP	Back propagation learning algorithm
COA	Centre of area
COG	Centre of gravity
DAMN	Distributed architecture for mobile navigation
DES	Discrete event systems
FLS	Fuzzy logic system
FSA	Finite state automata
I/O	Input/output address
IR	Infrared sensor
MAC	Mark and cover motion rule
MODM	Multiple objective decision making
MOM	Mean of the maximal value
R2D2	Relative robot direction detector
VLSI	Very large scale integration
SO	Self-organising behaviour

NOMENCLATURE

Chapter 2:

U	Total potential field
Uatt	Goal attractive potential field
Urep	Obstacle repletion potential field
BP	backpropagation
DA-FNN	fuzzy neural network with differentiable activation functions
FLC	fuzzy logic controller
FLS	fuzzy logic system
GMP	generalised modus ponens
GMT	generalised modus tolens
NB	negative big
NS	negative small

Chapter 4:

α_i	Relative applicability of the behaviour
$\alpha 1$ and $\alpha 2$	Relative applicability for behaviourl and behaviour 2
B1 and B2	Robot behaviours
u^*	Defuzzified crisp output
θ	Target angle
R1 and R2	Fuzzy rules
μ	Membership function degree
AND	Fuzzy conjunction connector
Min	Fuzzy implication

Chapter 5:

PB	positive big
$M_{X_I}^{J}$	The membership function of the jth term at the term set describing the ith fuzzy variable
m_{ij} and σ_{ij}	Centre (or mean) and width (or variance) of bell-shaped function
O_i^k	Activation function of the ith node at the kth layer of the network
u_i^n	Input of the nth layer node corresponding to the ith fuzzy variable (the superscript n corresponds to the layer number)
w_{ij}^n	Link weight at the nth layer
η	Learning rate

N_i	Number of terms at the ith term set
$\mathbf{T(x)} = \{T_x^1, T_x^2, \ldots, T_x^N\}$	Term set of x
w_{nij}^5	Link weight that connects the ith output numerator node to the output term nodes
w_{dij}^5	Link weight that connects the ith output denominator node to the output term nodes
δ_{ij}^5	Propagated error from layer five to the jth node at the ith term set at layer four
δ_{ri}^4	Propagated error from a node at the ith output term set at layer four to the rth rule node at layer three
δ_r^4	Total propagated error from layer four to the rth rule node at layer three
δ_{ijm}^3	Propagated error from the mth rule node of the rule nodes that share the same jth term node at the ith input term set at layer two
δ_{ij}^3	Total propagated error from layer three to the jth term node at the ith term set at layer two
δ_{ij}^2	Propagated error from the jth term node at the ith term set at layer two to the ith input node at layer one
δ_i^2	Total propagated error from layer two to the ith input node at layer one
η	Learning rate
$\mathbf{M(x)} = \{M_x^1, M_x^2, \ldots, M_x^N\}$ M_x^j	Set of membership functions that is associated with the linguistic terms at the term set
$f^k(.)$ and $a^k(.)$	Aggregation function and activation function of a node at the kth layer

m_{ij} and σ_{ij}	Centre (or mean) and width (or variance) of the bell-shaped function of the jth term of the ith input linguistic variable
w^6_{ni} and w^6_{di}	Link weights associated with each output variable node at layer six
δ^6_{ni} and δ^6_{di}	Propagated error from layer six to numerator and denominator nodes at layer five
Δw	The change of the error
x_i, f^1_i, u^1_i, a^1_i and w^1_i	The ith input, aggregation function, net input, activation function and weight link of the ith node at layer one of the network
f^2_{ij}, a^2_{ij} and w^2_{ij}	Aggregation function, activation function and weight Link of the jth node of the ith group of nodes at layer two
f^3_r, u^3_r and a^3_r	Aggregation function, net input and activation function of the rth rule node at layer three
f^4_{ij}, u^4_{ij} and a^4_{ij}	Aggregation function, net input and activation function of the jth node of the ith group of nodes at layer four
f^5_{ni}, u^5_{ni} and a^5_{ni}	Aggregation function, net input and activation function of the ith numerator node at layer five
f^5_{di}, u^5_{di} and a^5_{di}	Aggregation function, net input and activation function of the ith denominator node at layer five
f^6_i, y_i and a^6_i	Aggregation function, crisp output and activation function of the ith output node at layer six
$\mu_A(x)$, $\mu_B(x)$, $\overline{\mu_A(x)}$ and $\overline{\mu_B(x)}$	Two fuzzy membership functions and their complements
E	Error function

r_i, r_k and T1	Robot r_i is aware that another robot r_k is working with a certain task T1
$x_i(t)$ and $y_i(t)$	The input pattern and the desired output pattern
$y(t)$ and $y_{net}(t)$	The desired output and current network output
w	Adjustable free parameter
a_i and $\overline{a_i}$	The i^{th} argument of *softmin* and *softmax* functions and its complement.
x, y and z	Linguistic variables

Chapter 1

Introduction

1.1. Preliminaries

The increasing current interest in mobile robots and in particular, mobile robot teams, is due to their applicability to a wide range of tasks. Example tasks suitable for mobile robots include nuclear and hazardous waste cleanup, mining - including material removal, search and rescue operations, mine sweeping for both military and humanitarian purposes, space missions, lifting and carrying of materials, surveillance and sentry, as well as underwater excavation.

A multi-robot system or team consists of a group of robots that can take specific roles within that organisation. The team may be composed of individual robots that either differ or are similar in structure and capability, i.e., either heterogeneous or homogeneous. Furthermore, it might have co-operative individuals working together towards a mutual goal, or it could be composed of rivals competing for some limited resources.

Biological agents, for example social insects, have been manifestly successful in exploiting the natural environment in order to survive and reproduce. Scientists are interested in understanding the strategies and tactics adopted by such natural agents to improve the design and functionality of computer-based artificial agents (robots). They observe how these social insects locally interact and co-operate to achieve common goals. It seems that these creatures are programmed in such a way that the required global behaviour is likely to emerge even though some individuals may fail to carry out their tasks.

In this work, a biologically inspired collective behaviour for a team of co-operating mobile robots is proposed. This behaviour emerges by controlling the local interactions between a number of identical mobile robots performing a set of simple behaviours. A modification of the subsumption architecture is proposed for implementing the control of the individual robots. The context of tracking a dynamic target is used to illustrate the proposed approach.

Since the control in behaviour-based systems is distributed among a set of specialised behaviours, the behaviours of the robots have different aims, which may cause conflict. Therefore, it is necessary to obtain an appropriate trade-off between the objectives of the robots that can potentially conflict.

To address this issue, a fuzzy logic-based approach for behaviour coordination is proposed. Fuzzy logic is robust in the presence of system and external perturbations.
It is straightforward to design and implement and efficient at representing knowledge for systems that deal with continuous variables. The fuzzy rule format makes it easy to write simple and effective behaviours for a variety of tasks without having to use complex mathematical models. It is possible to adopt a weighted combination of behaviours, which gives smoother control. The local nature of fuzzy rules allows one to identify the fired rules immediately and makes it possible to modify specific behaviours quickly.

The focus of development in co-operative robotic systems is to construct teams of robots able to accomplish missions that cannot easily be achieved, if at all, using single robots. The potential advantages of co-operative systems over single robot solutions include increased fault tolerance, simpler robot design, widened application domains, and greater solution efficiency. However, the use of multiple robots introduces additional issues of robot control that are not present in single robot solutions. Foremost among these is the question of how to achieve globally coherent and efficient behaviours from the interaction of robots lacking complete global information.

The robots need to be responsive to continual changes in the capabilities of robot team members and to changes in the state of the environment and mission. They should be aware of the actions of their team mates and have the ability to adapt to these dynamic changes. To address this issue, a knowledge-based software architecture is proposed that enables these robots to perform co-operative behaviours and adapt their performance during the specified time of the mission. This improvement in team performance is achieved by updating the control of the robots based on knowledge acquired on-line.

1.2. Research Objectives

The overall aim of this research was to develop intelligent multi-agent robot teams capable of acting autonomously and collaborating in a dynamic environment to achieve team goals.

To reach the aim of the research, the following objectives were set:

1- To use the collective behaviour of simple creatures to enable a team of robots to accomplish a complex task, such as a dynamic target tracking task, and to modify the subsumption architecture to be suitable for controlling the robots.

2- To find a new technique for solving conflicts between the contradictory behaviours of each robot.

3- To generate a knowledge-based software architecture for a team of robots to enable them to select appropriate actions and adapt the mission performance to deal with uncertainties in the environment or changes in the capabilities of team members.

4- To construct a team of mobile robots for real experiments to investigate the proposed techniques.

1.3. Organisation of the Thesis

The remainder of the thesis is organised as follows. Chapter 2 reviews the background literature relevant to the work presented in this thesis. This includes literature on co-operating mobile robots, the collective behaviour of simple creatures such as social insect colonies, the action selection problem (ASP), behaviour coordination, robot awareness and the basic components of fuzzy logic systems.

Chapter 3 describes the modification of the subsumption architecture and examines the collective behaviour of social insects as applied to co-operating mobile robots in the context of dynamic target tracking.

Chapter 4 proposes a fuzzy logic approach for behaviour coordination in multi-robot systems.

Chapter 5 discuses the development of a knowledge-based system for multi-agent robot teams and proposes a software architecture to enable multiple robots to perform co-operative behaviours and adapt their performance during the specified time of the mission.

Chapter 6 focuses on the design of a team of mobile robots and real experiments to illustrate an implementation of some of the proposed ideas.

Chapter 7 concludes the thesis and suggests areas for further investigation.

Chapter 2

Background

2.1. Preliminaries

During the last few decades, major research efforts have been directed towards improving the performance of individual mobile robots through the use of advanced sensors and actuators and the application of intelligent control algorithms. This was mainly driven by the need to perform increasingly complex real time tasks. As a result, individual mobile robots have become very sophisticated. More recently, an alternative approach to achieving complex tasks using multiple co-operative autonomous mobile robots has been investigated (Hu and Gan, 2005; Melhuish et al., 1998; Alami et al., 1998; Hu et al., 1998; Arkin, 1990; Mataric, 1998, 1996). Groups of mobile robots have been constructed, with the aim of studying such issues as group architecture, resource conflict, mobile robots co-operation and learning.

Collaboration increases the performance of a robot team without requiring significant modifications to individual robot capacities. Collaboration may be obtained using communication schemes, implicit communication via the environment or simple explicit communication schemes. By these means, the task accomplished by the team can be more complex and its performance enhanced without losing the autonomy or increasing the complexity of individual robots. In some cases (Ghanea-Hercock and Barnes, 1996; Boehringer et al., 1995; Mataric et al., 1995; Martinoli, 1999a, 1999b), the task may require collaboration for it to be successfully performed at all, where a single robot is not able to carry out the task alone. Such tasks can be defined to be "strictly collaborative".

2.2. Co-operative Mobile Robotics Classification

Research in the field of co-operative mobile robotics has increased substantially in recent years. Most of this research has concentrated on how to obtain the desired interaction dynamics between agents (robots) to increase the overall team performance. This field can be broadly categorised into two groups: "collective" (swarm type) co-operation and "intentional" co-operation.

Collective robotics is usually behaviour-based and characterised by distributed control of homogeneous robot teams. The desired collective behaviour is obtained as an emergent property of the interaction mechanism designed into each robot. The approaches developed and the problems addressed are for homogeneous robot teams only, in which each robot has the same capabilities and control algorithm. Additionally, issues of efficiency are largely ignored. The types of tasks implemented take inspiration from social insect societies, such as ants and bees.

A number of researchers have investigated 'swarm' robotics. Steels (1990) presented simulation studies of several dynamic systems to achieve emergent functionality with application to the collection of rock samples on a distant planet. Drogoul and Ferber (1992) undertook simulations of foraging and chain-forming robots. Arkin et al. (1993) implemented research concerned with sensing and communication for tasks such as foraging. Mataric (1992) described the implementation of group behaviours for physical robots such as dispersion, aggregation and flocking. Kube and Zhang (1992) detailed an emergent control strategy applied to a group of physical robots performing the task of locating and pushing a brightly-lit box.

"Intentional" robotics achieves co-operation among a limited number of typically heterogeneous robots performing several distinct tasks. Such systems normally employ either central control or a mix of central and distributed control.

In an intentional co-operative system, the robots often have to deal with some kind of efficiency constraint that requires a more directed type of co-operation than is found in collective co-operative systems. Furthermore, the robots are usually required to perform several distinct tasks. These missions thus usually require a smaller number of robots involved in more purposeful co-operation, although the individual robots involved are typically able to perform useful tasks on their own. Such systems require a robust allocation of subtasks to robots, to maximise the efficiency of the team, and proper coordination among team members, to allow them to complete their mission successfully. Most existing work on heterogeneous physical robots uses a traditional artificial intelligence approach, whereby the robot controller is divided into modules for sensing, world modelling, planning, and acting. This is the so-called sense-model-plan-act paradigm, in contrast to the functional decomposition method used in behaviour-based approaches.

Many researchers have investigated these intentional co-operative systems. Noreils (1993) addressed one such sense-model-plan-act control architecture, which includes three layers of control. The planner level manages coordinated protocols, decomposes tasks into smaller sub-units, and assigns the sub-tasks to a network of robots. The control level organises and executes the tasks of the robots. The functional level provides controlled reactivity. This architecture was applied to two mobile robots performing box pushing.

Caloud et al. (1990) presented another sense-model-plan-act architecture, which includes a task planner, a task allocator, a motion planner and an execution monitor. Each robot had goals to achieve, either based on its own current situation or via a request by another team member.

Asama et al. (1992) described a robot system called ACTRESS, which addressed the issues of communication, task assignment and path planning among heterogeneous robotic agents. Their approach revolves primarily around a negotiation framework, which allows robots to recruit help when needed. They demonstrated their architecture on mobile robots performing a box-pushing task.

In general, co-operative (both swarm and intentional) approaches to robotics should include mechanisms within the control software of each robot that allows team members to recover from dynamic changes in their environment or in the robot team. Researchers have recognised that a more promising approach for the development of co-operative control mechanisms is by the inclusion of learning algorithms (Hu and

Gu, (2005, 2004); Cragg and Hu, 2005; Acosta and Hu, 2003a; 2003b). Much work in particular has been carried out in the field of multi-agent learning (Minguez and Montano, 2005; Elfwing, 2004; Weiss et al., 1996; Mataric et al, 1995). Applications include predator/prey scenarios (Korf, 1992; Tan, 1993; Gasser et al., 1989; Levy and Rosenschein, 1992; Stephens and Merx, 1990), multi-robot soccer teams (Duhaut et al., 1998), and box-pushing tasks (Stilwell and Bay, 1993; Kube and Zhang, 1992; Sen et al., 1994).

2.3. Collective Behaviour of Social Insects

Collective behaviour is demonstrated in any type of system where patterns are determined not by some centralised body, but instead by the interactions of a group of decentralised bodies (Fong et al., 2003; Kristina and Aram, 2002). There is no need for centralised authority at all, nor for explicit communication between interacting bodies.

Collective behaviour demonstrates also a fundamentally important principle that has been beneficial to nature and humans alike, namely that some objectives are easier to accomplish in a group rather than by an individual. This interaction does not necessarily require a high level of intelligence, or even communication between the participating bodies, yet objectives may be accomplished that are outside the scope of an individual. Many examples of collective behaviour can be found in nature, e.g. flocking of birds, termites building enormous mounds, and ants collectively carrying a large grasshopper back to the nest to be used as food. A flock of birds manoeuvring through the air is quite impressive. There is no a leader bird telling the other birds which way to move. Each bird simply has an instinctive behaviour to react to the other birds around it, and when they all fly together the result is a collective behaviour called flocking. Another example is ants, which have minimal forms of communication and are considered to have very low intelligence, yet army ants are able to move large objects thousands of times heavier than themselves back to their nest (Franks, 1989). One ant could not direct the all other surrounding ants to return that object, and could not move the object itself. It is also the case that an ant could not determine the weight of the entire object by simply tugging on it. However, the collective behaviour that results in successful completion of the ants' objective is due to a genetic trait possessed by the ants. As shown by these examples, collective behaviour provides a means for very simple creatures to accomplish complicated objectives.

Social insects can process many sensor inputs, modulate their behaviour according to many stimuli, including interactions with nest-mates, and take decisions on the basis of a large amount of information. The success of social insects lies mainly in their self-organising behaviour (SO), where complex behaviour emerges from the interactions of individuals that exhibit simple behaviour by themselves (Parker et al., 2005; Tarasewich and McMullen, 2002). They can also solve problems in a changing environment (flexibility) and give the highest level of performance even though some individuals fail to perform their tasks (robustness). More and more researchers are interested in this exciting way of achieving a form of artificial intelligence - swarm intelligence – in which it is attempted to link the functioning principles of insect colonies to the design principles of artificial systems. For example, Bay and Unsal

(1994) described the design and development of a class of small mobile robots intended to be simple, inexpensive and physically identical, thus constituting a homogeneous team of robots. They derive their usefulness from their group actions, performing physical tasks and making co-operative decisions as a coordinated team. Because of their behavioural resemblance to their insect counterparts, they have been named "army-ant" robots.

Bonabeau et al. (1999) and Kube and Bonabeau (2000) stated that social insects such as bees, ants and termites all function collectively as groups, and efficiently accomplish a range of tasks in order to maintain their societies.

Kube and Zhang (1992, 1994) examined the problem of controlling multiple autonomous robots based on observations made from the study of social insects. They proposed mechanisms that allowed populations of behaviour-based robots to perform tasks without centralised control or use of explicit communication.

Chantemargne and Hirsbrunner (1999) presented a collective robotics application whereby a pool of autonomous robots regroup objects that are distributed in their environment. There is no supervisor in the system, the global task is not encoded explicitly within the robots, the environment is not represented within the robots, and there is no explicit co-operation protocol between the robots. Instead, the global task is achieved by virtue of emergence and self-organisation.

Martinoli and Mondada, (1995; 1998), Martinoli et al., (1997a; 1997b; 1999a; 1999b) and Martinoli (1999) focused on the hardware tools needed to monitor team performances as well as those needed to achieve collective adaptive behaviours. They presented a simple bio-inspired collective experiment, namely the gathering and clustering of randomly distributed passive seeds. Vaughan et al. (2001a; 2001b) showed a team of real mobile robots that co-operated based on the ant-trail-following behaviour and the dance behaviour of bees to robustly transport resources between two locations in an unknown environment.

Ant-inspired solutions to various search problems have been demonstrated (Dorigo et al., 1996; Deneubourg et al., 1990, 1991; Beckers et al., 1994), as has chemical trail laying and following in robots (Sharpe and Webb, 1998; Russell, 1999).

Wagner et al. (1998) and Wagner and Bruckstein (1995) described an ant-inspired method for exploring a continuous unknown planar region. Such a method might employ robots with limited sensing capabilities but with the ability to leave marks on the ground to cover a closed region for the purposes of cleaning a floor, painting a wall, or demining a mine field. A mark and cover (MAC) rule of motion is proposed using temporary markers ("pheromones") as a means of navigation and indirect communication.

Ijspeert et al. (2001) investigated collaboration in a group of simple reactive robots through the exploitation of local interactions. A test-bed experiment is proposed in which the task of the robots is to pull sticks out of the ground – an action that requires the collaboration of two robots to be successful. The experiment is implemented in a physical set-up composed of a group of mobile robots, and in Webots, a three dimensional simulator of mobile robots, (Michel, 1998).

As mentioned above, through the use of collective behaviour inspired by social insects, simple tasks that require a small number of mobile robots working in uncluttered environments can be accomplished. The question of interest is whether this collective behaviour approach can help accomplish complex tasks, such as dynamic target tracking, which require more collaboration, interaction, coordination and awareness among a large number of robots working together in a highly cluttered and dynamic environment.

2.4. Behaviour Coordination

In behaviour-based robotics, the control of a robot is shared between a set of purposive perception-action units, called behaviours (Murrieta-Cid, 2003; Parker, 2002; Schultz and Parker, 2003; Arkin, 1999; Pirjanian and Christensen, 1997). Based on selective sensor information, each behaviour produces immediate reactions to control the robot with respect to a particular objective, i.e., a narrow aspect of the overall task of the robot such as obstacle avoidance or wall following. Behaviours with different and possibly incommensurate objectives may produce conflicting actions that are seemingly irreconcilable. Thus, a major issue in the design of behaviour-based control systems is the formulation of effective mechanisms for coordination of the behaviours in a robot. This is known as the action selection or behaviour coordination problem (Pirjanian, 1998).

Behaviour coordination is generally recognised as one of the major open issues in behaviour-based approaches to robotics. It can be split into two conceptually different problems: (1) how to decide which behaviour(s) should be activated at each moment;

and (2) how to combine the results from different behaviours into one command to be sent to the effectors of the robot. These are called the behaviour arbitration and the command fusion problems, respectively.

Numerous action selection mechanisms (ASMs) have been proposed over the last decade and these can be classified into a number of logical groups. Mackenzie et al. (1997) classified them into state-based and continuous mechanisms. With a state-based ASM, in a given state, only a relevant subset of the behaviour repertoire of the robot needs to be activated. With a continuous ASM, there are no discrete states and the whole behaviour repertoire is available for activation.

Saffiotti (1997) divided ASMs into arbitration and command fusion mechanisms, corresponding respectively to the state-based and continuous approaches of Mackenzie et al. Arbitration is concerned with "how to decide which behaviour to activate at each moment" and command fusion is concerned with "how to combine the results of different behaviours into one command to be sent to the effectors of the robot".

Based on these classifications, it seems that ASMs can be best classified according to one main characteristic, namely whether the ASM can handle only one or multiple behaviours simultaneously.

2.4.1. Arbitration ASMs

Arbitration mechanisms select one behaviour from a group of competing behaviours, and give it ultimate control of the system (the robot) until the next selection cycle. This approach is suitable for arbitrating between the set of active behaviours in accordance with the changing objectives and requirements of the system under varying environmental conditions. Arbitration mechanisms for action selection can be classified as priority-based, state-based and winner-takes-all. In priority-based mechanisms, an action is selected based on priorities assigned in advance. Thus, behaviours with higher priorities are allowed to take control of the robot. State-based mechanisms select a set of behaviours that is competent to handle the situation corresponding to the given state. Finally, in winner-takes-all mechanisms, action selection results from the interaction of a set of distributed behaviours that compete until one behaviour wins and takes control of the robot.

2.4.2. Command Fusion ASMs

Command fusion combines recommendations from multiple behaviours to form a control action that represents their consensus. This approach allows all the behaviours to contribute simultaneously to the control of the system in a co-operative rather than a competitive manner. Command fusion mechanisms can be divided into voting techniques, superposition techniques and multiple objective behaviour coordination techniques.

Voting techniques interpret the output of each behaviour as votes, and then select the action that receives the largest number of votes. Superposition techniques combine behaviour recommendations using linear combinations. Finally, multiple objective behaviour coordination techniques provide a formal theoretic approach to making decisions based on multiple objective decision theory.

2.4.3. Priority-Based Arbitration (Subsumption Architecture)

The subsumption architecture (Brooks, 1986) represents a priority-based arbitration mechanism, where behaviours with higher priorities are allowed to subsume the output of behaviours with lower priorities. This architecture is covered in more detail in chapter three.

2.4.4. State-Based Arbitration

2.4.4.1. Discrete Event Systems (DES)

Behaviour selection is accomplished using state-transition (Kosecka, 1993) where, upon detection of a certain event, a shift is made to a new state and thus to a new behaviour. Using this formalism, systems are modelled in terms of finite-state automata (FSA), where states correspond to the execution of actions or behaviours and where events, which correspond to observations, cause transitions between these states.

2.4.4.2. Temporal Sequencing

The temporal sequencing approach is also known as perceptual sequencing (Arkin and Mackenzie, 1994) and is very similar to the discrete-event systems approach. A finite-state automaton is used to sequence between a series of behaviours based on perceptual triggers. At each state, a distinct behaviour is activated and perceptual triggers cause transitions from one state to another. See the example in figure 2.1.

2.4.4.3. Bayesian Decision Analysis

The approach of sensor planning with Bayesian Decision Analysis (Kristensen, 1996) is used to address the problem of sensor selection, i.e., which sensors to use for which purpose. Sensor selection can be considered a special case of action selection, where the actions are certain sensor operations. It operates according to the purposive paradigm, where the system consists of a set of purposive modules similar to behaviours.

For example, the problem in figure 2.2 is to decide which sensors to allocate to which purposive modules in order to accomplish a given task, declared by the mission planning-module.

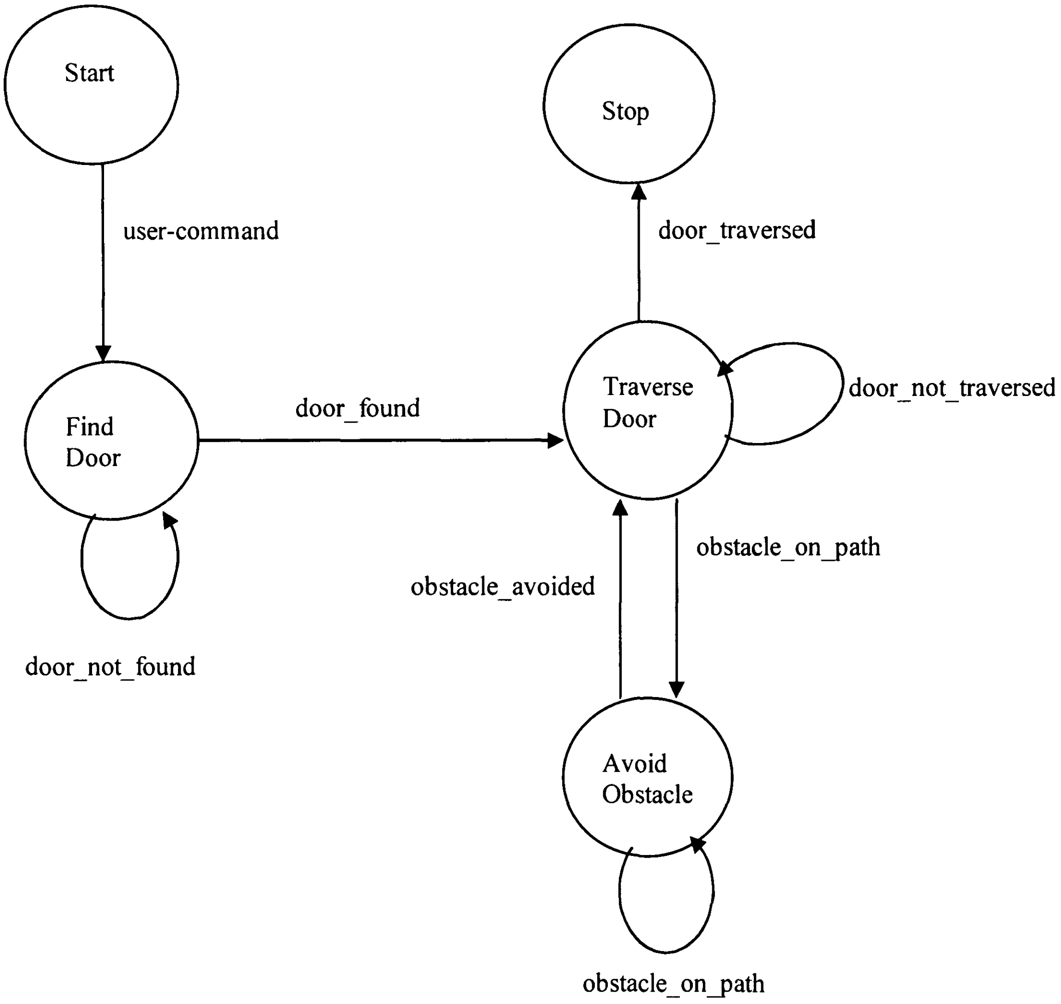


Figure 2.1: An example FSA encoding a door traversal operation (Pirjanian, (1999)

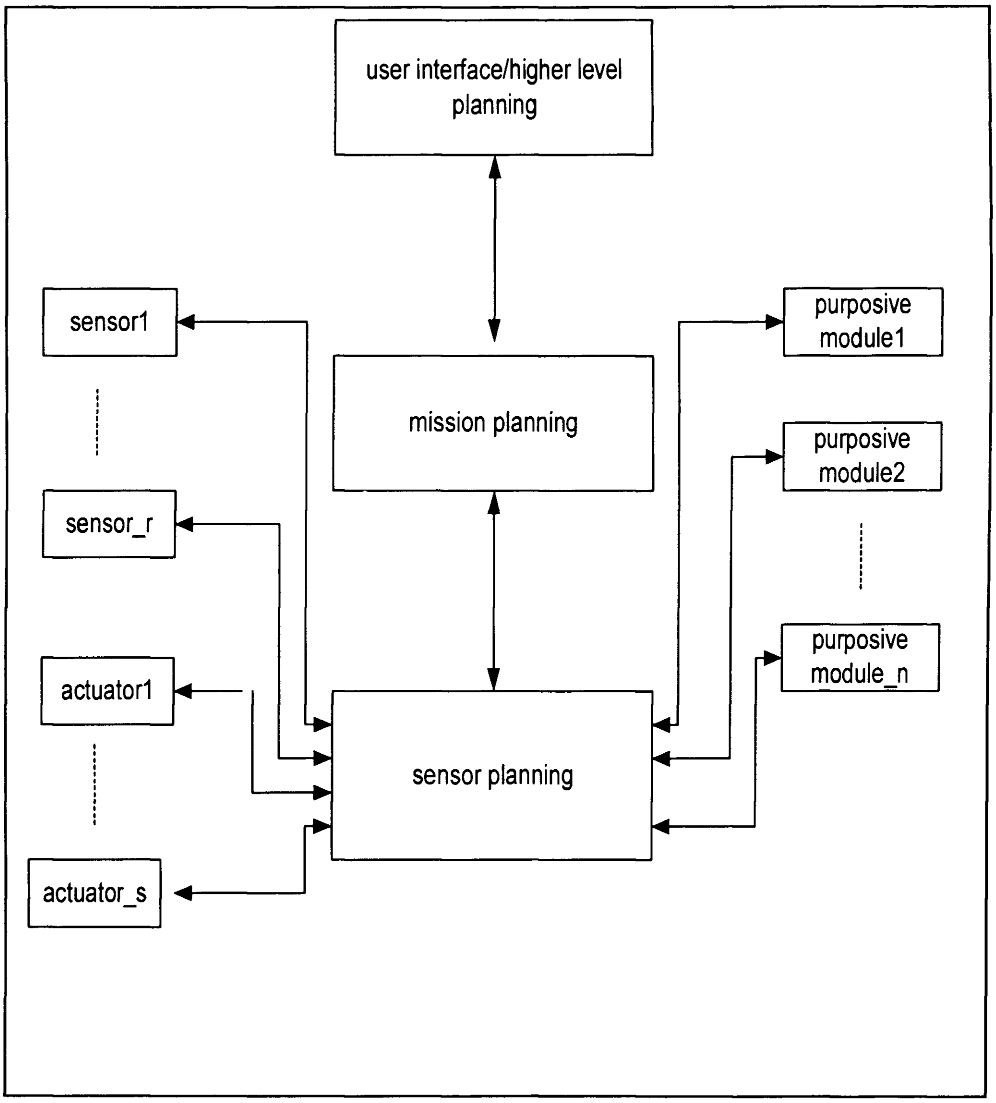


Figure 2.2: The architecture used in the sensor selection approach (Kristensen,1996)

2.4.4.4. Reinforcement Learning Approaches to Action Selection

A fundamentally different approach to action selection is to learn the action selection mechanism (Humphrys, 1997; Lin and Lu, 1996). Of the several learning approaches proposed, the most promising is reinforcement learning. Reinforcement learning in this context operates to induce, based on trial and error, a perception-to-action mapping that maximises some reward.

The robot learns the perception-action mapping, known as a policy, by exploring actions that lead to some reward. The reward function is designed so as to encourage desired behaviours and suppress unwanted ones. Thus, the robot will select actions that maximise the expected reward.

2.4.5. Winner-takes-All: Activation Networks

With this approach, the system consists of a set of behaviours or competence modules which are connected to form a network. In this network, each behaviour is described by the preconditions under which it is executable, the effects after successful execution in the form of add-lists and delete-lists and the activation level, which is a measure of applicability of the behaviour (Maes, 1989). When the activation level of an executable behaviour exceeds a specified threshold, it is selected to furnish its action.

2.4.6. Voting-Based Command Fusion

To manage the ongoing tasks of an agent so that action conflict is minimised and desired levels of compliance with overall goals are achieved, each behaviour votes for one action, which is suitable from its point of view. The votes received from all behaviours are summed for each action and the action with the largest number of votes is then selected. For example, DAMN is a distributed architecture for mobile robot navigation (Rosenblatt, 1997; Rosenblatt and Thorpe, 1995). It consists of a set of behaviours (figure 2.3) that pursue the system goals, based on the current state of the environment. Each behaviour votes for or against each action within the current possible set of actions. The action with the maximum weighted sum of received votes is then selected, where each behaviour is assigned a weight, which reflects the relative importance or priority of the behaviour in a given context.

2.4.7. Multiple Objective Behaviour Coordination

With this approach, multiple behaviours are blended into a single more complex behaviour that seeks to select the action that simultaneously satisfies all behavioural objectives as far as possible. In (Pirjanian and Christensen, 1997; Pirjanian and Mataric, 2000) mobile robot navigation and co-operative target acquisition examples are given, in which the principles of multiple objective decision-making (MODM) are demonstrated. Simulated as well as real-world experiments show that a smooth blending of behaviours according to the principles of MODM enables coherent robot behaviour.

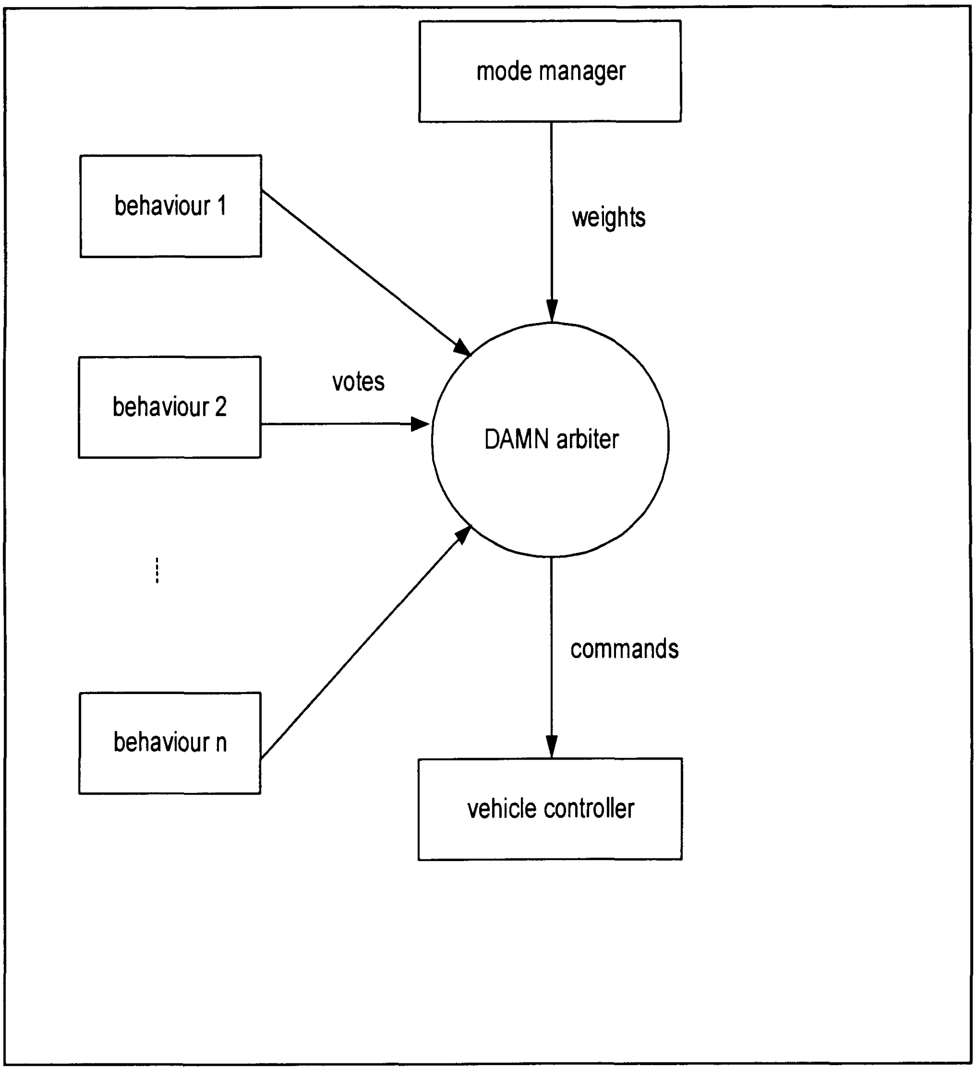


Figure 2.3: A distributed architecture for mobile robot navigation (Rosenblatt, 1995)

2.4.8. Superposition-Based Command Fusion (Potential Field)

The potential-field approach, introduced in (Khatib, 1986), is an approach to motion planning where the robot, represented as a point in configuration space, moves under the influence of an artificial potential field produced by an attractive force at the goal configuration position and repulsive forces at the obstacles. Action selection in this case corresponds to a move, at each configuration, in the direction indicated by the negative gradient of the total potential U. The potential function U is constructed as the sum of two potential functions:

$$U = U_{att} + U_{rep} \tag{2.1}$$

where $\boldsymbol{U_{att}}$ is the attractive potential associated with the goal and $\boldsymbol{U_{rep}}$ is the repulsive potential associated with the obstacles.

Much work has been carried out in behaviour coordination and action selection that does not directly relate to the above.

Saffiotti et al. (2000) espoused desirability functions as an effective way to express and implement complex behaviour coordination strategies within a single robot. The desirability function approach was extended to deal with the behaviours of teams of robots. The authors showed that desirability functions offer a convenient tool to incorporate and blend individual objectives and team objectives.

Yamada and Saito (1999) described an action selection method for multiple mobile robots performing box pushing in a dynamic environment. The robots are designed to need no explicit communication, and to be adaptive to dynamic environments by changing their active set of behaviours. The researchers proposed a mechanism that changed the active behaviour set depending on the situation.

Hu et al. (1998) presented a feasible solution for a team of autonomous mobile robots to function in a co-operative manner. To realise coordination, a multi-channel infrared communication system was developed to exchange messages among mobile robots. Two examples of flocking and shared experience learning were given to demonstrate the performance of the system.

Due to their co-operative nature, command fusion mechanisms promise improved performance over arbitration-based mechanisms. However, there are drawbacks that should be highlighted. Where a linear combination mechanism is employed, the obtained solution might be far from the required one. Command fusion systems are also costly both in computation time and hardware, and unnecessarily so if system accuracy is not critical. Furthermore, in multi-objective mechanisms, it is difficult to control the robots even heuristically to meet all objectives.

Fuzzy logic is suitable for a coordination scheme that allows all behaviours to contribute simultaneously to the control of the system in a co-operative rather than a competitive manner. This is therefore the solution proposed in this research for behaviour coordination in the context of dynamic target tracking.

When the output of a behaviour is represented by a fuzzy set, the problem of command fusion can be seen as an instance of the problem of combining individual preferences. Fuzzy operators can be used to combine the preferences of different behaviours into a collective preference, and finally to choose a command based on this collective preference. According to this view, command fusion is decomposed into two steps: (1) preference combination and (2) decision. Fuzzy logic offers many different operators to perform a combination and many defuzzification functions to select a decision. It is important to note that the decision taken from the collective preference can be different from the result of combining the decisions taken from the individual preferences. Figure 2.4 graphically illustrates this point in the case of two behaviours B1 and B2 both controlling the steering angle of a mobile robot. This explains why fuzzy command fusion is fundamentally different from vector summation.

Several proposals that use fuzzy logic to perform command fusion have appeared in the literature. Curiously enough, the first such proposal was made, in a naive form, by two roboticists who were unaware of fuzzy logic but were frustrated by the pitfalls of existing on-off arbitration schemas (Rosenblatt and Payton, 1989). Their suggestion was later restated in terms of fuzzy logic by Yen and Pfluger (1995). Other authors have proposed simplified forms of fuzzy command fusion. For instance, Goodridge and Luo (1994) used weighted singletons as fuzzy outputs and the centre of gravity (COG) method for defuzzification and Pin and Watanabe (1994) used symmetric rectangles and COG.

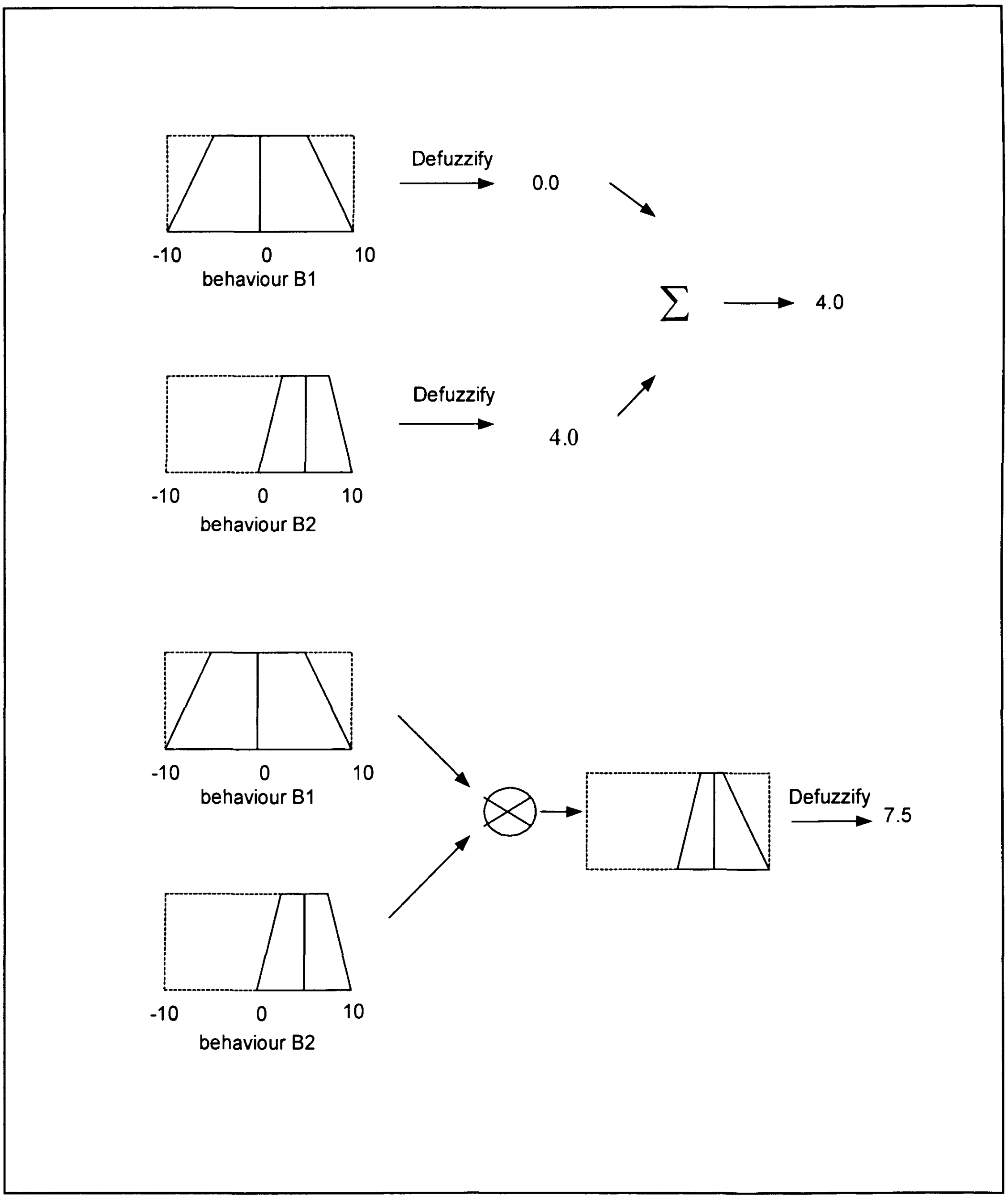


Figure 2.4: Two approaches to command fusion.
Top: combining individual decisions.
Bottom: combining individual preferences.

Even though there is not much work on behaviour coordination based on fuzzy logic, fuzzy logic is widely used in the controlling and learning mechanisms of mobile robots (Hu and Gu, 2005; Larsson, 2005; Lin and Mon, 2004; Abdessemed et al., 2004; Lin and Mon, 2004; Demirli and Molhim, 2004; Saffiotti and Wasik, 2003, Wasik and Saffiotti, 2002; Coradeschi et al., 2001; Buschka et al., 2000; Sossai, 2000; Hoffmann and Pfister, 1997; Surman et al., 1995; Pan, et al., 1995).

2.5 Awareness Effect on Mobile Robot Co-operation

Much existing work in the area of robot awareness addresses the problem of global coherence and efficiency by designing robotic teams that use sensor information to glean implicit information on the activities of other robot team members and/or the current state of the world (Deneubourg et al., 1990; Kube and Zhang, 1992). With these approaches, no explicit communication among robots is utilised. A more difficult approach requires the robots to use passive action recognition to observe the actions of their team-mates and modify their own actions accordingly (Huber and Durfee, 1993). A third, quite common, approach involves explicit co-operation among team members by employing direct communication between robots to relay information on robot goals and/or actions to other team members (Asama et al., 1992; Parker, 1994; 1995; 1996; 1999). These three approaches define a continuum in the degree of awareness of a robot of the actions or goals of its team-mates, from implicit awareness through the effect of a team-mate on the world, to passive observation of its actions or goals, to explicit communication of actions and/or goals. These approaches raise interesting questions concerning the impact of the awareness of the robot team members of the actions and/or goals of its team-mates.

For implicit co-operative systems and those using passive action recognition, the question is: What is the impact of a limited ability to sense the effect of robot actions on the world? For explicit communication systems, the question is: 'What is the impact of communication failure, which leads to the lack of awareness of team member actions/goals?' or, conversely: 'What benefits can be gained by using explicit communication to increase robot awareness of team member actions/goals?' Previous research concerning the effect of robot awareness, or recognition, of team member actions was usually described in terms of the effect of communication in co-operative robot teams (Balch and Arkin, 1994). However, Parker (2000) has used the phrase "robot awareness, or recognition, of team member actions" to describe precisely the issue of interest (awareness of team-mate actions), rather than the accessing of information that could possibly be communicated between team members. For example, the bid of a robot for an activity in a negotiation system may depend on the current local state of the environment near a given robot, or the sensed location of an intruder, etc. This shows that a robot may become aware of the actions of a team member without the use of explicit communication.

MacLennan (1991) investigated the evolution of communication in simulated worlds and concludes that the communication of local robot information can result in significant performance improvements. Balch and Arkin (1994) examined the importance of communication in robotic societies performing forage, consumption, and grazing tasks. They found that some communication could significantly improve performance for tasks, and that communication of the current robot state was almost as effective as communication of robot goals. Their research was performed primarily on real robots, rather than simulated robots.

Developing teams of robots that are able to perform their tasks over long periods requires the robots to be aware of and responsive to continual changes in the capabilities of the robot team members and to changes in the state of the environment and mission. Parker (1997; 2000; 2001; 2002) described the L-ALLIANCE architecture, which enables teams of robots dynamically to adapt their actions over time. This architecture, which is an extension of earlier work on ALLIANCE (Parker, 1994), is a distributed, behaviour-based architecture aimed at applications consisting of a collection of independent tasks. The key issue addressed in L-ALLIANCE is the determination of which task robots should select to perform during their mission, even where there are multiple robots with heterogeneous, continually changing capabilities present on the team. The L-ALLIANCE architecture is implemented on a team of heterogeneous real robots performing proof-of-concept box pushing experiments.

Due to the unreliability of the sensors and actuators employed and uncertainties in the environment, the approach of Parker of using a predefined time for each behaviour resulted in inconsistency, even if the behaviours are repeated and the values are averaged. This is because there is no guarantee that each robot will repeat the same behaviour at the same time. In this respect, it may be preferable to propose an architecture which does not rely on explicit communication or passive recognition and generate automatically the required time for a particular behaviour by accessing an on-line knowledge-base, updated by the use of neuro-fuzzy techniques, as will be discussed later.

2.6. Fuzzy Logic Systems (FLSs) Basic Structure and Design Elements

The basic structure of a FLS comprises four basic components (Lee, 1990a). They are the fuzzification interface, knowledge base, decision-making logic and defuzzification interface. Each component is responsible for a certain function in a FLS. In the following sections, the function and design parameters of each of these components are presented.

2.6.1. Fuzzification interface

Fuzzification is related to the vagueness and imprecision in natural languages. It is a mapping that transforms measurements into a subjective value, and hence it could be defined as a mapping from an observed measurement space into a subjective feature space. In fuzzy control applications, the observed data is usually crisp. Since the processed data in FLSs are based on fuzzy set theory, fuzzification is necessary during the early stages to transform the observed crisp data into fuzzy sets. A commonly used fuzzification approach is to transform this crisp data into fuzzy singletons within a certain universe of discourse. The transformation process begins with the normalisation or scaling of the crisp measurements to certain bounded range say $[-1,+1]$ using suitable scaling factors. The purpose of the normalisation process is to map the crisp input data into a universe of discourse with a finite range. Subsequently, the fuzzification interface transforms the normalised crisp input x_o into a fuzzy set A in universe X with the membership function $\mu_A(x)$ equal to zero for all $x \in X$ except at the point x_o, at which $\mu_A(x_o)$ equals one. In general, the role of the fuzzification interface can be summarised as follows (Keller et al., 1992):

a) It observes the crisp input values to a FLS.

b) It performs a scale transformation (normalisation) from the measurement space into the corresponding universe of discourse.

c) It performs the fuzzification function that converts the scaled input data into fuzzy sets.

2.6.2. Knowledge base

The knowledge base (Lee, 1990a) comprises knowledge concerning the application domain and the desired control objectives. It consists of a data base and a linguistic (fuzzy) control rule base. The data base provides necessary definitions, which are employed to define linguistic control rules and fuzzy data manipulation in FLSs. The rule base characterises the control objectives and the control policy of domain experts by means of a set of linguistic control rules.

2.6.2.1. Data base

The definitions associated with the data base are employed to characterise fuzzy control rules and fuzzy data manipulation in FLSs. These definitions are subjective in nature, which reflects engineering experience and judgement. These definitions comprise the normalisation of a fuzzy universe of discourse, the partition of a fuzzy universe of discourse and the definition of membership functions associated with fuzzy sets. In what follows, important definitions relating to the construction of the data base in FLSs are discussed.

A. Normalisation of a fuzzy universe of discourse

The normalisation of a universe of discourse involves a priori knowledge of the input-output universe of measurements. The normalisation process is a scale transformation of the input-output universe of measurements into a normalised closed interval universe. For example, if the measured input data ranges from -7.0 to +3.5, the universe of the input measurements can be normalised by a scale transformation into a normalised closed interval universe [-1, +1].

B. Fuzzy partition of the input-output universe

A linguistic variable in the antecedent or consequent of a fuzzy rule forms a fuzzy input or output feature space respectively. The input or the output feature space of each input or output linguistic variable is defined over a certain universe of discourse. Each feature space is internally partitioned into a number of clusters or fuzzy sets that define the term set of the input or output linguistic variables. Each fuzzy set is defined by a certain linguistic term, and usually has a meaning such as negative big (NB), negative small (NS), positive big (PB), etc. The number of partitions of the input and output feature spaces determines the maximum number of fuzzy rules that can be generated. Therefore the selection of the number of partitions influences the generated number of rules of FLSs. In most applications of FLSs, experience and engineering judgement are employed to choose the number of partitions of the fuzzy feature space.

C. Definition of the membership functions of fuzzy sets

There are two commonly used methods which define the membership functions of fuzzy sets depending on whether the universe of discourse is discrete or continuous (Lee, 1990b). The first method is a numerical definition where the grade of membership in a fuzzy set is represented as a vector of numbers. In this case, the membership function of each fuzzy set can be written as follows:

$$\mu_A (x) = [\mu_A (x_o) / x_o + \mu_A (x_1) / x_1 + \ldots\ldots\ldots\ldots\ldots + \mu_A (x_n) / x_n] \qquad (2.2)$$

where n is the number of supports of the discrete universe of discourse, x_n is the nth support of the discrete universe of discourse, and $\mu_A (x_n)$ is the membership grade of the nth support in fuzzy set A. The second method is a functional definition, which expresses the membership function of a fuzzy set in a functional form, typically a bell-shaped, triangle-shaped, trapezoid-shaped function, etc. For example the functional definition of the bell-shaped membership function can be written as follows:

$$\mu_A(x_o) = \exp[-(x_o - u)^2 / \sigma^2] \qquad (2.3)$$

where u and σ are respectively, the centre (or mean) and the width (or variance) of the bell-shaped function.

2.6.2.2. Rule base

A FLS is characterised by a set of linguistic statements based on expert knowledge. The expert knowledge is usually in the form of IF - THEN rules, which are easily implemented by fuzzy conventional statements in fuzzy logic. The collection of fuzzy rules that are expressed as fuzzy conditional statements forms the rule set or the rule base of a FLS. In this section, the following factors which influence the design and the implementation of a fuzzy rule base are discussed: the choice of the FLS input-output variables, the approaches employed to generate fuzzy rules, and the functional implementation of fuzzy rules.

A. Choice of the FLS input-output variables

It is important to choose suitable input and output variables for FLSs, because they influence the number of rules and the performance of FLSs. In several applications of FLSs, the selection of input-output variables relies on experience and control engineering judgement (Sugeno and Nishida, 1985). In some other applications, the selection is based on a deterministic method (Sugeno and Yasukawa, 1993).

B. Derivation of fuzzy rules

In general there are two common approaches to deriving fuzzy rules. These two approaches are not mutually exclusive, and it seems likely that a combination of them would be necessary to construct an effective method of deriving fuzzy rules. The first approach is to generate fuzzy rules based on expert experience and control engineering knowledge. This approach is mainly suitable for generating fuzzy rules for diagnosis systems including fault diagnosis and medical diagnosis systems. This approach is a heuristic approach, in which the fuzzy rules are obtained mainly from human experience. A human expert has to interpret his experience as linguistic relations between the input and output variables of the FLS. This approach can be successful if the human expert can perform this interpretation. However, if the human expert cannot express his experience linguistically, then the second approach which is based on the observed input-output data can be employed. This approach can be used to generate fuzzy rules for FLCs and for fuzzy process models. In the case of FLCs, the fuzzy rules can be generated based on observations of the human expert's control actions in terms of input-output data. In the case of fuzzy process models, the fuzzy rules are generated based on the process input-output data (Sugeno and Nishida, 1985; Takagi and Sugeno, 1983, 1985; Wang and Mendel, 1992).

C. Functional implementation of fuzzy rules

A rule base of a FLS consists of a set of fuzzy rules. For example, consider the following rules:

R_1: IF x is A_1 and y is B_1 THEN z is C_1

also R_2: IF x is A_2 and y is B_2 THEN z is C_2

....

....

also R_n: IF x is A_n and y is B_n THEN z is C_n

where x, y and z are linguistic variables and A_i, B_i and C_i are linguistic terms (fuzzy sets) of the linguistic variables x, y and z in the universes of discourse U, V and W respectively, with i = 1, 2,...., n. The i^{th} fuzzy rule is implemented by a fuzzy implication (fuzzy relation) R_i. This fuzzy relation is a fuzzy set in U x V x W and is defined for all u ∈ U, v ∈ V and w ∈ W as follows:

$$R_i = \{((u, v, w), \mu_{R_i}(u, v, w)) \mid (u, v, w) \in (UxVxW)\} \qquad (2.4)$$

and its membership function is given by:

$$\mu_{Ri}(u, v, w) \equiv \mu_{(Ai\ \mathit{and}\ Bi\ \rightarrow Ci)}(u, v, w) = [\mu_{Ai}(u)\ \mathit{and}\ \mu_{Bi}(v)] \rightarrow \mu_{Ci}(w) \qquad (2.5)$$

where "A_i *and* B_i" is a fuzzy set in the Cartesian product space U x V which can be defined based on the interpretation of the sentence connective "*and*" and, $R_i \equiv (A_i$ *and* $B_i) \rightarrow C_i$ is a fuzzy implication (relation) in the Cartesian product space U x V x W which can be defined based on the interpretation of the sentence connective "*and*" and the definition of the fuzzy implication function →.

The implication functions can be classified into two commonly used categories (Keller et al., 1992). The first category is the fuzzy conjunction that is defined for all u ∈ U and v ∈ V as follows:

$$A \rightarrow B = \int_{UXV} \mu_A(u) * \mu_B(v)/(u, v) \qquad (2.6)$$

where A and B are fuzzy sets in the universes of discourse U and V respectively, $A \to B$ is a fuzzy implication in the Cartesian product space U x V and $*$ is an operator that represents a triangular norm. The second category is the fuzzy disjunction that is defined for all $u \in U$ and $v \in V$ as follows:

$$A \to B = \int_{UXV} \mu_A(u) + \mu_B(v) /(u,v) \tag{2.7}$$

where A and B are fuzzy sets in the universes of discourse U and V respectively, $A \to B$ is a fuzzy implication in the Cartesian product space U x V and + is an operator that represents a triangular co-norm. In general, using the fuzzy conjunction along with the intersection and algebraic product triangular norms, the two commonly used fuzzy implication functions can be written as follows:

$$A \to B = \int_{UXV} \mu_A(u) \wedge \mu_B(v) /(u,v) \tag{2.8}$$

where $\mu_A(u) \wedge \mu_B(v) = \min[\mu_A(u), \mu_B(v)]$ is the intersection triangular norm.

$$A \to B = \int_{UXV} \mu_A(u) \cdot \mu_B(v) /(u,v) \tag{2.9}$$

where $\mu_A(u) \cdot \mu_B(v) = \mu_A(u)\mu_B(v)$ is the algebraic product triangular norm.

In most existing FLSs, the sentence connective "*and*" is usually implemented as a fuzzy conjunction in a Cartesian product space (Lee, 1990b). As an illustration, for two fuzzy sets A and B in the universes of discourse U and V respectively, “A *and* B” is defined by a fuzzy set A x B in the Cartesian product space U x V. If the sentence connective "*and*" is interpreted using the intersection triangular norm, the membership function of this fuzzy set is expressed as follows:

$$\mu_{AxB}(uxv) = \min[\mu_A(u), \mu_B(v)] \qquad (2.10)$$

Alternatively, if the sentence connective "*and*" is interpreted using the algebraic product triangular norm, the membership function of this fuzzy set is expressed as follows:

$$\mu_{AxB}(uxv) = \mu_A(u) \cdot \mu_B(v) \qquad (2.11)$$

On the other hand, the interpretation of the sentence connective "*also*" is based on the fact that different orders of fuzzy rules in the rule base should not influence the overall behaviour of a FLS. This requires that the sentence connective "*also*" should have the properties of commutativity and associativity. It has been reported in (Lee, 1990b) that the operators in triangular norms and co-norms (intersection, algebraic product, union, algebraic sum, etc) possess these properties and thus qualify as candidates for the interpretation of the connective "*also*". However, several investigations have been reported in (Lee, 1990b). These investigations studied FLS characteristics using different interpretations of triangular norms and co-norms. Based on these investigations, it has been concluded that the common interpretation of the connective "also" as the union operator $\cup$ yielded the best results. The union operator $\cup$ is a triangular co-norm defined using the max function (Lee, 1990b). Subsequently, considering the rule base of Subsection 2.1.2.2, the overall fuzzy relation R is defined as a fuzzy set in U x V x W for all u ∈ U, v ∈ V and w ∈ W as follows:

$$R = \{((u,v,w), \mu_R(u,v,w)) \mid (u,v,w) \in (UxVxW)\} \quad (2.12)$$

and its membership function is given by:

$$\mu_R(u,v,w) = \max_{i=1}^{n} \mu_{Ri}(u,v,w) \quad (2.13)$$

where U, V, and W are universes of discourse, R_i is the ith fuzzy relation of the ith rule in the rule base and $\mu_{R_i}(u,v,w)$ is as defined in Equation (2.4).

2.6.3. Decision making logic

FLSs may be regarded as a means of emulating a skilled human operator through an inference engine. More generally, the FLS inference engine may be viewed as another step towards modelling the human decision making process within the conceptual framework of fuzzy logic and approximate reasoning. The function of the FLS inference engine is to infer recommended solutions from fuzzy rules relevant to given inputs based on the employed inference strategy. Generally, there are two important inference strategies in approximate reasoning (Lee, 1990b). They are generalised modus ponens (GMP) and generalised modus tollens (GMT). Specifically, consider the following rule:

IF x is A THEN y is B

where x and y are linguistic variables and A and B are linguistic terms of the linguistic variables x and y in the universes of discourse U and V respectively. The GMP strategy can be defined as "given x is A' and the fuzzy relation R of the fuzzy rule then infer y = B' ". This inference strategy is a data-driven or forward chaining strategy, which is particularly useful in FLCs. On the other hand the GMT strategy is defined as "given y is B' and the fuzzy relation R of the fuzzy rule then infer x = A' ". This inference strategy is a goal-driven or backward chaining strategy, which is commonly used in expert fault diagnosis systems.

2.6.4. Defuzzification strategies

Most practical control applications require crisp control actions to drive the controlled process. Moreover, the output of most modelling and prediction systems has to be crisp. Defuzzification is the mapping from the linguistic fuzzy output defined over an output universe into a crisp output space. There are three commonly used defuzzification strategies (Shankir, 2001). The first strategy is the maximum criterion. The max criterion produces the point w_o in the output universe W that has the maximum degree of membership in the output fuzzy set $\max_{w \in W} \mu(w) = \mu(w_o)$. A problem arises with this method when more than one element of W possesses this maximal value and thus w_o is not uniquely determined. The second strategy is the Mean Of Maxima (MOM). If there is more than one element in W possessing the maximal membership value, then MOM produces the average value of the maxima. More specifically, let Z denote a set of w_i for which an output fuzzy set in a universe W attains maximum membership values that is $Z = \left\{ w_i : \max_{j=1,2,....} \mu(w_j) = \mu(w_i) \right\}$, and assume that the cardinality equals r; that is card(Z) = r . Then the defuzzified output w_o is written as follows:

$$w_o = \sum_{w_i \in W} \frac{w_i}{r} \qquad (2.14)$$

However, MOM does not take account of rules fired below the maximum level (Saade, 1996). The third and the most commonly used strategy is the Centre Of Area (COA) strategy. COA attempts to correct the drawback of MOM by considering rules that can be fired below the maximum level. COA generates the centre of gravity w_o of the possibility distribution of a control action as follows:

$$w_o = \frac{\sum_{j=1}^{n} \mu(w_j) \cdot w_j}{\sum_{j=1}^{n} \mu(w_j)} \qquad (2.15)$$

where n is the number of quantisation levels of a universe W and w_j is the point in the j^{th} quantisation level in a universe W at which $\mu(w)$ achieves its maximum value $\mu(w_j)$.

2.7. Summary

This chapter reviewed background literature relevant to the work presented in this thesis. The literature on co-operating mobile robots was examined from different perspectives. First, it was surveyed with a focus on the classification of co-operating mobile robots. Second, the focus shifted to the collective behaviour of social insects and the connection of the functioning principles of social insect colonies with the design principles of artificial systems. Third, the literature related to the action selection problem (ASP) and behaviour coordination was reviewed. Fourth, previous work on robot awareness and its effect on the performance of co-operating mobile robots was examined. Fifth, a brief review for the basic components of the fuzzy logic system was given.

Chapter 3

Biologically Inspired Collective Behaviour and Co-operating Mobile Robots

3.1. Preliminaries

Social insects such as ants and bees co-operate to achieve common goals with remarkable success. However, individually these insects are very simple creatures. For example, a bee is unlikely to have a global understanding of the collective task being performed. Instead, complex behaviours can emerge from a swarm of bees to provide solutions through the interactions of individual bees sensing and acting locally on the basis of simple rules. Collective tasks can thus be performed by the swarm even though some individuals might fail or the environment they operate in might change.

Studying this phenomenon might enable biologists to understand how living organisms work and engineers to develop new robust and adaptive technologies for dealing with complex problems that have defied conventional solution means.

This chapter focuses on the development of intelligent multi-agent robot teams that are capable of acting autonomously and of collaborating in a dynamic environment to achieve team objectives. It proposes a biologically-inspired collective behaviour for a team of co-operating mobile robots. This behaviour is achieved by controlling the local interactions among a set of identical robots with simple behaviours with the aim of tracking a dynamic target. The subsumption architecture is taken as the starting point for implementing the control of individual robots. With this architecture, behaviours are arranged in order of priority. When different behaviours are applicable simultaneously, the behaviour with the highest priority is activated. This so-called "competitive" architecture is adopted because it is computationally inexpensive and potentially suitable for low-level reactive and reflexive behaviours.

The remainder of the chapter is organised as follows. Collective dynamic target tracking is discussed in section 2. Section 3 describes a modified subsumption architecture. The simulation tool developed to test the proposed architecture is presented in section 4. Section 5 describes the experiments conducted using the tool and the results obtained.

3.2 Collective Dynamic Target Tracking

An important issue that arises in the automation of many security, surveillance and reconnaissance tasks is that of monitoring and tracking the movements of targets navigating in a bounded area of interest.

The collective dynamic target tracking task investigated here is based on the emergence of collective strategy in prey-predator behaviour, where the predators co-operate to catch the prey or the prey co-operate to defend themselves. The term collective is used in the sense of the collective motion of defence or attack. The dynamics of predator-prey interactions where the predators surround the prey to catch it using local sensor-based interactions among them have been implemented in the task of dynamic target tracking. The prey-capture task is a special case of pursuit-evasion problems, which consist of an environment with one or more prey and one or more predators. Pursuit - evasion tasks are interesting because they are ubiquitous in the natural world, and offer a clear objective that requires complex coordination with respect to the environment, and with respect to other agents with the same goal. They are therefore challenging for even the best learning systems, requiring accurate success measurement and good analysis and visualisation of the strategies that evolve.

Dynamic target tracking involves the following and capturing, by a group of mobile robots, of a moving object within a cluttered workspace, while avoiding collision with obstacles and with each other.

The target is unpredictable; i.e., its trajectory is not known in advance. It is however assumed to move with a bounded velocity that is comparable with the velocity of the tracking robots.

The research objective was to identify how these robots co-operate to search for, pursue, surround, and finally capture the target even though some individuals may fail to carry out their tasks. Another aim was to demonstrate how individual members of the collective team can perform the task in a distributed fashion so that the collective team as a whole meets its goal.

From the perspective of an individual robot, the task consists of searching for the target, broadcasting messages to other robots, receiving messages from other robots and approaching and capturing the target, as depicted in the state diagram of figure 3.1(a). In order to decide the direction from which it should approach the target, the robot is required to be aware of the actions of its partners.

The target-tracking task of the robot team, from a group perspective, can be described, at a high level, by the state diagram of figure 3.1(b). The accomplishment of this task is a function of the effective co-operation between the robots.

3.3. Modified Subsumption Architecture

Early strategies for controlling robots involved building a representation of their environment, and then planning their actions accordingly. Those strategies are expensive and cannot react well to changes in a dynamic environment. In the subsumption architecture (figure 3.2) there is no world model. Instead, a robot responds directly to information it receives through its sensors; the robot acts 'by reflex' and in this way the control software can be very simple. The architecture implements several independent 'behaviours' that react to sensory inputs, and provide control signals to the actuators (motors) of the robot. These 'behaviours' have different priorities, such that only one behaviour is allowed to command the motors at any one time.

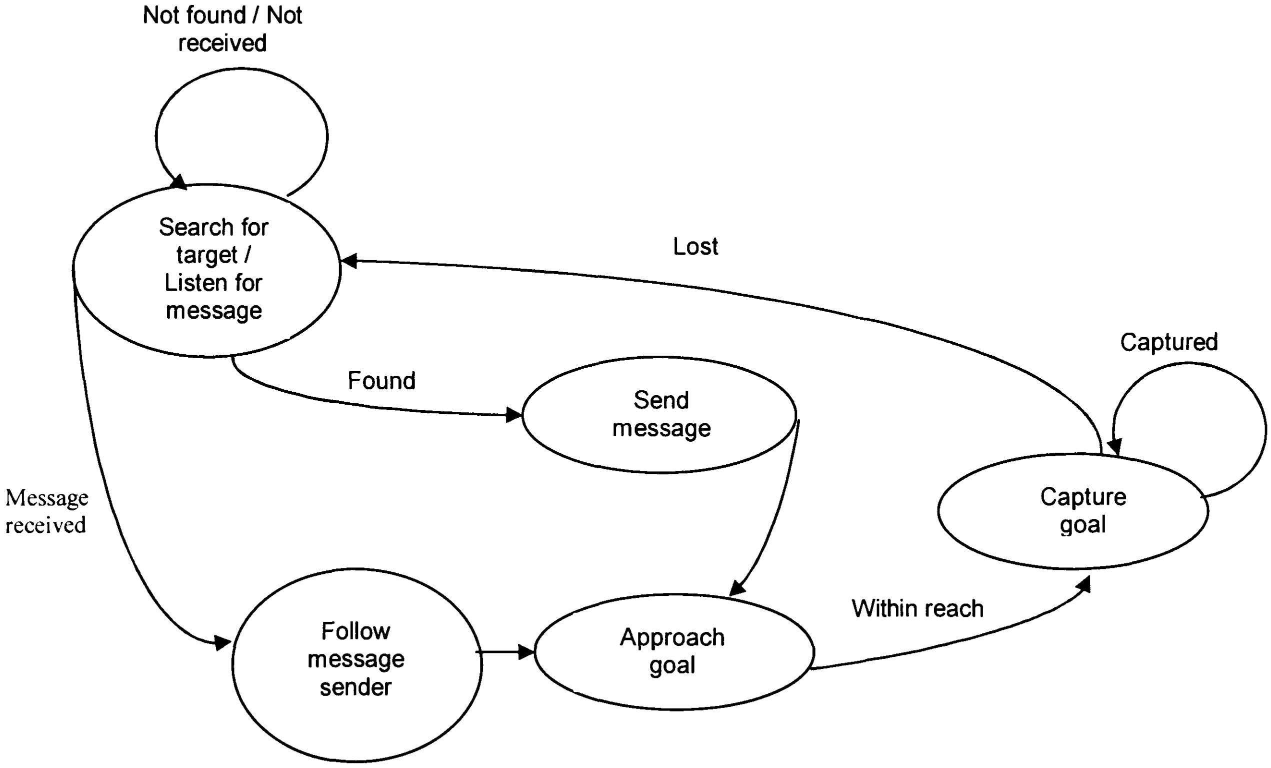


Figure 3.1(a): State diagram from the perspective of an individual robot

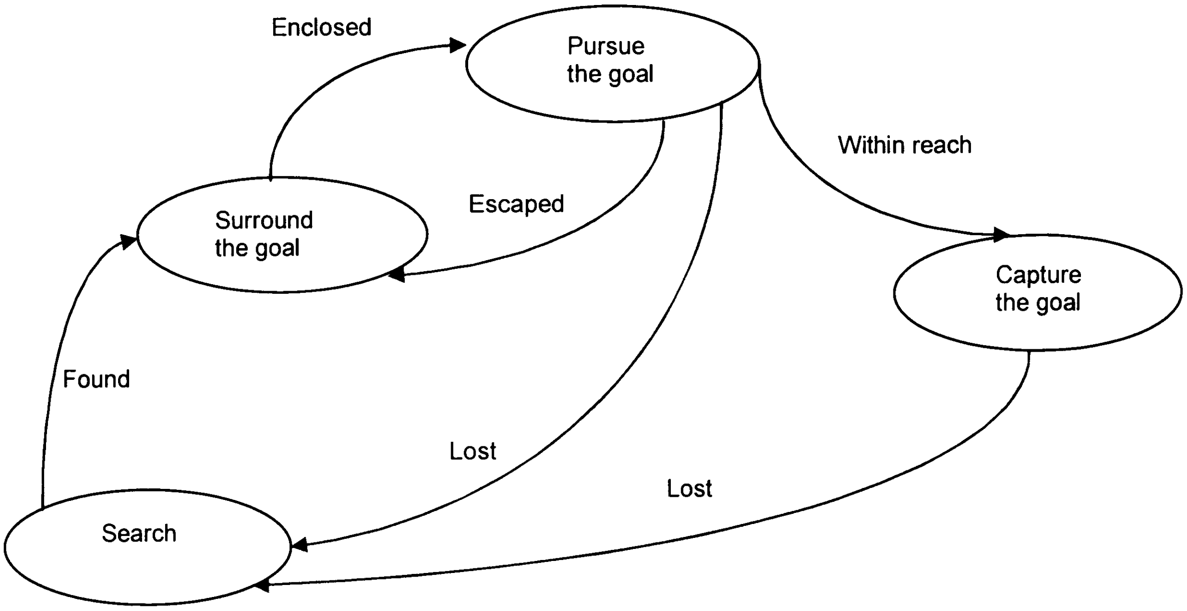


Figure 3.1(b): State diagram from the perspective of the group

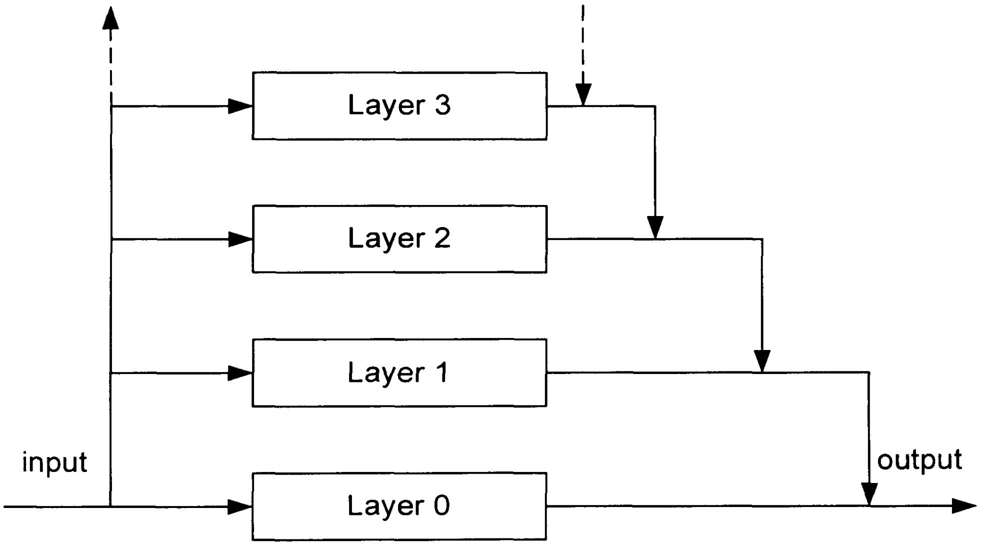


Figure 3.2: The Subsumption Architecture

The subsumption architecture comprises a hierarchy of controller layers where each layer is capable of instantly overriding all lower layers and taking control of the robot for as long as it wishes. It then relinquishes control to whatever lower layer was previously in command of the robot. The architecture control system is determined by the structure of the behaviours and their interconnections. The layer that subsumes control of the robot at any point needs no knowledge of what is currently controlling the robot at that point, and similarly the usurped module does not require any information about the subsumer.

As previously mentioned, one of the problems associated with subsumption is that only a single behaviour, one behaviour per layer, is active at any time. While this is satisfactory in many situations, there are times when a combination of more than one behaviour is required. In practice, it has also been found that this very loose coupling between layers is not sustainable (Pirjanian, 1998). Layers often need to pass information back and forth. Take, for example, the task of moving towards a target and avoiding obstacles. Each of these sub-tasks could be implemented as a single behaviour. So long as no obstacles are detected, the robot will gracefully head towards its target. If an obstacle is detected, however, the obstacle avoidance behaviour becomes active and steers the robot away from the obstacle. The problem with this is that the obstacle avoidance behaviour has no knowledge about the target, and therefore will not necessarily steer in a direction that takes the robot closer to its desired path.

In this work, the subsumption architecture is modified to comprise more than one behaviour module within one layer (figure 3.3). Those modules run in parallel and have the same priority. Based on information from the sensors, the activated behaviour module subsumes the others.

The design of the target-tracking controller begins by specifying the sensing requirements for the task. Collision free movement will require an obstacle sensor; to follow other robots needs a robot sensor; tracking the target will require a target or goal sensor.

To accomplish the task of tracking a dynamic target, each robot was given four main behaviours. The lowest priority default behaviours are the "search" and "listen for messages" behaviours. "Search" directs the robot to advance along its current path. Simultaneously, "listen for messages" makes the robot receptive to messages sent by other mobiles. No sensors are required to activate these behaviours.

The above default behaviours can be suppressed by the "follow message sender" behaviour if a message has been received from another robot (by means of the robot sensor on the current robot). "Follow message sender" causes the robot to move to its nearest sensed neighbour. The "send message" and "approach goal" behaviours are activated by the goal sensor. "Send a message" makes the robot issue a "target intercepted" message to the other mobiles and "approach goal" directs them towards the target. "Approach goal" causes the robots to turn a number of degrees towards the target while the goal sensor is active. The task is accomplished once several robots collectively have captured the target.

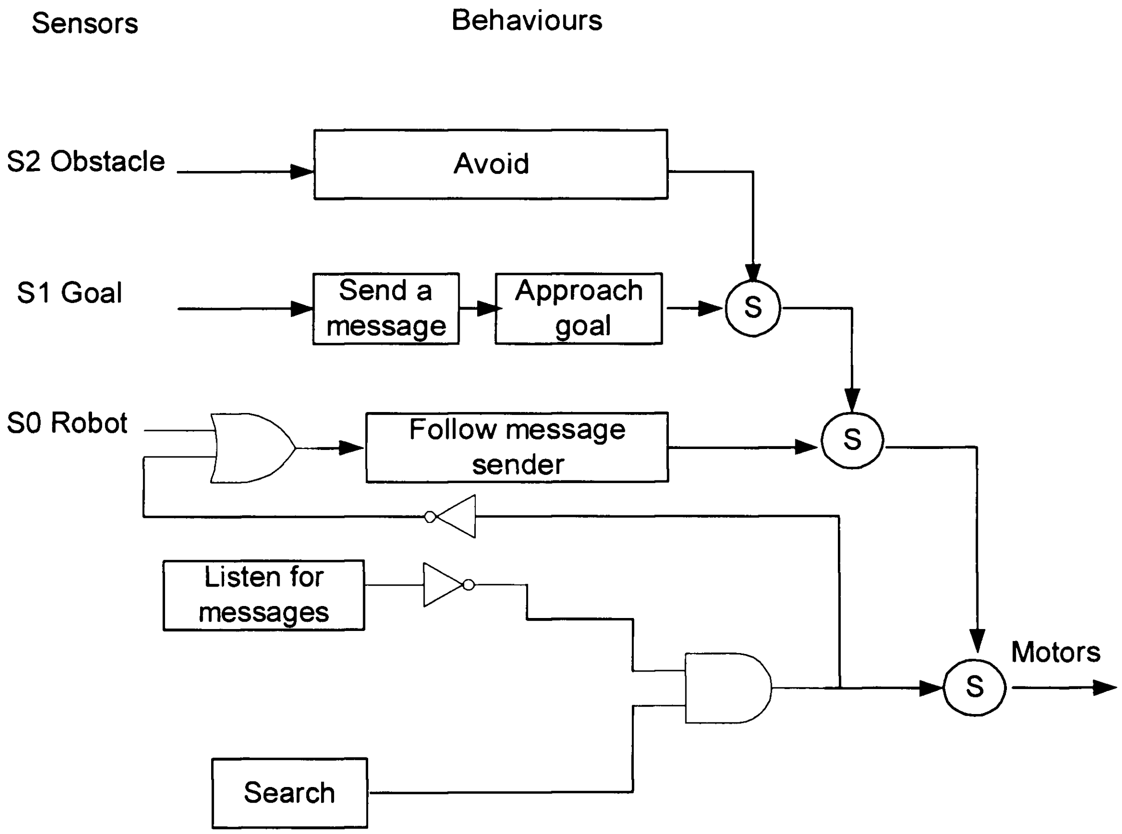


Figure 3.3: The behaviour architecture of a target - tracking robot

The “search”, “follow message sender” and “approach goal” behaviours can create motion resulting in collisions. To prevent them, the “avoid” behaviour is added. This highest priority behaviour becomes active and remains active as long as the obstacle sensor has detected an obstacle. Turning the robot a fixed number of degrees away from the sensed obstacles at each simulation time step prevents collisions.

3.4 Simulation

The objective of the developed simulation tool is to test the proposed architecture based on the context of the co-operative task of dynamic target tracking. For this, a simulated environment has been designed to model a large population of robots (a few thousand), different obstacles (e.g. in shape and size), and multiple dynamic targets.

Accomplishing tasks using a decentralised system of autonomous robots requires the control algorithms of each robot to make use of local information. This information is acquired by the on-board robot sensors and must be sufficient to ensure that the entire system of robots converges towards the desired goal.

Two kinds of sensors were simulated: obstacle detection sensors and target detection sensors.

The purpose of the obstacle detection sensors was to provide obstacle distance information to the robot. Three ultrasonic sensors were modelled to provide information on obstacles to the left and the right, and in front of the robot. The same models were used for the ultrasonic sensors fitted to the moving target.

Target detection was simplified by using an infrared source at the centre of the target and infrared target sensors mounted on the robots. The signal received by the sensors depended on the distance and the orientation between the robot and the target: the closer the distance the stronger the signal; similarly, the more directly the source and sensor were aligned, the more powerful the signal.

Two actuators were modelled, one for each motor (left and right). Steering of the robot was achieved by differentially turning the motors.

The behaviours mapped inputs from sensors to outputs to actuators to define a stimulus-response relationship. Sensors provided information to the behaviour modules, which then processed the data to provide commands to actuators.

During a simulation time step, each behaviour module read its related sensors and calculated an appropriate response, with the resulting command sent to a behaviour arbitration module, which decided the overall response.

3.5. Experiments and Discussion

Many factors determine the effectiveness of a co-operative multi-robot system for dynamic target tracking. Experiments were run with different numbers of robots and different obstacle densities. Each experiment on a collection of robots was performed thirty times and the results were averaged. Several sets of experiments were conducted to analyse the effects of various factors on performance. The first experiment analysed how varying the number of robots affected the time required to track (capture) the target. This experiment took place in a limited arena containing one small target (requiring at least two robots to track and capture it) and no obstacles. The average tracking time (measured as the number of steps by which the target has moved before being captured) versus the number of robots was examined. The second experiment differed from the first only by the addition of obstacles in the arena. Again, performance was analysed relative to the number of robots performing the task. Figure 3.4(a) shows one of the simulated environments before the experiment started. This contained fifteen robots (eleven in one corner and four in another corner), one target in the opposite corner, and different kinds of obstacles randomly distributed. Figure 3.4(b) depicts an intermediate stage of target tracking. The developed technique enabled the robots to complete their missions successfully even though in some trials some of them failed (became stuck for a long time to avoid an obstacle) to carry out their tasks as shown in figure 3.4(c). Figure 3.4(d) shows the final stage where the robots have captured the target. Figure 3.5(a-c) shows that increasing the number of robots reduced the time required to track the target. However, robot collision and interference tended to degrade the performance. Adding more robots did therefore produce a proportional increase in performance. Adding a very large number of robots causes the environment to be full of robots. Then, the target cannot move very far and tracking time does not change.

The third experiment was conducted by changing the environment complexity (obstacle density) with different target sizes (a small target that requires at least two robots to capture it, a medium-sized target needing at least four robots and a large target necessitating at least six robots). Figure 3.6 shows the results obtained. Adding obstacles in the environment increased the time required for capturing the target.

Furthermore, the robots took longer to capture the larger target because more robots were required for this.

Experiment four concerns multiple targets distributed in the environment. Again, performance was analysed relative to the number of robots performing the task in environments with different obstacle densities. Figure 3.7 shows that the robots co-operated to track the targets even in a cluttered environment. Compared with the results for tracking one target only, there is no significant difference in performance, except that the number of time steps is reduced as shown in figure 3.8. This is because moving more than one target in the environment reduces the tracking area available to the robots and hence increases the chance for successful capturing. However, in several trials the robots could not track all the targets as shown in figure 3.9. This is because there is no coordination among the behaviours within one robot and between the robots themselves. Also, there is no task allocation or task assignment techniques to allocate a suitable number of robots for each target.

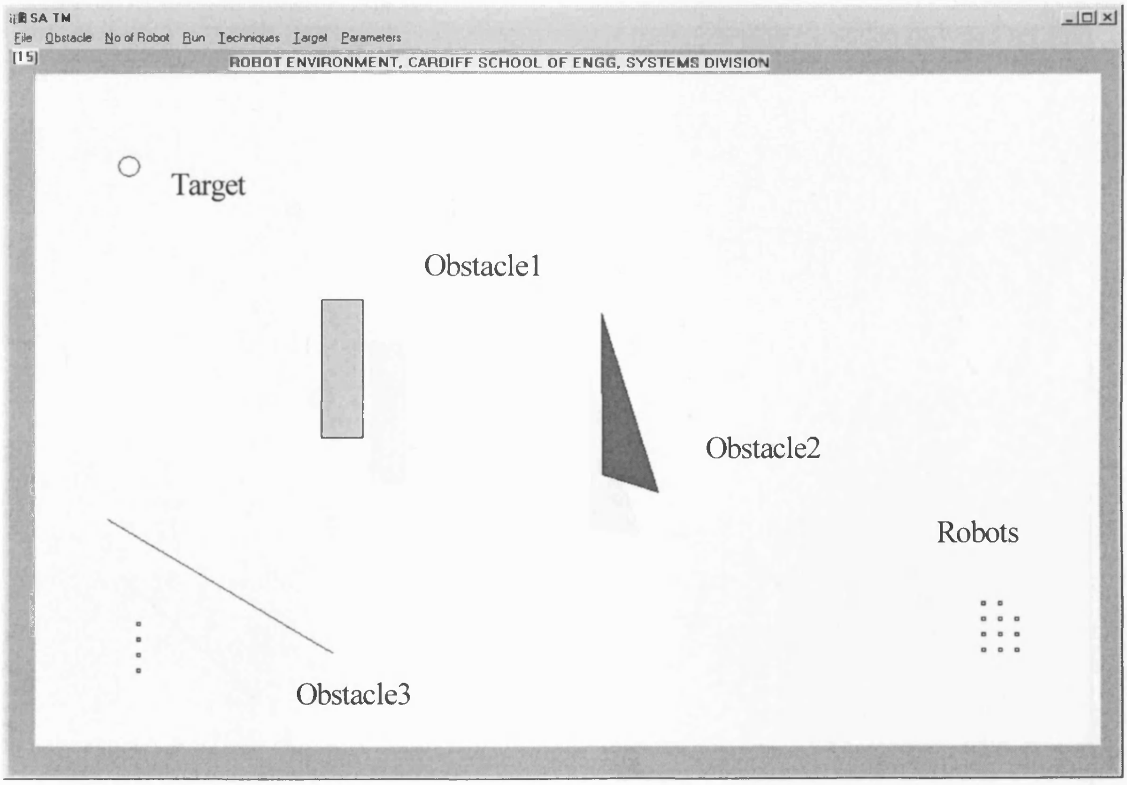

SA TM
File Obstacle No of Robot Run Techniques Target Parameters
[15]
ROBOT ENVIRONMENT, CARDIFF SCHOOL OF ENGG, SYSTEMS DIVISION
Target
Obstacle1
Obstacle2
Robots
Obstacle3

Figure 3.4(a): Initial environment (one target, different obstacles, and fifteen robots)

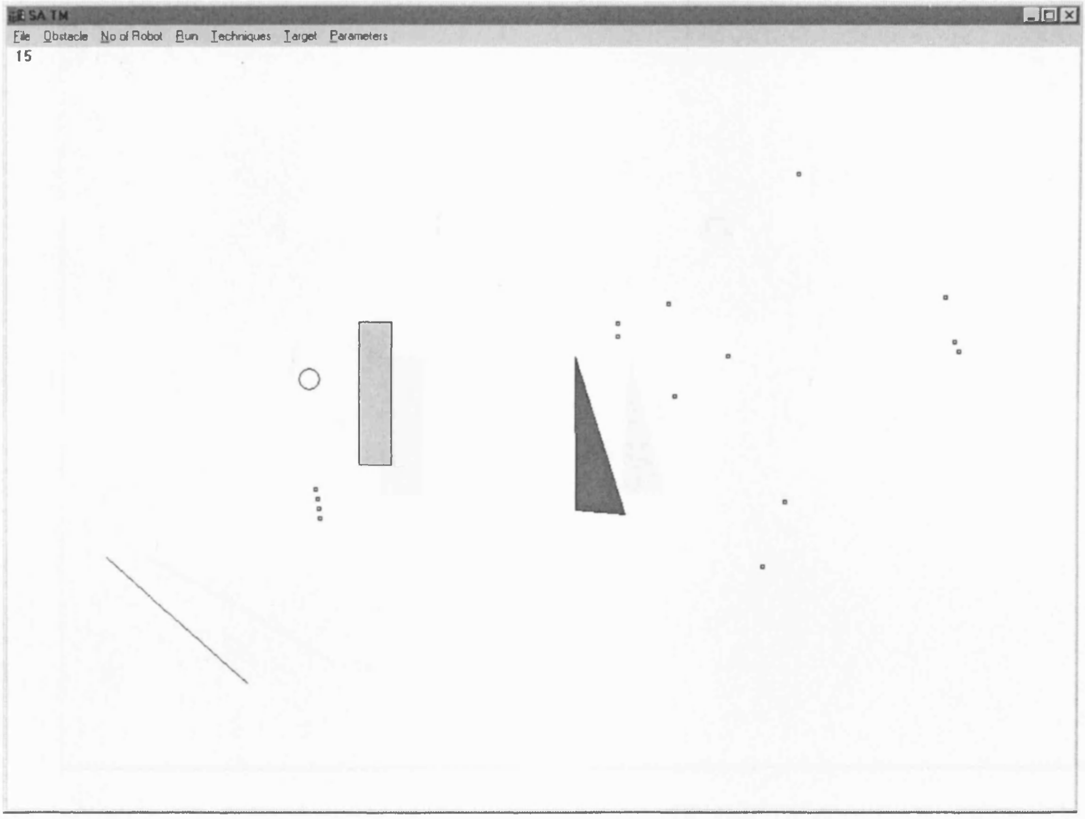


Figure 3.4 (b): Intermediate stage with robots tracking the target

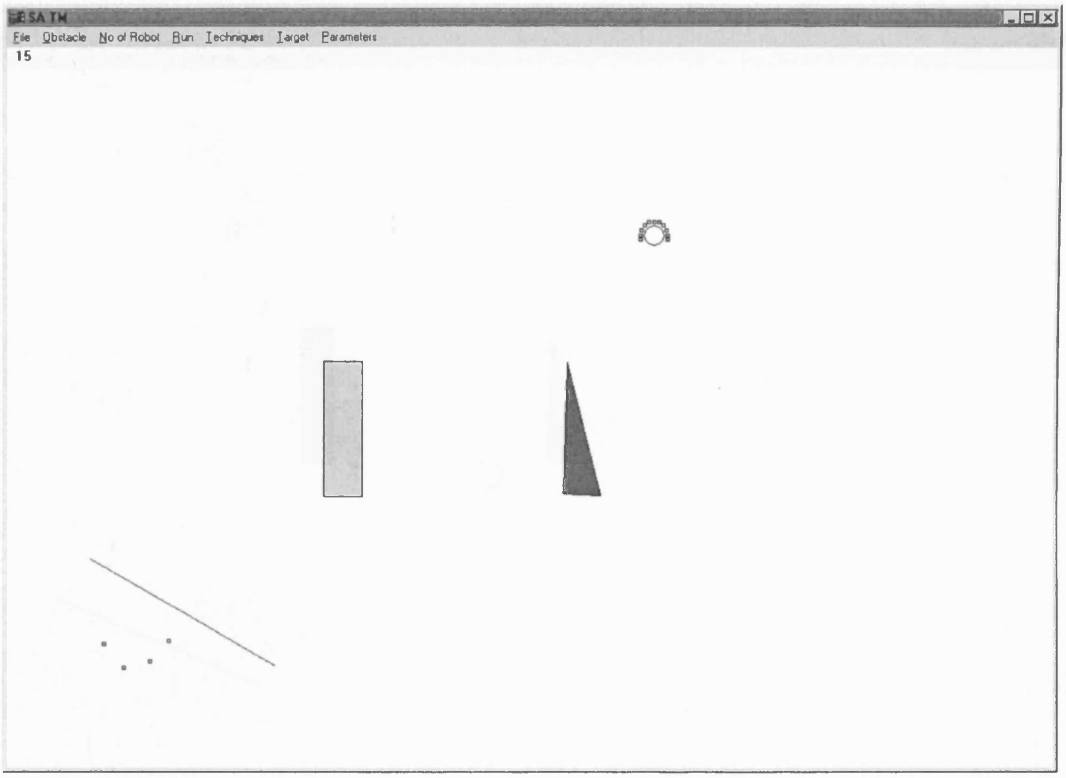


Figure 3.4(c): Robots having captured the target even though some of them failed to carry out their tasks

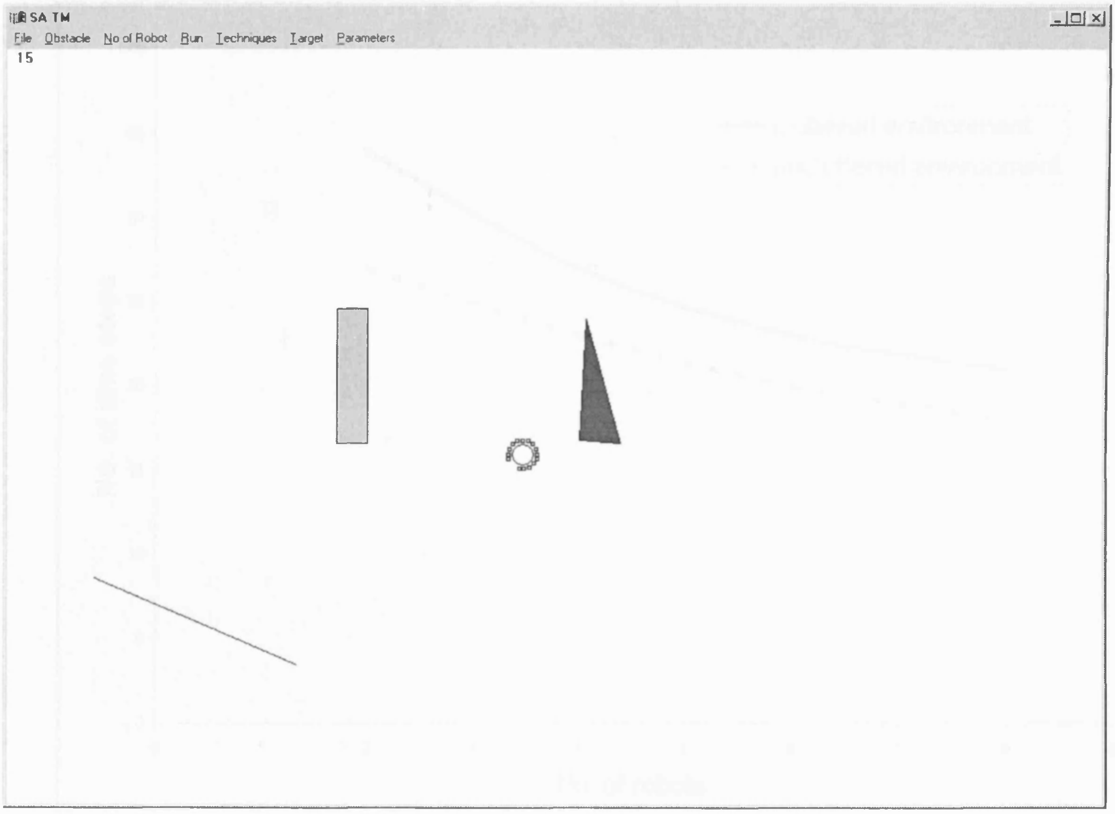


Figure 3.4(d): Final stage with all robots having captured the target

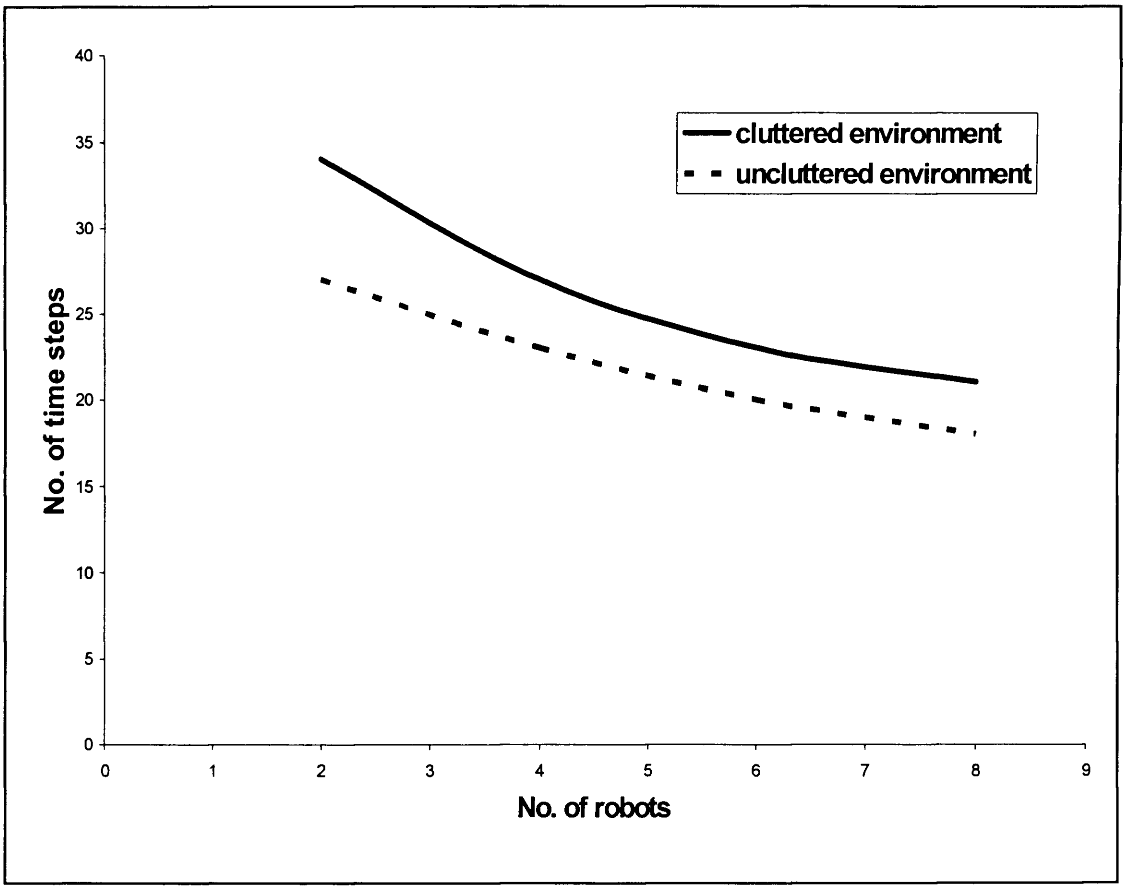


Figure 3.5(a): Mobile robots tracking a dynamic target
in cluttered and uncluttered environments

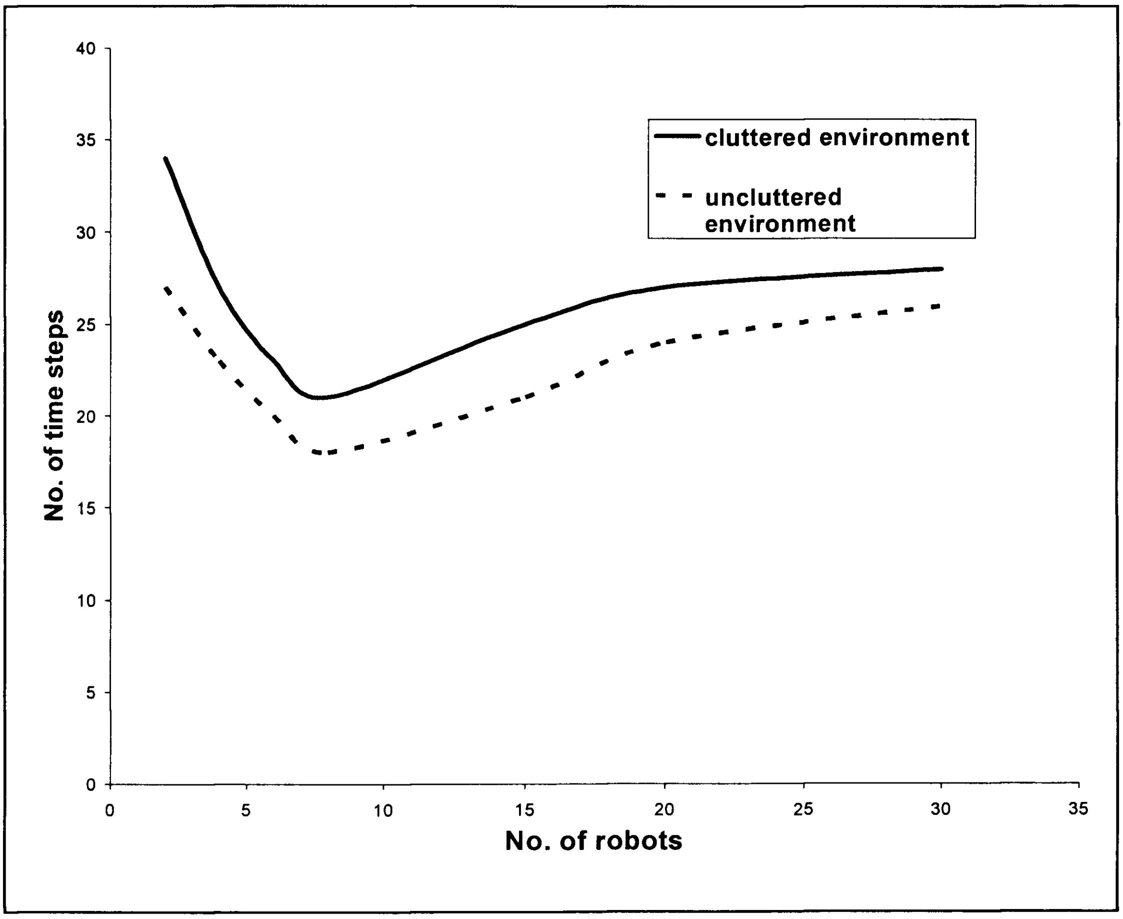


Figure 3.5(b): Effect of behaviour conflict on the performance of dynamic target tracking.

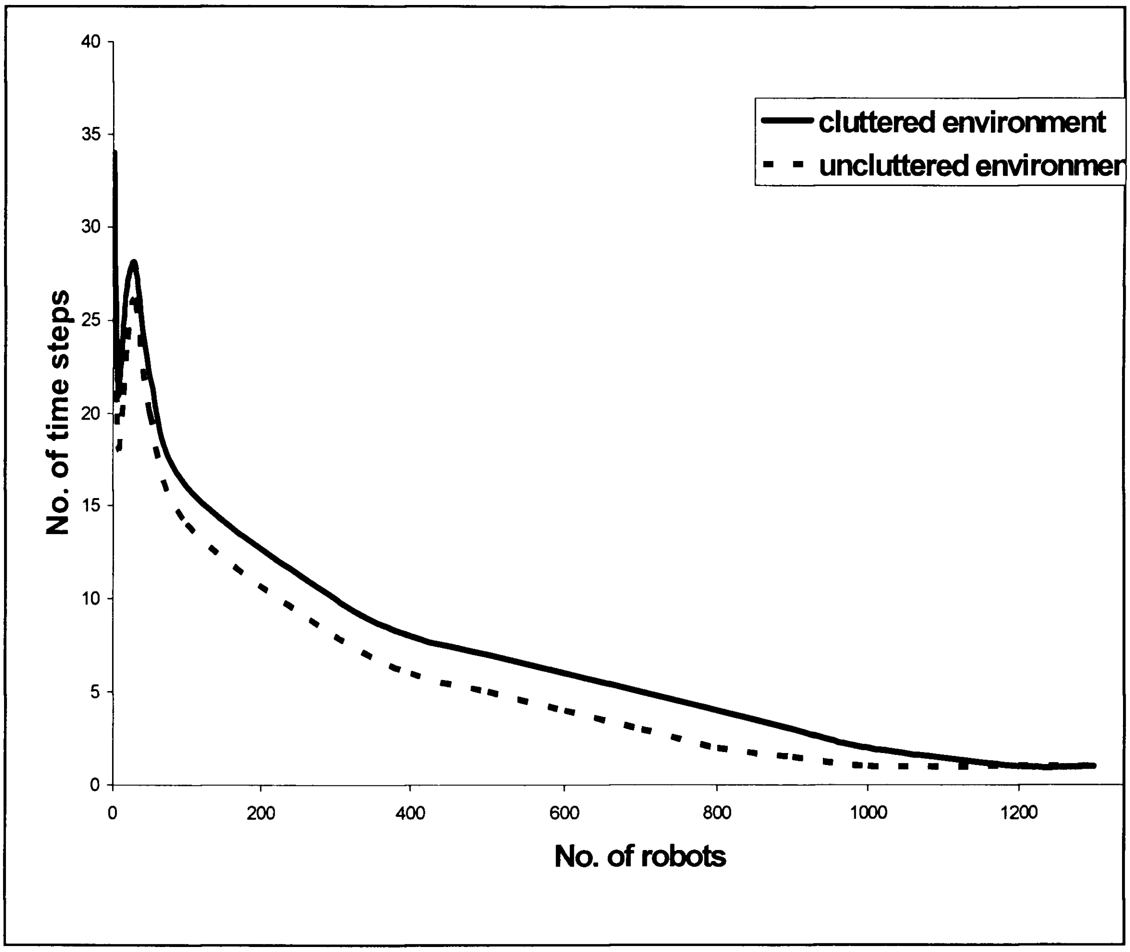


Figure 3.5(c): Mobile robots tracking a dynamic target in cluttered and uncluttered environments with varying numbers of robots

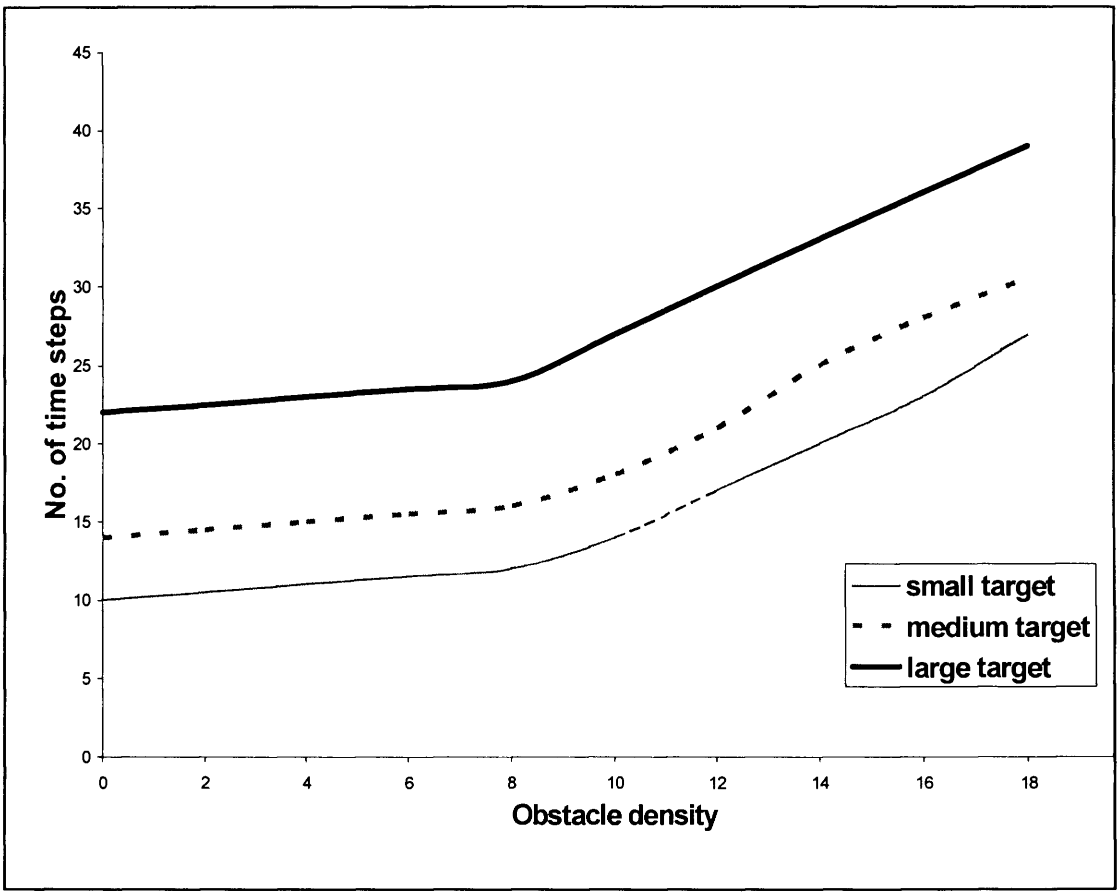


Figure 3.6: Effect of changing the obstacle density on dynamic target-tracking performance

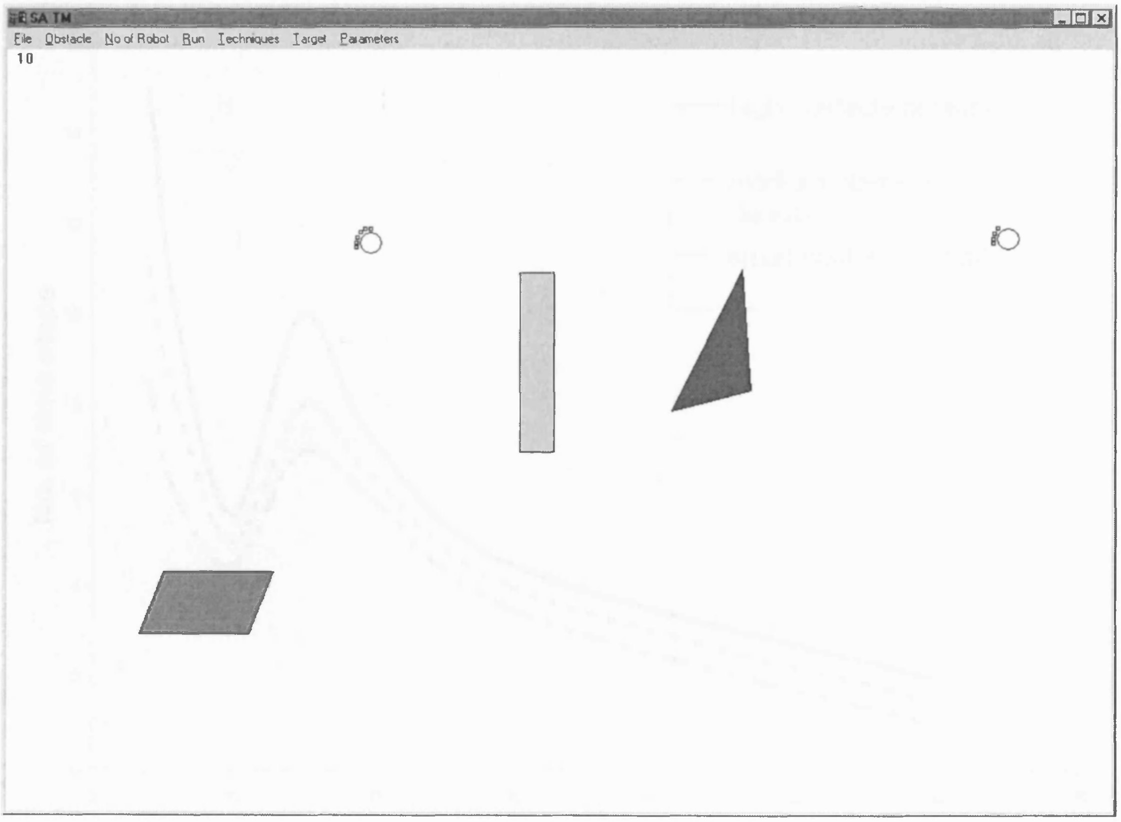


Figure: 3.7: Tracking dynamic multiple targets in a cluttered environment

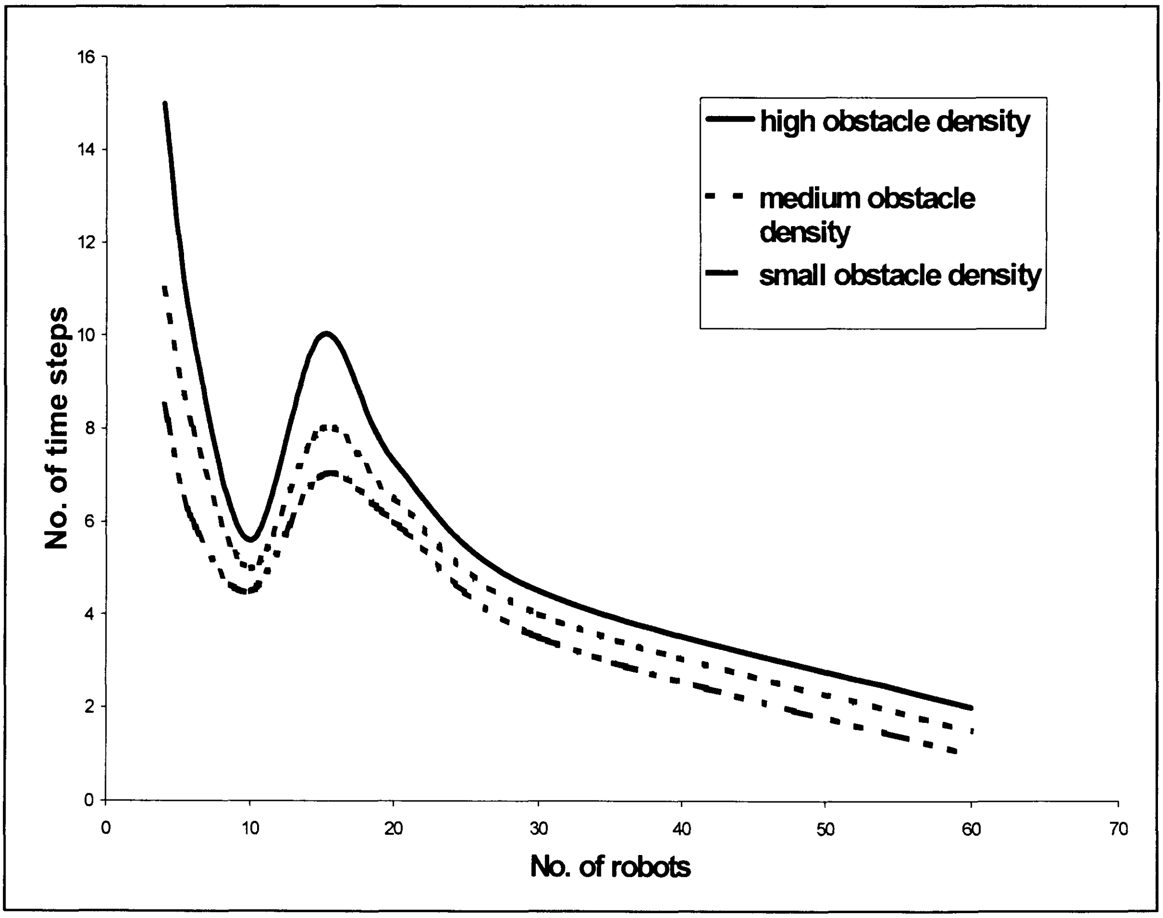


Figure 3.8:Multi-targets (two targets) tracking with varying the number of robots in different environments

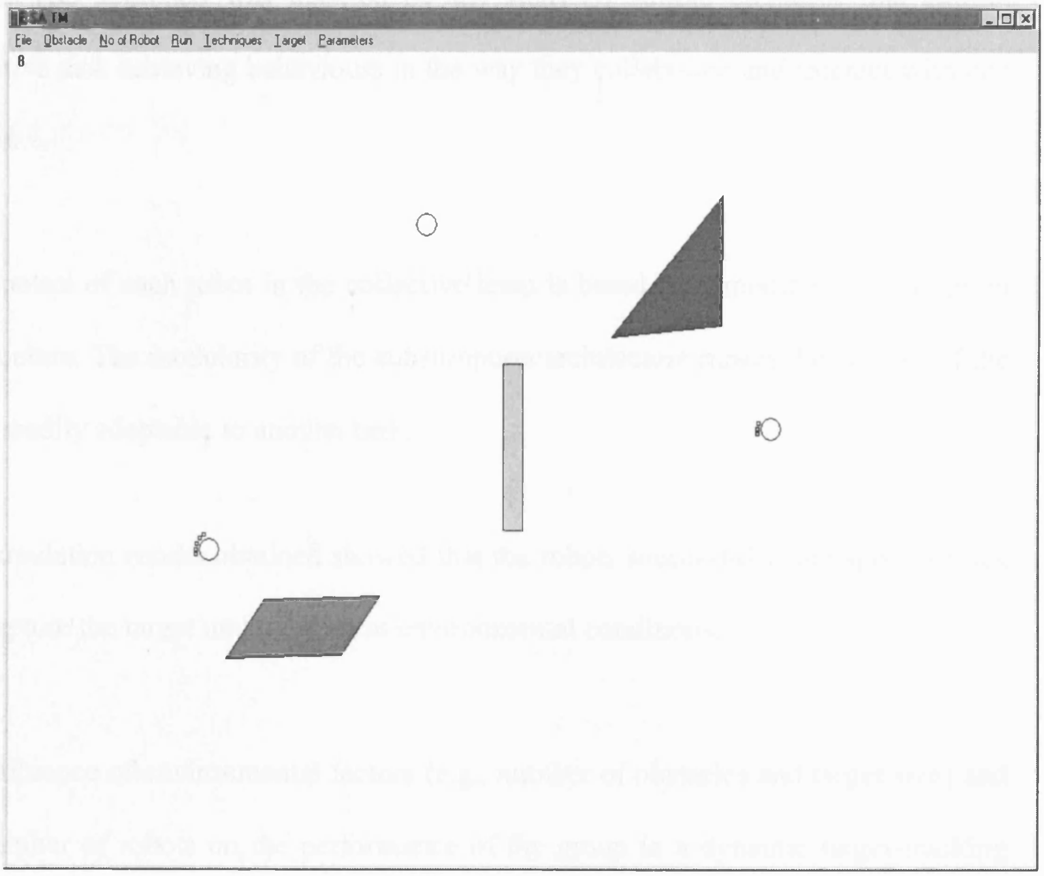


Figure 3.9: Robots could not track all the targets

3.6. Summary

This chapter has proposed an approach to controlling multiple robots that involves the use of collective behaviour resulting from the sensor-based behaviours of individual robots. The approach was inspired by the study of simple creatures that exhibit collective task achieving behaviours in the way they collaborate and interact with one another.

The control of each robot in the collective team is based on a modified subsumption architecture. The modularity of the subsumption architecture makes the control of the robot readily adaptable to another task.

The simulation results obtained showed that the robots successfully managed to track and capture the target under different environmental conditions.

The influence of environmental factors (e.g., number of obstacles and target size) and the number of robots on the performance of the group in a dynamic target-tracking task has been analysed. As expected, increasing the number of robots reduced the time required to track the target. However, robot collision and interference tended to degrade the performance. Continually adding more robots produced a proportional increase in performance.

For multiple targets, increasing the number of robots reduced the tracking time because the search space was reduced. However, the lack of coordination and task allocation algorithms caused some targets to escape.

Chapter 4

Fuzzy-Logic-Based Behaviour Coordination in Multi-Robot Systems

4.1 Preliminaries

Behaviour-based systems have proved to be useful in enabling robots to cope with the dynamics of real-world environments. The behaviour repertoire defines the skills available to a robot to enable it to react to situations encountered in its environment. A robot can exhibit multiple behaviours. Each behaviour is responsible for achieving or maintaining a particular objective. However, the objective of one behaviour might be in conflict with those of other behaviours and it is necessary to reach a compromise between conflicting objectives. This highlights the solution of actions to achieve the required trade-off as a major issue in the design of systems for control and co-ordination of multiple behaviours in a robot. For this purpose, a fuzzy logic technique for behaviour coordination is proposed. Fuzzy logic has been adopted as the basis of the technique because of its ability easily to combine the different individual behaviours in a robot with a modified subsumption-based control architecture.

The remainder of this chapter is organised as follows. Section 2 outlines the fuzzy logic technique for coordinating behaviours in each robot including command fusion and dynamic target tracking. The results obtained are presented and discussed in Section 3.

4.2 Fuzzy Behaviour Coordination

4.2.1 Command Fusion

The desired outcome can be achieved by integrating the outputs of the applicable behaviours, a process referred to as command fusion. For example, the outputs from the target following behaviour and the obstacle avoidance behaviour are combined to produce a heading that takes the robot towards its target whilst avoiding obstacles. An approach based on fuzzy sets operations is proposed here that takes into account the recommendations of all applicable behaviour modules. Behaviour coordination is achieved by weighted decision-making and rule-based (behaviour) selection. The weights used for weighted decision-making are the degrees of confidence placed on the different behaviours. They are empirical measures of applicability of particular behaviours.

To illustrate the process of behaviour coordination, assume there are just two behaviours B1 and B2 as shown in figure 4.1. The degrees of confidence for B1 and B2 are $\alpha_1 = 0.25$ and $\alpha_2 = 0.75$ respectively. The contribution of individual behaviours, each represented by a fuzzy set, is weighted by the corresponding degree of confidence. Thus, B1 and B2 are activated to degrees α_1 and α_2 respectively. Behaviour activation is accomplished via scalar multiplication of the output fuzzy sets by the appropriate degrees of confidence α_1 and α_2 in this example).

Multiplication of an output fuzzy set by a scalar α_i is equivalent to the conjunction of a set of uniform membership degree α_i with that output fuzzy set. The resulting fuzzy sets are then aggregated using an appropriate t-conorm operator (such as the 'Max' operator), and defuzzified to yield a crisp output u* that is representative of the intended behaviour.

In this procedure, the scalar α_i represents the weight of a behaviour in the aggregated control decision and multiplication by α_i expresses the applicability of the behaviour to the current situation. It is not necessary that the sum of the α_i's is equal to 1. This hypothetical example, illustrated in figure 4.1, reveals that the output of the system is influenced more by its dominant behaviour B2 as intended. Control recommendations from each applicable behaviour module are considered in the final decision. In general, the resultant control action can be thought of as a consensus of recommendations offered by multiple experts.

In some instances, it may be evident from current sensory data that only one particular behaviour is fully applicable (α_i associated with that behaviour is equal to 1). In this case, coordination simply reduces to behaviour selection, a process also referred to as switching coordination since behaviours are alternately switched on and off.

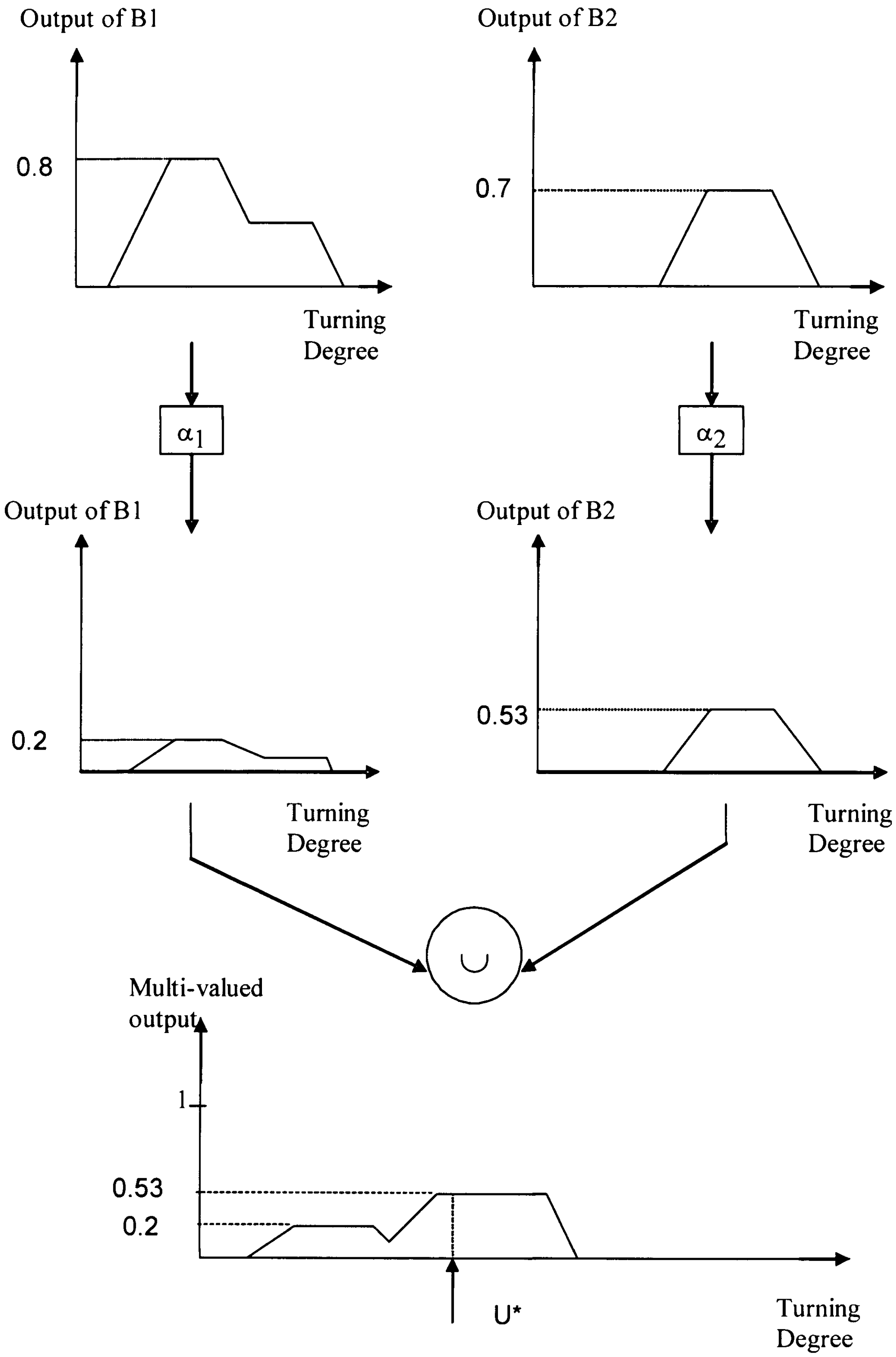


Figure 4.1: Fuzzy coordination of behaviours

4.2.2. Fuzzy-Logic-Based Dynamic Target Tracking Behavioural Architecture

As with other behavioural approaches, the fuzzy-logic-based architecture for mobile robots, in the context of a dynamic target tracking system, consists of several behaviours, such as target following and obstacle avoidance. Each behaviour relates sensor data and robot status to control recommendations, as shown in figure 4.2, and has two components: a set of fuzzy rules and a fuzzy inference module. The fuzzy rules of a behaviour explicitly represent its control strategy in the form of linguistic statements. As illustrated in figure 4.2, multiple behaviours could share a common fuzzy inference module. Fuzzy control recommendations generated by all behaviours are fused and defuzzified to generate a final crisp control command.

The basic algorithm executed in every control cycle by the architecture consists of the following four steps: (1) the target following behaviour determines the desired turning direction; (2) the obstacle avoidance behaviour determines the disallowed turning directions; (3) the command fusion module combines the desired and disallowed directions and (4) the combined fuzzy command is converted into a crisp command through a defuzzification process. The desired and disallowed directions are maintained in fuzzy set form to reduce possible loss of information during command fusion. Figure 4.3 shows an example of a situation in which the target following behaviour suggests that the robot should turn left, but the robot must continue straight on a little longer to avoid the obstacle on the left (i.e., obstacle A). This example is used in the following subsections to describe and demonstrate each step of the algorithm.

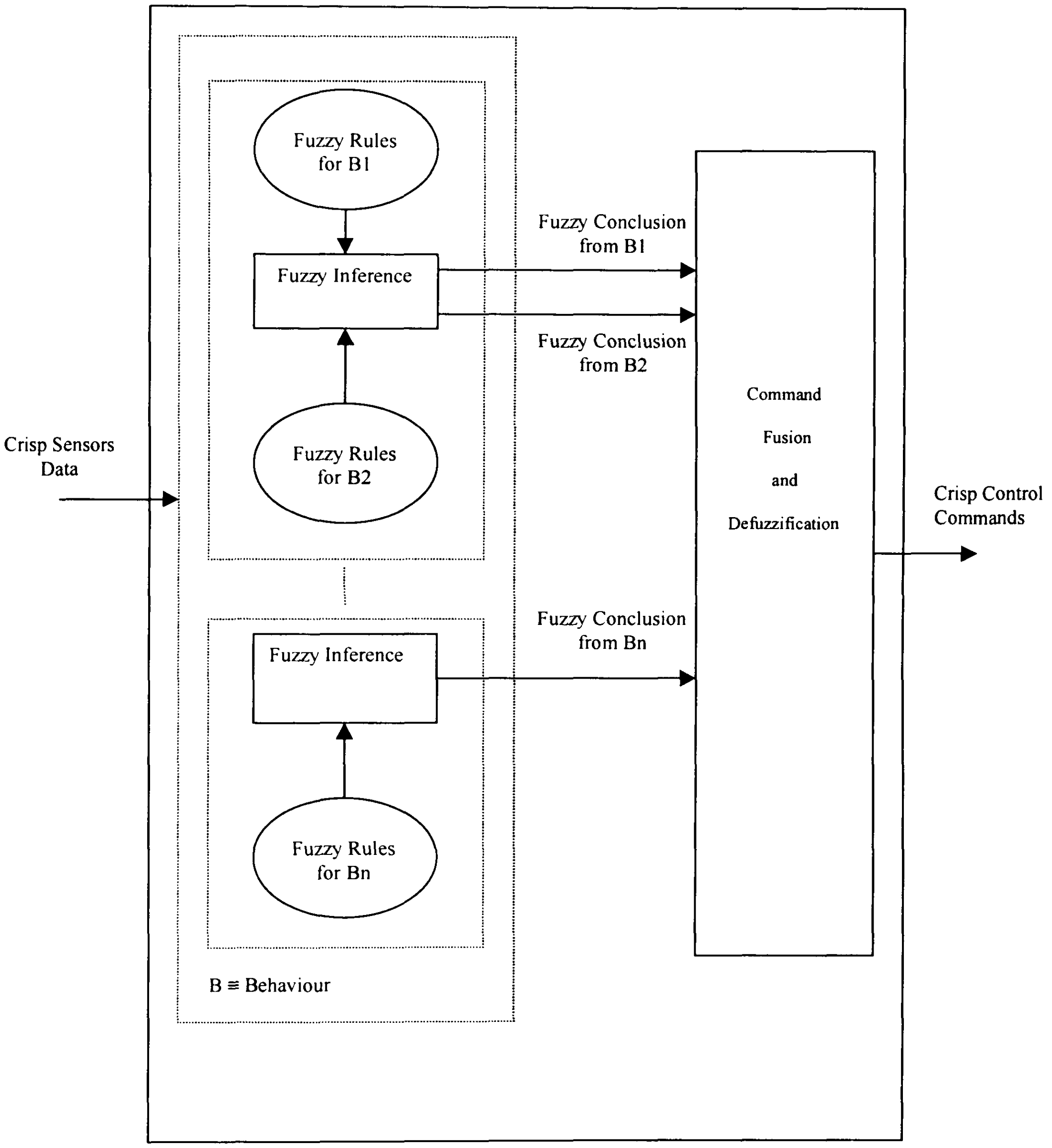


Figure 4.2: Fuzzy logic based behavioural architecture

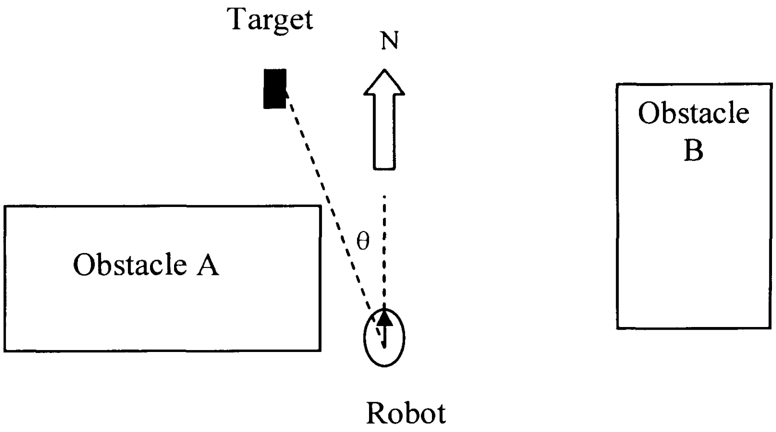
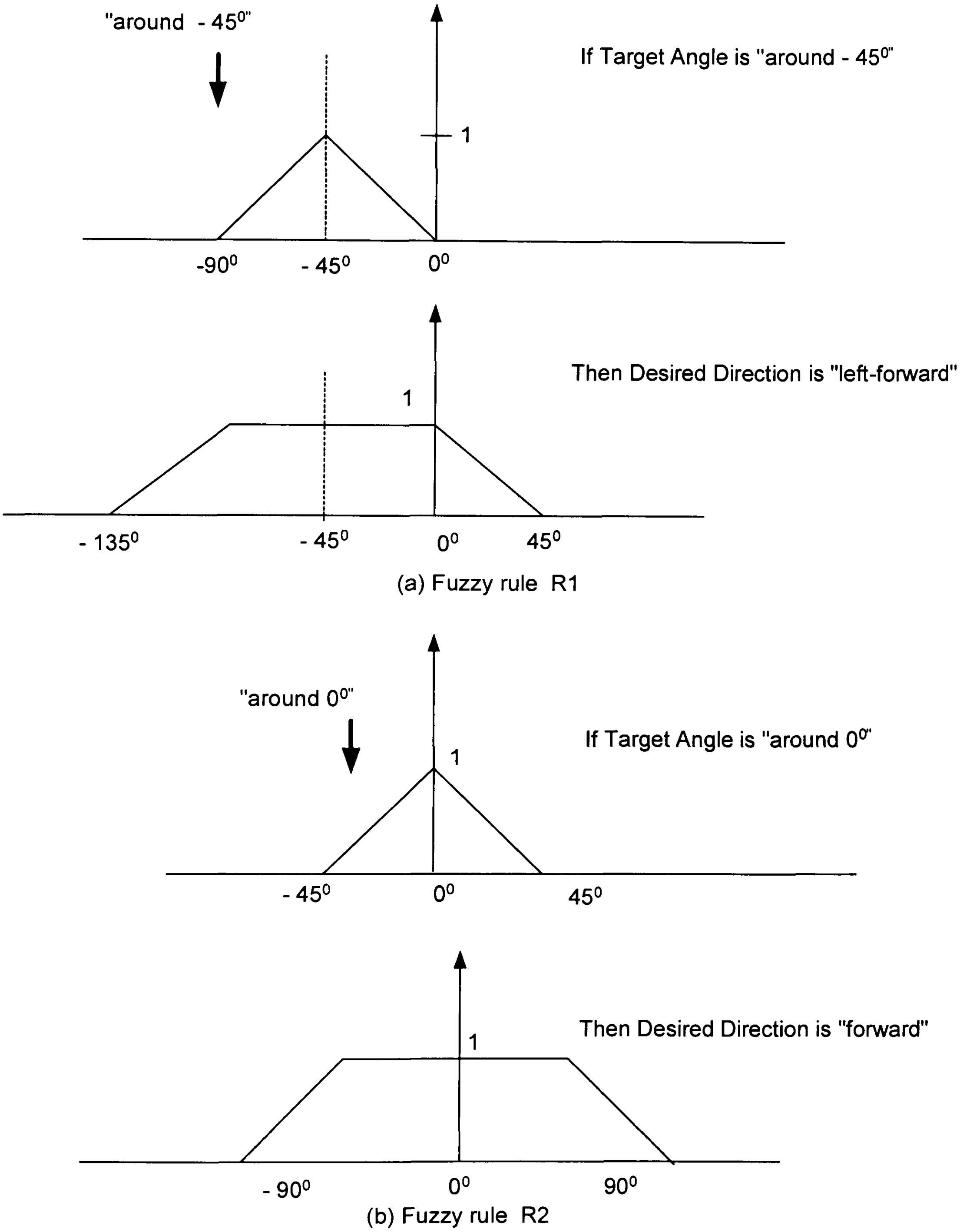


Figure 4.3: An example of a situation where a robot wants to move towards a target in the proximity of obstacles

4.2.2.1. Target Following Behaviour

The target following behaviour generates a desired turning direction based on the current location of the target and its current bearing. The target following behaviour determines the desired steering direction in three steps. First, it senses the target. Second, the behaviour computes the target angle, which is the angle between the current direction of the robot and a vector from its current location to the target. For the example given, with the robot heading North, the target angle θ is –30 degrees. Third, the behaviour uses a set of fuzzy rules to change the specific target angle into a general desired direction, which gives the robot more flexibility in avoiding obstacles while still following the target. Figure 4.4 shows the two fuzzy rules R1 and R2 employed in the example by the target following behaviour. The fuzzy inference module of the target following behaviour combines the desired directions recommended by all target following behaviour fuzzy rules using weighted decision-making as explained previously. The process is illustrated in figure 4.5 for a target angle of –30 degrees using R1 and R2 from figure 4.4. The antecedent membership functions (i.e., "around 0 degrees" and "around – 45 degrees") are designed to overlap such that the sum of their membership values in that region is 1.0.


"around - 45⁰"
If Target Angle is "around - 45⁰"
1
-90⁰
- 45⁰
0⁰
Then Desired Direction is "left-forward"
1
- 135⁰
- 45⁰
0⁰
45⁰
"around 0⁰"
If Target Angle is "around 0⁰"
1
- 45⁰
0⁰
45⁰
Then Desired Direction is "forward"
1
- 90⁰
0⁰
90⁰

(a) Fuzzy rule R1

(b) Fuzzy rule R2

Figure 4.4: Fuzzy rules for target following

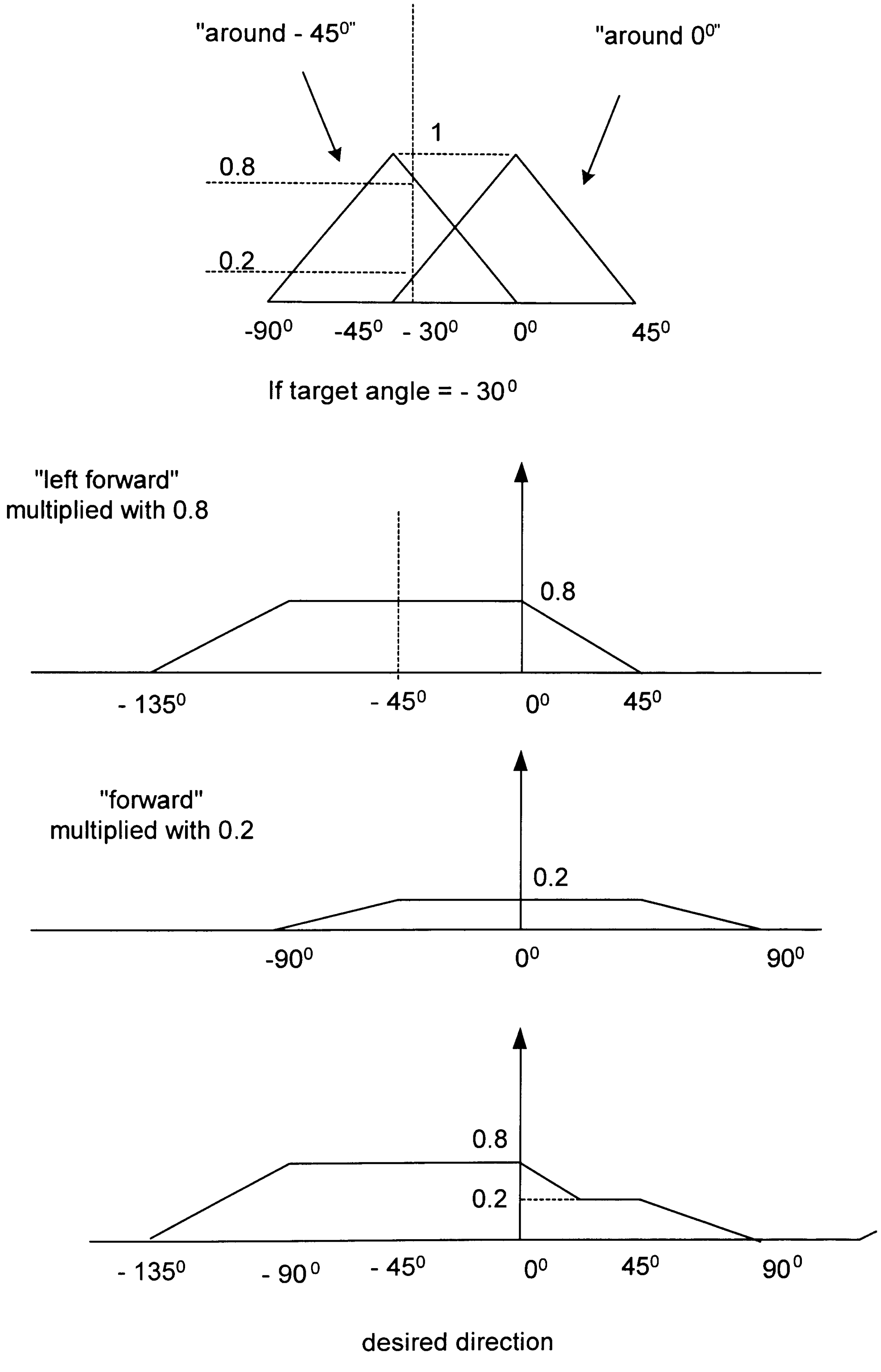


Figure 4.5: An example of computing desired direction

4.2.2.2. Obstacle Avoidance Behaviour

The obstacle avoidance behaviour uses ultrasonic data to generate a fuzzy set that represents the disallowed directions of travel (i.e., directions that lead, in the short term, to or near an obstacle). The behaviour operates by first comparing the distance of the closest obstacle detected by each direction sensor to a fuzzy set, "Near", associated with the sensor. Based on the result of the comparison, the behaviour determines the degree to which the general direction of each sensor is considered disallowed. Examples of fuzzy rules used by the obstacle avoidance behaviour are shown in figure 4.6. The membership functions of disallowed turning directions associated with a sensor have been designed such that: (1) they partially overlap those of neighbouring sensors and (2) they have a major influence on the direction of the sensor.

Once all the fuzzy rules associated with the obstacle avoidance behaviour have been fired, their fuzzy conclusions are combined using the Max operator. Figure 4.7 shows an example of this combination with sensor inputs (three ultrasonic sensors) based on the situation in figure 4.3. Here, the fuzzy inference module of the behaviour uses the Max operator instead of other t-conorm operators (such as the arithmetic sum) because it is consistent with the intuitive idea that the degree to which a direction is disallowed should be determined by the sensor source that has the strongest opinion about it.

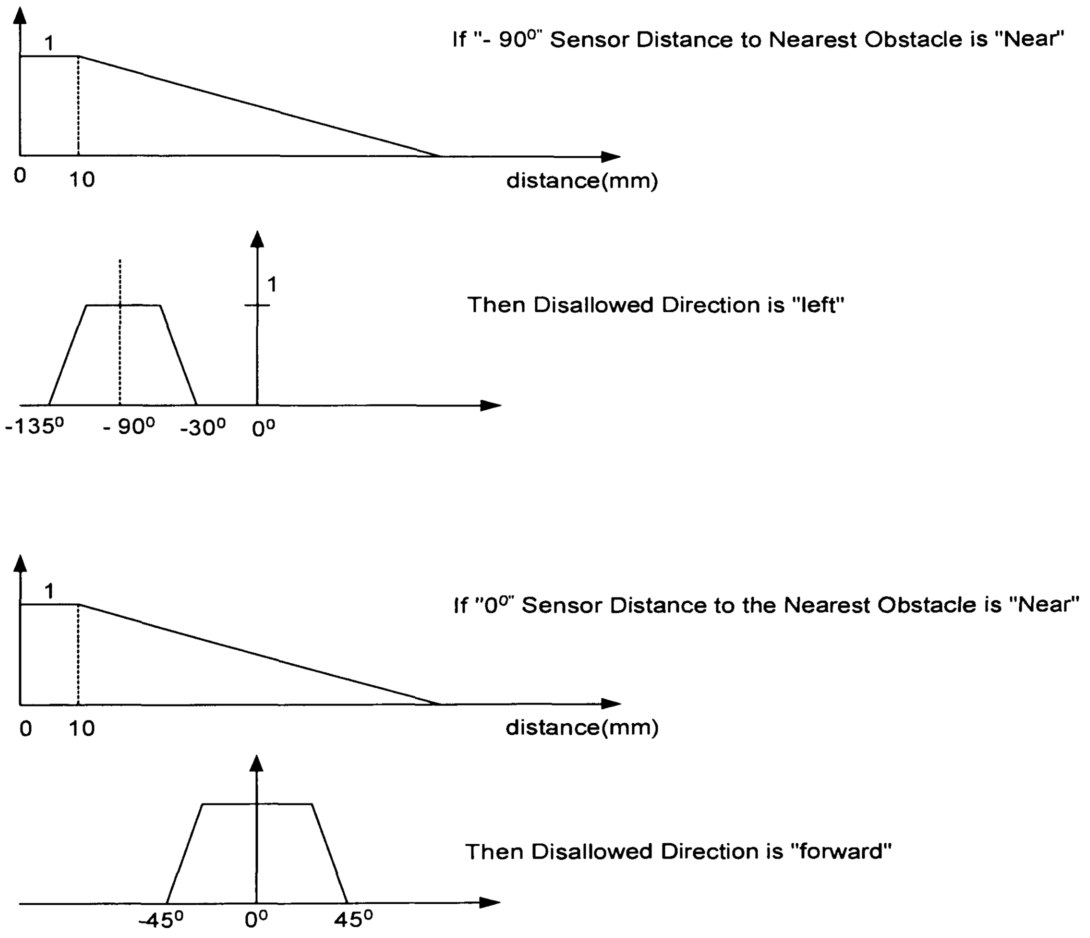
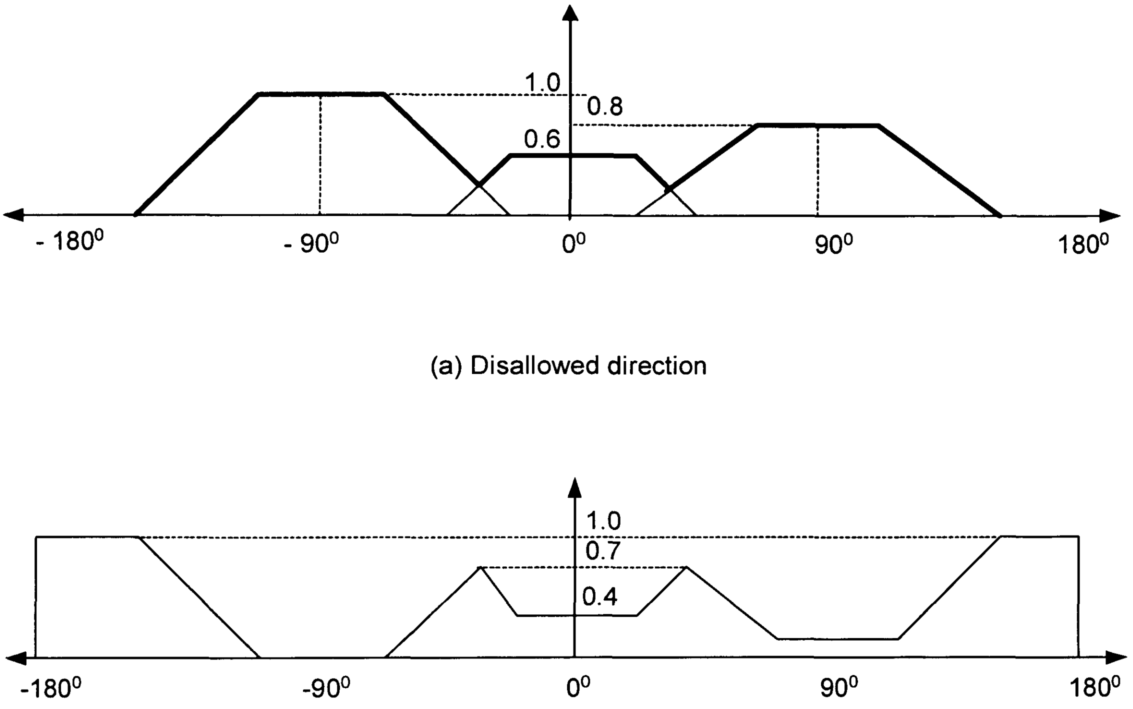

1
If "- 90°" Sensor Distance to Nearest Obstacle is "Near"
0
10
distance(mm)
1
Then Disallowed Direction is "left"
-135°
- 90°
-30°
0°
1
If "0°" Sensor Distance to the Nearest Obstacle is "Near"
0
10
distance(mm)
Then Disallowed Direction is "forward"
-45°
0°
45°

Figure 4.6: Fuzzy rules used for obstacle avoidance



(a) Disallowed direction

(b) Allowed direction (complement of (a))

Figure 4.7: Robot turning directions

4.2.2.3. Fuzzy Command Fusion

The third component of the mobile robot controller combines the fuzzy conclusions about the desired direction and the disallowed direction into a single fuzzy control command. Since the final robot direction should be both desired from the target following viewpoint and not disallowed by obstacle avoidance considerations, the command fusion module uses the Min operator in fuzzy logic to form a conjunction of the output of the two behaviours as follows:

$$
\begin{aligned}
\mu_{\text{Turning-Direction}(x)} &= \mu_{\text{Desired}(x) \text{ AND Not Disallowed}(x)} \\
&= \min\{\mu_{\text{Desired }(x)}, \mu_{\text{Not Disallowed }(x)}\} \\
&= \min\{\mu_{\text{ Desired}(x)}, \mu_{\text{ Allowed}(x)}\}
\end{aligned}
$$

For convenience, the negated Disallowed Direction will be referred to as the Allowed Direction of travel.

With reference to the example in figure 4.3, figure 4.8 illustrates the command fusion and defuzzification steps under consideration.

4.2.2.4. Defuzzification

For the case shown in figures 4.3 and 4.8, by using the centre of area (COA) strategy, the crisp turning direction is found to be – 20 degrees. This angle combines the recommendations of both the target following and obstacle avoidance behaviours (see figure 4.4 and 4.7b) and enables the robots to approach the target without colliding with obstacles.

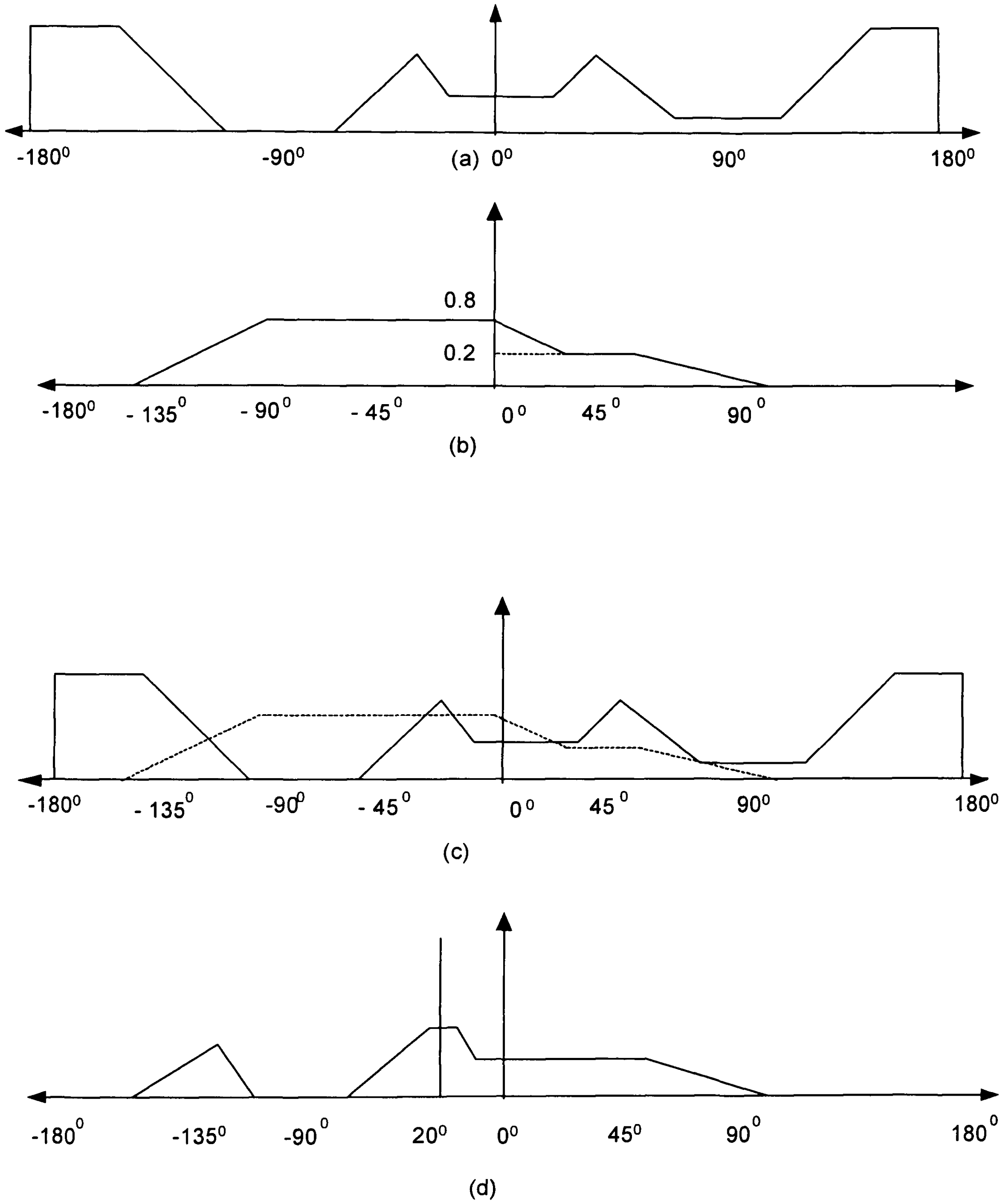


Figure 4.8: Steps to determine the proper turning direction
(a) Allowed direction
(b) Desired direction
(c) Fusion between allowed and desired directions
(d) Defuzzification by using centre of area

4.3. Experiments and Discussion

The experiments reported in chapter three, concerning tracking a dynamic target in a limited arena with a different number of robots and different obstacles, were repeated to demonstrate the fuzzy logic technique for behaviour coordination. In addition, situations in which the robots face conflicting behaviours, such as obstacle avoidance and target tracking, have been illustrated, to prove the reliability of the technique. Without coordination (as in chapter three), when the robots face obstacles while tracking a target, as shown in Figures 4.9a - 4.11a, obstacle avoidance has the highest priority. Even though the robots subsequently lose the target, the robots avoid the obstacles and one another as shown in Figures 4.9b - 4.11b. Subsequently, the robots start searching again, wasting much time. However, with behaviour coordination (the resolution of conflicts between contradictory behaviours), obstacle avoidance and target tracking are achieved by selecting an action that represents the consensus among the behaviours and that best satisfies the decision objectives that they encode.

As shown in Figures 4.9c - 4.11c the robots avoid the obstacles and moved directly towards the target to follow it. After these specific instances showed improvement in the system performance, the experiments described in chapter three were repeated and evidence of the successful application of fuzzy logic for behaviour coordination is shown in figure 4.12. Where conflict among contradictory behaviours is correctly managed by behaviour coordination, the tracking time reduces with increasing numbers of robots.

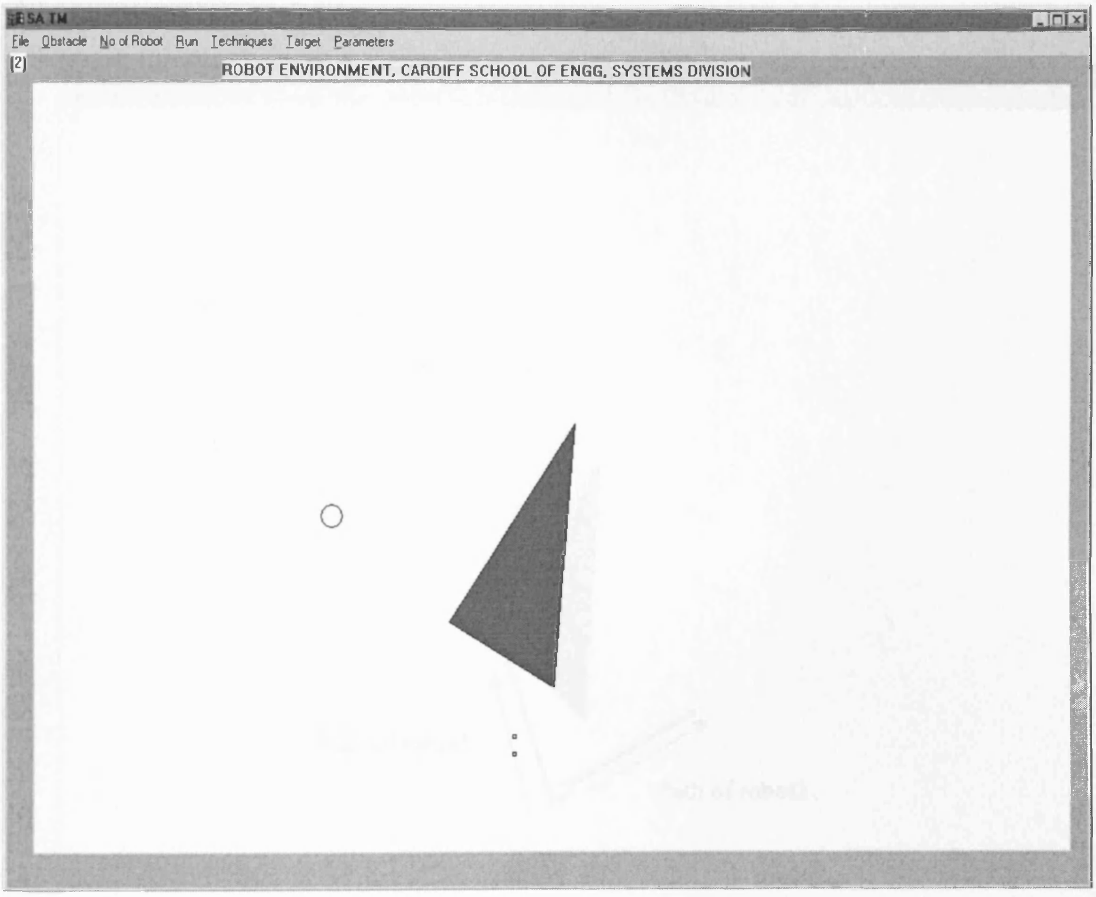

ESA TM
File Obstacle No of Robot Run Techniques Target Parameters
[2]
ROBOT ENVIRONMENT, CARDIFF SCHOOL OF ENGG, SYSTEMS DIVISION

Figure 4.9(a): Initial environment (one target, one obstacle, and two robots)

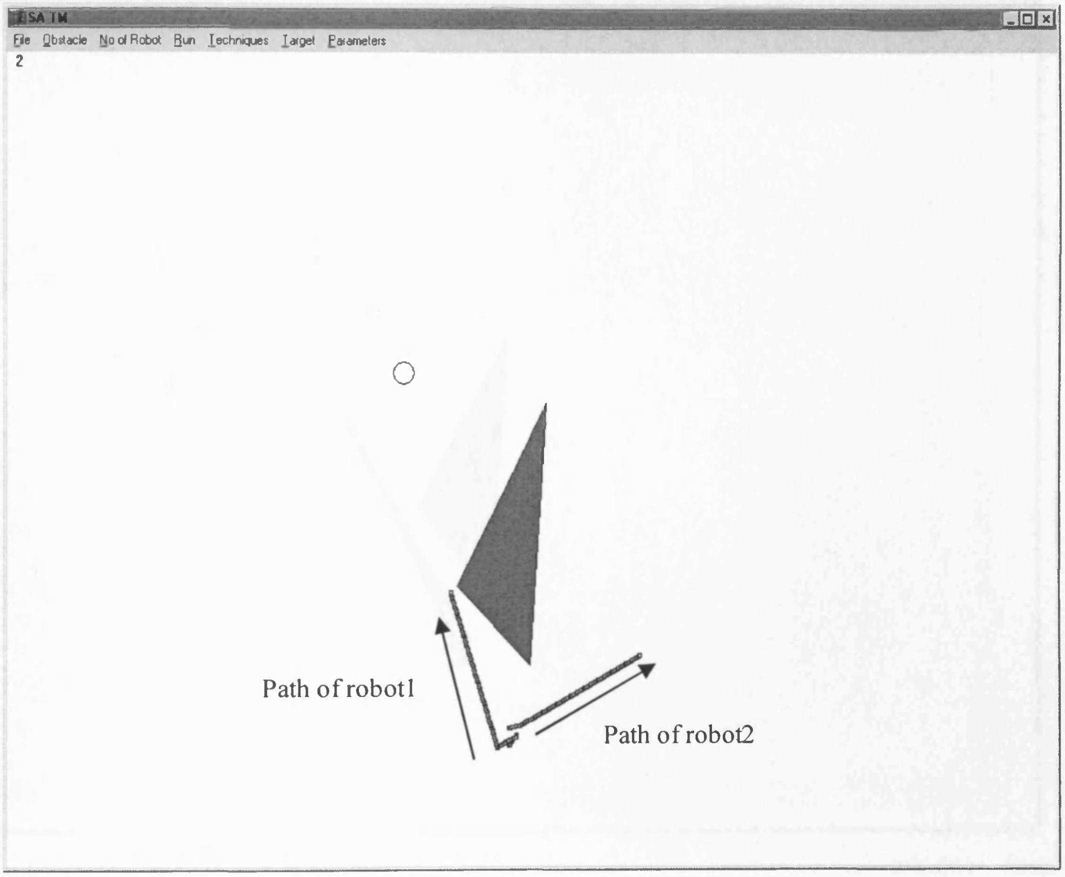


Figure 4.9(b): Missing of target when there is no coordination

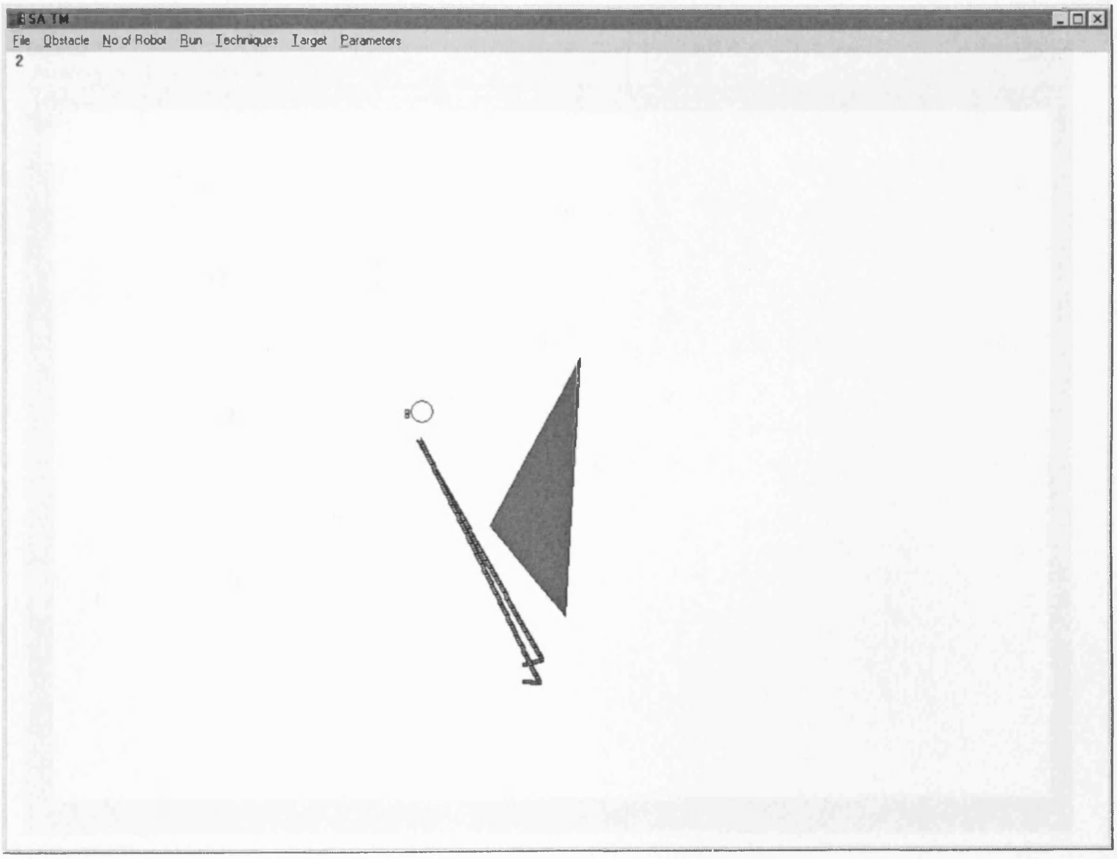


Figure 4.9(c): Successful capturing of target when there is coordination

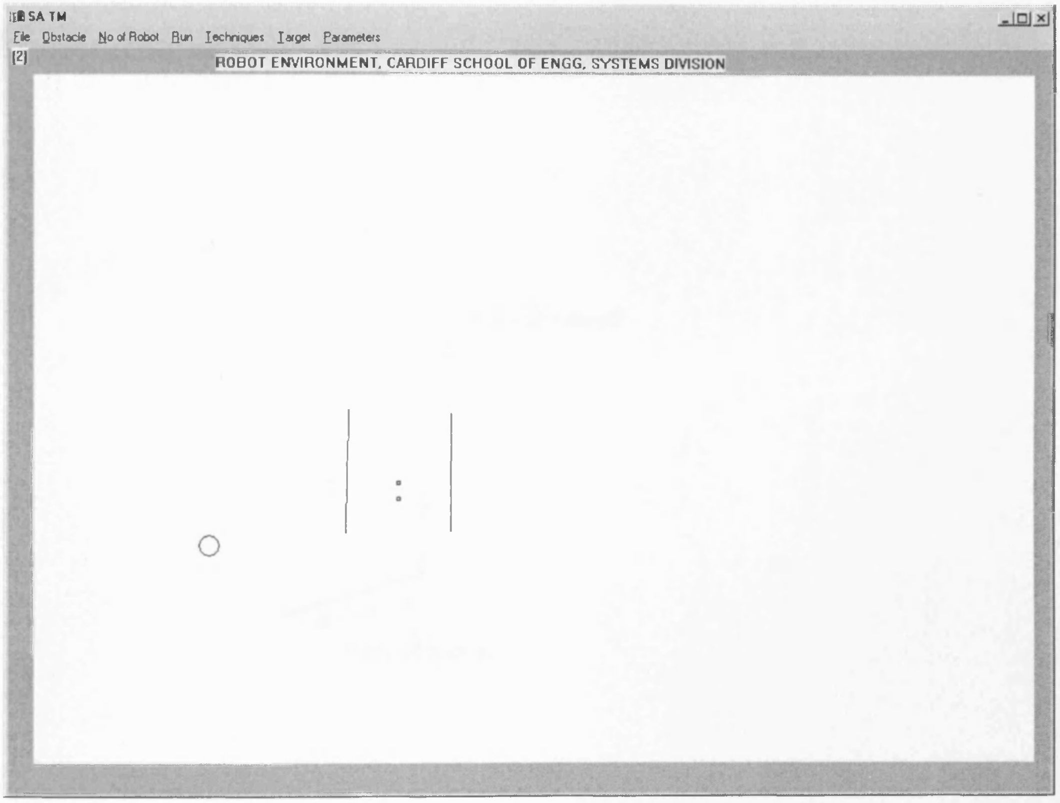

SA TM
File Obstacle No of Robot Run Techniques Target Parameters
[2]
ROBOT ENVIRONMENT, CARDIFF SCHOOL OF ENGG, SYSTEMS DIVISION

Figure 4.10(a): Initial environment (one target, two obstacles, and two robots)

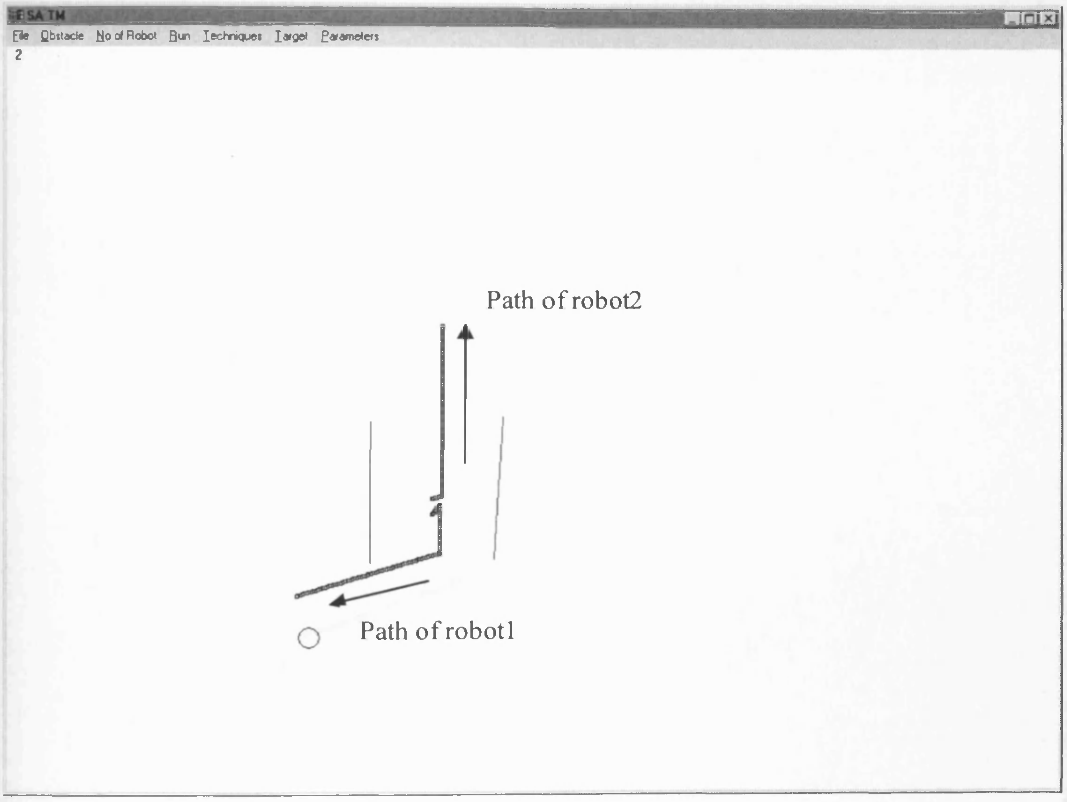


Figure 4.10(b): Missing of target when there is no coordination

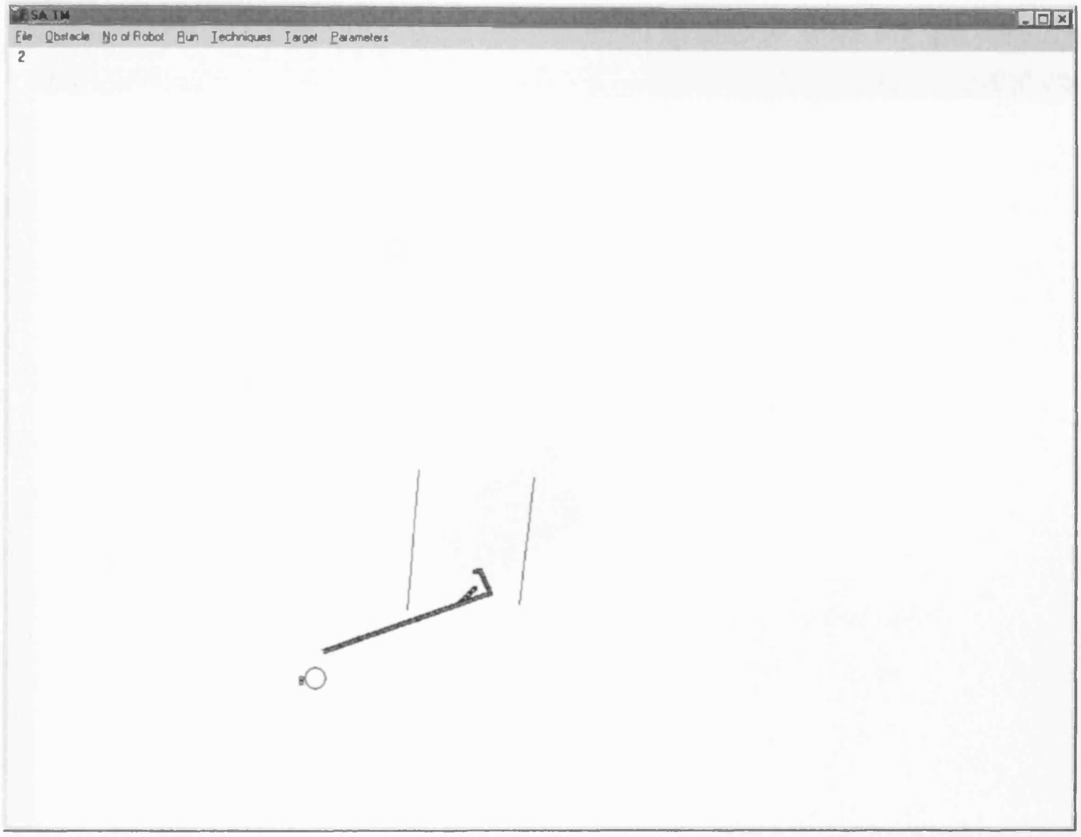


Figure 4.10(c): Successful capturing of target when there is coordination

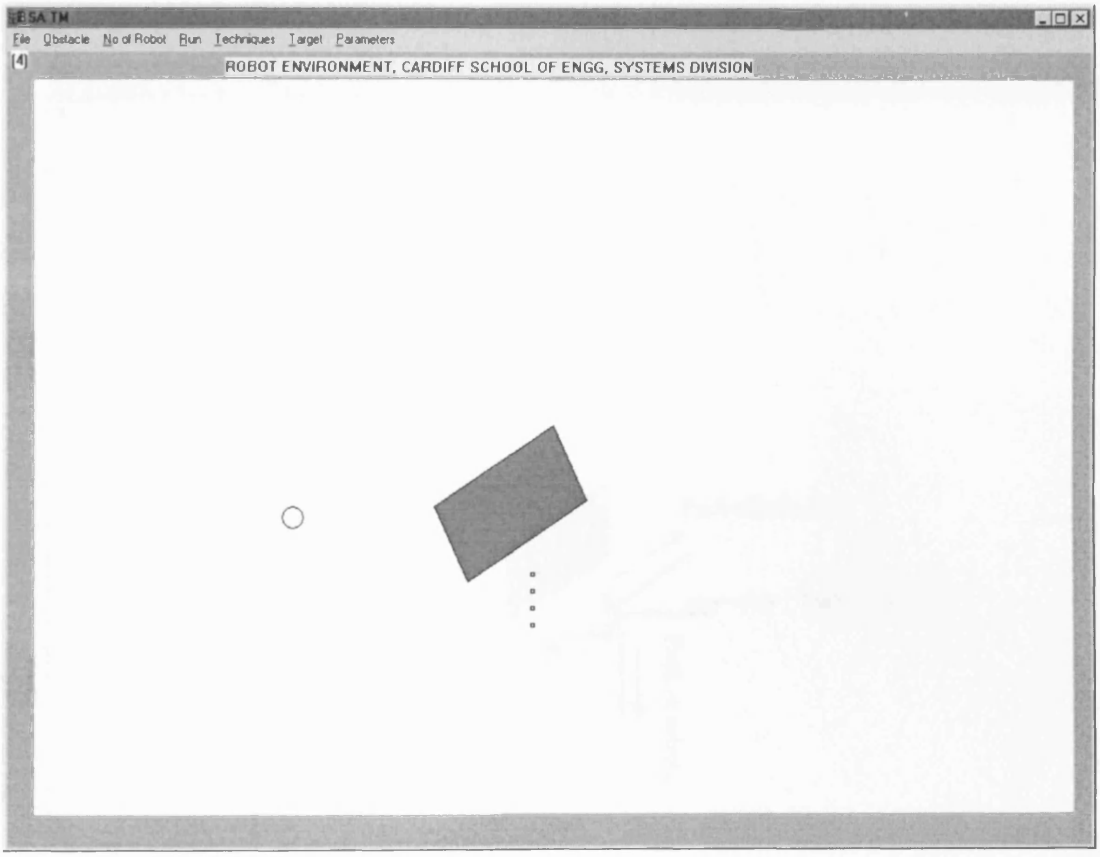


Figure 4.11(a): Initial environment (one target, one obstacle, and four robots)

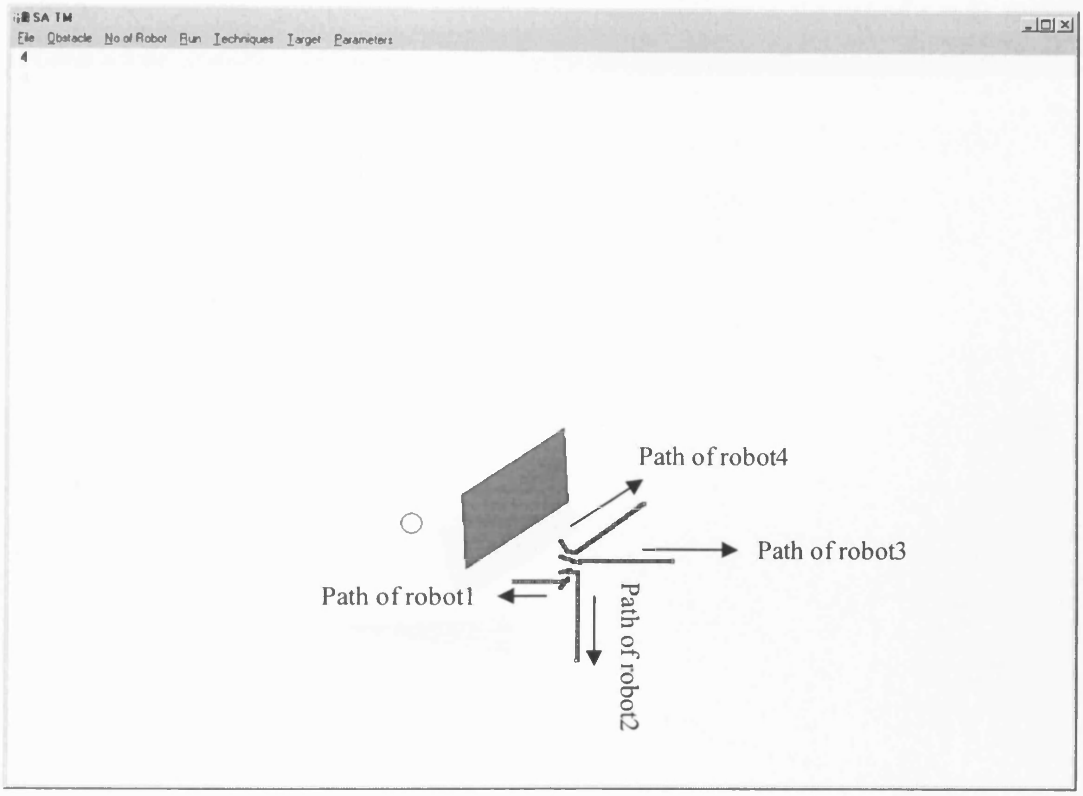


Figure 4.11(b): Missing of target when there is no coordination

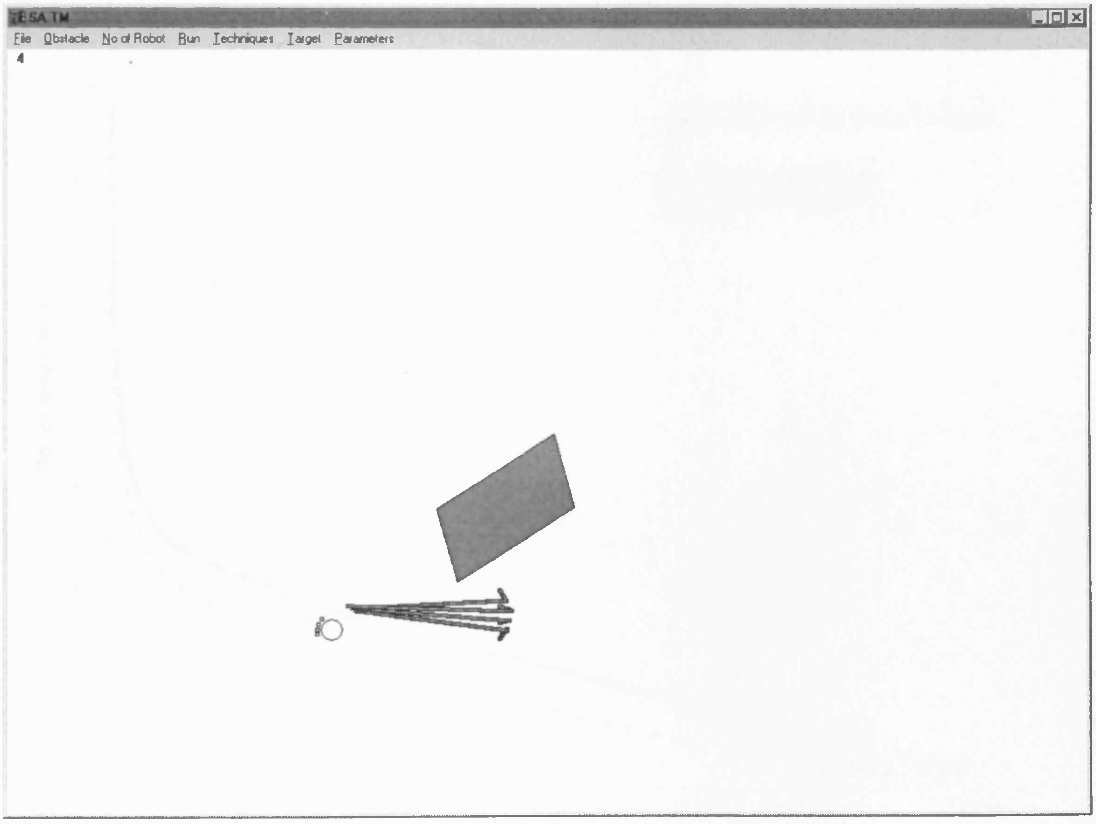


Figure 4.11(c): Successful capturing of target when there is coordination

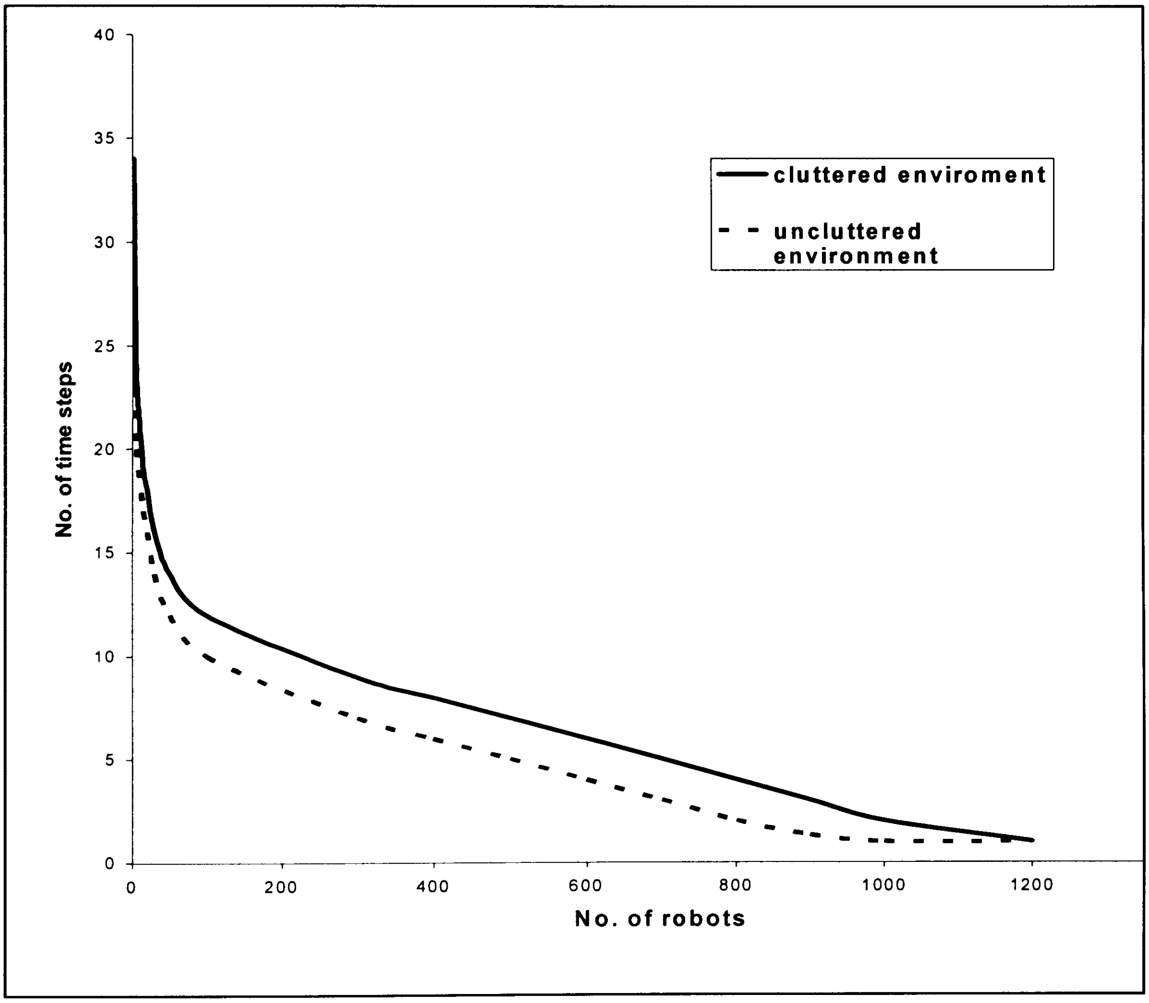


Figure 4.12: Tracking a dynamic target with varying numbers of robots

PRIFYSGOL

4.4. Summary

This chapter has demonstrated how a fuzzy logic technique enables the resolution of conflicts between contradictory robot behaviours by selecting an action that represents the consensus among the behaviours and that best satisfies the decision objectives that they encode. The results show an improvement in the global performance of a multiple robot system.

Chapter 5

Knowledge-Based Software Architecture for Adaptive Co-operative Mobile Robots

5.1 Preliminaries

Multi-robot teams can increase the reliability, flexibility, robustness and efficiency of automated solutions by taking advantage of the redundancy and parallelism of multiple team members. Before multi-robot teams can become widely used in practise, it is necessary to develop automated techniques that enable robot team members automatically to adapt their actions over time in response to changes in their environment or in the robot team itself.

Achieving adaptive co-operative robot behaviour is more challenging. Many issues must be addressed in order to develop a working co-operative team; these include action selection, task allocation, coherence, communication, resource conflict resolution, and awareness. Awareness of other members of the robot team is a necessary component of co-operation; however this causes an increase in the search space dimension (Touzet, 2000).

A knowledge-based software architecture is proposed to enable robot agents to accomplish collective behaviours and adapt their performance during the specified time of the mission. The improvement in team performance is achieved by updating the control of the robots based on knowledge acquired on-line.

The remainder of the chapter is organised as follows. Section 2 outlines the proposed adaptive co-operative action selection architecture. Section 3 explains the performance evaluation and monitoring modules of the architecture. The control strategy, comprising off-line and on-line learning phases, is described in Section 4. The feed-forward neuro-fuzzy technique and parameter learning algorithms are described in Section 5 and Section 6. Section 7 presents simulation results for a box pushing exercise using existing simulation software.

5.2 Adaptive Co-operative Action Selection Architecture

The major design goal in the development of this architecture is to address the real-world issues of behaviour coordination, fault tolerance and adaptivity when using teams of fallible robots equipped with noisy sensors and effectors. The architecture must also allow the building of robot teams able to cope with failures and uncertainty in action selection and action execution, and with changes in a dynamic environment. Furthermore, in order to maintain a purely distributed co-operative control scheme which affords an increased degree of robustness, individual agents must always be fully autonomous, with the ability to perform useful actions even amidst the failure of the other robots.

The architecture is developed to be fully distributed, and giving all robots the capability to determine their own actions based upon their current situation, the activities of other robots and the current environmental conditions. No centralised control is utilised, so that it is possible to investigate the power of a fully distributed robotic system to accomplish group goals.

The components of this architecture (Figure 5.1) will be explained in this chapter, with the exception of the fuzzy-logic-based action selection arbiter which was explained in Chapter 4.

5.2.1. Assumptions

In this architecture, it is not required that a robot be able to determine the actions of its team-mates through passive observation, which can be difficult to achieve. Instead, robots are enabled to learn about the actions of their team-mates through an explicit communication mechanism, whereby the robots broadcast information concerning their current activities to the rest of the team.

Furthermore, it is assumed that the robots are built to work as a team, and are neither in direct competition with one another, nor attempting to subvert the actions of their team-mates, although conflict may arise at a low level due to, for example, the sharing of compatible goals, (Note, however, that some multi-robot team applications, such as robot soccer and military battles, may require the ability to deal with adversarial teams).

It is further assumed in the architecture that robots do not have access to some centralised store of knowledge, and that no centralised agent is available that can monitor the state of the entire robot environment and make control decisions based upon this information.

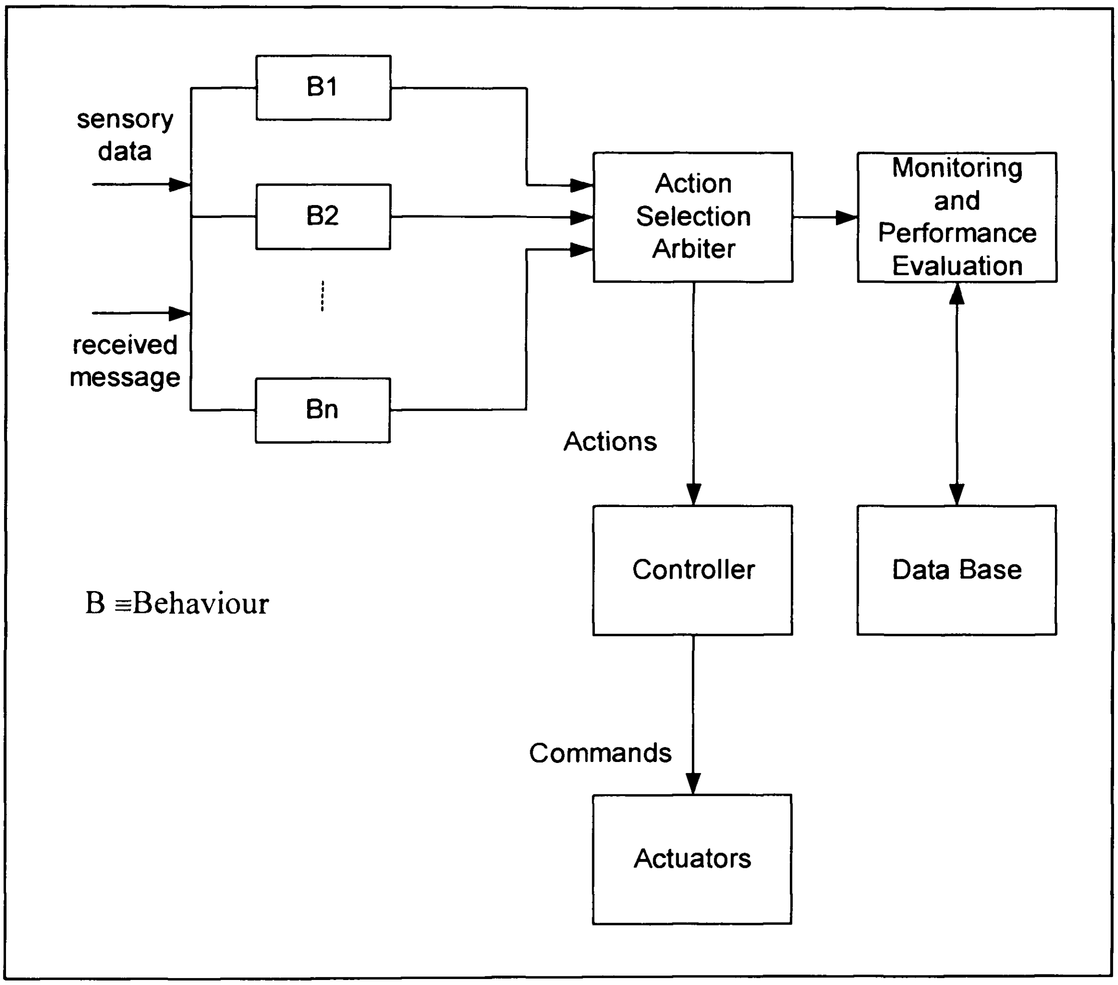


Figure 5.1: Adaptive co-operative action selection architecture

5.2.2. Architecture Mechanism

At all times during the mission, the motivation for each robot to activate a certain behaviour set is based on receiving sensory input and inter-robot communication. When these inputs are valid, the behaviour set becomes active.

Intuitively, the motivation of robot r_i to activate any given behaviour set is initialised to 0. Over time, the motivation increases quickly as long as the task corresponding to that behaviour set is not being accomplished, as determined from sensory feedback.

However, robots also have to be responsive to the actions of other robots, adapting their task selection to the activities of team members. Thus, if robot r_i is aware that another robot r_k is working on a certain task T1, then r_i should be satisfied for some time (based on knowledge learned on-line) that the task will be accomplished even without its own participation, and thus go on to some other applicable actions. Its motivation to activate the behaviour set (addressed by another robot) still increases, but at a slower rate. This characteristic prevents any robot from replicating the actions of the others and wasting energy. Of course, detecting and interpreting the actions of the other robots (sometimes called action recognition) is not a trivial problem, and often requires perceptual abilities that are not yet possible with current sensing technology. Thus, to enhance the perceptual abilities of the robots, the architecture utilises a simple form of broadcast communication to allow robots to inform other team members of their current activities, rather than relying totally on sensory capabilities. At a pre-specified rate, each robot r_i broadcasts a statement of its current action, which other robots may listen to or ignore as they wish. Two-way communication is not employed in this architecture.

Each robot is designed to be somewhat impatient, in that a robot r_i is only willing for a certain amount of time to allow the messages from another robot to affect its own motivation to activate a given behaviour. Continued sensory feedback indicating that a task is not accomplished thus overrides the statements of another robot performing that task. This characteristic allows robots to adapt to failures of other robots, causing them to ignore a robot that is not successfully completing its task.

A complementary characteristic in these robots is acquiescence (compliance). Just as the impatience characteristic of a robot reflects the recognition that other robots may fail, the acquiescence characteristic recognises that the robot itself may fail. This feature operates as follows. As a robot r_i performs a task, its willingness to give up that task increases over time provided that the sensory feedback indicates that the task has not been accomplished. As soon as some other robot r_k signals it has begun that same task and r_i feels that it (r_i) has attempted the task for an adequate length of time, the unsuccessful robot r_i gives up its task in an attempt to find an action at which it is more productive. Additionally, even if another robot r_k has not taken over the task, robot r_i may give up its task anyway if it is not completed within a time limit. This allows r_i the possibility of working on another task that may prove to be more fruitful rather than attempting in vain to perform the original task forever. With this acquiescence characteristic, therefore, a robot is able to adapt its actions to its own failure.

As a simple illustrative example, consider a team of two robots A and B unloading boxes from a truck and placing them on one of two conveyor belts, depending upon the labelling on the box. Both robots have the ability to unload boxes from the truck to a temporary storage location, and the ability to move them from the temporary storage location to the appropriate conveyor belt, (it is assumed that, due to the way the loading dock is designed, the robots cannot move boxes immediately from the truck to the conveyor belt). At the beginning of the mission, say robot A elects to unload the boxes from the truck. Robot B is then satisfied that the boxes will be unloaded, and proceeds to move the boxes from the temporary location to the correct conveyor belt. As the mission progresses, it is assumed that the mechanism of robot A for unloading the truck fails. Since no more boxes are arriving at the temporary location, robot B becomes increasingly impatient to take over the task of unloading boxes, even though robot A is still attempting to accomplish that task - unaware that its sensor is returning faulty readings. Following a predetermined number of unsuccessful attempts at unloading boxes and receipt of a signal from robot B that it has begun the unloading task, being complicit, robot A abandons that task and turns its attention to the task of loading the conveyors instead.

5.3. Monitoring and Performance Evaluation

One item of central importance to the learning mechanism used is the requirement for robots to monitor and evaluate the performance of team members in executing tasks.

Without this ability, a robot must rely on human-supplied measurements of the performance of robot team members that are unlikely to be responsive to changes occurring over time. In either case, once these performance measurements are obtained, the robot team members have a basis for determining the preferential activation of one behaviour over another, either for the sake of efficiency and long-term adaptation, or to determine when a robot failure has occurred.

The monitor function, implemented within each robot, is responsible for observing and evaluating the performance of any robot team member (including itself) whenever it performs a behaviour. Thus each robot monitors the performance of other robots for tasks that it itself is able to accomplish, recording information on the time of task completion. Robots do not monitor all tasks by all robots – only tasks that they themselves have the ability to perform.

During a live mission, each robot chooses the most suitable action to execute based on sensory feedback and the current system situation. It then broadcasts that action to its team mates to avoid duplication. The monitor function will observe its progress. If there is no progress during the expected time, the robot must either leave this task, or request help from the other robots. At the same time, its team mates also monitor its performance when they receive its broadcast message. If there seems to be no progress in task performance, they start to negotiate with that robot to help it or to take over the task. This procedure is repeated until the task is completed.

5.4. Architecture Control Strategy

The degree to which robot team members can actively pursue knowledge concerning team member abilities depends on the type of mission in which they are engaged. If they are on a training mission, whose sole purpose is to allow robots to become familiar with themselves and with their team mates, then the robots have more freedom to explore their capabilities without concern for possibly not completing the mission. On the other hand, if the robots are on a live mission that may continue for a long time, then the team has to ensure that the mission is completed as efficiently as possible, while continuing to adapt their performance over time as the capabilities of their team mates change. During training missions, the robots will be in an off-line learning mode whereas during live missions, they will be in an on-line learning mode.

5.4.1 Off-line Learning Phase

The best way that the robots can independently learn about their own abilities and those of their team mates is by activating as many of their behaviours as possible during a mission, and monitoring their own progress and the progress of team members during task execution. On any given mission, not all of the available behaviour sets may be appropriate, so it is usually not possible to learn complete information about the capabilities of the robots from just one mission scenario.

However, this learning phase allows the team to obtain as much information as possible to allow each robot to select its next action properly. This action is one of the actions that is currently incomplete, as determined from the sensory feedback, and not being executed by another robot, as determined from the broadcast communication messages. All this information is used as a common knowledge base from which fuzzy rules are generated. These rules are then fine-tuned using a feed-forward neuro-fuzzy technique explained later in this chapter.

5.4.2. On-line Learning Phase

When a robot team is on a mission, it cannot afford to allow members to attempt to accomplish tasks for long periods with little or no demonstrable progress.

The team members have to make a concerted effort to accomplish the mission with whatever knowledge is available about team member abilities, and must not tolerate long episodes of robot actions that do not contribute to the task execution. However, each robot continues to observe robot performance during this phase, and to update the common knowledge base (built during the off-line phase) if required. For example, due to the unreliability of the sensors and actuators and uncertainties in the environment, there might be a small, but acceptable, variation in the time required for robots to implement the same behaviour. Accordingly, the knowledge base has to be updated. Furthermore, if a new situation occurs, a suitable action will be executed, monitored, evaluated and, if appropriate, added to the knowledge base.

5.5. Feed-Forward Neuro-fuzzy Technique

One major disadvantage of fuzzy approaches is that there are no clear guidelines as to how to fine-tune the fuzzy membership functions. However, learning techniques are being developed that can help in this process.

Updating the knowledge-base affects the current rules and hence the system outputs. Therefore, a neuro-fuzzy technique has been proposed to fine-tune these rules and minimise the total error between the desired output and the fuzzy controller output.

Mamdani-model-based fuzzy neural networks (FNNs) represent more transparent neurofuzzy systems compared with Takagi Sugeno-model-based FNNs (Shankir, 2001). The reason is that the rule base of the Mamdani-model is more understandable to human users. Also, it is more general in terms of how its rule base is created, because the latter can be constructed using human experience and numerical data. However, a disadvantage of this model is that it does not allow easy mathematical analysis due to the logical nature of its inference functions, e.g., the logic min/max functions. Also, it does not allow the simple application of BP as one of the most powerful learning algorithms, due to the non-differentiable min/max functions employed.

In this chapter, a Mamdani-model-based FNN with Differentiable Activation functions (DA-FNN) is described. A differentiable alternative to the logic min and logic max functions termed *softmin and softmax* (Shankir, 2001) are presented. These two differentiable functions (*softmin* and *softmax*) are employed instead of the two non-differentiable functions (logic min and logic max) to implement the decision-making mechanism of DA-FNN. Using these differentiable functions allows the effective application of Back propagation (BP) for the parameter learning of DA-FNN.

Figure 5.2(a) presents the structure of the proposed neuro-fuzzy system. The structure is a six-layer feed-forward connectionist representation of a Mamdani-model-based fuzzy logic system (FLS) (Mamdani, 1974). In general, a node in any layer has some finite "fan-in" of connections represented by weight values from other nodes and a "fan-out" of connections to other nodes (see Figure 5.2(b)). Associated with the fan-in of a node is an aggregation function, f, that serves to combine information, activation, or evidence from other nodes. Using the same notations as in (Lin and Lee, 1992), the function that provides the net input to such a node is written as follows:

$$\text{net input} = f^{k}\left(u_1^k, u_2^k, \ldots\ldots\ldots, u_p^k; w_1^k, w_2^k, \ldots\ldots\ldots, w_p^k\right) \tag{5.1}$$

where p is the number of fan-ins of the node, w is the link weight associated with each fan-in, u is an output of a node in the preceding layer associated with the fan-in and the superscript indicates the layer number. A second action of each node is to output an activation value as a function of its net input,

$$\text{output} = o_i^k = a^k\left(f^k\right) \tag{5.2}$$

where a^k denotes the activation function in layer k. The functions of the nodes in each of the six layers of the proposed structure are described next.

Layer 1: Nodes in Layer 1 are input nodes that represent input linguistic variables. Layer one contains N nodes, each of which receives a crisp input vector $\mathbf{x} = (x_1, \ldots, x_N)$. The nodes in this layer simply transmit input values to the next layer directly. That is,

$$f_i^1 = u_i^1 = x_i \quad \text{and} \quad a_i^1 = f_i^1 \tag{5.3}$$

The link weights in Layer 1 are fixed at unity.

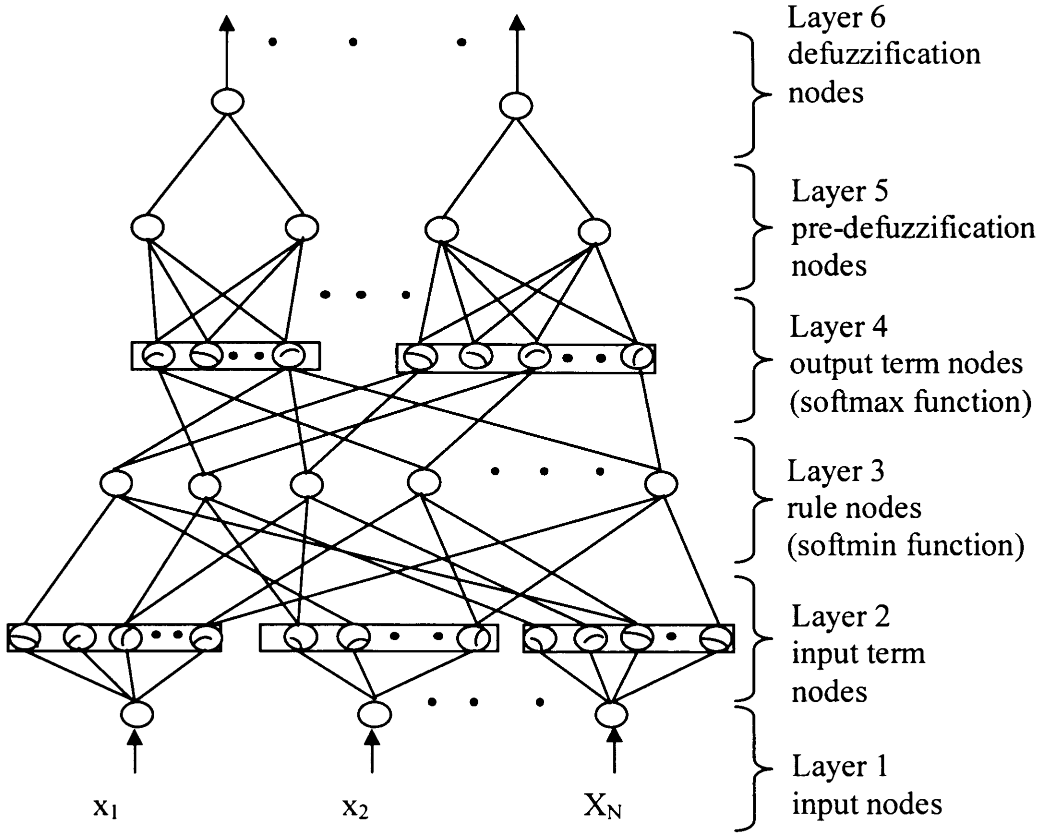


Figure 5.2(a): Structure of the proposed neuro-fuzzy system

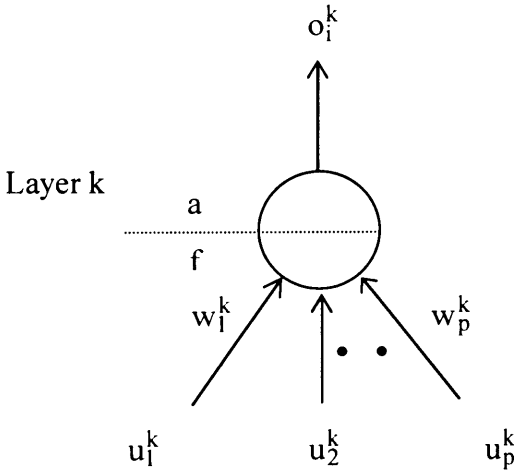


Figure 5.2(b): Basic structure of a node in the network

Layer 2: Nodes in Layer 2 are input term nodes which act as membership functions. An input linguistic variable **x** in a universe of discourse **U** is characterised by $\mathbf{T(x)} = \{T_x^1, T_x^2, ..., T_x^N\}$ and $\mathbf{M(x)} = \{M_x^1, M_x^2, ..., M_x^N\}$, where **T(x)** is the term set of **x**; that is the set of names, e.g., (small, medium, large), of the linguistic values of **x** and **M(x)** is the membership function, e.g., (triangular, trapezoidal, bell-shaped), defined on a universe **U.** The bell-shaped function is chosen because it is differentiable function. The function of each node j in a term set i is to calculate the degree of membership of input x_i with respect to the membership function $M_{x_i}^j, j = 1,2,....,N_i$, associated with the term set $T(x_i)$ according to the following bell-shaped function:

$$f_{ij}^2 = M_{x_i}^j(mij, \sigma ij) = \frac{\left((w_{ij}^2 * a_i^1) - m_{ij}\right)^2}{\sigma_{ij}^2} \quad \text{and} \quad a_i^2 = e^{-f_{ij}^2} \qquad (5.4)$$

where m_{ij} and σ_{ij} are, respectively, the centre (or mean) and the width (or variance) of the bell-shaped function of the j^th^ term of the i^th^ input linguistic variable x_i.

Layer 3: The nodes in Layer 3 are rule nodes, where each node associates one term node from each term set to form a condition part of one fuzzy rule. In this structure, the *softmin* function (Berenji and Khedkar, 1992) and its complement *softmax* function (Shankir, 2001) are used.

$$\mathit{softmin}(a_i, i = 1,2,...,n) = \frac{\sum_{i=1}^{n} a_i e^{-k a_i}}{\sum_{i=1}^{n} e^{-k a_i}} \qquad (5.5)$$

$$\mathit{softmax}\,(a_i, i = 1,2,\ldots,n) = \overline{\mathit{softmin}(\overline{a_i}, i = 1,.2,..,n)} = \left[1 - \frac{\sum_{i=1}^{n} \overline{a_i}\, e^{-k\overline{a_i}}}{\sum_{i=1}^{n} e^{-k\overline{a_i}}}\right] \qquad (5.6)$$

where $a_i = \mu$, $\overline{a_i} = 1 - a_i$ and the parameter k controls the "hardness" of the *softmin* function.

Therefore, the function of the rth rule node using *softmin* can be written as follows:

$$f^3_r = \mathit{softmin}(u^3_1, u^3_2, \ldots\ldots, u^3_N) \quad \text{and} \quad a^3_r = f^3_r \qquad (5.7)$$

where r = 1,..., R, and R is the number of rules or rule nodes in layer three. However, in this layer, there are no link weights to be adjusted because all the link weights are fixed at unity.

Layer 4: The nodes in this layer are output term nodes which act as membership functions to represent the output terms of the respective L linguistic output variables. The nodes in Layer 4 should integrate the fired rules that have the same consequent. The *softmax* function is used to perform the integration. Therefore, the function of each term node j in the output term set i can be written as follows:

$$f^4_{ij} = \quad f^4_{ij} = \mathit{softmax}\left(a^3_m \;, m = 1,\ldots,p\right) \quad \text{and} \quad a^4_{ij} = f^4_{ij} \qquad (5.8)$$

where p is the number of rules sharing the same consequent (the same output term node). Hence, the link weights in Layer 4 are fixed at unity.

Layer 5: The number of nodes in layer 5 is 2L, where L is the number of output variables, i.e. there are two nodes for each output variable. The function of these two nodes is to calculate the denominator and the numerator of a quasi Centre Of Area (COA) defuzzification value for each output variable. The functions of the two nodes of the ith output variable are:

$$f^5_{ni} = \sum_j a^4_{ij} * m_{ij} * \sigma_{ij} \quad \text{and} \quad a^5_{ni} = f^5_{ni} \tag{5.9}$$

$$f^5_{di} = \sum_j a^4_{ij} * \sigma_{ij} \quad \text{and} \quad a^5_{di} = f^5_{di} \tag{5.10}$$

where f^5_{ni} and f^5_{di} are, respectively, the node functions of the numerator and the denominator nodes of the ith output variable.

Layer 6: The nodes in layer 6 are defuzzification nodes. The number of nodes in this layer equals the number of output linguistic variables. The function of the ith node corresponding to the ith output variable can be written as follows:

$$f^6_i = \frac{w^6_{ni} * a^5_{ni}}{w^6_{di} * a^5_{di}} \quad \text{and} \quad a^6_i = f^6_i \quad \text{and} \quad y_i = a^6_i \tag{5.11}$$

where w^6_{ni} and w^6_{di} are layer 6 link weights associated with each output variable node.

In order to build a neuro-fuzzy system based on the above description, three main steps have to be considered. The first step is to specify the input and output variables of the network. The second step is to divide the input-output universes into a suitable number of partitions (fuzzy sets) and to specify a membership function for each partition. A linguistic term has to be assigned to each membership function and the parameters of the membership function (centre and width) have to be specified initially. The third step is to generate fuzzy rules to perform the input-output mapping of the FLS. After the construction of the network, a parameter learning phase has to be conducted. The algorithm for that phase is explained next.

5.5.1. Parameter Learning Algorithm

Following the construction phase, the network then enters the parameter learning phase to adjust its free parameters. The adjustable free parameters were selected to be the centres (m_{ij}s) and widths (σ_{ij}s) of the term nodes in layer 4 as well as the link weights in layers 2 and 6. A supervised learning technique is employed along with the back propagation (BP) learning algorithm (Berenji and Khedkar, 1992) to tune these parameters. The problem can be stated as: 'Given n input patterns $x_i(t)$, i = 1,...., n, m desired output patterns $y_i(t)$, i = 1,....., m, the fuzzy partitions, and the fuzzy rule base, adjust the free parameters of the network optimally'. In the parameter learning phase, the goal is to minimise the following error function:

$$E = \frac{1}{2}\left(y(t) - y_{net}(t)\right)^2 \tag{5.12}$$

where $y(t)$ is the desired output, and $y_{net}(t)$ is the current network output. For each training data set, starting at the input nodes, a forward pass is made to compute the activity levels of all the nodes in the network. Then, starting at the output nodes, a backward pass is followed to compute the rate of change of the error function with respect to the adjustable free parameters for all the hidden nodes. Assuming that w is an adjustable free parameter in a node, then the general learning rule can be written as follows:

$$\Delta \mathrm{w} = \eta \left(-\frac{\partial \mathrm{E}}{\partial \mathrm{w}} \right) \tag{5.13}$$

$$\mathrm{w}(\mathrm{t}+1) = \mathrm{w}(\mathrm{t}) + \Delta \mathrm{w} \tag{5.14}$$

where η is the learning rate. Using the chain rule, the partial derivative can be defined thus:

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w}} = \frac{\partial \mathrm{E}}{\partial(\text{net}-\text{input})}\frac{\partial(\text{net}-\text{input})}{\partial \mathrm{w}} = \frac{\partial \mathrm{E}}{\partial f}\frac{\partial f}{\partial \mathrm{w}} = \frac{\partial \mathrm{E}}{\partial \mathrm{a}}\frac{\partial \mathrm{a}}{\partial f}\frac{\partial f}{\partial \mathrm{w}} \tag{5.15}$$

The calculation of the back-propagated errors as well as the updating of the free parameters can be described starting at the output nodes as follows.

Layer 6: Using Equation (5.15) and Equation (5.11), the adaptive rule for the Layer 6 weights is derived below:

$$\frac{\partial E}{\partial w_{ni}} = \frac{\partial E}{\partial a_i^6} * \frac{\partial a_i^6}{\partial f_i^6} * \frac{\partial f_i^6}{\partial w_{ni}} = -\left(y(t) - y_{net}(t)\right) * \frac{a_{ni}^5}{w_{di} a_{di}^5} \tag{5.16a}$$

$$\mathrm{w}_{\mathrm{ni}}(\mathrm{t}+1) = \mathrm{w}_{\mathrm{ni}}(\mathrm{t}) + \eta_6 \left(-\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{ni}}} \right) \tag{5.16b}$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{di}}} = \frac{\partial \mathrm{E}}{\partial \mathrm{a}_{\mathrm{i}}^6} * \frac{\partial \mathrm{a}_{\mathrm{i}}^6}{\partial f_{\mathrm{i}}^6} * \frac{\partial f_{\mathrm{i}}^6}{\partial \mathrm{w}_{\mathrm{di}}} = -\left(\mathrm{y}(\mathrm{t}) - \mathrm{y}_{\mathrm{net}}(\mathrm{t})\right) * -\frac{\mathrm{w}_{\mathrm{ni}} \mathrm{a}_{\mathrm{ni}}^5}{(\mathrm{w}_{\mathrm{di}})^2 \mathrm{a}_{\mathrm{di}}^5} \tag{5.17a}$$

$$\mathrm{w}_{\mathrm{di}}(\mathrm{t}+1) = \mathrm{w}_{\mathrm{di}}(\mathrm{t}) + \eta_6 \left(-\frac{\partial \mathrm{E}}{\partial \mathrm{w}_{\mathrm{di}}} \right) \tag{5.17b}$$

where η_6 is the learning rate of the link weights in layer 6. The propagated errors from Layer 6 to the numerator and the denominator nodes in layer 5 are derived as follows:

$$\delta_{ni}^{6} = \frac{\partial E}{\partial a_{ni}^{5}} = \frac{\partial E}{\partial a_{i}^{6}} * \frac{\partial a_{i}^{6}}{\partial f_{i}^{6}} * \frac{\partial f_{i}^{6}}{\partial a_{ni}^{5}} = -\left(y(t) - y_{net}(t)\right) * \frac{w_{ni}}{w_{di} a_{di}^{5}} \tag{5.18a}$$

$$\delta_{di}^{6} = \frac{\partial E}{\partial a_{di}^{5}} = \frac{\partial E}{\partial a_{i}^{6}} * \frac{\partial a_{i}^{6}}{\partial f_{i}^{6}} * \frac{\partial f_{i}^{6}}{\partial a_{di}^{5}} = -\left(y(t) - y_{net}(t)\right) * -\frac{w_{ni} a_{ni}^{5}}{w_{di} \left(a_{di}^{5}\right)^{2}} \tag{5.18b}$$

Layer 5: An adjustment is required for the link weights w_{nij}^{5} which represent the centres m_{ij} of the output membership functions. Also, an adjustment is required for the free parameter σ_{ij} that represents the width of the output membership functions. Consequently, using Equation (5.9) and Equation (5.15), the adaptive rule to tune the free parameters in Layer 5 is derived. First, the adaptive rule to tune the centres of the output membership functions can be obtained as follows:

$$\frac{\partial E}{\partial m_{ij}} = \frac{\partial E}{\partial a_{ni}^{5}} * \frac{\partial a_{ni}^{5}}{\partial f_{ni}^{5}} * \frac{\partial f_{ni}^{5}}{\partial m_{ij}} = \delta_{ni}^{6} * \sigma_{ij} * a_{ij}^{4} \tag{5.19a}$$

$$m_{ij}(t+1) = m_{ij}(t) + \eta_5 \left(-\frac{\partial E}{\partial m_{ij}} \right) \tag{5.19b}$$

Second, the adaptive rule to tune the widths of the output membership functions is given by:

$$\frac{\partial E}{\partial \sigma_{ij}} = \frac{\partial E}{\partial a_{ni}^{5}} * \frac{\partial a_{ni}^{5}}{\partial f_{ni}^{5}} * \frac{\partial f_{ni}^{5}}{\partial \sigma_{ij}} = \delta_{ni}^{6} * m_{ij} * a_{ij}^{4} \tag{5.20a}$$

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) + \eta_5\left(-\frac{\partial E}{\partial \sigma_{ij}} \right) \qquad (5.20b)$$

where η_5 is the learning rate of the adjustable parameters σ_{ij} and m_{ij} in layer 5. The propagated error from layer 5 to the j th node in the i th term set in layer 4 is:

$$\delta^5_{ij} = \left(\frac{\partial E}{\partial a^5_{ni}} * \frac{\partial a^5_{ni}}{\partial f^5_{ni}} * \frac{\partial f^5_{ni}}{\partial a^4_{ij}} \right) + \left(\frac{\partial E}{\partial a^5_{di}} * \frac{\partial a^5_{di}}{\partial f^5_{di}} * \frac{\partial f^5_{di}}{\partial a^4_{ij}} \right) = \left(\delta^6_{ni} * m_{ij} * \sigma_{ij}\right) + \left(\delta^6_{di} * \sigma_{ij}\right) (5.21)$$

Layer 4: No adjustment is required for the link weights of layer 4. Only the error signals δ^4_r are required to be calculated and propagated to a rule node r in layer 3. Each one of these error signals is a summation of L propagated error signals δ^4_{ri}, one error signal from a specific node j of each term set i, where i =1,…., L and L is the number of output variables (or term sets). Using Equation (5.15), the error signal δ^4_r is then:

$$\delta^4_r = \sum_i \delta^4_{ri} = \sum_i \left(\delta^5_{ij} * \frac{\partial a^4_{ij}}{\partial f^4_{ij}} * \frac{\partial f^4_{ij}}{\partial a^3_r} \right) \qquad (5.22)$$

From Equations (5.8) and (5.5)

$$\frac{\partial a^4_{ij}}{\partial f^4_{ij}} = 1 \qquad (5.23)$$

and

$$\frac{\partial f_{ij}^{4}}{\partial a_{r}^{3}} = \frac{e^{-k\bar{a}_{r}^{3}}\left[\left(1 - k\bar{a}_{r}^{3}\right)\sum_{m=1}^{p} e^{-k\left(\bar{u}_{ijm}^{4}\right)} + k\sum_{m=1}^{p} \bar{u}_{ijm}^{4} e^{-k\left(\bar{u}_{ijm}^{4}\right)}\right]}{\left(\sum_{m=1}^{p} e^{-k\left(\bar{u}_{ijm}^{4}\right)}\right)^{2}}, \tag{5.24}$$

if the jth term node at the ith term set in layer 4 is connected to the rth rule node in Layer 3. Otherwise,

$$\frac{\partial f_{ij}^{4}}{\partial a_{r}^{3}} = 0 \tag{5.25}$$

where p is the number of rules sharing the same jth output term node, and $\bar{u}_{ijm}^{4}$ is the complement of the mth input to the jth output term node at the ith term set in Layer 4.

Layer 3: As with layer 4, no adjustment is required for link weights in layer 3. Only the error signals δ_{ij}^{3} are required to be calculated and propagated from the rth rule node in layer 3 to the jth term node at the ith term set in layer 2. Each one of these error signals is a summation of p propagated error signals δ_{ijm}^{3} from layer 3, where $m = 1,\ldots,p$, and p is the number of rules which share the same jth term node at the same ith input term set in layer 2. Using Equation (5.15), the error signal δ_{ij}^{3} can be calculated as follows:

$$\delta_{ij}^{3} = \sum_{m} \delta_{ijm}^{3} = \sum_{m}\left(\delta_{m}^{4} * \frac{\partial a_{m}^{3}}{\partial f_{m}^{3}} * \frac{\partial f_{m}^{3}}{\partial a_{ij}^{2}}\right) \tag{5.26}$$

From Equations (5.7) and (5.6)

$$\frac{\partial a_{m}^{3}}{\partial f_{m}^{3}} = 1 \tag{5.27}$$

and

$$\frac{\partial f_{m}^{3}}{\partial a_{ij}^{2}} = \frac{e^{-k a_{ij}^{2}}\left[\left(1 - k a_{ij}^{2}\right)\sum_{i=1}^{N} e^{-k u_{mi}^{3}} + k\sum_{i=1}^{N} u_{mi}^{3} e^{-k u_{mi}^{3}}\right]}{\left(\sum_{i=1}^{N} e^{-k u_{mi}^{3}}\right)^{2}}, \tag{5.28}$$

if the jth term node at the ith input term set in layer 2 is connected to the rule node m in layer 3; otherwise,

$$\frac{\partial f_{m}^{3}}{\partial a_{ij}^{2}} = 0 \tag{5.29}$$

where N is the number of input term sets and u_{mi}^{3} is the ith input to the rule node m in Layer 3.

Layer 2: Using Equation (5.13) and Equation (5.4), the adaptive rule to tune the weights in layer 2 is given by:

$$\frac{\partial E}{\partial w_{ij}^{2}} = \frac{\partial E}{\partial a_{ij}^{2}} * \frac{\partial a_{ij}^{2}}{\partial f_{ij}^{2}} * \frac{\partial f_{ij}^{2}}{\partial w_{ij}^{2}} = \delta_{ij}^{3} * e^{f_{ij}^{2}} * \frac{-2 a_{i}^{1}\left(a_{i}^{1} w_{ij}^{2} - m_{ij}\right)}{\sigma_{ij}^{\;\;2}} \tag{5.30a}$$

$$w_{ij}^{2}(t+1) = w_{ij}^{2}(t) + \eta_{2}\left(-\frac{\partial E}{\partial w_{ij}^{2}}\right) \tag{5.30b}$$

where η_2 is the learning rate of the link weights in layer 2. The propagated error from Layer 2 to the ith input node in layer 1 is derived as:

$$\delta_i^2 = \sum_j \delta_{ij}^2 = \sum_j \left[\frac{\partial E}{\partial a_{ij}^2} * \frac{\partial a_{ij}^2}{\partial f_{ij}^2} * \frac{\partial f_{ij}^2}{\partial a_i^1} \right] = \sum_j \left[\delta_{ij}^3 * f_{ij}^2 * \frac{-2 w_{ij}^2 \left(a_i^1 w_{ij}^2 - m_{ij} \right)}{\sigma_{ij}^{\;2}} \right] \qquad (5.31)$$

Following the construction phase and the learning phase, an optimally tuned FLS is developed to perform a specific mapping function. This mapping function may represent a function of a dynamic system or a control function.

5.5.2. Pattern and Batch Modes of Training

In the practical application of the back propagation algorithm to the multi-layer perceptron, learning results are obtained from many presentations of a prescribed set of training examples to the network. One complete presentation of the entire training set during the learning process is called an epoch. The learning process is maintained on an epoch-by-epoch basis until the synaptic weights and threshold levels of the network stabilise and the average squared error over the entire training set converges to some minimum value. It is good practice to randomise the order of presentation of training examples from one epoch to the next. For a given training set, back-propagation learning may proceed in one of two basic ways, pattern or batch mode.

In the pattern mode of back propagation learning, weight updating is performed after the presentation of each training example. Each example in the epoch is presented to the network, and a sequence of forward and backward computations is performed resulting in certain adjustments to the synaptic weights and threshold levels of the network.

In batch mode, weight updating is performed after the presentation of all training examples that constitute an epoch.

The pattern mode was adopted in this work because it is simpler to implement and it can still give trained networks producing outputs very close to the desired outputs.

5.6. Experiments and Discussion

This architecture has been applied to a simulated team of mobile robots performing a proof-of-concept co-operative box pushing task. That task was chosen because it enabled the features of the proposed architecture to be demonstrated. The objective of the co-operative box-pushing task is to find a box, randomly placed in the environment of the robots, and push it across the 'room'. The box is heavy and long and one robot alone cannot (continuously) push the box to move it across the room. It is necessary to synchronise the pushing of the box by robots at the two ends, so that the task is defined in terms of two recurring subtasks. These subtasks are: push the left end a little and push the right end a little – neither of which can be activated (except for the first time) unless the opposite side has just been pushed.

The Webots simulation shell (Michel, 1998) was used to implement this task. This is a three-dimensional simulation tool with a good graphical interface to display the simulation results. Using Webots, robots equipped with actuators and sensors for detecting the box and obstacles and a set of behaviour modules that map sensor inputs to actuator outputs at an environment containing a long box and obstacles have been modelled.

The two experiments performed in (Parker, 2001) were repeated in this investigation for comparison. In the first experiment, two robots co-operate to find a box and push it across the room with no obstacles in the environment. The second experiment differs by adding an obstacle, located at one of the corners, that obstructs one of the robots to study how the other robot dynamically reselects its actions in response to changes in the mission situation.

At the start, the robots are situated randomly in the environment (figure 5.3(a)) and they begin to locate the box. After both of them have reached the box (figure 5.3(b)), it is assumed that the robot at the left starts to push first (figure 5.3(c)). The box then needs to be pushed from the right side, so the robot on the right starts to push and broadcasts that action to the robot on the left. During the expected time for that action, the robot on the left monitors the performance of its team mate. The procedure then repeats itself. Finally, the robots complete the task (figure 5.3(d)). In the second experiment, one of the robots is stuck behind an obstacle added to the environment while the other reaches the box (figure 5.3(e)). Because there is no contribution from the other robot (sensors readings are unchanged and no messages are received), the robot that reached the box starts to push it at one end (figure 5.3(f)). It then moves to the other end to push (figure 5.3(g)). It continues its back and forth movements (figure 5.3(h)), executing the tasks of pushing the left end of the box and pushing the right end of the box for as long as it fails to hear that another robot is performing the pushing task at the other end of the box.

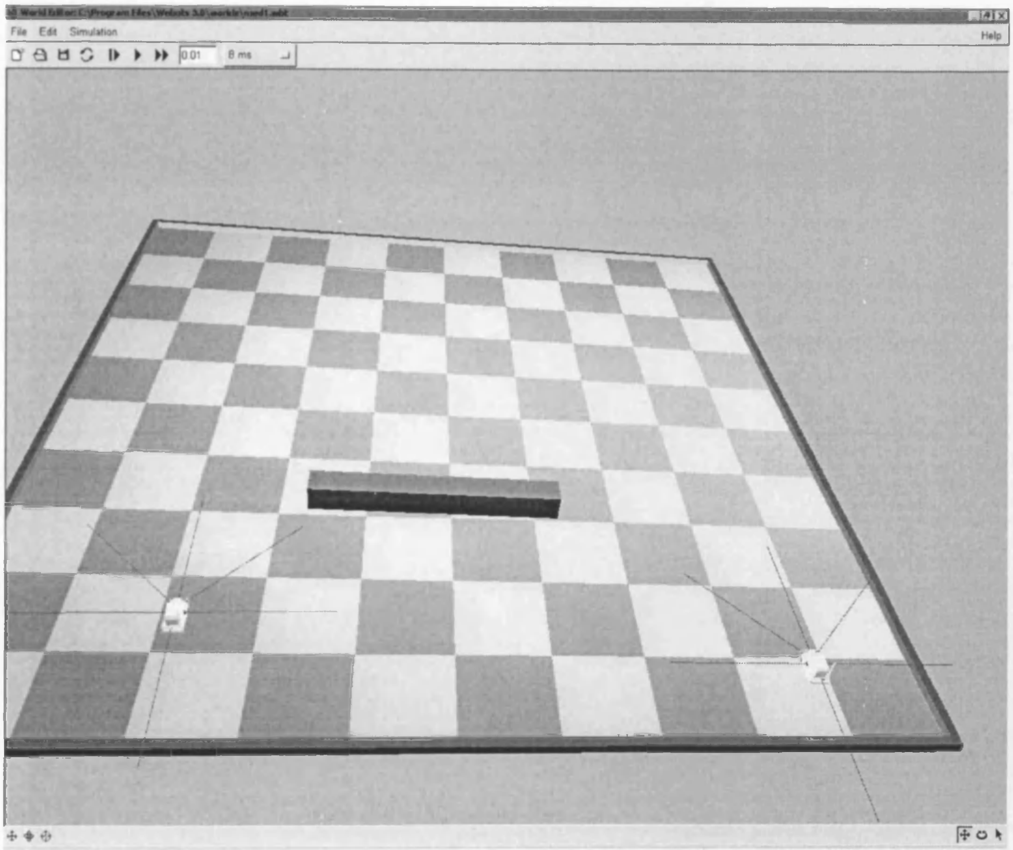


Figure 5.3(a): Initial environment (two robots randomly situated in an environment with a long box)

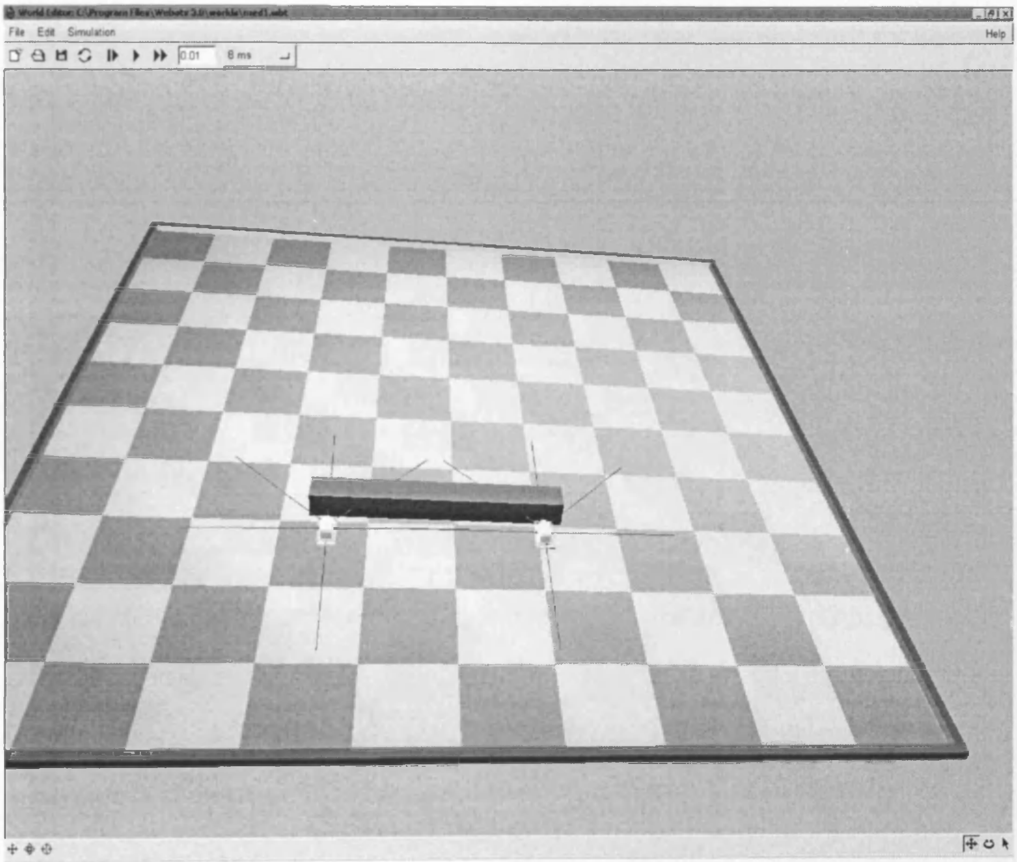


Figure 5.3(b): The robots reach the box

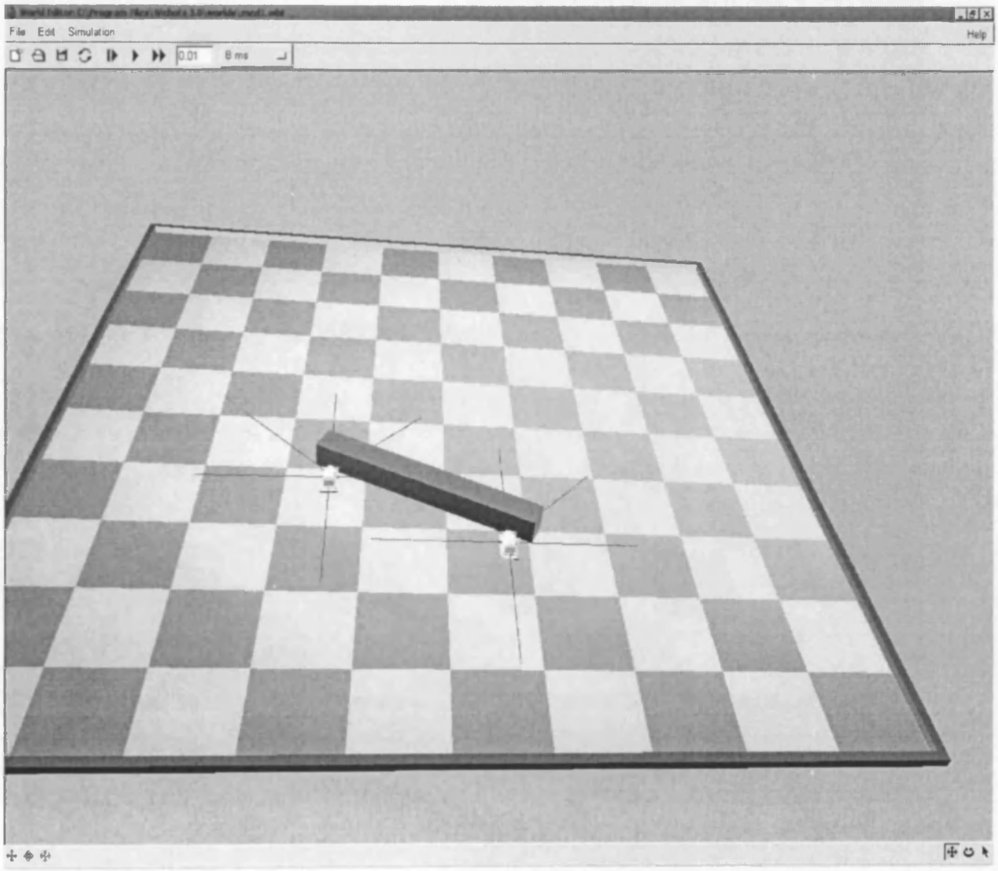


Figure 5.3(c): The left robot starts to push the box

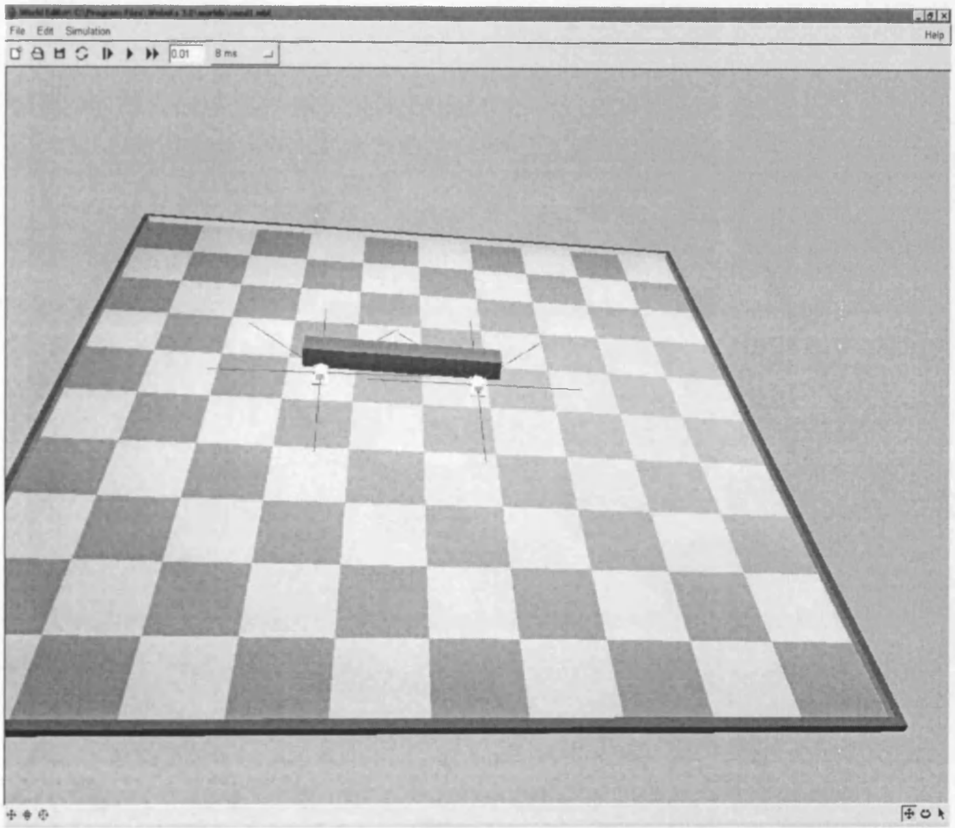


Figure 5.3(d): Both robots successfully complete their mission

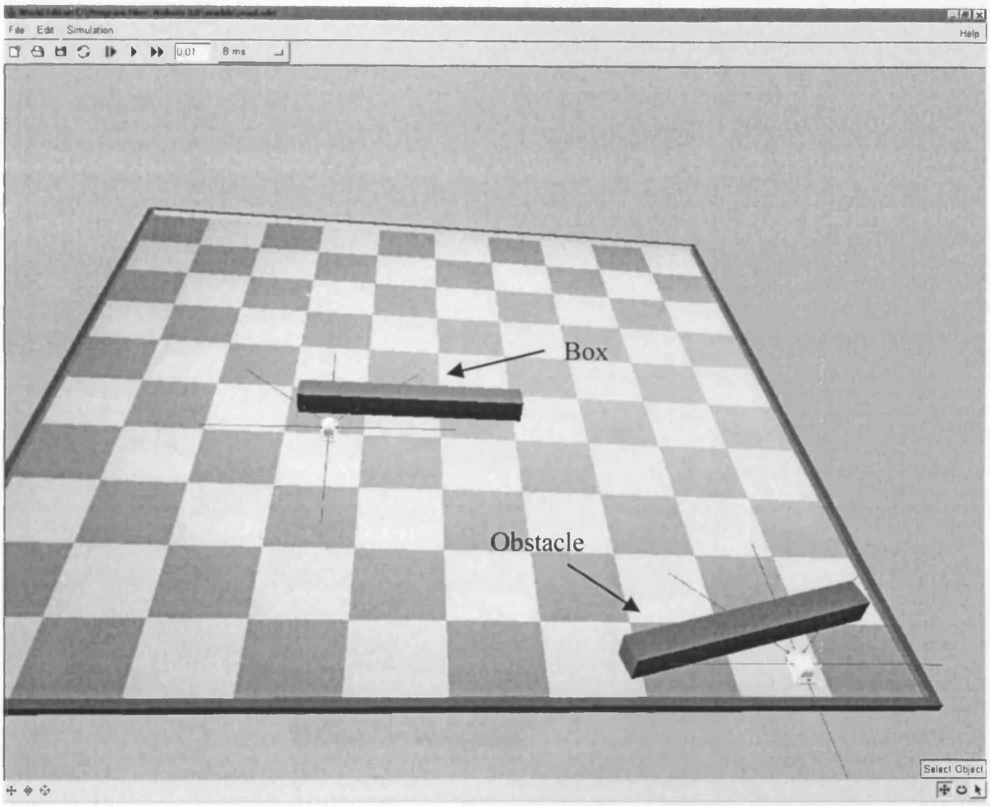


Figure 5.3(e): Only one robot reached the box because of the obstacle

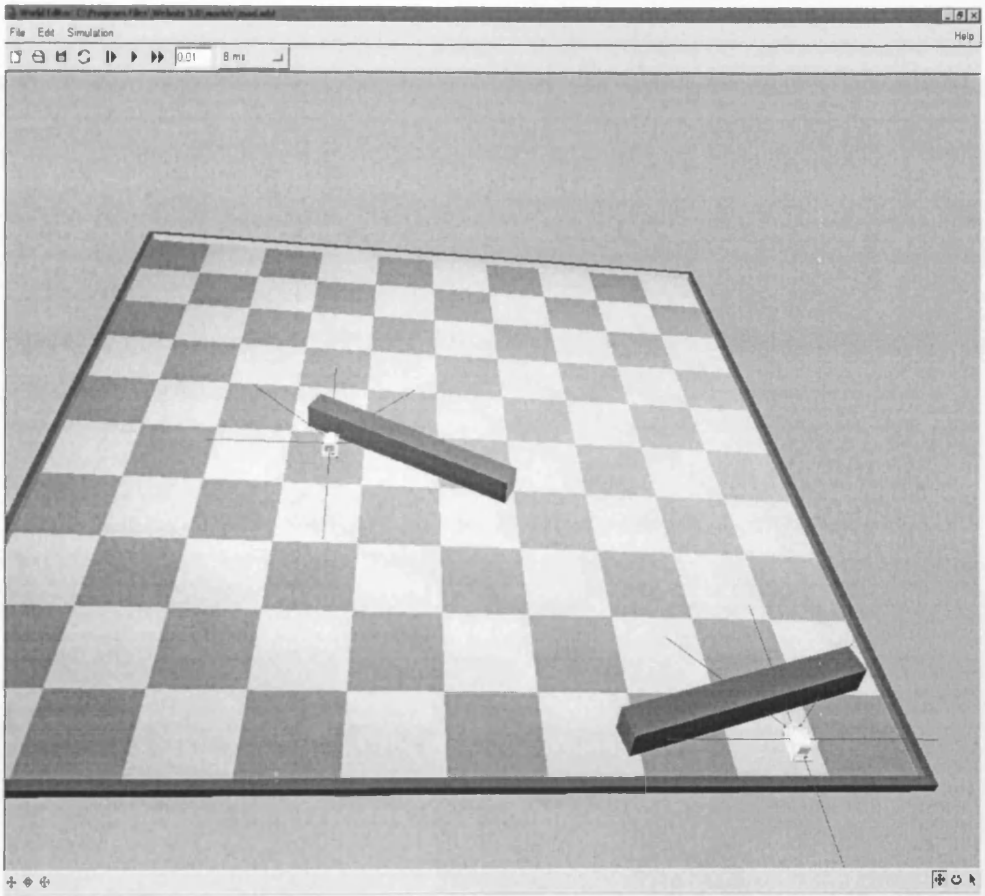


Figure 5.3(f): The robot starts to push from the left side

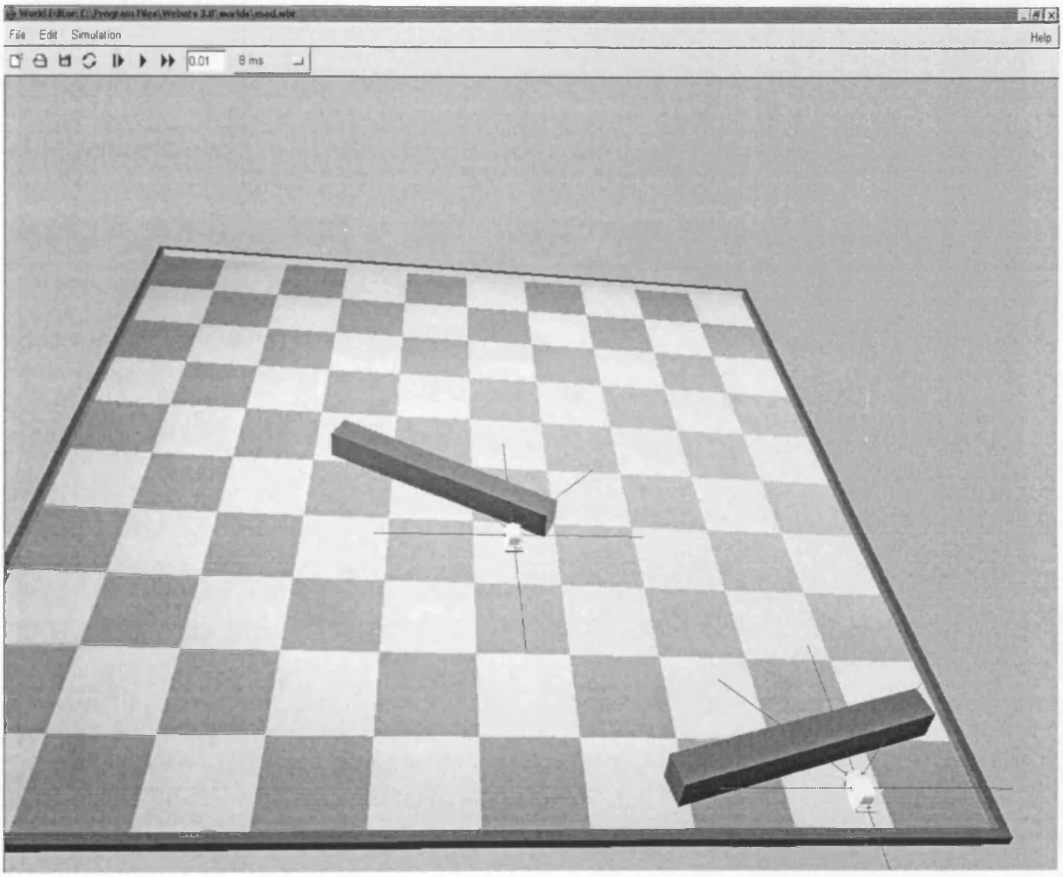


Figure 5.3(g): The robot moves to the other end to push

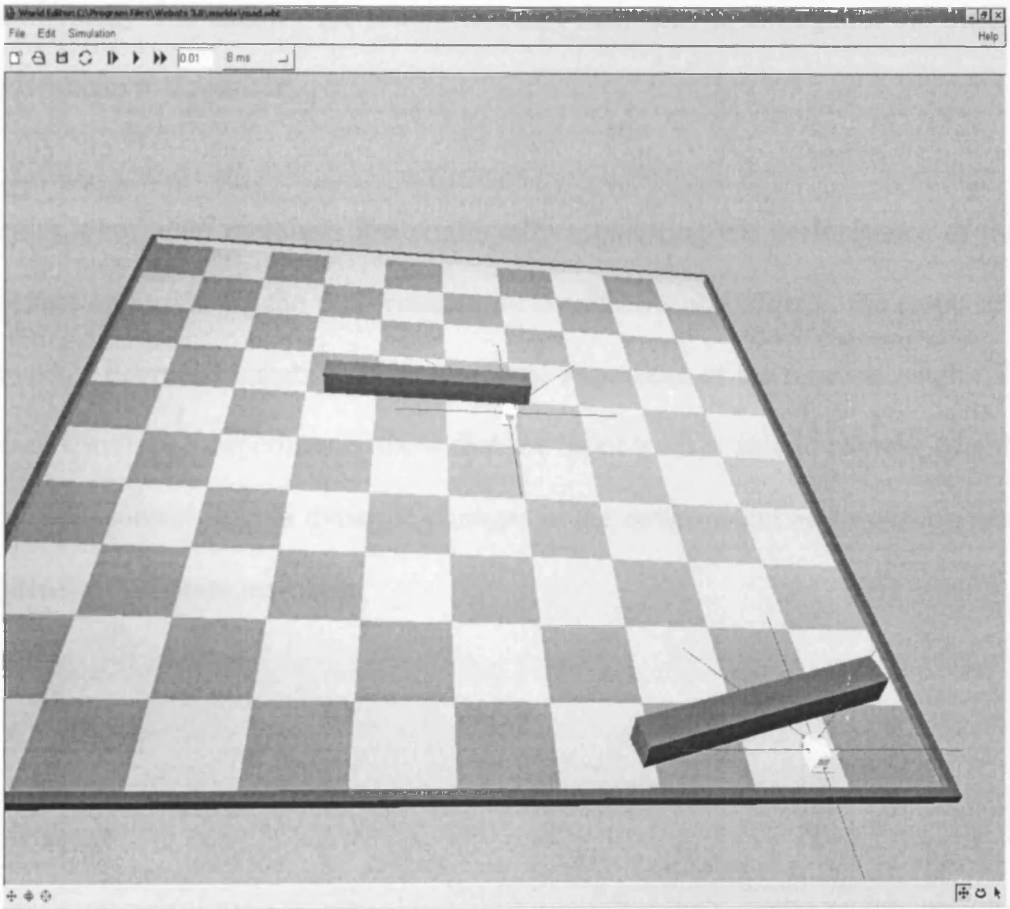


Figure 5.3(h): The robot continues the back and forth movement at the left-right alternating pushing

5.7. Summary

The knowledge-based architecture presented in this chapter was used to create a robot team that can perform missions over long periods, even when the environment or the robot team itself changes. An important component of these robotic systems is a control strategy that enables the robots to adapt their actions throughout the mission without human intervention.

Since the robot team members are continually monitoring the performance of their team mates and updating the performance measurements accordingly, the response to improved or degraded capabilities is automatic regardless of the mission length. The results of simulation experiments show that the robot team is able to achieve adaptive co-operative control despite dynamic changes in the environment and variation in the capabilities of the team members.

Chapter 6

Mobile Robot Hardware and Experiments

6.1 Preliminaries

The original objective was to develop a research platform, including both software and hardware components, for the evaluation of control algorithms for multi robot systems. The developed software should enable users to develop their own control algorithms. These algorithms could then be tested both in simulation environments and also on real mobile robots. The developed hardware should enable the creation of multiple autonomous mobile robots with the following requirements:

- Extendibility – the robot hardware must allow future expansion.
- Compatibility – the robot design must be based on commercially available standard components.

The remainder of the chapter is organised as follows. The construction of a team of mobile robots is covered in Section 2. Experiments are discussed in section 3. The experiments were devised to demonstrate practical implementations of some of the ideas proposed in previous chapters.

6.2 Construction of a team of mobile robots

Small radio-controlled toy cars and a small radio-controlled toy tank were adapted to provide the mechanical structures for the mobile robots and moving target, respectively (see Figure 6.1). The toy cars and the tank were driven by small dc motors and were capable of a maximum speed of 3 m/s and 2 m/s respectively. Both types of toys are inexpensive (costing under £10 each) and readily available. This contributes to achieving the compatibility and cost requirements.

The control system for the robots and target was purpose designed for this application as the existing radio-operated controllers in the toy cars and toy tank were not suitable. The control system for the robots comprises a main board and sensor board. The heart of the main board is a Priority Interrupt Controller (PIC) of type 18F252, used to control the robot/target actuators and process the sensor signals. A circuit diagram for the sensor board is shown in Figure 6.2. The sensor board contains circuitry to enable the robots to detect the target and other obstacles. An emitter and four sensors were designed to be incorporated in the sensor board (Figure 6.3). However, due to the size of the robots, the dimensions of the control system (and thus the main and sensor board) are very limited. This does not permit the fitting of many sensors. Hence, the rear facing sensor which was not considered essential was omitted. Also, there was no room to fit a system for the robots to detect or communicate with one another and that was why only the dynamic target tracking task could be implemented (Figure 6.3). However, as the emitter of a robot is front facing, it cannot differentiate between another robot and an obstacle in front of it.

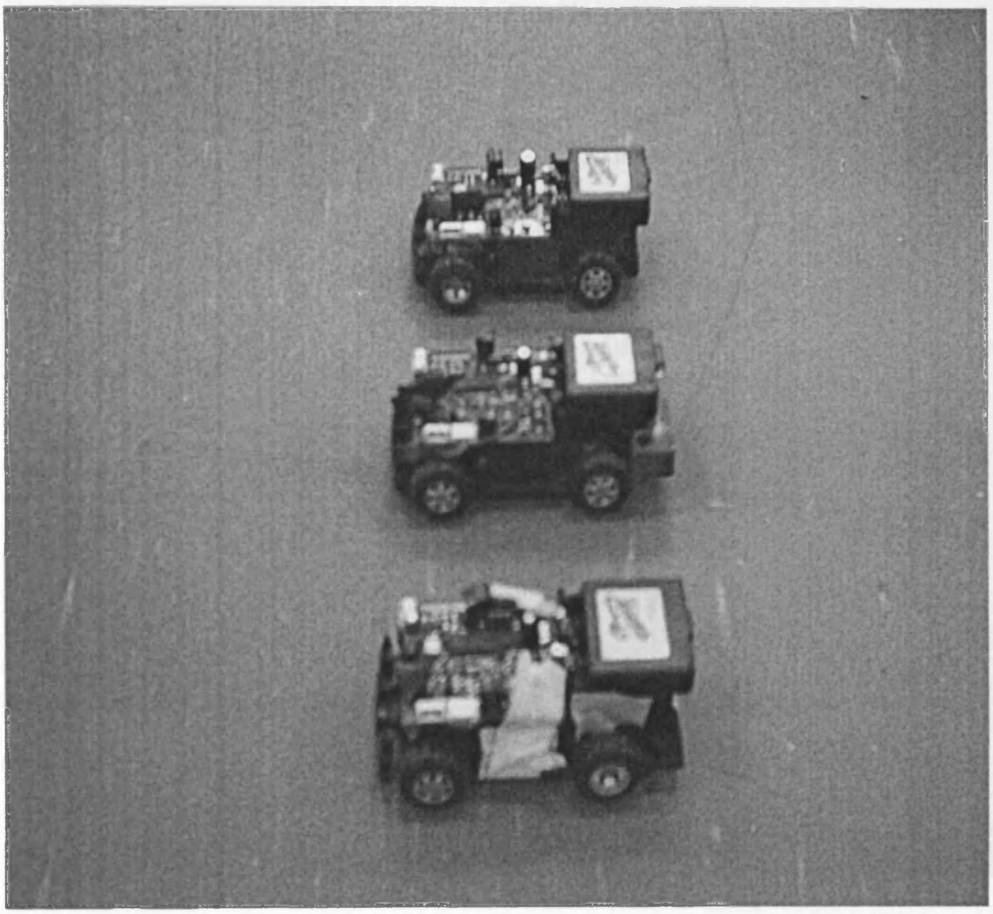


Figure 6.1 (a): The mobile robots

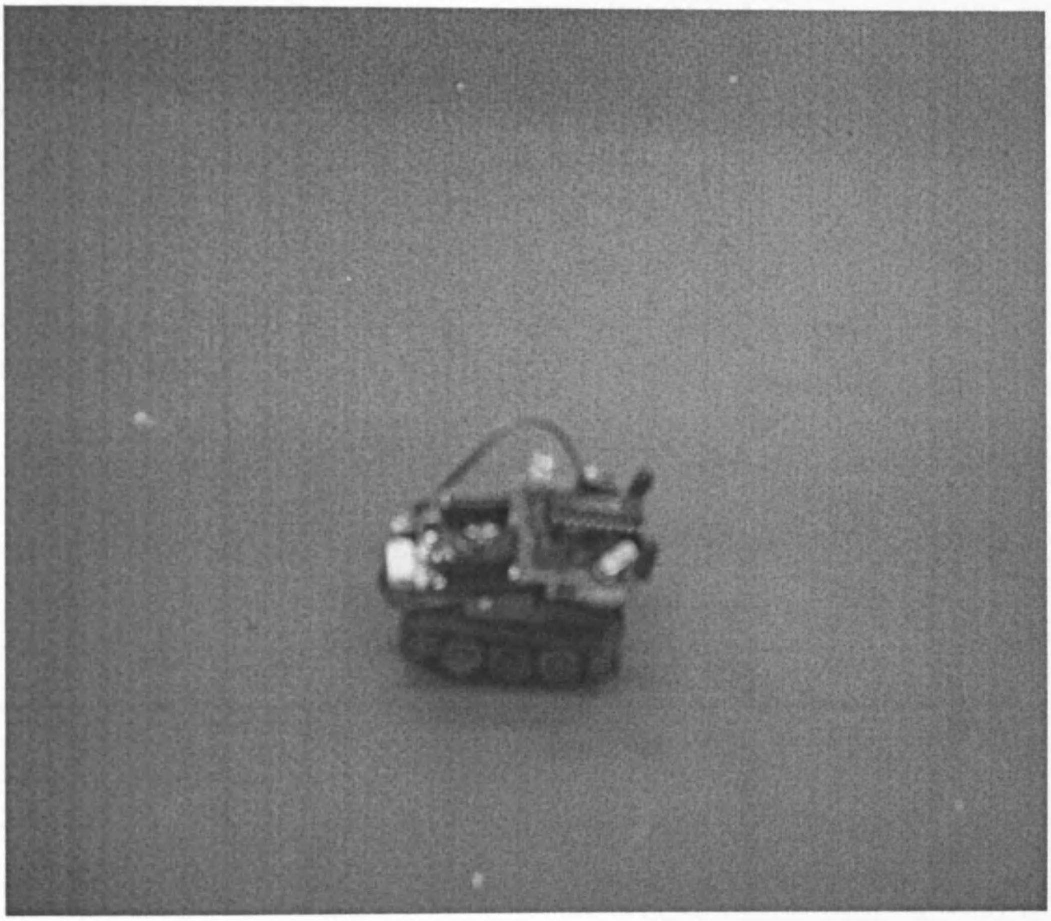


Figure 6.1 (b): The target

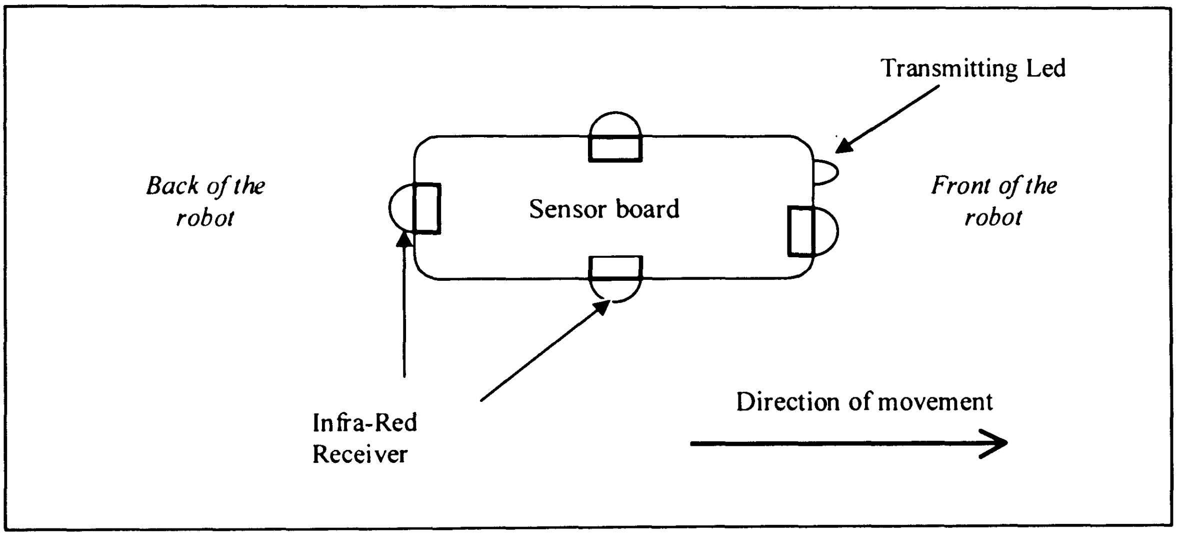


Figure 6.2: Robot sensors board

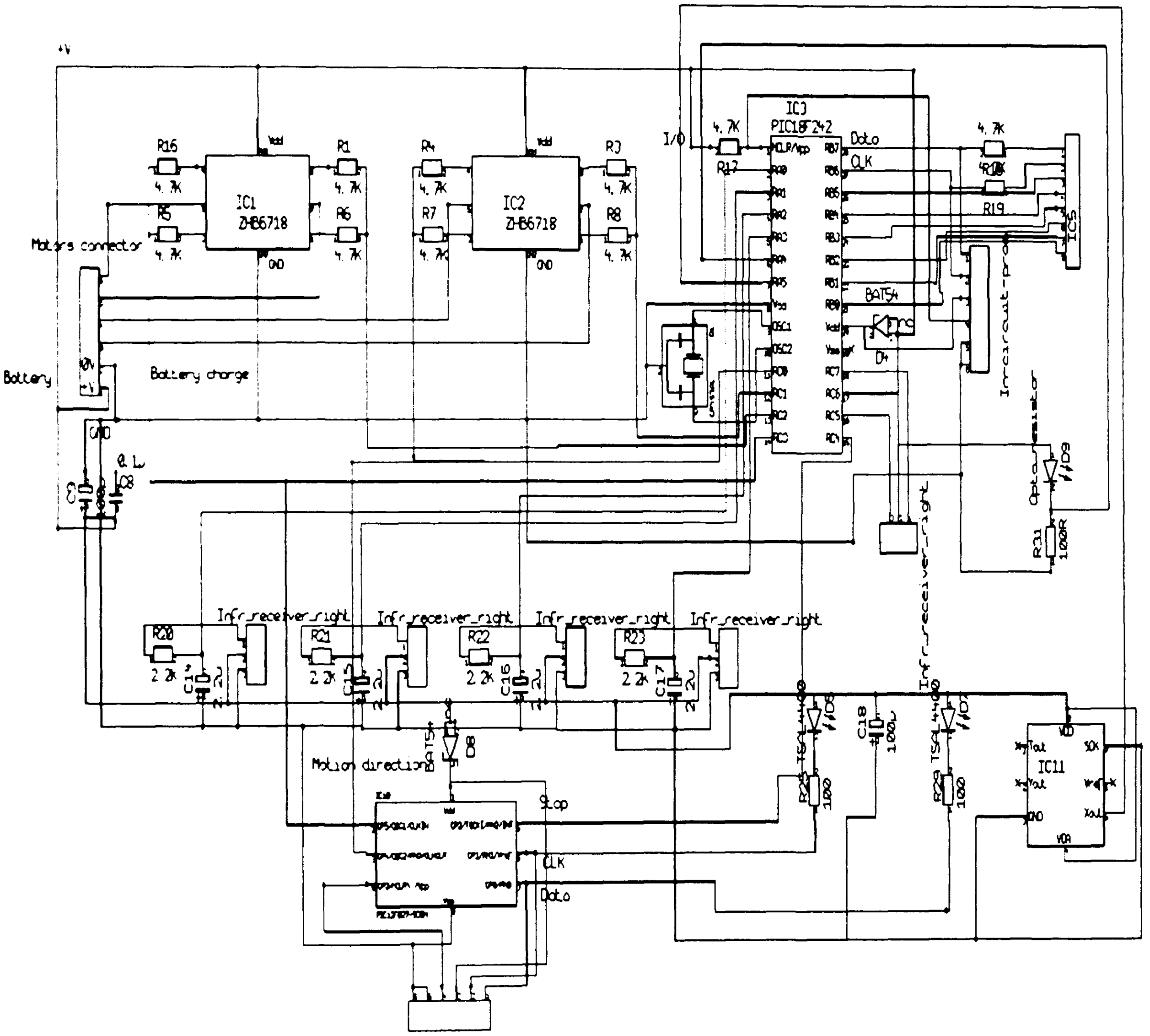
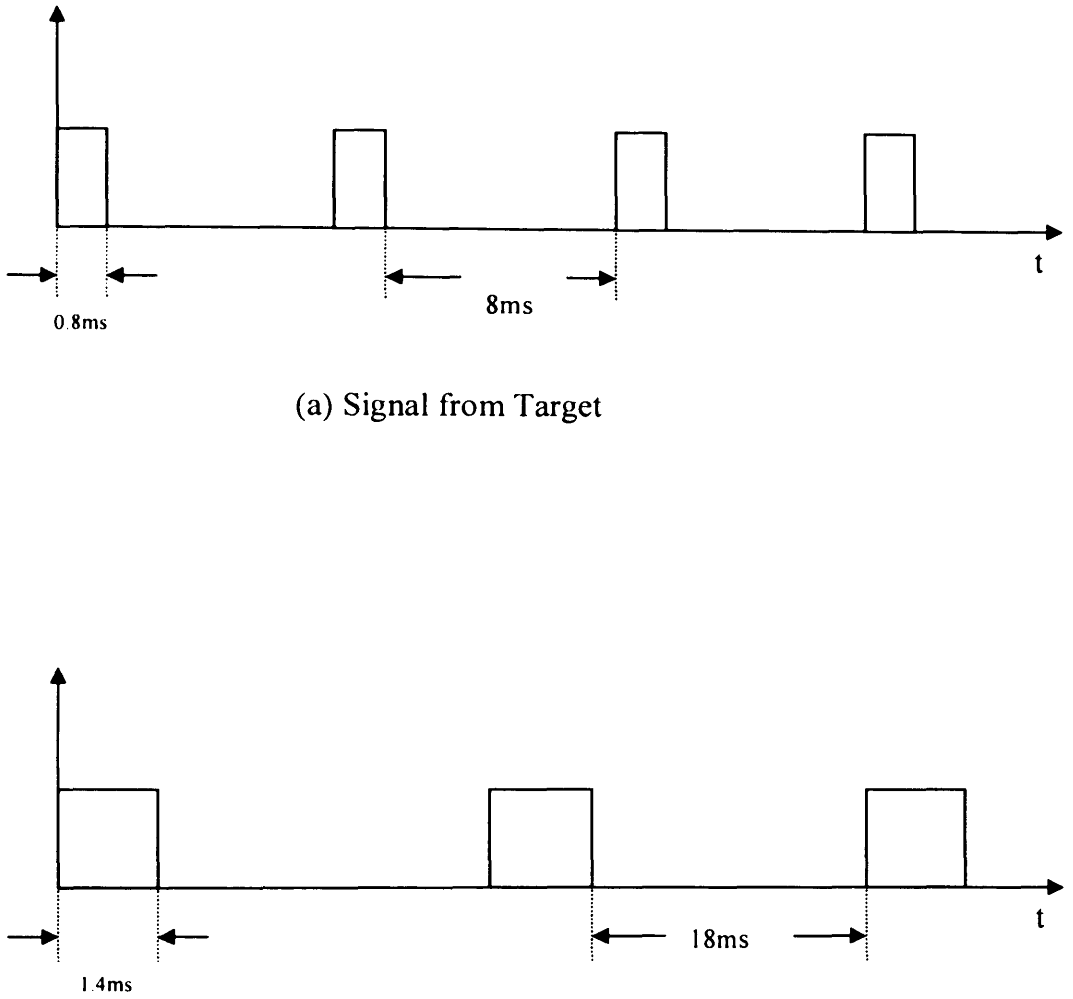


Figure 6.3: Schematic diagram for the sensors board

However, a robot can detect another robot approaching it from either side of it because one of the side sensors of the first robot will be activated by the signal emitted by the approaching robot. A robot can also distinguish between a target and another robot located on either side of it because of the different signals they emit. This help the robots to coordinate their movements based on the direction of motion of their team mates and that of the target.

Two factors restricted the choice of the infrared receivers and transmitting LEDs: the size of the robots and the current consumption. Both the transmitting LED and receivers operate at the same frequency signal chosen at 38 kHz. For obstacle detection, the robot uses the transmitter to send a signal which would be reflected and detected by one of the receivers if there is an obstacle near by. Obstacles can be detected as far as 30 cm and the angle of the detection is approximately 45 degrees. The target could be sensed from a distance of 1m.

To differentiate between target and obstacle, two different codes are used in the emitted signal one for target detection and the other for obstacle detection. Figure 6.4(a) shows the signal emitted by the target. A square wave at 38 kHz is emitted for a duration of 0.8 ms and then stopped for 8 ms. Figure 6.4(b) shows the signal emitted by the robot for obstacle detection. This is also a square wave signal at 38 kHz, but emitted for a duration of 1.4 ms and then stopped for 18 ms. These values were determined experimentally. To find the range for obstacle and target detection,



(a) Signal from Target

(b) Signal from robot to detect obstacles

Figure 6.4: The emitted signal (a) for the target (b) for the obstacles

experiments were conducted under different conditions and the output of the PIC controller measured. The detection ranges were used by the control programs to enable the robots to move safely and perform their tasks successfully.

6.3 Experiments and Discussion

Experiments were run with two or three robots, different obstacles and one target. Each experiment was performed six times and the results are averaged. The first set of experiments analysed how varying the number of robots affected the time required for tracking the target. This experiment took place in a limited arena containing one target and no obstacle. The average tracking time versus the number of robots was noted. The second set of experiments differed from the first set only by the addition of obstacles in the arena. Again, performance was analysed relative to the number of robots performing the task.

Figures 6.5(a) shows one of the environments before the experiment started. This contained two robots in one corner, and one target in the opposite corner. Figure 6.5(b) depicts an intermediate stage of target tracking. Figure 6.5(c) shows the final stage when the robots have cooperated and captured the target. Figures 6.6(a), 6.6(b) and 6.6(c) demonstrate the same scenario for three robots. Figures 6.7 (a), 6.7(b) and 6.7(c) show the same scenario with the addition of obstacles.

Even though the performance of the robots in some trials was not as expected (the robots spent too long to track and capture the target), this might be due to the narrow beam angle of the emitters, so that it was difficult for the robots to find the target.

However, the robots managed in some trials to track and capture the target even in cluttered environments. It was found that the time required to track and capture the target using three robots was approximately 2 minutes. With only two robots, the required time was about 4 minutes. In the case where obstacles were included, the time was 7 minutes. The tracking time can be reduced by increasing the number of robots or the speed of the robots. However the speed of the robots was kept slow (0.5 m/s) to give them time to respond to the signal emitted from the infrared emitters.

Due to the limited capabilities of the robots as previously mentioned in addition to the robots are light and does not have enough force, the box pushing task could not be implemented in real.

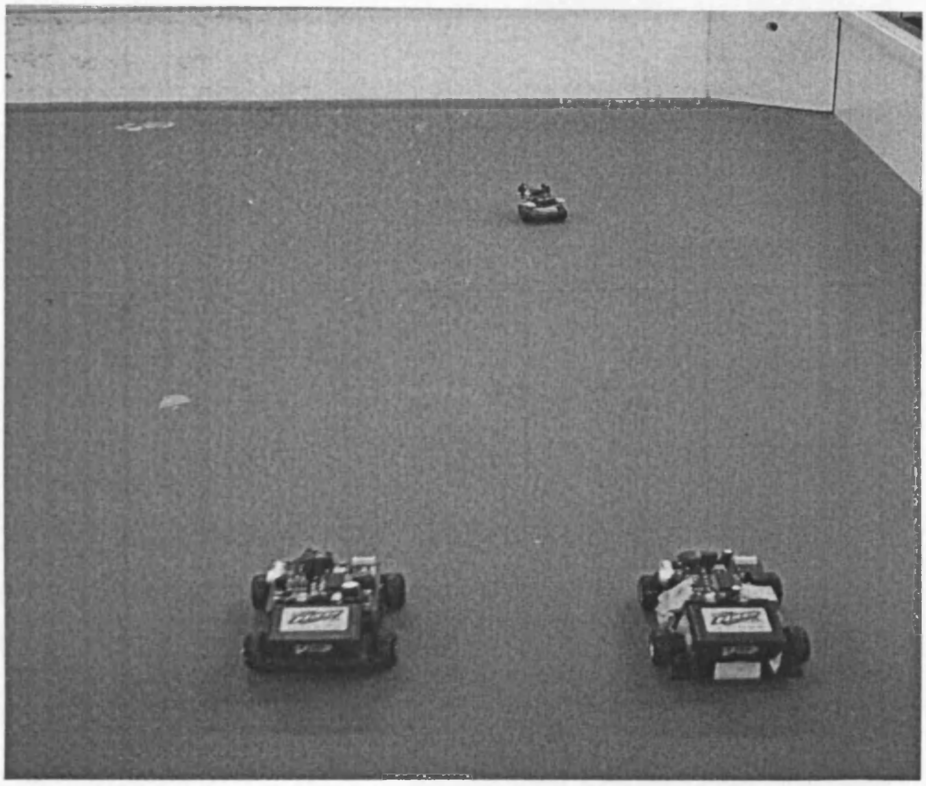


Figure 6.5(a): Initial environment (one target and two robots)

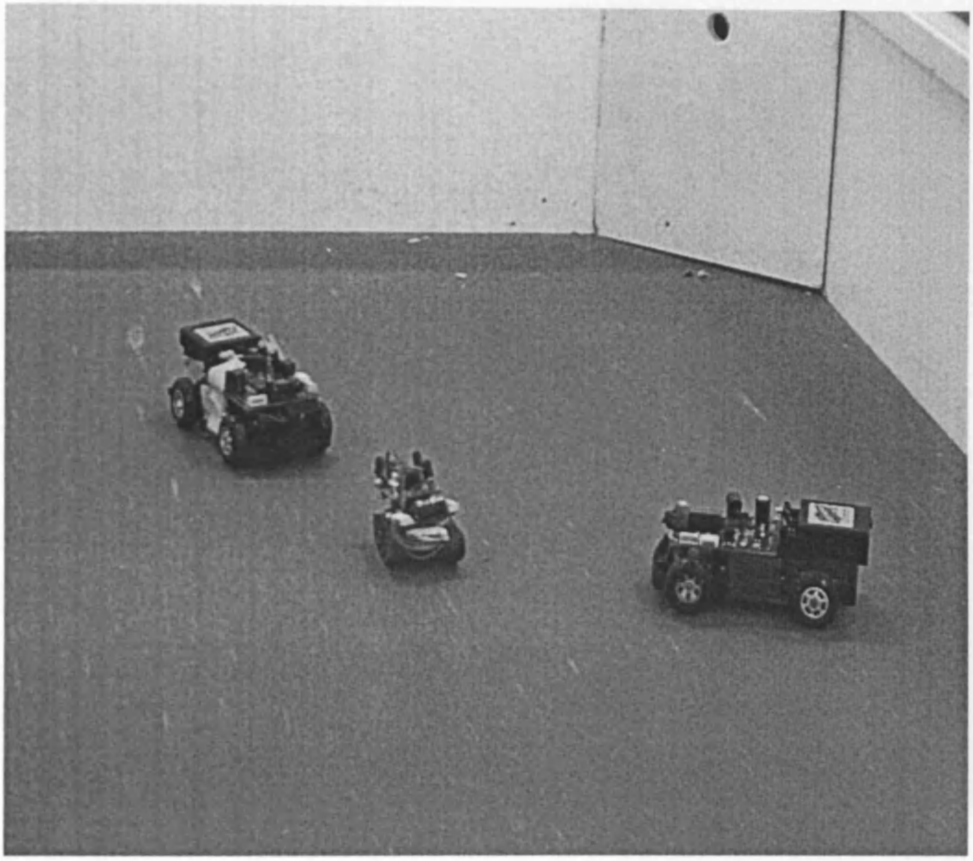


Figure 6.5(b): Intermediate stage with robots tracking the target

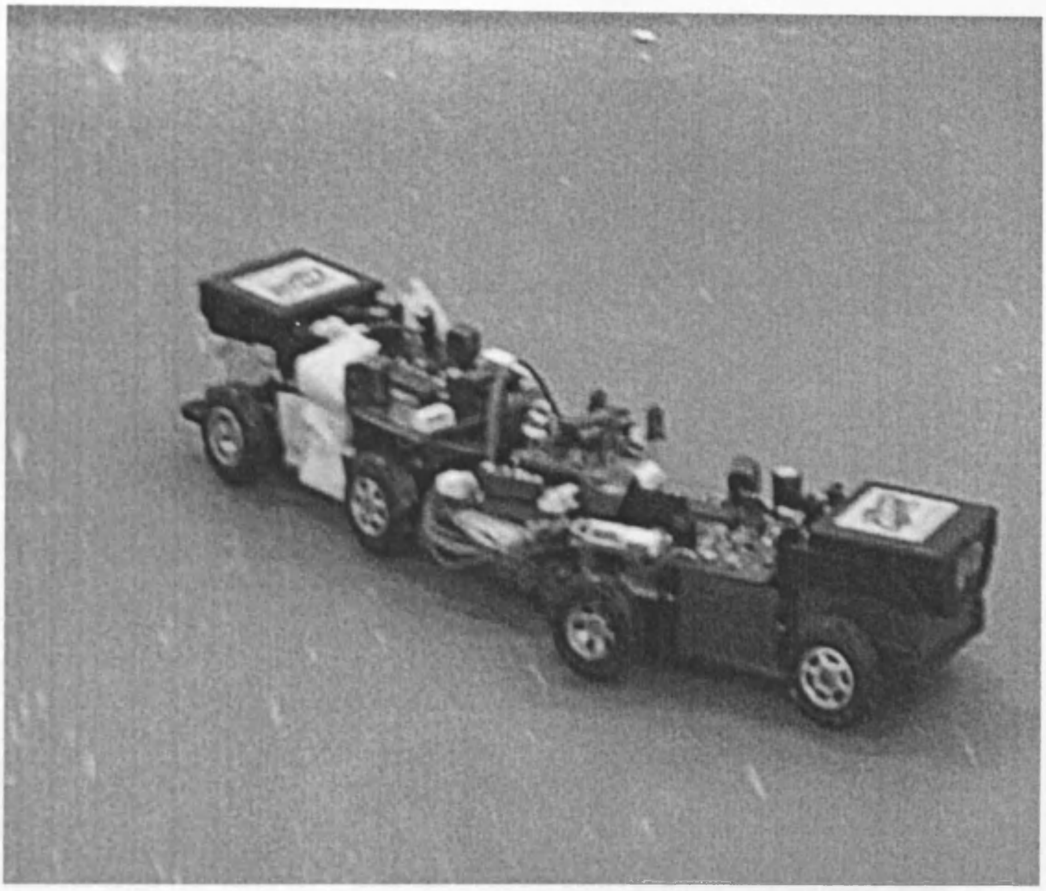


Figure 6.5(c): Final stage with the robots having captured the target

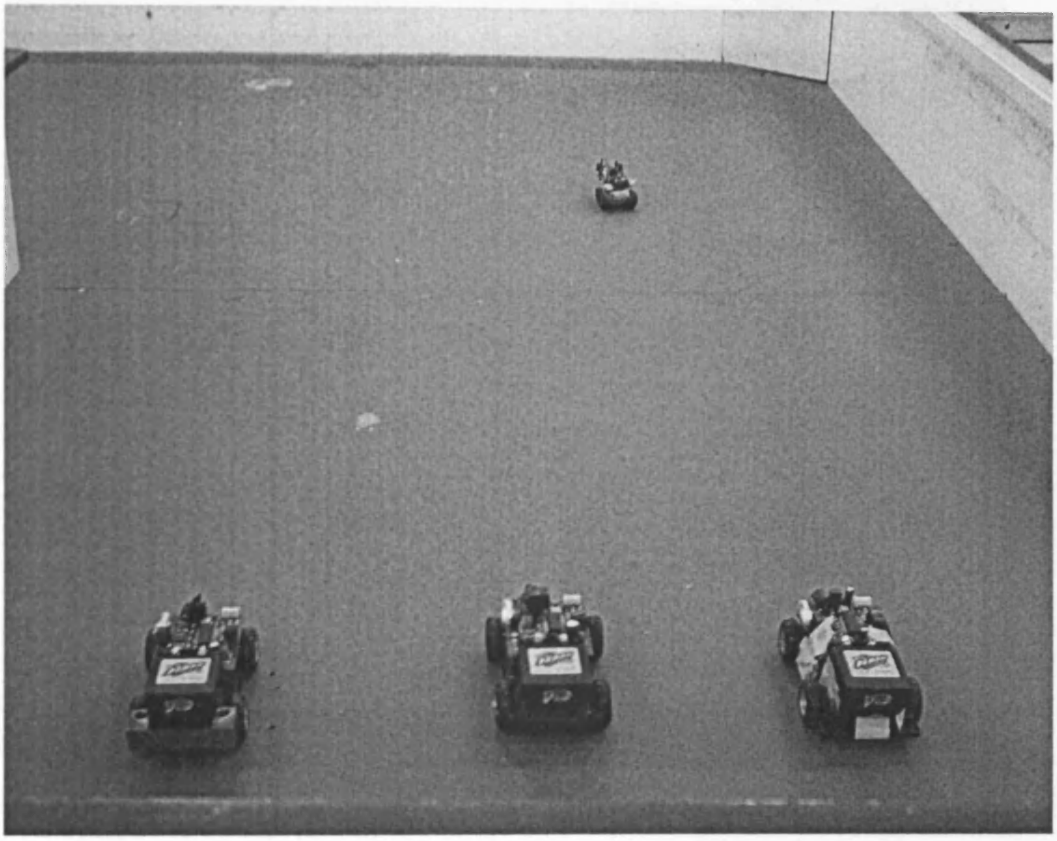


Figure 6.6(a): Initial environment (one target and three robots)

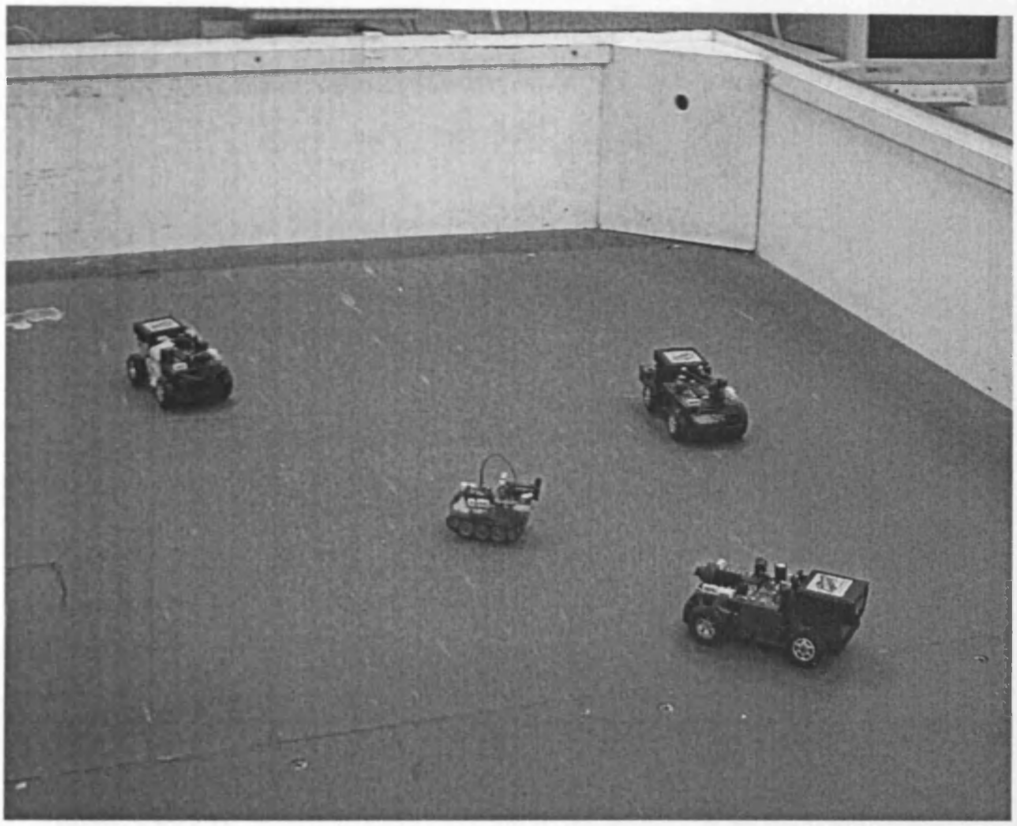


Figure 6.6(b): Intermediate stage with robots tracking the target

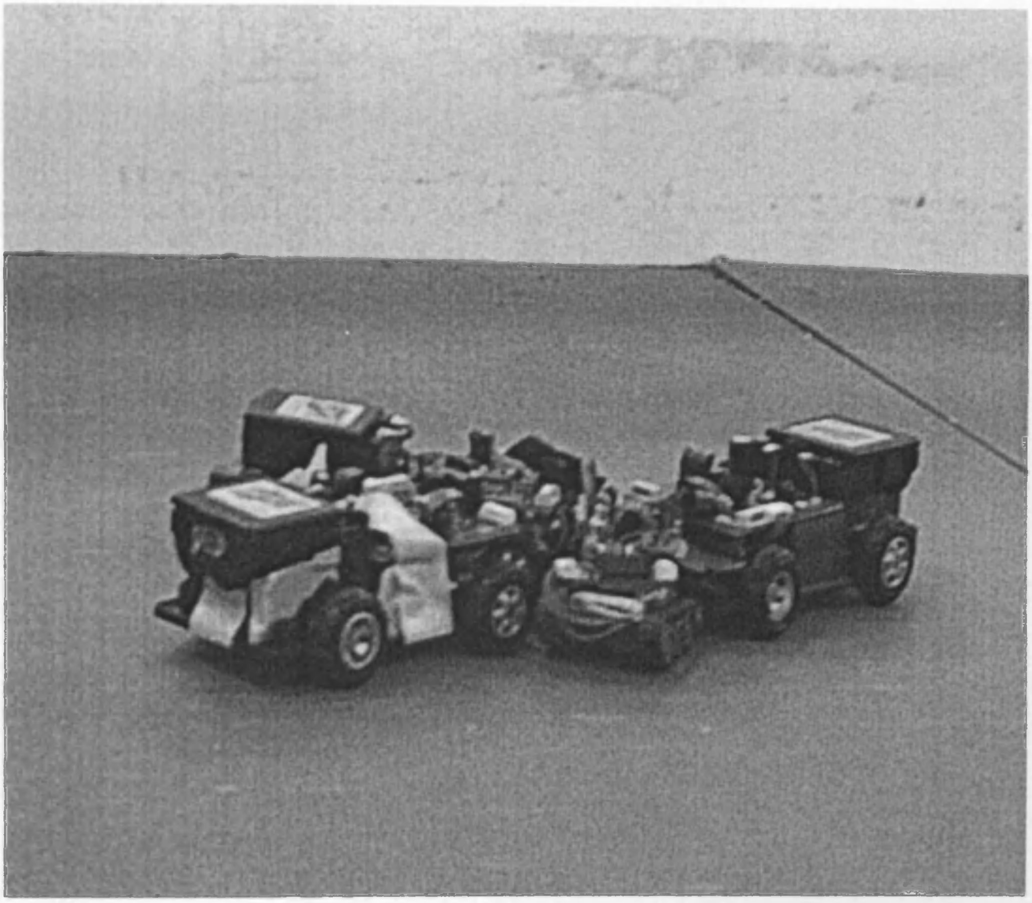


Figure 6.6(c): Final stage with the robots having captured the target

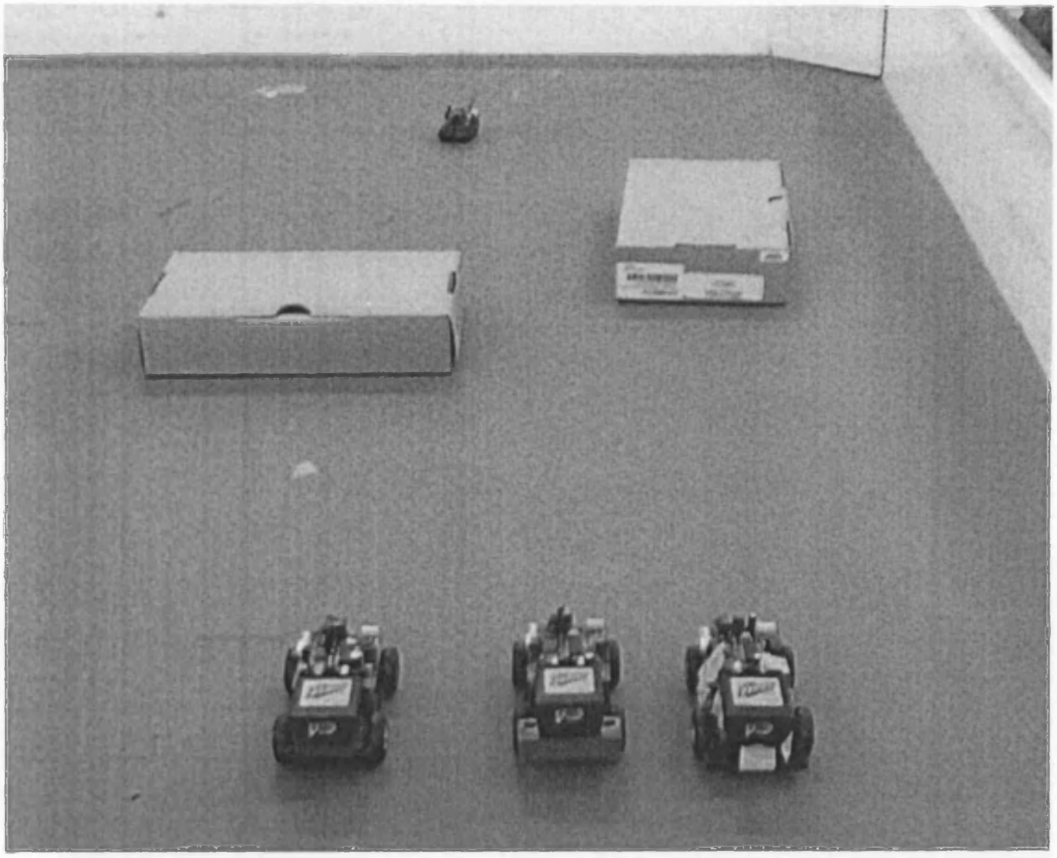


Figure 6.7(a): Initial environment (one target, different obstacles, and three robots)

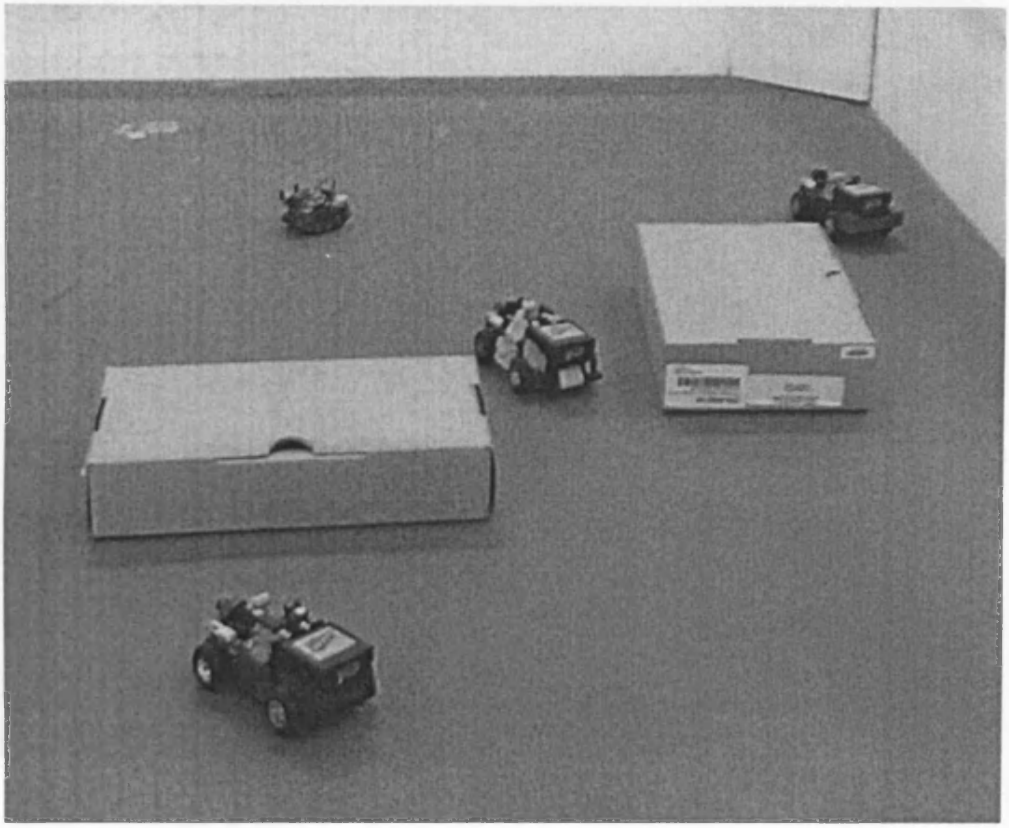


Figure 6.7(b): Intermediate stage with robots tracking the target

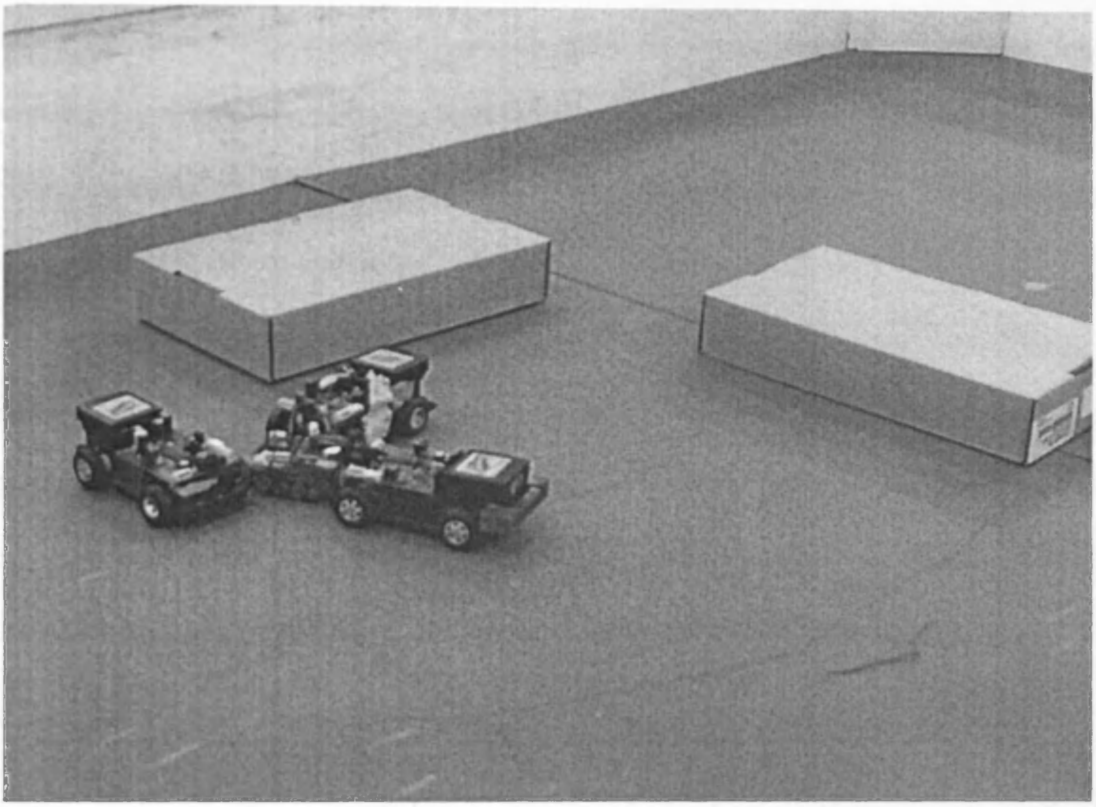


Figure 6.7(c): Final stage with the robots having captured the target

6.3 Summary

This chapter has focused on the design and construction of a team of mobile robots for tracking and capturing a dynamic target.

Electronic control and sensing circuits for driving three small radio-controlled cars and a toy tank were adapted to act as three mobile robots and a moving target, respectively. Electronic circuitry was fitted to these vehicles to enable them to detect obstacles, signal their presence (in the case of the target and robots approaching the sides of another robot) and detect the target (in the case of the robots).

Experiments showed that the robots successfully managed to track and capture the target. However, due to the limited number of emitters installed, the performance of the robots in some trials was not satisfactory. As expected, the more robots that tracked a dynamic target, the shorter was the time required to capture it.

Chapter 7

Conclusions and Further Work

This chapter summarises the main contributions made and the conclusions reached in this work and proposes topics for further investigation.

7.1. Contributions

1- A modification to the subsumption robot control architecture has been proposed to enable the control of multiple robots using the collective behaviour resulting from individual sensor-based behaviours.

2- A fuzzy logic technique has been developed to enable the resolution of conflicts between contradictory behaviours by proposing an action that represents the consensus among the behaviours and that best satisfies the decision objectives that they encode.

3- A knowledge-based software architecture has been implemented for cooperating mobile robots to update their behaviours based on knowledge acquired on-line.

4- A group of low cost miniature mobile robots has been developed to enable some of the proposed ideas to be demonstrated.

7.2. Conclusions

1. This work has proposed an approach to controlling multiple robots that involves the use of collective behaviour resulting from several sensor-based behaviours. The influence of environmental factors and the number of robots on the performance of the group in a dynamic target-tracking task has been analysed. As would be expected, increasing the number of robots reduced the time required to track the target. However, robot collision and interference tended to degrade the performance. Continually adding more robots therefore did not produce a proportional increase in performance.

2. The use of fuzzy logic enabled the resolution of conflicts between contradictory behaviours by selecting an action that represents the consensus among the behaviours and that best satisfies the decision objectives encoded in them.

3. The proposed co-operative robot architecture has been shown to allow robot teams to perform real-world missions over long periods, even while the environment or the robotic team itself changes. An important component is the control strategy that enables the robots to adapt their actions throughout a live mission without human intervention. The improvement in team performance was achieved by updating the control of the robots based on knowledge acquired on-line. Since the robot team members continually monitor the performance of their team-mates and update the performance measures accordingly, the response to improved or degraded capabilities is automatic, regardless of mission length. The results show that the robot team is able to achieve adaptive co-operative control despite

dynamic changes in the environment and variation in the capabilities of the team members.

7.3. Further Work

1- The output of the fuzzy logic rules is not optimal because the rules and membership functions are developed heuristically. The rules and membership functions could be optimised by learning based on some advanced search methods such as genetic algorithms.

2- The developed robot team is only able to track one target at a time. For multiple targets tracking, a task allocation algorithm is required to direct an appropriate number of robots to track each target.

3- The idea of having robots to learn how to accomplish a task, rather than being told explicitly is an appealing one. It seems easier and much more intuitive for the programmer to specify what the robot should do, and then let it learn the fine details of how to do it. A technique such as reinforcement learning is required for optimising the interaction with an environment or control of a system.

4- The robots developed in this work are few in number and very simple due to cost constraints. If a larger budget is available, larger teams of more advanced robots could be built to enable more complex tasks to be attempted.

Publications

1. Pham, D. T. and Awadalla, M. H., (2002). Fuzzy logic based behaviour coordination in multi-robot system for dynamic target tracking. The Third CIRP Int. Seminar on Intelligent Computation in Manufacturing Engineering (ICME 2002), Ischia (Naples), Italy, pp. 551-556.

2. Pham, D. T. and Awadalla, M. H., (2002). Neuro-Fuzzy Based Adaptive Cooperative Mobile Robots. In the Proc. of IECON'02. Sevilla, Spain, pp. 2962-2967

3. Pham, D. T. and Awadalla, M. H., (2004). Fuzzy logic based behaviour coordination in multi-robot system. Proc. Instn Mech. Engrs Vol. 218 Part B: J., pp. 583-598.

References

Abdessemed, F., Benmahammed, k. and Monacelli, E. (2004). Fuzzy-based reactive controller for a non-holonomic mobile robot. In Robotics and Autonomous Systems, Vol. 47, No. 1, pp. 31-46.

Acosta, C.A. and Hu, H., (2003a). Goals and Actions: Learning by Imitation. Proc. of AISB 2003 Convention: Cognition in Machines and Animals, University of Wales, Aberystwyth, UK, pp. 112-115.

Acosta, C.A. and Hu, H., (2003b). Robotics Societies: Elements of Learning by Imitation, Proc. of IASTED Int. Conf. on Artificial Intelligence and Applications, Innsbruck, Austria, pp. 315-320.

Alami, R., Fleury, S., Herrb, M., Ingrand, F., and Robert, F., (1998). Multi-robot co-operation in the MARTHA project. IEEE Robotics and Automation Magazine, Vol. 5, No.1, pp 36-47.

Andrew, H., Lotspeich, D., Hoff, J., and George, A., (1996). Rapid reinforcement learning for reactive control policy design. Proc. of the World Congress on Neural Networks, San Diego, California, pp. 18-26.

Arambula, F. and Castañeda, M.A. (2004). Autonomous robot navigation using adaptive potential fields. Mathematical and Computer Modelling, Vo. 40, Issues 9-10, pp. 1141-1156.

Arkin, R.C. and Mackenzie, D., (1994). Temporal coordination of perceptual algorithms for mobile robot navigation. IEEE Transactions on Robotics and Automation, Vol.10, No.3, pp. 276-286.

Arkin, R.C., (1990). Integrating behavioural, perceptual and world knowledge in reactive navigation. Journal of Robotics and Automation Systems, Vol. 6, pp. 105-122.

Arkin, R.C., (1999). Behaviour-based Robotics. MIT Press.

Arkin, R.C., Balch, T., and Nitz, E., (1993). Communication of behavioural state in multi-agent retrieval tasks. Proc. of 1993 IEEE Int. Conf. on Robotics and Automation, Atlanta, GA, Vol. 3, pp. 588-594.

Asama, H., Ozaki, H., Matsumoto, A., Ishida, Y., and Endo, I., (1992). Development of task assignment system using communication for multiple autonomous robots. Journal of Robotics and Mechatronics, Vol. 4, No. 2, pp. 122-127.

Balch, T. and Arkin, R.C., (1994). Communication in reactive multi-agent robotic systems. Autonomous Robots, Vol.1, No.1, pp. 1-25.

Bay, J. S. and Unsal, C., (1994). Group dynamics in many-robot systems. IEEE Sympoium on Intelligent Control, Columbus, Ohio, pp. 249-254.

Beckers, R., Holland, O.E., and Deneubourg, J.L., (1994). From local actions to global tasks: stigmergy and collective robotics. In Proc. of the Fourth Workshop on Artificial Life, Boston, pp. 181-189.

Berenji, H. R. and Khedkar, P., (1992). Learning and tuning fuzzy logic controllers through reinforcements. IEEE Transactions on Neural Networks, Vol.3, No.5, pp. 724-740.

Boehringer, K., Brown, R., Donald, B., Jennings, J., and Rus, D., (1995). Distributed robotic manipulation: Experiments in minimisation. In Proc. of the Fourth Int. Sympoium on Experimental Robotics, Standford, pp. 11-252.

Bonabeau, E., Dorigo, M., and Theraulaz, G., (1999). Swarm intelligence: from natural to artificial systems. Santa Fe Institute Studies in the Sciences of Complexity. Oxford University Press.

Brooks, R.A., (1986). A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1, pp. 14-23.

Buschka, P., Saffiotti, A. and Wasik, Z., (2000). Fuzzy Landmark-Based Localization for a Legged Robot. Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), Takamatsu, Japan, pp. 1205-1210.

Caloud, P., Choi, W., Latombe, J.C, Pape, C.L., and Yim, M., (1990). Indoor automation with many mobile robots. In the Proc. of the IEEE Int. Workshop on Intelligent Robots and Systems, Tsuchiura, Japan, pp. 62-72.

Chantemargue, F., and Hirsbrunner, B., (1999). A collective robotics application based on emergence and self-organisation. In Proc. of Fifth Int. Conf. for Young Computer Scientists, Nanjing, China, pp.1-8.

Coradeschi, S., Driankov, D., Karlsson, L., and Saffiotti, A., (2001). Fuzzy Anchoring. Proc. of the IEEE Intl. Conf. on Fuzzy Systems, Melbourne, Australia, pp. 111-114.

Cragg, L. and Hu, H. (2005). Application of reinforcement learning to a mobile robot in reaching recharging station operation, Pro. of IPROMS 2005 Virtual Conference, http://conference.iproms.org.

Demirli, K. and Molhim, M. (2004). Fuzzy dynamic localization for mobile robots. Fuzzy Sets and Systems, Vol. 144, Issue 2, pp. 251-283.

Deneubourg, D., Goss, S., Sandini, G., Ferrari, F., and Dario, P., (1990). Self-organising collection and transport of objectives in unpredictable environments. In Japan-U.S.A. Symposium on Flexible Automation, Kyoto, Japan, pp. 1093-1098.

Deneubourg, J.C., Goss, S., Franks, N., Sendova, A., Franks, A., Detrin, C., and Chatier, L., (1991). The dynamics of collective sorting: Robots-like ant and ant-like robot. In J.A. Mayer and S.W. Wilson, editors, Simulation of Adaptive Behaviour: from Animals to Animates, pp. 356-365, MIT Press.

Djakov, D.B., (1996). An intelligent sensor for direction detection. MSc thesis, Intelligent systems laboratory, Manufacturing Engineering Centre, Cardiff University.

Dorigo, M., Maniezzo, V., and Colorni, A., (1996). The ant system: Optimisation by a colony of co-operating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, pp.29-41.

Drogoul, A. and Ferber, J., (1992). From tom thumb to the dockers: some experiments with foraging robots. In Proc. of the Second Int. Conf. on Simulation of Adaptive Behaviour, Honolulu, Hawaii, pp. 451- 459.

Duhaut, M., Asama, H., and Suzuki, S., (1998). The RoboCup physical agent challenge: Phase I. Applied Artificial Intelligence (AAI), Vol. 12, pp. 251-265.

Elfwing, S., Uchine, E., Doya, K., and Christensen, H. (2004). Multi-agent reinforcement learning. In IROS-04, IEEE/JSR, pp. 3164-3169.

Fong, T., Nourbakhsn, I. and Dautenhahn K., (2003). A survey of socially interactive robots. Robotics and Autonomous Systems, Vol. 42, pp. 143-166.

Franks, N., (1989). Army ants: A collective intelligence. American Scientist, Vol. 77, pp. 139-145.

Gasser, L., Rouquette N., Hill, R. and Lieb, J., (1989). Representing and using organisational knowledge in DAI systems. In Les Gasser and Michael N. Huhns editors, Distributed Artificial Intelligence, Pitman, Vol. 2, pp. 55-78.

Ghanea-Hercock, R. and Barnes, D.P., (1996). An evolved fuzzy reactive control system for co-operating autonomous robots. In Proc. of the Fourth Int. Conf. on Simulation of Adaptive Behaviour, (SAB), Cap Cod, pp. 599-607.

Goodridge, S.G. and Luo, R.C., (1994). Fuzzy behaviour fusion for reactive control of an autonomous mobile robot: MARGE. In Proc. of the IEEE Int. Conf. on Robotics and Automation, San Diego, CA, pp. 1622-1627.

Hoff, J. and Bekey, G., (1995). An architecture for behaviour coordination learning. In IEEE Int. Conf. on Neural Networks, Perth, Australia, Vol. 5, pp. 2375-2380.

Hoffmann, F. and Pfister, G., (1997). Evolutionary Design of a Fuzzy Knowledge Base for a Mobile Robot. Int. Journal of Approximate Reasoning (Elsevier, North Holland),vol. 17, no. 4, pp. 447- 469.

Hu, H. and Gan, J.Q., (2005). Sensors and data fusion algorithms in mobile robotics, Research report CSM-422, University of Essex, ISSN 1744-8050.

Hu, H., Kelly, I. and Keating, D., (1998). Coordination of multiple mobile robots via communication. In Proc. of SPIE'98, Mobile Robots XIII Conf., Boston, pp.94-103.

Hu, H. and Gu, D. (2005). Hybrid Learning Architecture for Fuzzy Control of Quadruped Walking Robots, Int. Journal of Intelligent Systems, Vol. 20, No. 2, pp. 131-152.

Hu, H. and Gu, D. (2004). Teaching robots to coordinate their behaviours, Proc. of IEEE Int. Conf. on Robotics and Automation, Riverside Hilton & Towers, New Orleans, LA.

Huber, M. and Durfee, E., (1993). Observational uncertainty in plan recognition among interacting robots. In Proc. of the 1993 IJCAI Workshop on Dynamically Interacting Robots, Chambery, France, pp. 68-75.

Humphrys, M., (1997). Action selection methods using reinforcement learning. PhD thesis, Computer Laboratory, University of Cambridge.

Huosheng, H., Keating, I.D. and Vinagre, D. (1998). "Coordination of multiple mobile robots via communication". In Proc. of SPIE. Mobile Robots XIII and Intelligent Transportation Systems, Boston, Massachusetts, pp. 94-103.

Ijspeert, A.J., Martinoli A., Billard A. and Gambardella, L.M., (2001). Collaboration through the exploitation of local interactions in autonomous collective robotics: the stick pulling experiment, Autonomous Robots.Vol.11, No.2, pp 149-171, Kluwer Academic.

Keller, J. M., Yager, R. R. and Tahani, H. (1992). Neural Network Implementation of Fuzzy Logic. Fuzzy Sets and Systems, Vol.45, pp.1-12.

Khatib, O., (1986). Real–time obstacle avoidance for manipulators and mobile robots. The Int. Journal of Robotics Research, Vol.5, No.1, pp. 90-98.

Kirchner, W.H. and Towne, W.F., (1994). The sensory basis of the honeybee's dance language. Scientific American, Vol. 270, No. 6, pp. 74-80.

Korf, R., (1992). A simple solution to pursuit games. In working papers of the 11th Int. Workshop on Distributed Artificial Intelligence, Boston, pp. 183-194.

Kosecka, J. and Bajcsy, R., (1993). Discrete event systems for autonomous mobile agents. In Proc. of Intelligent Robotic Systems '93, Zakopane, pp. 21-31.

Kristensen, S., (1996). Sensor planning with bayesian decision analysis. PhD thesis, Department of Medical Informatics and Image Analysis, Aalborg University.

Kristina, L. and Aram, G., (2002). Mathematical Model of Foraging in a Group of Robots: Effect of Interference. Autonomous Robots, Vol.13, No. 2, pp. 127-141,2002.

Kube, C. R. And Zhang, H., (1992). Collective robotic intelligence. In Proc. of the Second Int. Workshop on Simulation of Adaptive Behaviour, Honolulu, Hawaii, pp. 460-468.

Kube, C. R. and Zhang, H., (1994). Stagnation recovery behaviours for collective robotics. In 1994 IEEE/RSJ/GI Int. Conf. on Intelligent Robots and Systems, Vol. 3, pp.1883-1890.

Kube, C. R. and Bonabeau, E., (2000). Cooperative transport by ants and robots. Robotics and Autonomous Systems, Vol. 30, pp. 85-101.

Larsson, J., Broxvall, M. and Saffiotti A. (2005). A Navigation System for Automated Loaders in Underground Mines. Proc. of the 5th Int. Conf. on Field and Service Robotics (FSR-05). Port Douglas, Australia, pp. 112-124.

Lee, C. C. (1990a). Fuzzy Logic in Control Systems: Fuzzy Logic Controller Part I. IEEE Trans. Systems, Man. and Cybern, Vol.20, No.2, pp.404-417.

Lee, C. C. (1990b). Fuzzy Logic In Control Systems: Fuzzy Logic controller Part II. IEEE Trans. Systems, Man. and Cybern, Vol.20, No.2, pp.514-527.

Levy, R. and Rosenschein, J., (1992). A game theoretic approach to the pursuit problem. In working papers of the 11th Int. Workshop on Distributed Artificial Intelligence, Boston, pp. 195-213.

Lin, C. and Mon, Y., (2004). Hybrid adaptive fuzzy controllers with application to robotic systems Fuzzy Sets and Systems, Vol. 139, No.1, pp. 151-165.

Lin, C.T. and Lu, Y.C., (1996). A Neural fuzzy system with fuzzy supervised learning, IEEE Transactions. Systems, Vol. 26, No. 5, pp.745-753.

Mackenzie, D.C, Arkin, R.C. and Cameron, J., (1997). Specification and execution of multi-agent missions. Autonomous Robots, Vol. 4, No. 1, pp. 276-292.

MacLennan, B., (1991). Synthetic ethology: An approach to the study of communication. In Proc. of the Second interdisciplinary workshop on synthesis and simulation of living systems, Redwood City, pp. 631-658.

Maes, P., (1989). How to do the right thing. Connection Science Journal, Vol. 1, No. 3, pp. 291-323.

Mamdani, E. H. (1974), Application of Fuzzy Algorithms for Control of Simple Dynamic Plant, Proc. of IEE, Vol. 121, No. 12, pp.1585-1588.

Martinoli A., Ijspeert A.J., and Gambardella L.M., (1999b). A probabilistic model for understanding and comparing collective aggregation mechanisms. Proc. of the Fifth European Conf. on Artificial Life (ECAL99), Berlin, pp. 575-584.

Martinoli, A. and Mondada, F., (1995). Collective and co-operative behaviours: biologically inspired experiments in robotics, Pre-prints of the fourth Int. Symposium on Experimental Robotics, ISER'95, Stanford, CA, pp. 2-7.

Martinoli, A. and Mondada, F., (1998). Probabilistic modelling of a bio-inspired collective experiment with real robots. In Proc. of the Fourth Int. Symposium. On Distributed Autonomous Robotic Systems DARS-98, Karlsruhe, Germany, pp. 289-308.

Martinoli, A., (1999). Swarm intelligence in autonomous collective robotics: from tools to the analysis and synthesis of distributed control strategies. Unpublished doctoral dissertation, PhD Thesis No 2069, EPFL.

Martinoli, A., Franzi, E., and Matthey, O. (1997a). Towards a reliable set-up for bio-inspired collective experiments with real robots. In Proc. of the Fifth Int. Symposium. On Experimental Robotics ISER-97, Barcelona, Catalonia, pp. 597-608.

Martinoli, A., Ijspeert, A. J. and Mondada, F., (1999 a). Understanding collective aggregation mechanisms: from probabilistic modelling to experiments with real robots. Invited paper, Special Issue on Distributed Autonomous Robotic Systems, Journal of Robotics and Autonomous Systems, Vol. 29, pp. 51-63.

Martinoli. A., Yamamoto, M., and Mondada, F., (1997b). On the modelling of bio-inspired collective experiments with real robots. In Proc. of the fourth European Conf. on Artificial Life ECAL'97, Brighton, UK, pp.432-441.

Mataric, J., (1992). Designing emergent behaviours: from local interactions to collective intelligence. In the second Int. Conf. on Simulation of Adaptive Behaviour, Honolulu, Hawaii, pp. 432-441.

Mataric, M., Nilsson, M., and Simsarian, K., (1995). Co-operative multi-robot box pushing. In the Proc. of the 1995 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IEEE Computer Society, pp. 556-561.

Mataric, M.J., (1996). Learning in multi-robot systems. In Adaptation and Learning in Multi-Agent Systems, Gerhard Weiss and Sandip Sen, editors. Lecture Notes in Artificial Intelligence (LNAI), 1042, Springer-Verlag, pp.152-163.

Mataric, M.J., (1998). Coordination and learning in multi-robot systems. IEEE Intelligent Systems, pp.6-8.

Melhuish, C., Holland, O., and Hoddell, S., (1998). Collective sorting and segregation in robots with minimal sensing. In Proc. of the Fifth Int. Conf. of Simulation of Adaptive Behaviour, Zurich, pp. 345-363.

Michel, O., (1998). Webots: Symbiosis between virtual and real mobile robots. In Proc. of the first Int. Conf. on Virtual Worlds, VW' 98, San Mateo, CA, pp. 254-263, Morgan Kaufmann.

Minguez, J. and Montano, L. (2005). Sensor-based robot motion generation in unknown, dynamic and troublesome scenarios. Robotics and Autonomous Systems, Vol. 52, Issue 4, pp. 290-311.

Murrieta-Cid, R., Sarmiento, A. and Hutchinson, S. A. (2003). Distributed motion coordination strategy for multiple nonholonomic mobile robots in co-operative hunting operations. Robotics and Autonomous Systems, Vol. 43, No. 4, pp. 234-242.

Noreils, F.R., (1993). Toward a robot architecture integrating cooperation between mobile robots: application to indoor environment. Int. Journal of Robotics Research, Vol.12, No.1, pp. 79-98.

Pan, J., and Pack, D.J., Kosaka, A., and Kak, A.C., (1995). FUZZY-NAV: A vision-based robot navigation architecture using fuzzy inference for uncertainty reasoning. In Proc. of the World Congress on Neural Networks, Washington, DC, pp. 602-607.

Parker, L. E., (2002). Distributed algorithms for multi-robot observation of multiple moving targets. Autonomous Robots, Vol. 12, No. 3, pp. 231-255.

Parker, L.E., (1994). ALLIANCE: An architecture for fault tolerance, co-operative control of heterogeneous mobile robots. In Proc. of the 1994 IEEE/RSJ/GI Int. Conf. on Intelligent Robots and Systems (IROS'94), Munich, Germany, pp. 776-783.

Parker, L.E., (1995). The effect of action recognition and robot awareness in co-operative robotic teams. Proc. of the 1995 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS '95), Vol. 1, pp. 212-219.

Parker, L.E., (1996). On the design of behaviour-based multi-robot teams. Advanced Robotics, Vol. 10, No.6, pp.547-578.

Parker, L.E., (1997). L-ALLIANCE: task-oriented multi-robot learning in behaviour-based systems. Advanced Robotics, Special Issue on Selected Papers from IROS '96, Vol. 11, No. 4, pp. 305-322.

Parker, L.E., (1999). Adaptive heterogeneous multi-robot teams. Neuro-computing, special issue of NEURAP '98: Neural Networks and Their Applications, Vol. 28, pp.75-92.

Parker, L.E., (2000). Lifelong adaptation in heterogeneous multi-robot teams: Response to Continual Variation in Individual Robot Performance. Autonomous Robots, Vol. 8, No. 3, pp. 239-267.

Parker, L.E., (2001). Evaluating success in autonomous multi-robot teams: experiences from ALLIANCE architecture implementations. Journal of Theoretical and Experimental Artificial Intelligence, Vol. 13, pp. 95-98.

Parker, L.E., Schneider, F. and Schultz, A. (2005). Multi-Robot Systems: From Swarms to Intelligent Automata. Vol. III, Springer, 2005 (ISBN: 1-4020-3388-5).

Pham, D. T. and Dayal R. Parhi, (2003). Navigation of multiple mobile robots using a neural network and a Petri Net model. Journal of Robotica, Vol. 21, No. 1, pp. 79-93.

Pin, F.G. and Watanabe, Y., (1994). Navigation of mobile robots using a fuzzy behaviourist approach and custom-designed fuzzy inference boards. Robotics, Vol. 12, pp. 491-503.

Pirjanian, P. and Mataric, M., (2000). Multi-robot target acquisition using multiple objective behaviour coordination. In Proc. of the 2000 IEEE Int. Conf. on Robotics and Automation, San Francisco, CA, pp. 101-106.

Pirjanian, P. and Christensen, H.I., (1997). Behaviour coordination using multiple-objective decision-making. SPIE Conf. on Intelligent Systems and Advanced Manufacturing, Pittsburgh, Pennsylvania USA, pp.78-89.

Pirjanian, P., (1998). Multiple objective action selection & behaviour fusion using voting. Ph.D thesis, Faculty of Technical Sciences, Aalborg University, Denmark, pp. 257-282.

Pirjanian, P., (1999). Behavior Coordination Mechanisms - State-of-the-art. Tech-report IRIS-99-375, Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California.

Rosenblatt, J. and Thorpe, C., (1995). Combining multiple goals in a behaviour-based architecture IEEE Conference on Intelligent Robots and Systems, Human Robot Interaction and Cooperative Robots (IROS '95), Pittsburgh, PA, Vol. 1, pp. 136 - 141.

Rosenblatt, J., (1997). The Distributed Architecture for Mobile Navigation. In Journal of Experimental and Theoretical Artificial Intelligence, Vol. 9, No. 2/3, pp.339-360.

Rosenblatt, J.K. and Payton, D.W., (1989). A Fine-Grained Alternative to the Subsumption Architecture for Mobile Robot Control. In Proc. of the IEEE/INNS International Joint Conference on Neural Networks, Washington DC, Vol. 2, pp. 317-324.

Ruspini, E.H., (1991). On the semantics of fuzzy logic. Int. Journal of Approximate Reasoning, Vol. 5, pp. 45-88.

Russell, R.A., (1999). Ant trails – an example for robots to follow? In Proc. 1999 IEEE Int. Conf. on Robotics and Automation, pp. 2698-2703.

Saffiotti, A. and Wasik, Z., (2003). Using Hierarchical Fuzzy Behaviours in the RoboCup Domain. C. Zhou, D. Maravall and D. Ruan (Eds) Autonomous Robotic Systems. Springer, Berlin, pp. 235-262.

Saffiotti, A., (1997). The uses of fuzzy logic in autonomous robot navigation: a catalogue raisonne. Technical report 2.1, IRIDINA, Universite Libre de Bruxelles, 50 av. F. Rossevelt, CP 194/6, B-1050 Brussels, Belgium.

Saffiotti, A., Zumel, N.B. and Ruspini, E.H., (2000). Multi-robot team coordination using desirabilities. Proc. of the sixth Int. Conf. on Intelligent Autonomous Systems (IAS), Venice, Italy, pp. 107-114.

Schultz, A., Parker, L. E. and Schneider, F., (2003). Multi-Robot Systems: From Swarms to Intelligent Automata, Vol. 2, Kluwer, 2003 (ISBN: 1-4020-1185-7).

Sen, S., Sekaran, M., and Hale, J., (1994). Learning to coordinate without sharing information. In Proc. of the Twelfth National Conf. of Artificial Intelligence, Seattle, WA, pp. 426-431.

Shankir, Y., (2001). Fuzzy logic systems and fuzzy neural networks for dynamic systems modelling and control. PhD thesis, Intelligent systems laboratory, Manufacturing Engineering Centre, Cardiff University.

Sharpe, T. and Webb, B., (1998). Simulated and situated models of chemical trail following in ants. In Proc. of the Fifth Int. Conf. of Simulation of Adaptive Behaviour, Cambridge, London, pp. 195-204.

Sossai, C., Chemello, G., and Saffiotti, A., (2000). A Note on the Role of Logic in Fuzzy Logic Controllers. Proc. of the 8th Int. Conf. on Information Processing and Management of Uncertainty, Madrid, SP, pp. 717-723.

Steels, L., (1990). Cooperation between distributed agents through self-organisation. In Proc. of the IEEE Int. Workshop on Intelligent Robots and Systems '902 Vol. 2, pp. 8-14, Amsterdam: Elsevier Science (North-Holland).

Stephens, L. and Merx, M. (1990). The effect of agent control strategy on the performance of a DAI pursuit problem. In Proc. of the 1990 Distributed AI Workshop, Bandera, Texas, pp. 324-331.

Stilwell, D. and Bay, J., (1993). Toward the development of a material transport system using swarms of ant-like robots. In Proc. of IEEE Int. Conf. on Robotics and Automation, Atlanta, GA., pp. 766-771.

Sugeno, M. and Nishida, M. (1985). Fuzzy Control of Model Car. Fuzzy Sets and Systems, Vol.16, pp.103-113.

Sugeno, M. and Yasukawa, T. (1993). A Fuzzy-Logic-Based Approach to Qualitative Modelling. IEEE Trans. Fuzzy Systems, Vol.1, No.1, pp.7-31.

Surman M., Juser J. and Peters L., (1995). Guiding and controlling mobile robots with a fuzzy controller. In Proc. of the fourth IEEE Int. Conf. on Fuzzy Systems Yokohama, Japan, pp.83-88.

Takagi, T. and M. Sugeno (1985), Fuzzy Identification of Systems and Its Applications to Modelling and Control, IEEE Trans. Syst., Man. and Cybern., Vol.SMC-15, No.1, pp.116-132.

Tan, M., (1993). Multi-agent reinforcement learning: Independent vs. Co-operative agents. In proc. of the Tenth Int. Conf. on Machine Learning, Amherst, MA, pp. 330-337.

Tarasewich, P., and McMullen, P. R., (2002). Swarm Intelligence. Communications of the ACM, Vol. 45, No. 8, pp. 62-67.

Touzet, C., (2000) Robot Awareness in Cooperative Mobile Robot Learning. Autonomous Robots, Vol. 8, No. 1, pp. 93-105.

Vaughan, R., Stoey, K., Sukhatme, G., and Mataric, M., (2000a). Blazing a trail: insect-inspired resource transportation by a robot team. In Proc. of the fifth

Int. Symposium on Distributed Autonomous Robotic Systems (DARS), Knoxville, TN, pp. 111-120.

Vaughan, R.T., stoy, K., Sukhatme, G.S. and Mataric, M.J., (2000b). Whistling in the dark: Co-operative trail following in uncertain localisation space. In Proc. of the Fourth Int. Conf. on Autonomous Agents. Boston, pp. 187-194.

Wagner, I. A. and Bruckstein, A. M., (1995). Co-operative cleaners - a study in ant robotics, In the Proc.of the third French-Israeli Symposium on Robotics. Herzlia, pp. 289-308.

Wagner, I. A., Lindenbaum, M. and Bruckstein, A. M., (1998). Efficiently exploring a continuous unknown domain by an ant inspired process. The second Int. workshop on Randomisation and Approximation Techniques in Computer Science, Barcelona, Spain, pp. 116-130.

Wang, L. X. and Mendel, J. M., (1993). Fuzzy Adaptive Filters with Application to Nonlinear Channel Equalisation. IEEE Trans. Fuzzy Systems, Vol.1, No.3, pp.161-170.

Wasik, Z. and Saffiotti., A., (2002). A Fuzzy Behavior-Based Control System for Manipulation. Proc. of the *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems* (IROS), Lausanne, CH, pp. 1596-1601.

Weiss, G., Sen, S., and Editors, (1996). Adaptation and learning in multi-agent systems. In the eleventh Int. Workshop on Distributed Artificial Intelligence, pp. 183-194.

Yamada, S. and Saito, J., (1999). Adaptive action selection without explicit communication for multi-robot box-pushing. 1999 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'99), pp. 1444-1449.

Yen, J., and Pfluger, N. (1995). A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation. IEEE Transactions. On Systems, Man, and Cybernetics, Vol. 25, No. 6, pp. 971-978.

CAR ★ PRIFYSGOL CAERDYDD ★