# Interaction and Intelligent Behavior

by

Maja J Matarić

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1994



Author .....
Department of Electrical Engineering and Computer Science
May 12, 1994

Certified by .....
Rodney A. Brooks
Professor of Computer Science and Engineering
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chairman, Committee on Graduate Students

# Interaction and Intelligent Behavior

by

Maja J Matarić

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 1994, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

## Abstract

This thesis addresses situated, embodied agents interacting in complex domains. It focuses on two problems: 1) synthesis and analysis of intelligent group behavior, and 2) learning in complex group environments.

*Behaviors* are proposed as the appropriate level for control and learning. **Basic behaviors** are introduced as building blocks for synthesizing and analyzing system behavior. The thesis describes the process of selecting such basic behaviors, formally specifying them, algorithmically implementing them, and empirically evaluating them. All of the proposed ideas are validated with a group of up to 20 mobile robots using a basic behavior set consisting of: *avoidance*, *following*, *aggregation*, *dispersion*, and *homing*. The set of basic behaviors acts as a substrate for achieving more complex high-level goals and tasks. Two behavior combination operators are introduced, and verified by combining subsets of the above basic behavior set to implement collective *flocking* and *foraging*.

A methodology is introduced for automatically constructing higher-level behaviors by learning to select among the basic behavior set. A novel formulation of reinforcement learning is proposed that makes behavior selection learnable in noisy, uncertain multi-agent environments with stochastic dynamics. It consists of using **conditions** and **behaviors** for more robust control and minimized state-spaces, and a reinforcement shaping methodology that enables principled embedding of domain knowledge with two types of shaping functions: **heterogeneous reward functions** and **progress estimators**. The methodology outperforms two alternatives when tested on a collection of robots learning to forage. The proposed formulation enables and accelerates learning in complex multi-robot domains. The generality of the approach makes it compatible with the existing reinforcement learning algorithms, allowing it to accelerate learning in a variety of domains and applications.

The presented methodologies and results are aimed at extending our understanding of synthesis, analysis, and learning of group behavior.

Thesis Supervisor: Rodney A. Brooks
Title: Professor of Computer Science and Engineering

# Acknowledgments

Since this is neither a Nobel Prize nor an Oscar acceptance, the acknowledgements need not be short. There are many people to thank for contributing to seven great years I've had at the MIT Artificial Intelligence Lab.

I thank Rod Brooks for being a great advisor, an excellent motivator, a rebel, a role model, and most importantly, never a conservative. I am grateful for his advice and patience, and his continuing support. I am also grateful to Gerry Sussman, Lynn Stein, and Patrick Winston for good thesis-related, job-related, and life-relate advice during my last year at MIT.

Unlimited thanks go to Mike Bolotski and Jose Robles, who have been great friends from the beginning until the end, participated in kayaking classes, heated discussions, and LaTeX hacking in the last hours of thesis preparation. Together with Brian Eberman and Sundar Narasimhan, they have kept me honest through the years by asking hard questions and not accepting mediocre answers. Special thanks to Mike for proof-reading numerous papers, encouraging me to learn Russian, introducing me to the heymarket, and being a superb friend to have in Vancouver. Special thanks to Jose for being a good friend, dance partner, and source of good advice all the way back to the days of Toto and the Masters' thesis to the formalization of the basic behaviors in this document.

Numerous thanks to Anita Flynn for being an inspiring force, a fantastic cheerleader for running and research, and always great to talk to about all aspects of life. Warmest thanks to Mike Erdmann for being a wonderfully supportive officemate during the first two years at MIT, for initiating me into the culture, for talking and walking and running and sailing and for unenumerable good things he has done. Endless gratitude to Nancy Pollard for being a patient, fun, and supportive officemate for four great years. Thanks for sharing everything, including running, squash, research dilemmas, and plant watering.

Many Slavic thanks to Matt Marjanović for being crazy in just the right way, for managing as my officemate during the job interview and thesis season, for making me

put the ć back into my last name, and of course, for the Pig. Thanks to Ian Horswill for being a great companion for flights to far-away conferences, for being a patient source of good advice, and for introducing me to the weird world of comics. Sibling thanks to Paul Viola for being a "bro" in the early years at the Lab, for giving me outstanding advice about theses committees, research directions, people to talk to about work, and most of all for not hesitating in saying exactly what he is thinking.

Thanks to Cindy Ferrell for having an endless supply of good cheer, great spirit, excellent taste (especially in tech report covers), and enthusiasm for both research, sports, and exotic vacations. Thanks to Pattie Maes for being a friend and a part-time advisor on all matters imaginable, for having impeccable taste (again in all matters imaginable), and for being a superb role model. Thanks to Lynne Parker for understanding the robots' quirks better, and for having endless patience with both them and me. She was a great person with whom to share research views as well as hardware frustrations.

Many thanks to three hard-working UROPS: Matt Marjanović, Stanley Wang, and Owen Wessling, for making the unruly Nerd Herd behave. Thanks to Phillip Alvelda for rock climbing and rappeling lessons, bad movie advice but good home movies, and endless enthusiasm. Thanks to David Beymer, Gideon Stein, Sandy Wells, and Matt Williamson for being great running and talking partners. Added thanks to Matt for the most hilarious squash games and the Scottish accent. Thanks to Carl de Marken for writing memorable Girl Scout Benefit announcements and for being an expert and a patient teacher for every outdoor activity known to mankind. Many thanks to Adonis Stassinopoulos for unsurpassed Greek desserts, great taste in movies, and for always being excellent company.

Many thanks to Bruce Blumberg, David Brock, Barb Moore Bryant, and Amy Bruckman for great discussions, to Joanna Bryson for her enthusiasm and organizational skills, to Mike Caine for bizarre humor through the years, and of course for the broken talking clock, to the great people in the Cog Group, to Trevor Darrell for always having something interesting and new to say on a variety of topics, and for always keeping in touch, to Lisa Dron for good dinner parties, to Charles Isbell for great

enthusiasm and unique quotes, to Tom Knight for being at the Lab at weird hours, knowing about and always being willing to discuss any and all topics, and for loaning me "green slime" for building a ceiling-swinging robot, to Tina Kapur and Lilly Lee for not minding me adopting the fridge in their office, to Marina Meila for being a great travel-mate and wonderful to talk to about machine learning, to Henry Minsky for being an irreplaceable cow supplier, to Annika Pfluger for all administrative help through the years and great talent for cello, to Robert Ringrose for saintly patience in helping with the Creature Library and the video equipment, to Ruth Schonfeld for being a great friend to talk to even if we loose touch for a few months at a time, to Laurel Simmons for keeping the Lab from collapsing, to Patrick Sobalvarro for great literary skills and movie advice, and to Jeanne Speckman for organizing the dance group, and to everybody I've left out because it's 2am.

Deepest thanks to Liba and Bora Mikić, who served as my Boston family, and were always ready to overfeed me and listen to my stories. Great thanks to Milan Radovanov for being the smart uncle I could talk about and talk with, and for always playing devil's advocate to keep me on my toes.

Warmest and most enduring thanks to Rich Roberts, for over a decade of love, support, and inexplicable, engaging silliness. He made every minute more fun, more memorable, and more worthwhile.

Infinite and incomparable gratitude to Mira Matarić, the most wonderful mother I could have. She taught all lessons by example, worked harder for and at everything in life than anybody else I have known, and had superhuman energy and perseverance that inspired everything I have accomplished. This thesis is dedicated to her.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Overview of the Thesis

One of the goals of AI is to gain insight into natural intelligence through a synthetic approach, by generating and analyzing artificial intelligent behavior. In order to glean an understanding of a phenomenon as complex as natural intelligence, we need to study complex behavior in complex environments.

Traditionally, AI has concerned itself with complex agents in relatively simple environments, in the sense that they could be precisely modeled and involved little or no noise and uncertainty. In contrast, reactive and behavior–based systems have placed agents with low levels of cognitive complexity into complex, noisy and uncertain environments. This thesis describes work that attempts to simultaneously scale up along both dimensions. The environmental complexity is scaled up by introducing other agents, and agent cognitive complexity is scaled up by introducing learning capabilities into each of the agents (figure 1-1).

This thesis addresses two problems:

1. synthesis and analysis of intelligent group behavior

2. learning in complex group environments

I will present a methodology for generating various robust group behaviors, including following, homing, and flocking (figure 1-2). I will also introduce a formulation

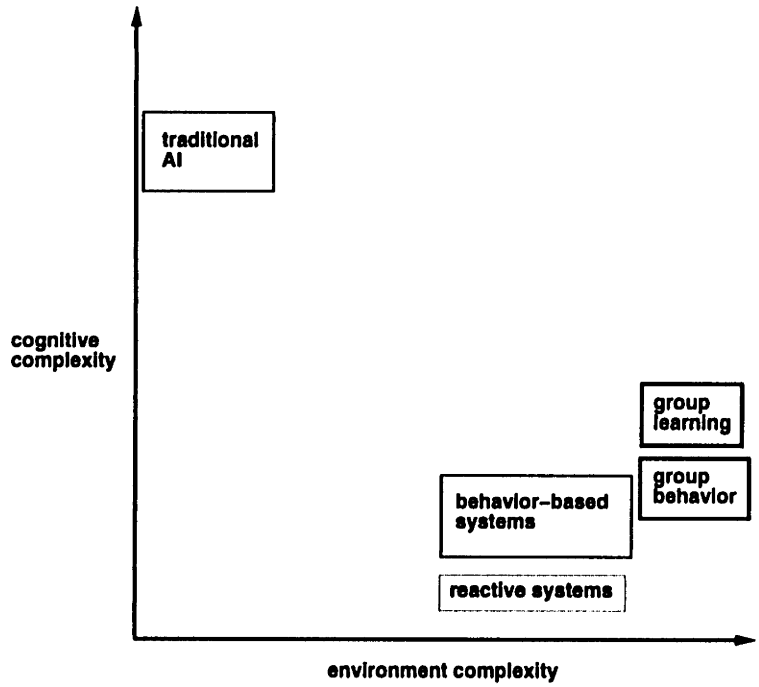
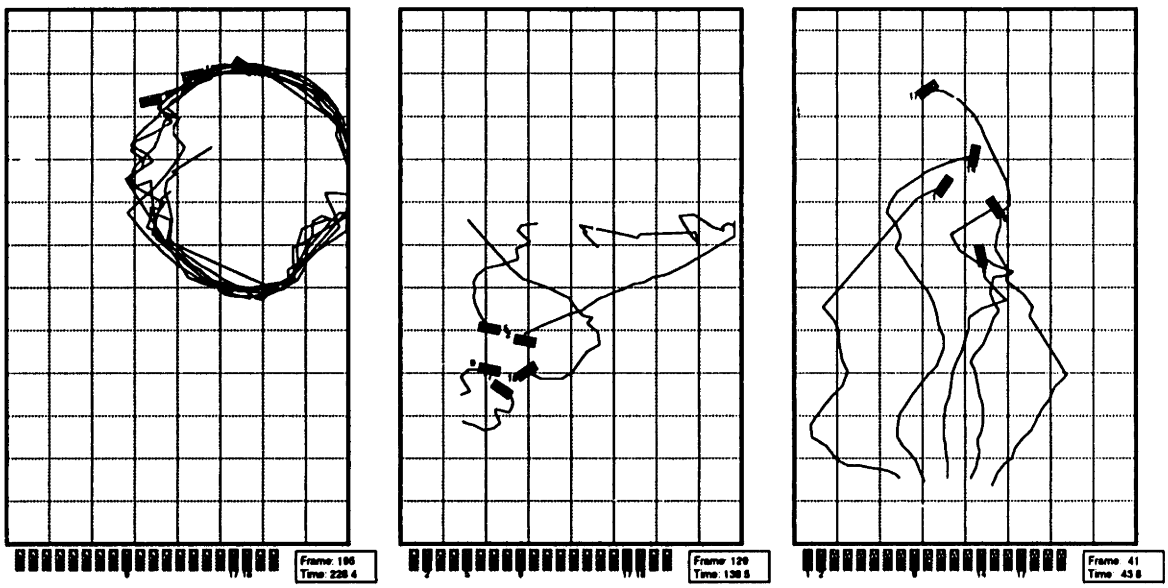



Figure 1-1: Traditional AI has addressed complex agents in simple environments while reactive and behavior-based approaches have dealt with simple agents in noisy and uncertain worlds. This work attempts to scale up along both dimensions simultaneously, by addressing synthesis and learning of complex group behavior.



Following Homing Flocking

Figure 1-2: This figure shows examples of physical robot data for three different group behaviors: following, homing, and flocking. The robots are plotted as black rectangles, with white arrows indicating their heading. The dark robots in the row of rectangles at the bottom shows the robots that were used in the experiment. Boxes on the lower right indicate frame numbers and the elapsed time for each of the runs.

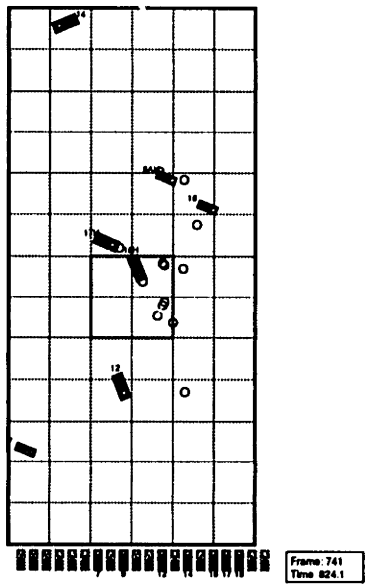


Figure 1-3: An example of the foraging behavior of 7 robots, shown after 13.7 minutes of running. About eight pucks have been delivered to the home region, marked with a grey box. The two robots near home are following each other on the way to the drop-off. Other robots are searching for additional pucks.

of reinforcement learning that allows a group of agents to learn complex tasks such as foraging (figure 1-3). Finally, I will validate the proposed approaches with experiments on groups of mobile robots.

This chapter gives a brief summary of the novel approaches, the experimental data, and the implications of the thesis. The organization of the remainder of the thesis is outlined at the end of the chapter.

## 1.1 Synthesis and Analysis of Group Behavior

This thesis is based on the assertion that intelligent collective behavior in a decentralized system results from *local interactions* which are based on simple rules. **Basic behaviors** are proposed as a methodology for structuring those rules through a principled process of synthesis and evaluation. A behavior is a control law that achieves and maintains some goal. For example, *wall following* is a behavior that maintains

an agent within a given range of distances from a wall.

For each domain, a set of behaviors can be found that are **basic** in that they are required for generating other behaviors, as well as being a minimal set the agent needs to reach its goal repertoire. The process of choosing the set of basic behaviors for a domain is dually constrained. From the bottom up, the process is constrained by the agent and environment dynamics. From the top down, the process is constrained by the repertoire of the agent's goals.

The example of group interactions between mobile robots will be used to illustrate this process. High-level goals of the system are defined as consisting of collectively manipulating objects (pucks) in the environment in an efficient fashion. Efficiency is defined in terms of minimizing energy. The agents are embodied, and endowed with specific mechanical, sensory, and effector constraints.

An effective set of basic behaviors in the spatial domain should enable the agents to employ a variety of flexible strategies for puck manipulation, collection, and distribution. The effectiveness of all such strategies depends on minimizing inter-agent interference while achieving the necessary goals.

The following set of basic behaviors is proposed:

- *avoidance* - minimizes collisions between agents
- *following* - minimizes interference by structuring movement of any two agents
- *aggregation* - gathers the agents
- *dispersion* - dissipates the agents
- *homing* - enables the agent to find an arbitrary location

The above behavior set is minimal in that its members are not further reducible to each other. Additionally, it will be shown that they are sufficient for achieving the set of pre-specified goals. The described basic behaviors are defined with respect to

the group. A number of other utility behaviors can be a part of an agent's repertoire, such as *grasping*, *dropping*, and *searching*, the only other behaviors used in this work.

The basic behavior set is evaluated by giving a formal specification of each of the behaviors, and comparing the collection of those specifications to a formal specification of the set of global tasks required for the group.

Once a basic behavior set is established, it can be implemented with a variety of algorithms. The first step in the verification of basic behavior algorithms is a comparison between the formal behavior specification and the formal correctness of the algorithm. It will be argued that it is difficult to prove properties of the exact behavior of individual agents within a group, but it is possible to evaluate and predict the behavior of the ensemble as a whole. Algorithms utilizing a centroid operator will be proposed, well suited to the group domain as they have statistical properties that allow for making predictions about the group behavior. Basically, centroid computations allow for averaging the inputs of each of the agents and minimizing small perturbations of individuals.

This thesis provides detailed specifications and algorithms for each of the basic behaviors. Instead of analytical proofs, it provides empirical evaluations of the performance of each of the algorithms, based on the following criteria:

- repeatability: how consistent is the behavior over different trials?
- stability: does the behavior oscillate under any conditions?
- adaptability: how robust is the behavior in the presence of sensor and effector error and noise?
- scalability: how is the behavior effected by increased and decreased group sizes?

The above criteria were applied to the data obtained by running at least 50 trials of each basic behavior. The experiments were performed on two different multi-agent environments, in order to minimize domain biases. The first environment was

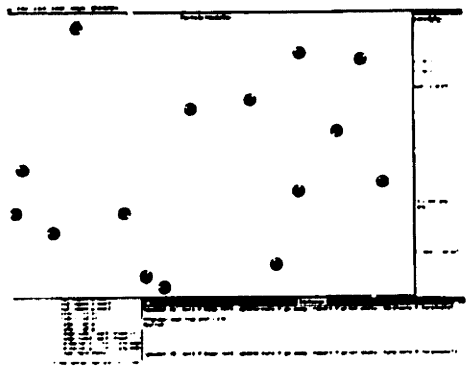


Figure 1-4: The simulator environment called the Interaction Monitor was used to validate the methodologies for synthesizing and analyzing group behavior described in the thesis. The agents are shown as black circles, with white markers indicating their heading. The large rectangle represents the agents' workspace.

a multi-agent simulator (the Interaction Monitor) featuring up to 50 agents with local sensing and distributed, local control (figure1-4).

The second environment was a collection of 20 physical mobile robots equipped with local sensors and local control (figure 1-5). Each of the robots is equipped with a suite of infra-red sensors for collision avoidance and puck detection and stacking, and micro switches and bump sensors for contact detection. In addition to the local sensors, the robots were equipped with radios and sonars for triangulating their position relative to two stationary beacons, and for broadcasting that position within a limited radius. The radios were used to detect other robots and gather data for local centroid computations.

The basic behaviors, each consisting of one or a small set of simple rules, generated robust group behaviors that met the prespecified evaluation criteria. A small subset of the data is shown here, using the Real Time Viewer[1] which allows for displaying and replaying each of the robots runs, for plotting their positions over time, and for displaying each frame and the elapsed time for each experiment. The figures show

[1]Written by Matthew Marjanović.


Figure 1-5: Some of the 20 mobile robots used to validate the group behavior methodologies described in the thesis. These robots demonstrated group avoidance, following, aggregation, dispersion, flocking, foraging, and docking.

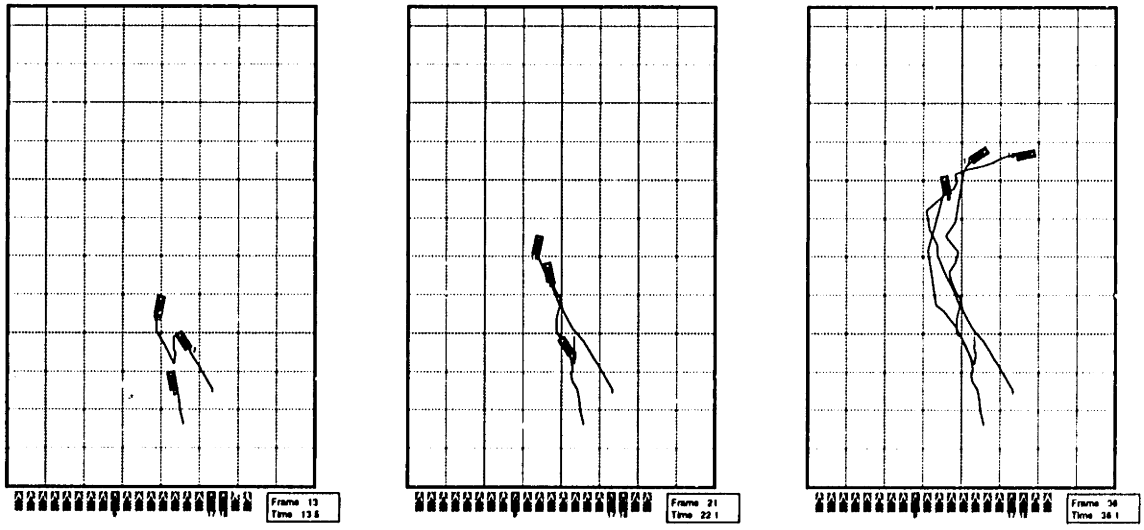


Figure 1-6: Continuous following behavior of 3 robots. The entire time history of the robots' positions is plotted.

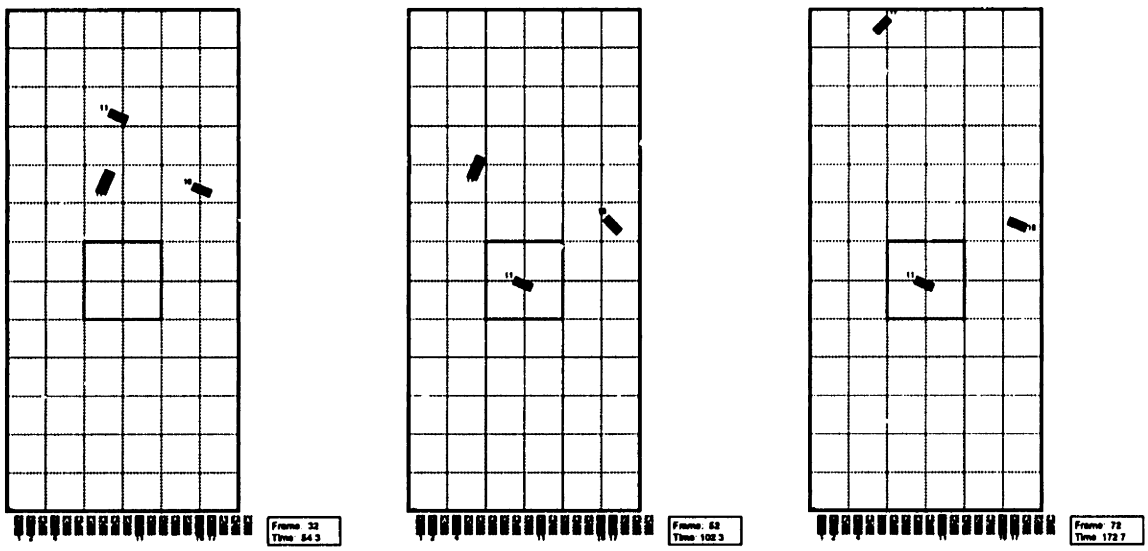


Figure 1-7: Dispersion behaviors of 3 robots.

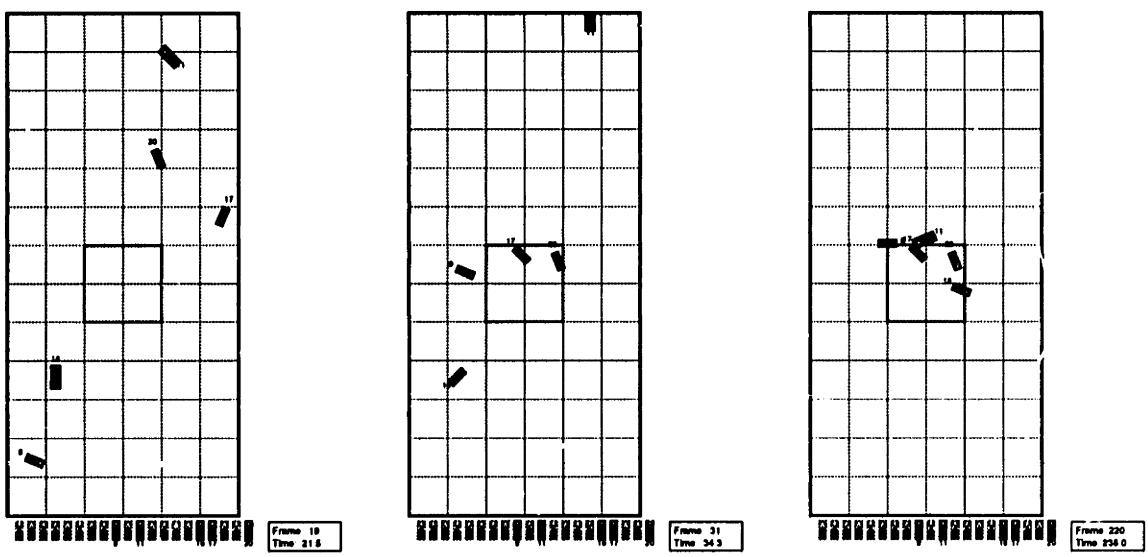


Figure 1-8: Homing behaviors of 5 robots. Four of the five robots reach home quickly and the fifth joints them about 60 second later.

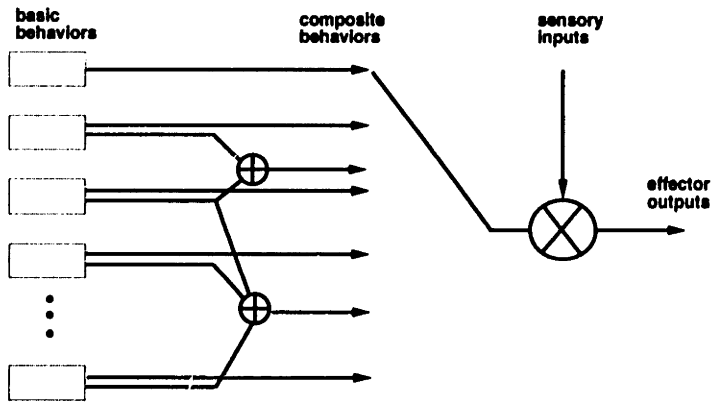



Figure 1-9: The control architecture for generating group behaviors consists of direct and temporal combinations of subsets from a fixed basic behavior set. Direct combinations are marked with $\oplus$, temporal combinations with $\otimes$.

following (figure 1-6), dispersion (figure 1-7), and homing (figure 1-8). More of the data, the algorithms, the specifications, and a detailed evaluation can be found in Chapter 5.

Basic behaviors are intended as building blocks for achieving higher–level goals. They are embedded in an architecture that allows two types of combination of the basic behaviors: direct (by summation) and temporal (by switching) (see figure 1-9). Both types of combination operators were tested empirically. A simple and robust *flocking* behavior was generated by summing the outputs of *avoidance*, *aggregation*, and *homing* (figure 1-10). A more complex behaviors, called *foraging* was implemented by switching between *avoidance*, *dispersion*, *following*, and *homing*, and the addition of puck manipulation and searching. The controller for foraging uses the basic behaviors to generate a global behavior of group collecting of pucks and delivering them to a home location (figure 1-11).

In addition to empirical testing of the behaviors and their combinations, the proposed methodology for generating decentralized group behavior was compared to a centralized, "total knowledge" approach, and found to be only a constant factor slower in the case of dispersion and aggregation algorithms.

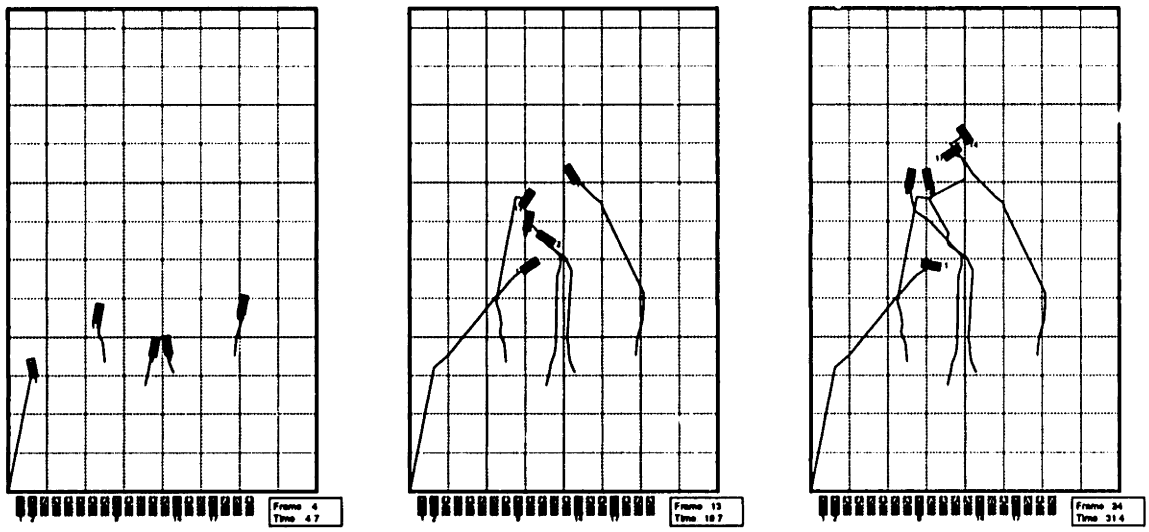


Figure 1-10: Flocking behavior of 5 robots. The robots are started out in a linear dispersed state. They quickly establish a flock and maintain it as the positions of the individual robots within the flock fluctuate over time.

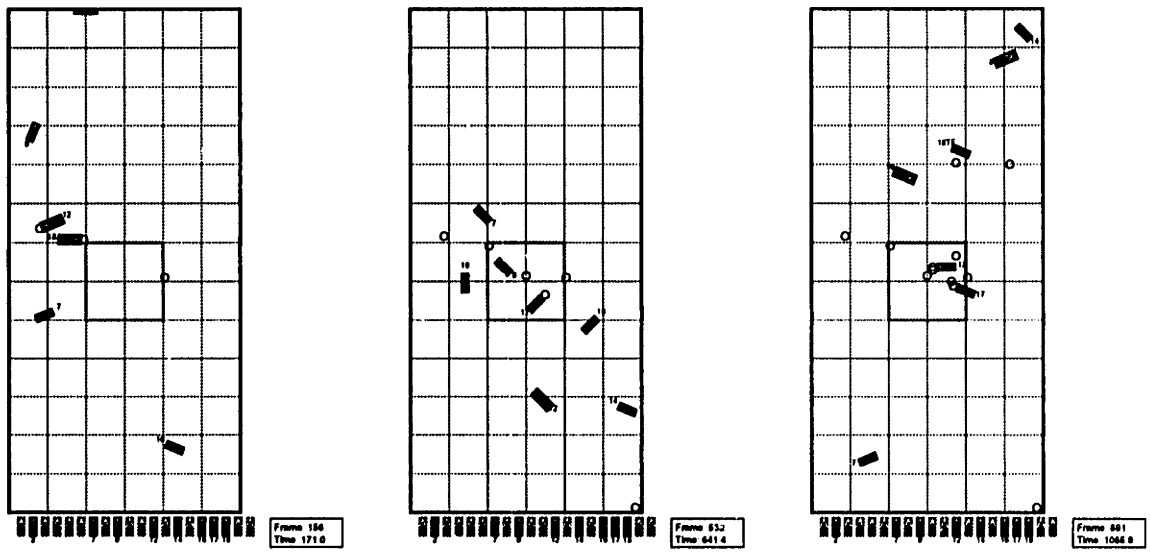


Figure 1-11: An example of the foraging behavior of 6 robots. About eight pucks have been delivered to the home region, marked with a grey box. Two of the robots are dropping off pucks while the others are searching for additional pucks to pick up and deliver home.

## 1.2 Learning in Complex Group Environments

The first part of the thesis introduces basic behaviors as a methodology for structuring simple rules into flexible and effective repertoires of group behavior. It also presents combination operators that allow for constructing and achieving higher-level goals. The second part of the thesis, starting with Chapter 7, describes a methodology for automatically generating higher-level behaviors by having the agents learn through their interactions with the world and other agents, i.e., to improve by **reinforcement learning**.

Reinforcement learning (RL) is a very popular learning methodology that has been successfully applied to a variety of domains. However, most domains that have been considered to date have been modeled as Markovian, such that the agent-environment interaction can be described as a Markov Decision Process (MDP). Unfortunately, that assumption does not directly apply to the stochastic, noisy, and uncertain multi-agent environment addressed in this thesis.

The traditional formulation of RL problems as states, actions, and reinforcement required a reformulation in order to be applied to our complex domain. The notion of state as a monolithic descriptor of the agent and the environment did not scale up to the multi-agent domain used here, given the continuous and discrete aspects describing the agent (e.g., velocity, IR sensors, radio data), and the existence of many other agents in the environment. The traditional notion of actions was inappropriate since atomic actions were too low level and had unpredictable and noisy effect to be useful to a learning algorithm. Finally, delayed reinforcement also proved to be inappropriate.

To make learning possible in this domain, a reformulation is proposed that, instead of using actions, learns at the level of *behaviors*. Behaviors are more appropriate since they hide low-level control details, and are thus more general and robust. Furthermore, they fit directly into the described basic behavior framework . If actions

Figure 1-12: The mobile robots used to validate the group behavior and learning methodologies described in this thesis. These robots demonstrated learning to forage by using group avoidance, following, searching, and resting behaviors.

are replaced with behaviors, states can be replaced with *conditions*, the necessary and sufficient subsets of state required for triggering the behavior set. Conditions are many fewer than states, thus greatly diminishing the state space and speeding up any RL algorithm.

In addition to the use of behaviors and conditions, two ways of shaping the reinforcement function are proposed, in order to aid the learner in a nondeterministic, noisy, and dynamic environment. *Heterogeneous reward functions* are introduced, which partitioned the task into subgoals, each of which could provided more immediate reinforcement. Within a single behavior (i.e., .. single goal), *progress estimators* are introduced as functions associated with particular conditions that provided some metric of the learner's performance. Progress estimators, or internal critics, decrease the learner's sensitivity to noise, minimize thrashing among inappropriate behaviors by encouraging exploration, and minimize fortuitous rewards by correlating some domain knowledge about progress with appropriate behaviors the agent has taken in the past. The details of the reformulation are given in Chapter 8.

The proposed formulation was validated on learning to associate the conditions and behaviors for group foraging with a collection of robots. The behaviors in-

cluded the foraging subset of basic behaviors, augmented with grasping, dropping, and searching, as well as with *resting*, a new internally generated behavior.[2] The condition set was reduced to the power set of the following: *have-puck?*, *at-home?*, *night-time?*, and *near-intruder?*.

A new, smaller group of robots with more reliable hardware was used for the learning experiments. In terms of sensors and effectors, the robots were functionally identical to the first set (figure 1-12), and the implemented basic behaviors and combinations were shown to be directly portable to the new robots.

Three learning algorithms were implemented and tested on the foraging task. The first was standard RL Q-learning, while the other two simply summed the reinforcement received over time.

Q-learning received a reward whenever a robot dropped a puck in the home region. The second algorithm was based on the reinforcement received from heterogeneous reward functions based on reaching subgoals including grasping and dropping pucks, and reaching home. The third algorithm used reinforcement both from the heterogeneous reward functions and from two progress estimators: one monitoring progress in getting away from an intruder, and the other monitoring progress toward home. The two progress estimators were found to be necessary and sufficient for making the given learning task possible. The absence of either one disabled the robots from learning the complete policy for foraging, and the two were sufficient for consistent and complete learning performance.

The performance of each of the three algorithms was averaged over 20 trials (figure 1-13). The analysis of the learning performance showed that the parts that were not learned by the first two algorithms relied on the progress estimators and were successfully learned in the third case. Detailed analysis of the results is given in Chapter 9.

[2]Resting is merely homing at appropriate intervals.

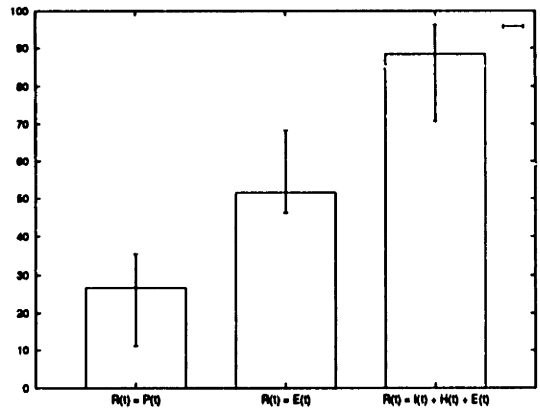



Figure 1-13: The performance of the three reinforcement strategies on learning to forage. The x-axis shows the three reinforcement strategies. The y-axis maps the percent of the correct policy the agents learned, averaged over twenty trials.

## 1.3 Thesis Outline

The preceding sections briefly summarized the contributions of the thesis. This section outlines the structure of the thesis and summarizes each of the chapters.

Chapters 2 through 6 deal with synthesizing and analyzing group behavior. Chapters 7 through 10 address learning in multi–agent domains. All newly introduced, frequently used, or ambiguous terms are defined in Appendix C. Readers interested in moving directly to the details of the basic behavior approach should skip to Chapter 4. Those interested in going directly to the learning part of the thesis should skip to chapter 7. The following are summaries of the chapter contents.

Chapter 2 describes the biological, sociological, and pragmatic motivation behind this work, and presents an overview of related work in robotics, simulation, Artificial Life, and Distributed AI.

Chapter 3 discusses the issues in designing and analyzing the behavior in multi–agent systems. It begins by describing individual–agent control as the foundation for most multi–agent work. It then discusses the difficulties of analyzing such multi–agent behavior and overviews related work in analysis. It then introduces an approach to

estimating interference in such systems and using it as a measure of global behavior. Finally, it describes the role of positive and negative feedback in describing and controlling multi-agent systems.

Chapter 4 introduces the basic behavior approach, defines key terms, and describes and motivates the imposed constraints.

Chapter 5 describes the methodology for selecting basic behaviors, and illustrates the process by defining the basic behaviors for a collection of mobile agents interacting in the plane. The chapter describes the experimental environments, basic behavior specifications and algorithms, and the empirical data and the criteria for evaluating the performance of each of the behaviors as well as their efficacy relative to centralized alternatives.

Chapter 6 describes two methodologies for combining basic behaviors into more complex, higher-level behaviors. The methodologies are demonstrated by combining the basic behaviors described in Chapter 5 to generate two kinds of higher-level behaviors, and demonstrate and evaluate their performance. This chapter also discusses methods for minimizing interference between behaviors within an agent.

Chapter 7 motivates learning in situated agents and reviews the existing learning work based on the type of information being acquired by the agent. It then defines the group learning problem discussed in the thesis as an instance of reinforcement learning (RL) and overviews existing RL models and algorithms as applied to the situated agent domain.

Chapter 8 describes a formulation of RL that enables and facilitates learning in multi-agent domains. It introduces the use of behaviors and their conditions in place of actions and states, and describes a method for shaping the learning process through the use of heterogeneous reward functions and progress estimators.

Chapter 9 presents the experimental robot environment and the learning task used to validate the methodologies proposed in Chapter 8. It describes the experimental

design, the three learning algorithms that were implemented and compared, and discusses the results. In conclusion, the chapter addresses the problem of learning social rules and multiple concurrent tasks.

Chapter 10 summarizes the thesis.

# Chapter 2

# Motivation and Related Work

## 2.1 Biological and Sociological Motivation

**Why study multiple agents?**

The motivation for this work comes from two quite different but complementary directions: the desire to understand and analyze natural systems and the need to design and synthesize artificial ones. The following observations constitute the natural motivation.

*Intelligence is a social phenomenon.* Most intelligent animals live, obey the rules, and reap the benefits of a society of kin. Societies vary in size and complexity, but have a key common property: they provide and maintain a shared culture.

*Culture is both a result and a cause of intelligent behavior.* Intelligent creatures create and refine social rules in order to perpetuate the society. These rules constitute a culture which is communicated and shared by the society, and has important effects on its individual members.

*Culture enables genetic parsimony.* Social interaction is used to transfer information across generations, though social learning. Thus, less genetic investment is necessary, as fewer abilities need to be innate. Interestingly, as culture adapts, growing complexity of social rules makes increased demands on individual intelligence,

specifically on the ability to absorb and adapt to the culture. Humans are an extreme example of cultural complexity, requiring the longest learning and training developmental period of all animals.

*Culture allows for faster adaptation.* As an alternative to evolution, culture allows for testing and adapting social behaviors at a much shorter time scale. Social interactions can be created and destroyed within a single generation. For example, elephants have been shown to learn to avoid humans even if no harm was inflicted for generations, based on a distant cultural memory of past abuse (Gould 1982).

*Culture allows for Lamarckian evolution.* It enables the direct transfer of learned information to future generations. A single individual's discovery can be adopted by an entire population and passed on. For example, an individual Japanese macaque monkey discovered washing of sweet potatoes. The practice was transmitted culturally through the society and on to later generations (Gould 1982).

*Culture makes up for genetic deficiencies.* Social interactions can compensate for individual limitations, both in terms of physical and cognitive capabilities. For example, pack organization allows animals to attack larger prey. At the cognitive end, nature abounds with examples of group information sharing, including bee dance, warning calls and signals, and pheromones.

In order to be understood, individual intelligence must be observed and analyzed within its social and therefore cultural context. In contrast to traditional AI, which addresses intelligence as an individualistic phenomenon, this work is based on the belief that intelligent behavior is inextricably tied to its cultural context and cannot be understood in isolation. The emphasis is similar to the principles of ethology, the study of animal behavior. Unlike the behaviorist branch of biology, which studies animals in controlled laboratory settings, ethology observes animals in their natural habitats. Analogously, this research attempts to study intelligent behavior in its natural habitat: situated within a culture.

The complexity of culture results from the interactions among individuals. This research will focus on exploring simple social interactions which result in purposive group behaviors. These behaviors, which are the building blocks of culture, will be studied with the goal of:

1. understanding social and group behavior in nature, and
2. developing a methodology for principled design of group behavior in artificial systems.

The study of social agents and culture as a basis and structure of intelligent behavior, is necessarily exploratory. Thus, the part of the thesis that addresses that domain is phenomenological, but hopefully also scientific in its attempt to understand natural phenomena and explain them in principled terms.

## 2.2 Pragmatic Motivation

While nature offers challenges for analysis, engineering demands synthesis. In particular, it strives for efficient, automated, reliable, and repeatable methods of synthesizing useful systems.

Discoveries about systems of multiple interacting agents can be applied to many parallelizable computational problems. The idea of applying multiple computational (or physical) agents to a variety of distributed domains, from terrain exploration and mapping, to fire fighting, harvesting, and micro surgery, has been around for many years. However, in spite of the potentially numerous applications, the distributed, multi-agent approach is an exception rather than the rule in most domains.

Parallel, decentralized, non-hierarchical computation requires a paradigm shift (Resnick 1992). Regardless of the domain of application, this approach raises a number of difficult issues. The particular few that motivate this research and are addressed in this thesis include:

- What common properties and principles of organization are shared among different domains of application of multi–agent systems?

- How do the interactions of the individuals affect the behavior of the group?

- How does the group get the job done?

- How much does each individual need to know about the group, the task, the environment, and the other agents?

This research is aimed at finding common properties across various domains of multi–agent interaction. Identifying these properties allows for classifying group behaviors into common categories and thus simplifies the process of both design and analysis.

The behavior of a group is determined by the interactions between the agents. These interactions are usually local, but have global consequences. As this work is focused on decentralized computational models, the question of global behavior from local interactions is critical.

In a distributed model of control, no central focus of control oversees the progress of the group with respect to the particular task. In addition to the global–from–local problem, which addresses getting the job done by local control, the issues of monitoring progress and correcting errors must also be addressed at the local level.

Given all those demands on the individual agents, what are the limits on the simplicity each agent can have? What information about itself, its neighbors, and the global task does an agent in such a system need to have? How much does it need to communicate with others in order to succeed? The thesis will address these questions by focusing on a particular type of a multi–agent system, synthesizing and analyzing behaviors in its domain, and studying their generality.

## 2.3 Related Group Behavior Work

This thesis focuses on the problems involved in synthesizing and analyzing intelligent group behavior. In particular, the work described here applies to agents that are embodied in physically constrained bodies and situated in physically constrained worlds, inhabited by other agents of the same kind, and dealing with multiple goals ranging from basic survival to accomplishing one or more tasks. The experimental environments in which the work was validated included mobile robots and multi-agent simulations.

Consequently, this work is related to a number of lines of research within and outside of AI, including mobile robotics, intelligent control, simulations of multi-agent systems, distributed artificial intelligence, artificial life, machine learning, ethology, and cognitive science. This section presents an overview of the work in these related fields, with the exception of machine learning, which is covered in the second part of the thesis.

### 2.3.1 Control of Multiple Physical Robots

The last decade has witnessed a shift in the emphasis of robotics in general and mobile robotics in particular away from theoretical problems and toward physical implementations. Most of the work in robotics so far has focused on control of a single agent. Few projects have dealt with control of multiple physical robots. Fukuda, Nadagawa, Kawauchi & Buss (1989) and their subsequent work describe an approach to coordinating multiple homogeneous and heterogeneous mobile robotic units, and demonstrate it on a docking task. Caloud, Choi, Latombe, LePape & Yim (1990), Noreils (1992) and Noreils (1993) remain faithful to the state-based framework, and apply a traditional planner-based control architecture to a box-moving task implemented with two robots in a master-slave configuration. Kube (1992) and Kube & HongZhang (1992) describe a series of simulations of robots performing a collec-

tion of simple behaviors which are being incrementally transferred to physical robots. Barman, Kingdon, Mackworth, Pai, Sahota, Wilkinson & Zhang (1993) report on a preliminary testbed for studying control of multiple robots in a soccer-playing task. Parker (1993*b*) and Parker (1994) describes a behavior-based task-sharing architecture for controlling groups of heterogeneous robots, and demonstrates it on a set of physical robots performing toxic waste cleanup and box pushing. Donald, Jennings & Rus (1993) report on the theoretical grounding for implementing a cooperative manipulation task with a pair of mobile robots.

In these early stages of research with multiple physical robots, approaches have largely fallen along two ends of the cooperation and communication spectrum. One direction includes work on systems designed to be cooperative. In these, the two or more robots are aware of each other's existence, can sense and recognize each other directly or through communication. This type of research explores explicit cooperation, usually through the use of directed communication and is represented by Caloud et al. (1990), Noreils (1992), and Parker (1993*a*).

The other category includes work on implicit cooperation, in which the robots usually do not recognize each other but merely coexist and indirectly cooperate by having identical or at least compatible goals. Such work includes Dallas (1990) and Kube (1992). The work described in this thesis falls nearer this end of the spectrum, but is focused on agents that can discriminate each other from the rest of the world, and use this ability as a basis for social behavior.

### 2.3.2 Simulations of Multiple Agents

The problem of multi-agent control has been treated mostly in simulation and under two major categories: simulations of situated systems and simulations of abstract agents.

Simulations of situated systems involve some degree of faithfulness to the physical

world, at least to the extent of employing the simplest models of sensors, effectors, and physical laws. A number of simulations of behavior–style controlled systems have been implemented. For instance, Steels (1989) describes a simulation of simple robots using the principles of self–organization to perform a gathering task. Brooks, Maes, Matarić & Moore (1990) report on a set of simulations in a similar task domain, with a fully decentralized collection of non–communicating robots. Arkin (1992) describes a schema–based approach to designing simple navigation behaviors, used for programming multiple agents working in a simulated environment with future extensions to physical agents; Arkin, Balch & Nitz (1993) apply the approach to a multi–agent retrieval task. Brock, Montana & Ceranowicz (1992) describe SIMNET simulations of large numbers of tank–like robots performing avoidance and formation following. Kube, Zhang & Wang (1993) propose a behavior–arbitration scheme that will be tested on physical robots.

In contrast to simulations of multiple robots, "swarm intelligence" refers to simulations of abstract agents dealing with more theoretical problems of communication protocols, the design of social rules, and strategies for avoiding conflict and deadlock often in societies with with large numbers of simple agents. Representative work includes Fukuda, Sekiyama, Ueyama & Arai (1993), Dario & Rucci (1993), Dudek, Jenkin, Milios & Wilkes (1993), Huang & Beni (1993), Sandini, Lucarini & Varoli (1993), Kurosu, Furuya & Soeda (1993), Beni & Hackwood (1992), Dario, Ribechini, Genovese & Sandini (1991), and many others. This work is also related to DAI (see below) but in contrast to DAI it deals with agents of comparatively low cognitive complexity.

### 2.3.3 Artificial Life

The field of Artificial Life (Alife) focuses on bottom–up modeling of complex systems. Alife work relevant to this thesis is often labeled "swarm intelligence" and

features simulations of colonies of ant–like agents, as described by Corbara, Drogoul, Fresneau & Lalande (1993), Colorni, Dorigo & Maniezzo (1992), Drogous, Ferber, Corbara & Fresneau (1992), Travers (1988), and many others. Deneubourg, Goss, Franks, Sendova-Franks, Detrain & Chretien (1990), Deneubourg & Goss (1989), and Deneubourg, Goss, Pasteels, Fresneau & Lachaud (1987) have experimented with real and simulated ant colonies and examined the role of simple control rules and limited communication in producing trail formation and task sharing. Deneubourg, Theraulax & Beckers (1992) define some key terms in swarm intelligence and discuss issues of relating local and global behavior of a distributed system. Assad & Packard (1992), Hogeweg & Hesper (1985) and other related work also report on a variety of simulations of simple organisms producing complex behaviors emerging from simple interactions. Schmieder (1993) reports on an experiment in which the amount of "knowledge" agents have about each other is increased and decreased based on local encounters. Werner & Dyer (1990) and MacLennan (1990) describe systems that evolve simple communication strategies. On the more theoretical end, Keshet (1993) describes a model of trail formation that fits biological data.

Work in Artificial Life is related to the work in this thesis in that both are concerned with exploiting the dynamics of local interactions between agents and the world in order to create complex global behaviors. However, work in Alife does not usually concern itself with agents situated in physically realistic worlds. Additionally, it usually deals with much larger populations sizes that the work presented here. Finally, it most commonly employs genetic techniques for evolving the agents' simple control systems.

### 2.3.4 Distributed Artificial Intelligence

Distributed Artificial Intelligence (DAI) is another field that deals with multi–agent interactions (see Gasser & Huhns (1989) for an overview). DAI focuses on negoti-

ation and coordination of multi–agent environments in which agents can vary from knowledge–based systems to sorting algorithms, and approaches can vary from heuristic search to decision theory. In general, DAI deals with cognitively complex agents compared to those considered by the research areas described so far. However, the types of environments it deals with are relatively simple and low complexity in that they feature no noise or uncertainty and can be accurately characterized.

According to Rosenschein (1993), DAI can be divided into two subfields: Distributed Problem Solving (DPS) and Multi–Agent Systems (MAS). DPS concerns itself with centrally designed systems solving global problems and using built–in cooperation strategies. In contrast, MAS work deals with heterogeneous, not necessarily centrally designed agents faced with the goal of utility–maximizing coexistence.

For example, Ephrati (1992) describes a master–slave scenario between two agents with essentially the same goals. Miceli & Cesta (1993) describe an approach to using an estimate of the usefulness of social interactions at the individual agent level in order for agents to select what other agents to interact with. This decision is based on an estimate of possible future payoff in terms of help given the agents' attitudes and skills. Unfortunately, the estimation of dependence relations scales poorly with the size of the group, and as is the case of most DAI work, is best suited for a small number of highly deliberative, non–situated knowledge–based agents. Along similar lines, Kraus (1993) describes negotiations and contracts between selfish agents. Durfee, Lee & Gmytrasiewicz (1993) discuss game–theoretic and AI approaches to deals among rational agents. The paper describes the added advantages of introducing meta–level information for the purposes of delineating knowledge.

Aspects of DAI work are purely theoretical and deal with the difficulty of multi–agent planning and control in abstract environments. For example, Shoham & Tennenholtz (1992) discuss the complexity of automatically deriving social saws for agent groups. They show that the problem is NP–complete but can, under a number of

restrictions, be made polynomial.

Some DAI work draws heavily from less abstract mathematical results in the field of parallel distributed systems. In particular, Huberman (1990) describes the effects of information exchange on the performance of a collection of agents applied to a class of search problems. He also addresses the ubiquity of log-normal distributions of performance found across different domains, and hypothesizes a universal law of distribution for all large systems of interdependent agents using resources allocated based on perceived progress. Clearwater, Huberman & Hogg (1991) present related work on cooperative strategies for solving constraint satisfaction problems.

Decker & Lesser (1993*a*) is a good example of work in distributed problem solving. It addresses the task of fast coordination and reorganization of agents on a distributed sensor network with the goal of increasing system performance and decreasing performance variance. Hogg & Williams (1993) is another good example showing how parallel search performs better with distributed cooperative agents than with independent agents.

DAI and Alife merge in the experimental mathematical field that studies computational ecosystems, simulations of populations of agents with well defined interactions. The research is focused on global effects and the changes in the system as a whole over time. This process of global changes is usually referred to "co-evolution" (Kephart, Hogg & Huberman 1990). Often the systems studied have some similarities to the global effects found in biological ecosystems, but the complex details of biological systems cannot be reasonably addressed. Co-evolution experiments are used to find improved search-based optimization techniques. For example, Hillis (1990) demonstrates how co-evolution can be used to overcome local maxima in evolving optimal sorting algorithms.

## 2.4 Summary

The work in this thesis shares motivations and goals with a number of related fields, but does not cleanly fit into any of them. It deals with attaining and maintaining goals in ways quite different from those employed by traditional Artificial Intelligence. In contrast to traditional robotics, it does not attempt to solve a particular problem but instead proposes a methodology for dealing with synthesis and analysis of group behavior in general. Unlike most of mobile robotics, it is concerned with multiple mobile agents. Different from both types of Distributed AI, it deals with less cognitively complex but embodied agents situated in physically constrained, noisy, uncertain and thus complex worlds. Finally, unlike Artificial Life, it deals not with large populations but with relatively small groups of realistic agents coordinating, cooperating, and learning in real-time rather than at the genetic level.

The next chapter focuses on specific related single-agent and multi-agent work in order to set the stage for the theory and experiments presented in the rest of the thesis.

# Chapter 3

# Multi–Agent Control

This chapter describes the multi–agent control problem by first overviewing approaches to individual agent control, and then discussing their extensions to multiple agents. Many issues in and methods for behavior analysis are presented and their limitations are discussed, in order to set the stage for the new model, presented in the next chapter.

Domains for multi–agent research include a vast array of natural and artificial systems ranging from the brain, to operating systems, bird flocks, and collection of robots. For the purposes of this work, an *agent* is a process capable of perception, computation, and action within its world[1]. A *multi–agent system* consists of two or more such agents.

The problem of *multi–agent control* can be viewed at the individual agent level and the collective level. The two levels are interdependent and the design of one is, or should be, strongly influenced by the other. However, multi–agent control has grown out of individual agent control, and this history is often reflected in the control strategies at the collective level. The next section describes the main approaches to individual agent control and their extensions and applicability to multi–agent domains.

[1]The world may or may not be physical

## 3.1 Individual Agent Control

At one extreme of the agent control spectrum lie traditional *planner-based* strategies that use a centralized world model for verifying sensory information and generating actions in the world (Giralt, Chatila & Vaisset 1983, Chatila & Laumond 1985, Moravec & Cho 1989, Laird & Rosenbloom 1990). The information in the world model is used by the planner to produce the most appropriate sequence of actions for the task at hand. Planner-based approaches have been criticized for scaling poorly with the complexity of the problem and consequently not allowing for reaction in real-time (Brooks 1990*b*, Brooks 1991*c*).

Various attempts at achieving real-time performance have been proposed. Perhaps the most prominent are *purely reactive* approaches which implement the agent's control strategy as a collection of preprogrammed condition-action pairs with minimal state (Brooks & Connell 1986, Agre & Chapman 1987, Connell 1990). These systems maintain no internal models and perform no search, but simply look-up and command the appropriate action for each set of sensor readings. They rely on a direct coupling between sensing and action, and fast feedback from the environment. Purely reactive strategies have proven effective for a variety of problems that can be well defined at design-time, but are inflexible at run-time due to their inability to store information dynamically (Matarić 1992*a*).

Rosenschein & Kaelbling (1986) presented work on situated automata that achieve real-time performance by compiling all of the system's goals and the ways of their achievement into a language that compiles into circuits with constant-time computation properties. Subsequently, Schoppers (1987) proposed a related idea of precompiling the entire control system as a decision graph into a collection of reactive rules ("universal plans") in order to achieve real-time performance.

*Hybrid* architectures attempt a compromise between purely reactive and planner-based approaches, usually by employing a reactive system for low-level control, and

a planner for higher-level decision making. Hybrid systems span a large and diverse body of research. It includes *reactive planning or reactive execution* exemplified by Firby (1987), who introduced RAPs (Reactive Action Packages), higher-level primitives for planning which hide and take care of the details of execution. Similarly, Georgeff & Lansky (1987) proposed PRS (Procedural Reasoning System), an architecture for flexible control rule invocation. Alternative approaches were proposed by Arkin (1989), Payton (1990), and Connell (1991), among others. These systems tend to separate the control system into two or more communicating but otherwise independent parts. In most cases, the low-level reactive process takes care of the immediate safety of the robot, while the higher level uses the planner to select action sequences.

*Behavior-based* approaches are an extension of reactive systems that also fall between the purely reactive and the planner-based extremes. The approach grew out of reactive work of Brooks (1986) and was extended by Maes (1989) and others. Although often confused in the literature, behavior-based strategies are strictly more powerful than purely reactive approaches since they have no fundamental limitations on internal state. While behavior-based systems embody some of the properties of reactive systems, and usually contain reactive components, their computation is not limited to look-up. Other than centralized reasoning engine and representation, these systems may use different forms of distributed internal representations and perform distributed computations on them in order to decide what effector action to take (Matarić 1992*a*).

The choice of control architecture is largely based on individual biases rather than on objective criteria of evaluation. Indeed, a comparative classification of above methodologies based on domains of applicability has not yet been undertaken.

## 3.2 Multi–Agent Control

Having overviewed single–agent control, this section discusses how the described approaches scale to multi–agent problems.

Extending the planing paradigm[2] from single–agent to multi–agent domains requires expanding the global state space to include the state of each of the agents. Such a global state space is, in the worst case, exponential in the number of agents. Specifically, the size of the global state space $G$ is: $|G| = s^a$ where $s$ is the size of the state space of each agent, here assumed to be equal for all agents, or equivalently the maximum for all agents, and $a$ is the number of agents. Exponential growth of the state space makes the problem of global on–line planning intractable for all but the smallest group sizes, unless control is synchronized and has SIMD form[3]. Further, since global planning requires communication between the agents and the controller, the bandwidth also grows with the number of agents. Additionally, the uncertainty in perceiving state grows with the increased complexity of the environment. Consequently, global planner–based approaches to control do not appear well suited for problems involving multiple agents acting in real–time based on uncertain sensory information.

At the other end of the control spectrum, extending the reactive and behavior–based approaches to multi–agent domain results in completely distributed systems with no centralized controller. The systems are identical at the local and global levels: at the global level the systems are a collection of reactive agents each executing task–related rules relying only on local sensing and communication. Since all control in such distributed systems is local, it scales well with the number of agents, does not require global communication, and is more robust to sensor and effector errors. However,

[2]The planning paradigm includes includes traditional and hybrid systems. In terms of multi–agent extensions, hybrid systems fit into the planner–based category since their collective behavior is generally a result of a plan produced by a global controller.

[3]All agents perform the same behavior at the same time.

global consequences of local interactions between agents are difficult to predict.

The following table summarizes the properties of these two approaches to multi-agent control:

| centralized approaches | distributed approaches |
|---|---|
| can optimize global parameters | can only optimize locally |
| scale poorly | scale well |
| require global sensing | use local sensing |
| require global communication | may not require communication |
| contain a bottleneck | have no bottleneck |
| impose hierarchical control | use flat control |
| may degrade poorly | are usually redundant |

Table 3.1: A comparative summary of typical centralized and distributed approaches.

The work described in this thesis was dually motivated: in part by the disadvantages of the centralized approach and its variations, and in part by the ubiquitous apparent advantages of distributed approaches to group behavior in nature. Consequently, this thesis will focus on fully distributed systems for multi-agent control. The next section describes the key properties of such systems.

## 3.3 Distributed Multi-Agent Control

This thesis will focus on *fully distributed multi-agent systems*, those in which the behavior of each agent is determined by its own control system rather than by an external controller. Such systems are by definition complex, either because they are composed of a large number of elements, or because the inter-element interactions are not simple. Multi-agent systems consisting of several situated agents with uncertain sensors and effectors fit into this category. This section addresses how these properties affect their behavior and its analysis.

### 3.3.1 Behavior Analysis

The exact behavior of an agent situated in a nondeterministic world, subject to real error and noise, and using even the simplest of algorithms, is impossible to predict exactly. By induction, the exact behavior of each part of a multi–agent system of such nature is also unpredictable. However, according to Simon (1969), a system is analyzable, and thus well designed, if it is *decomposable* into non–interacting modules. Thus, minimizing inter–module interactions is considered good engineering and principled AI, and most of traditional Artificial Intelligence relies on this style of top–down modularity. In contrast, nature abounds with complex systems whose global behavior results from precisely the type of interactions that current research methodologies try to avoid. These effects can be found at all scales, from the subatomic (Gutzwiller 1992), to the semantic (Minsky 1986), to the social (Deneubourg et al. 1990).

Global behavior of complex multi–agent systems is determined by the local interactions between individuals. These interactions merit careful study in order to understand the global behavior. In natural systems, such interactions result in the evolution of complex and stable behaviors that are difficult to analyze using traditional, top–down approaches. In order to reach that level of complexity synthetically, such behaviors must be generated through a similar, interaction–driven, incrementally refined process.

Precise analysis and prediction of the behavior of a single situated agent, specifically, a mobile robot in the physical world, is an unsolved problem in robotics and AI. Previous work has shown that synthesis and analysis of correct plans can be intractable even in highly constrained domains (Lozano-Pérez, Mason & Taylor 1984, Canny 1988, Erdmann 1989). Physical environments pose a greater challenge as they usually do not contain the structure, determinism, and thus predictability usually required for formal analysis (Brooks 1991*c*, Brooks 1991*b*). Predicting the

behavior of a multi-agent system is more complex than the single-agent case. The difficulty in analyzing comes from two properties intrinsic to complex systems:

1. the actions of an agent depend on the states/actions of other agents,

2. the behavior of the system as a whole is determined by the interactions between the agents rather than by individual behavior.

In general, no mathematical tools are available for predicting the behavior of a system with several, but not numerous, relatively complex interacting components, namely a collection of situated agents. In contrast to physical particle systems, which consist of large numbers of simple elements, multi-agent systems in nature and AI are defined by comparatively small groups of much more complex agents. Statistical methods used for analyzing particle systems do not directly apply as they require minimal interactions between the components (Weisbuch 1991). While systems with large numbers of simple and simply interacting components can be analyzed this way (Wiggins 1990), no tools are currently available for systems consisting of comparatively few but more complex components with complex interactions.

Instead of attempting to analyze arbitrary complex behaviors, this work focuses on providing a set of analyzable behavior primitives that can be used for synthesizing and analyzing a particular type of complex multi-agent systems. The primitives provide a kind of a programming language for designing analyzable control programs and an interaction calculus for analyzing the resulting group behaviors.

### 3.3.2 Emergent Behavior

*Emergent behavior* is a popular topic of research in the field of complex systems (see Forrest (1989), Chris G. Langton (1989), Langton (1990), and Steels (1994) for overviews). Such behavior is characterized by the following properties: 1) it is manifested by global states or time-extended patterns that are not explicitly programmed

in but result from local interactions between a system's components, and 2) it is considered interesting by some observer-established metric. Because emergent phenomena are by definition observed at a global level, they depend on the existence of an observer.

Emergent behavior can be observed in any sufficiently complex system, i.e., a system which contains local interactions with temporal and/or spatial consequences. Perhaps because of their pervasiveness, emergent phenomena have been objects of interest, although perhaps not objects of analytical study, for a long time. The property of observer-dependence make emergent phenomena more difficult to study. Kolen & Pollack (1993) eloquently describe why in general the complexity of a physical system is not an intrinsic property but is dependent on the observer, and further why traditional measures of complexity do not apply to physical systems. Subjective evaluation is also discussed by Bonabeau (1993).

Emergent phenomena are appealing to some researchers because they appear to provide something for nothing. These types of systems are referred to as "self-organizing" because of their apparent ability to create order. In reality, the dynamics of such self-organizing systems are carefully crafted (usually by eons of evolution) to produce the end-results. Theoretical analysis of multi-agent systems of the type used in this research is difficult, and, as will be shown, exact prediction of the behavior of such systems is not currently within reach. Consequently, work on situated group behavior can benefit from synthesis and experimentation.

In order to structure and simplify this process of experimental behavior design, this work will provide a set of basic group behaviors and methods for synthesizing them from local rules. These basic behaviors and their combinations are emergent in that they result from the local interactions, but are predictable and well understood.

### 3.3.3 Limits of Analysis

The difficulty in analyzing complex multi-agent systems lies in the level of system description. Descriptions used for control are usually low level, detailed, and continuous. In contrast, planning and analysis are usually done at a high level, often using an abstract, discrete model. A more desirable and manageable level may lie in between those two (figure3.2).

| approach | level of description |
|---|---|
| complex dynamics | microscopic & continuous |
| <?> | macroscopic & quasi-continuous |
| state spaces | macroscopic & discrete |

Table 3.2: A desirable level of system description for control and analysis lies between the commonly employed ends of the spectrum.

In general, this work is concerned with predicting the global behavior of the system rather than the precise behavior of any of its components. At the high level of precision requiring a low level of description, most interactions are chaotic and unpredictable. The goal of analysis is to gain predictive power by modeling the system at the right level. In the case of complex systems, however, it is not possible to determine that level without generating and testing the system itself.

For the case of a fully deterministic agent and world, it is possible, but usually not realistic, to enumerate all trajectories the agent can take in its behavior space. This is equivalent to elaborating the agent's phase space. Early AI methods for proving correctness consisted of showing that, for a given set of possible initial conditions, usually expressed as discrete states, the agent would, through a series of actions, reach the desired terminal (often designed to be goal) state. Search-based methods for plan or action generation are particularly amenable to this type of analysis (Fikes & Nilsson 1971). However, besides the scaling problem, this approach to behavior

analysis fails in more realistic worlds in which both the agent and the environment are not deterministic.

Nondeterministic worlds can be modeled probabilistically (e.g., Doyle & Sacks (1989)) but obtaining appropriate values for the probabilities is in general very difficult since it requires a complete and accurate model of the world. Even small inaccuracies in the values can accrue and result in artifactual dynamics of the system as a whole. Another way of looking at it is that the behavior space is complex enough that even a small amount of error in the estimated trajectory can introduce a perturbation which causes the system to diverge from the actual trajectory. Consequently, most probabilistic models fail to capture the stochastic dynamics of the kinds of complex behavior this work is concerned with.

This need for accurate descri tion is again related to the level of system description. Quantitative analysis is extremely difficult for any but the simplest of deterministic systems. On the surface, this does not appear to be a problem, as most researchers would be satisfied with knowing the system's "global, qualitative behavior." However, such behavior is generally defined in quantitative terms since qualitative analysis requires first a quantitative description which can then be abstracted to a higher level of description. For instance, qualitative descriptions of spatial information are derived from quantitative ones whether it be on macroscopic scale of building maps (e.g., Chatila & Laumond (1985)) or on the microscopic scale of particle interactions (e.g., Abraham & Shaw (1992)).

The only path to a qualitative description of a system is indirect, through abstracting away the details or through clustering analytical, quantitative information. A qualitative description is a collection of non-analytic symbols (i.e., words instead of numbers) with complicated associated semantics. When these semantics are defined, they are either stated in terms of other symbols or eventually grounded in numerical terms.

Given the difficulty of the problem, most analysis approaches to date have been limited to constrained, special case, scenarios. This is not surprising since any general method for analyzing complex systems with nonlinear interacting components is unlikely to be powerful enough to provide useful predictions. The next section overviews a selection of representative approaches to analysis of multi-agent systems that have been considered to date.

### 3.3.4 Related work on analysis

As described earlier, Distributed Artificial Intelligence (DAI) deals with multi-agent negotiations and coordination in a variety of abstract environments. Decker & Lesser (1993*b*) is an example of a DAI approach to modeling a distributed system. It depends on the ability to specify the agents' beliefs, intentions, and their quality and duration. These types of models do not scale well to large groups of agents. Further, in order to apply at all they need to abstract away the "low-level" properties of the system such as the exact noise and error models and side-effect. However, as has been shown in physical systems, these "low-level" properties determine the high-level behavior.

Similarly, Kosoresow (1993) describes a probabilistic method for agent coordination based on Markov processes. This method relies on specifying agents inference mechanisms (as chains), and having agents with compatible and specifiable goals and preferences. This type of approach applies to domains where the problem of resource allocation can be clearly specified. However, the ability to predict agents' behavior in order to assess the resource allocation problem is not achievable in a physical system. If it were, a number of mathematical and game-theoretic paradigms would apply.

The classical robotics field of motion-planning has dealt with the problem of planning for multiple objects. For example, Erdmann & Lozano-Pérez (1987) describe theoretical results on the motion-planning problem for multiple polygonal moving objects. The presented solution searches the two-dimensional representation of space-

time slices to find a safe path. These results depend on having only one object move at a time, a constraint that cannot be easily enforced in situated systems. Furthermore, the proposed strategy is too computationally intensive to be applied for real-time control.

Donald et al. (1993) discuss motion-planning algorithms for coordinated manipulation with different numbers of agents and different amounts of *a priori* knowledge about the object to be moved. The theoretical aspect of the work focuses on computing the information requirements for performing particular robot tasks. The work is directly applicable to manipulation tasks, such as box-pushing, and solutions that can apply one or more cooperating robots as "force-appliers." In contrast, the work in this thesis does not focus on algorithms for explicit cooperation on tasks such as object manipulation, but instead on distributed solutions to problems that do not necessitate cooperation but can benefit from it.

Strategies for proving distributed algorithm correctness are tangentially related to analyzing multi-agent behavior. Lynch & Tuttle (1987), for example, describe such methods for distributed systems with hierarchical components. More closely related is work by Lynch (1993), which uses a simulation method for reasoning about real-time systems modeled as general automata. This work is targeted at proving properties of message-passing protocols, most of which are more constrained and less uncertain than communication among distributed physical agents.

Work on stochastic analysis of qualitative dynamics, such as that by Doyle & Sacks (1989), is appealing for its qualitative nature. However, the proofs depend on the ability to represent the system as a series of transitions in a graph and the system's dynamics as a Markov chain over that graph. The difficulty lies in establishing such a model for a multi-agent system. It is in general difficult to obtain the values for the transition probabilities that capture the complex dynamics of such systems. Simpler models can be constructed but fail to contain enough detail to conserve the dynamics.

Related work on analysis of group behavior has been conducted in branches of biology. For example, Belić, Skarka, Deneubourg & Lax (1986) present a model for honeycomb constructions based on partial differential equations describing the bee density distribution in the hive and their wax distribution behavior. Less structured group behavior, such as exploration and foraging, have also been addressed. For instance, Benhamou & Bovet (1990) describe a probabilistic model for foraging. The work closest to the domains addressed in this thesis is done by Deneubourg, Aron, Goss, Pasteels & Duernick (1986), Deneubourg et al. (1987), Calenbuhr & Deneubourg (1992), etc. The authors propose strategies for describing and analyzing various collective behaviors in ants. Their work is closest in nature to the kind of analysis proposed here as viable for describing group behavior of situated, embodied agents.

Since prediction of group behavior is too difficult from the individual perspective, approaches that focus on describing and analyzing ensemble properties appear better suited for the domains addressed in this work. The next section describes an approach to assessing and predicting global behavior by measuring interference, a local property that has collective consequences.

## 3.4 Interference and Conflict

*Interference* is any influence that opposes or blocks an agents' goal-driven behavior. In societies consisting of agents with identical goals, interference manifests itself as competition for shared resources. In diverse societies, where agents' goals differ, more complex conflicts can arise, including goal clobbering[4], deadlocks, and oscillations.

Two functionally distinct types of interference are relevant to this work: interference caused by multiplicity, which will be called *resource competition*, and interference

[4]The term is used in the same sense as in Sussman & McDermott (1972) and Chapman (1987).

caused by goal-related conflict, which we will call *goal competition.*

Resource competition includes any interference resulting from multiple agents competing for common resources, such as space, information, or objects. This type of interference is the cause of decline in performance in multi-agents systems as more agents are added. It is also the primary impetus for social rules.

Resource competition is caused by physical coexistence among potentially identically behaving agents. In contrast, goal competition arises between agents with different behavior. Such agents may have identical high-level goals (such as a family, for example), but individuals can pursue different and potentially interfering subgoals at any particular instance, i.e., they can be "functionally heterogeneous." Such heterogeneity does not arise in SIMD-style groups[5] in which all agents are executing exactly the same program at each point in time.

Goal competition is studied primarily by the Distributed AI community (Gasser & Huhns 1989). It usually involves predicting other agents' goals and intentions, thus requiring agents to maintain models of each other (e.g., Huber & Durfee (1993) and Miceli & Cesta (1993)). Such prediction abilities require computational resources that do not scale well with increased group sizes[6]. In contrast, in the work discussed here, goal competition, and thus the need for agents to model each other, is minimized by agent homogeneity, and the thesis largely focuses on dealing with direct resource competition.

### 3.4.1 Individual vs. Group Benefit

Social rules attempt to eliminate or at least minimize both resource and goal competition. In particular, their purpose is to direct behavior away from individual greediness

[5]Single instruction multiple data, groups of functionally identical agents.

[6]The problem of maintaining internal models or so called theories of mind is discussed in detail in section 4.3.3.

and toward global efficiency[7]. In general, agents in groups must give up individual optimality in favor of collective efficiency because greedy individualistic strategies perform poorly in group situations since resource competition grows with the size of the group.

Since social rules are designed for optimizing global resources, it is in the interest of each of the individuals to obey them. However, since the connection between individual and collective benefit is rarely direct, societies can harbor deserters who disobey social rules in favor of individual benefit. Game theory offers elaborate studies of the effects of deserters on individual optimality (Axelrod 1984), but domains treated in game theory are much more cleanly constrained than those treated here. In particular, game theory deals with rational agents capable of evaluating the utility of their actions and strategies. In contrast, this work is concerned with situated agent domains where, due to incomplete or nonexistent world models, inconsistent reinforcement, and noise and uncertainty, the agents cannot be assumed to be rational.

Furthermore, the goal of this work is not to devise optimal strategies for a specific group behavior but to provide methodologies for finding efficient approaches to a variety of related problems. Optimality criteria for agents situated in physical worlds and maintaining long-term achievement and maintenance goals are difficult to characterize and even more difficult and largely impossible to achieve. For example, while in game theory interference is a part of a competing agent's predictable strategy, in the embodied multi-agent domain interference is largely a result of direct resource competition, which can be moderated with relatively simple social rules. For example, complex traffic jams can be alleviated through the appropriate use of yielding.

So far in this document interference was viewed as a destructive, and thus undesirable, influence on group behavior. However, in many situations controlled interference can be used to regulate group behaviors, in particular those based on positive

[7]Also referred to, in cultural contexts, as "the common good."

feedback. Appropriate interfering actions can be used to slow down or completely terminate undesirable feedback[8]. Such regulatory feedback is discussed in more detail in the next section, following an approach to estimating interference.

### 3.4.2 Estimating Interference

Understanding interference is an integral part of synthesizing and analyzing group behavior. In synthesis, the task must be distributed over multiple agents in a way that minimizes interference, or the benefits of concurrent execution are lost. In analysis, interference must be taken into account in order to characterize the realistic behavior of a distributed system as well as motivate the existence of social rules and protocols.

Attempting to precisely predict inter-agent interference is equivalent to trying to predict the system's exact behavior. As has been argued about analysis in general, this level of prediction is impossible to reach. This section proposes a qualitative alternative that can be applied to obtain useful estimates.

Agent density is a key parameter for estimating interference since it measures likelihood of interaction. The higher the density the higher the probability that any two agents will interact. Even without evaluating the outcome of interaction, being able to predict its estimated frequency is a useful part of describing the dynamics of a group. For example, the probability of interaction based on density determines how "collectively-conscious" an agent must be, or how much greedy behavior it can get away with. Not surprisingly, particular densities are critical for the nucleation of group behaviors based on positive-feedback.

Density estimation is straight-forward. I define *group density* to be the ratio of the sum of the agents' footprints and the size of the available interaction space. An agent's *footprint* is the sphere of its influence. In the spatial domain, for example, an agent's footprint is based on its geometry, its motion constraints, and its sensor

[8]In many societies, including human, non-conformists either become leaders or social outcasts.

range and configuration. The size of the interaction space is the area of the physical space the agents can inhabit. The same idea applies in more abstract domains as well. In many such domains the interaction space is time, and the agent's footprint is the duration of information exchange. For instance, in a telecommunications domain density can be estimated as the ratio of the duration of all the calls in one day. Highway traffic is another example in which the relevant space of interactions is time. The density can be represented by the ratio of the sum of the agents' footprints and the total surface area of the road.

The density metric allows for computing how much interaction space is necessary for a group to perform any task, and whether a specific amount of interaction space is sufficient. In the spatial domain, for example, using the number and size of the agents is enough to compute the *mean free path* of an agent and use it to estimate how many collisions are expected between agents executing random walks. Similarly, for the telecommunications domain the average uninterrupted call duration relative to the average number of calls per unit time can be computed, which gives an estimate of how much "phone interaction space" is available for the given parameters. Finally, for the highway domain the same computation yields the average length of free speeding[9].

Such an approximate measure of density can then be used to estimate how much interaction space, on average, is required for the system, even before the specifics of the task are considered. By bringing the constraints of the task into the computation, the expected interference over the duration of the task can also be estimated. For most tasks, interference will vary depending on the fluctuations of the density over the lifetime of the task. This temporal density distribution demonstrates which parts of the task require social rules. Although the exact computation of relevant density is dependent on the particular domain and task, a rough approximation provides useful metrics for estimating the dynamics of the group and the time-evolution of the

[9]This model does not include stationary police cars.

system's behavior as a whole.

Agent density is a particularly important parameter in group behaviors that rely on positive and negative feedback. The next section describes such behaviors and their relevance to this thesis.

## 3.5 Feedback and Group Behavior

Situated behavior is based on the interaction with, and thus feedback from, the environment and other agents. Both negative and positive feedback are relevant. Negative feedback has a regulatory effect, damping the system's response to external influences, while positive feedback has an amplifying effect, increasing the system's response. In multi-agent environments, negative feedback controls the local structure among the agents while positive feedback recruits more agents into the structure.

Behaviors based on positive feedback usually require a critical mass to initiate and accelerate with increased group size. All of these behaviors are variations on *recruitment*; the more agents that are engaged in an activity, the more agents that join in. Such behaviors are usually unstable as they are sensitive to the particular conditions and resources required to maintain the recruitment effect. Numerous natural group behaviors are based on positive feedback: lynch mobs, public polls, popularity ratings, and traffic jams all fit in this category. Ant trail formation and worker recruitment in both ants and bees are also instances of positive feedback (Camazine 1993, Deneubourg et al. 1990, Deneubourg et al. 1986).

The behavior of a group of interacting agents is a dynamical system. Most such systems have a few stable states and some transient ones. Negative and positive feedback behaviors correspond to the different regimes of the system. For example, Miramontes, Sole & Goodwin (1993) present a framework for describing ant behavior as individually chaotic but collectively stable and periodic. Spatial distributions of activity display similar symmetries. Brown & McBurnett (1993) describe a model

of a simple political voting system which displays a large array of group behaviors based on simple local feedback (i.e., recruitment or persuasion) mechanisms. The system has only two stable states: a homogeneous distribution and a collection of invariant blocks. Intuitively, this is an analogy of an equal power distribution, in which any imbalance results in a transient instability. Camazine (1993) shows an analogous pattern for honey-comb population, nectar foraging, and brood sorting. DeAngelis, Post & Travis (1986) demonstrates how most aggregation-type behaviors can be shown to fit this pattern.

Another form of common feedback-based behavior involves the synchronization of rhythmic patters of activity. For example, Meier-Koll & Bohl (1993) describe the synchronization of circadian rhythms of in human and animal subjects and models them as a collection of coupled oscillators. Analogous effects are commonly observed in hormonal cycles (Vander, Sherman & Luciano 1980). In such systems, the synchronized state is a stable behavior, as is the evenly dispersed equal-power state, while all other states are transient. Sismondo (1990) reports on similar synchronization behavior in insect rhythmic signaling and proposes a similar model of the behavior.

The work in this thesis is based on the notion of regulatory negative-feedback group behaviors. The next chapter will introduce a methodology that uses such behaviors as a basis of synthesis and analysis of higher-level interactions in multi-agent systems. The behaviors have a regulatory purpose of maintaining goal-driven behavior through local minimization of interference. Furthermore, these behaviors are not sensitive to or dependent on external resources and are thus more stable.

## 3.6 Summary

This chapter has described the individual and multi-agent control problem. It also focused on the complexities introduced by the coexistence of multiple situated agents. It was proposed that precise predictive analysis of distributed, decentralized, embod-

ied multi-agent systems of the type studied in this thesis is currently out of reach, but that methods for predicting the ensemble behavior of a group are available. The issues of interference and conflict were discussed as possible means of qualitative analysis. The next chapter outlines an approach to analysis and synthesis of group behaviors through the use of building blocks at the appropriate level of description.

# Chapter 4

# The Basic Behavior Approach

One of the hardest problems in AI is finding the right level of system description for effective control, learning, modeling, and analysis. This thesis proposes a particular description level, instantiated in so-called **basic behaviors**, building blocks for synthesizing and analyzing complex group behavior in multi-agent systems.

Basic behaviors are intended as a tool for describing, specifying, and predicting group behavior. By properly selecting such behaviors one can generate repeatable and predictable group behavior. Furthermore, one can apply simple compositional operators to generate a large repertoire of higher-level group behaviors from the basic set.

The idea behind basic behaviors is general, but particular sets of such behaviors are domain-specific. In order to demonstrate the methodology, basic behaviors for group interaction in the spatial domain will be derived, combined, analyzed theoretically, and tested empirically.

The following table summarizes the research goals, the approach, and the experimental methodology:

| | |
|---|---|
| **problem** | synthesis and analysis of intelligent group behavior<br>in order to understand the phenomenon (science)<br>and apply it (engineering) |
| **assertion** | complex group behavior results from<br>local interactions based on simple rules |
| **approach** | propose basic behaviors for structuring<br>such simple rules |
| **validation** | implement robot group behaviors using<br>a basic behavior set and combinations |

Table 4.1: A summary of the group behavior problem being addressed in the thesis, and the structure of the proposed solution.

## 4.1 Defining Behavior

The notion of "behavior" is the main building block of this work. In the last few years the use of behaviors has been popularized in the AI, control, and learning communities. Approaches labeled "behavior-based AI" and "behavior-based control" are becoming mainstream, but behavior is yet to be cleanly defined and circumscribed.

I define *behavior* to be a control law for reaching and/or maintaining a particular goal. For example, in the robot domain, *following* is a control law that takes inputs from an agent's sensors and uses them to generate actions which will keep the agent moving within a fixed region behind another moving object.

The above definition of behavior specifies that a behavior is a type of an operator that guarantees a particular goal. In order to serve as general building blocks, basic behaviors must be capable of dealing with both attaining and maintaining goals. *Goals of attainment* are terminal states; having reached a goal, the agent is finished with all related subgoals. Such goals include reaching a home region and picking up an object. *Maintenance goals* persist in time, and are not always representable

with terminal states, but rather with dynamic equilibria that must be maintained. Examples include avoiding obstacles and minimizing interference. Maintenance goals can usually be expressed as sequences of achievement goals but may require fine granularity of description. Situated agents generally have multiple concurrent goals, including at least one goal of attainment, and one or more goals of maintenance.

This thesis will attempt to show that behaviors are a natural, convenient, and efficient abstraction for control, planning, and learning for situated agents. Behaviors hide the low-level details of control that deal with precise control parameters. They allow for specifying robot tasks and goals in terms of higher-level primitives that cut down on the state space and are more intuitive for the user. Finally, they are a good basis for learning in noisy and uncertain situated domains.

The next two chapters will focus on designing behaviors for controlling groups of agents, and combining such behaviors into higher-level aggregates. Chapters 7, 8, and 9 will focus on learning to select among such behaviors in order to automatically produce coherent and efficient higher-level composite behaviors.

## 4.2 Defining Interaction

Interaction is another foundational concept in this work. Typically, interaction is viewed as any influence that affects an agent's behavior. By this definition, an agent interacts with everything it can sense or be affected by, since all of its external and internal state can have an impact on its actions.

This work is largely concerned with the interaction that takes place between agents. Thus I propose a stricter definition: *interaction is mutual influence on behavior.* Consequently, objects in the world do not interact with agents, although they may affect their behavior. The presence of an object affects the agent, but the agent does not affect the object since objects, by definition, do not behave, only agents do.

## 4.3 Domain Description

Having defined the key concepts of the thesis, behavior and interaction, we turn to the specification of the domain being addressed.

In order to focus and constrain the study of group behavior, this work focuses on interactions among agents situated in physical worlds. Some key constraints were imposed on the experimental domain in order to structure the exploration while still providing sufficient variety of behaviors to study. The following are the key constraining properties:

- Agents are homogeneous.
- Agents do not use explicit models of each other.
- Agents do not use directed communication or explicit cooperation.

The reasons for and implications of each of the constraints are described and discussed in the following sections.

### 4.3.1 Implications of Homogeneity

This work focuses on groups of agents that are *homogeneous* in that they are situated in the same world and have the same goal structure.[1] Homogeneous agents will also be referred to as *similar*, as distinct from *identical*, a property that can be ascribed to SIMD-style agents. Homogeneity has some important implications.

#### Predictability

The fact that all agents are similar makes their behavior *predictable* to one another in that they do not require internal explicit models of each other. This predictability

[1] Furthermore, the agents in this work are embodied with similar dynamics. While the dynamics of simulated agents can be made identical, those of physical robots often vary enough to significantly effect their group behavior. The section on hardware limitations explains this in detail. The terms homogeneous and similar will be used interchangeably.

can be used explicitly, allowing agents to infer other agents' actions and use that information to make individual decisions, or implicitly, to simplify the control of each individual. This work focuses on the latter approach. For example, identical control laws can take advantage of inherent symmetry, in spatial as well as in more abstract domains.

Homogeneity eliminates goal-related conflicts and resulting strategies such as deserting and cheating. Furthermore, homogeneity allows for leaving much of the information about the world implicit. Although the agents in this work do not use explicit expectations about other agents' behavior, their decision process implicitly takes that information into account.

Given their similarity, agents do not need identities and thus do not require abilities for identification. This presents a significant cognitive savings. As homogeneity and similarity greatly reduce individual cognitive requirements, they can be used for simplifying the synthesis and understanding of group behavior.

Finally, homogeneity can result in increased global robustness through redundancy. Failure of any subset of agents should not seriously affect the system, since the agents are similar and thus interchangeable, and no particular agent or group of agents is critical for the accomplishment of the task. To preserve robustness, no specific roles, such as leaders and followers are designated *a priori*. However, temporally (and spatially) local, replaceable leaders may emerge in various situations.

### 4.3.2 A Necessary Condition: Recognition of Kin

Taking advantage of homogeneity depends on a critical property: the agents must be able to recognize other similar agents. The ability to distinguish the agents with whom one is interacting from everything else in the environment is a *necessary condition* for intelligent interaction and group behavior. This ability is innate and ubiquitous in nature, and enables almost all creatures to distinguish others of their own kind, and

more specifically to recognize kin from others (McFarland 1985, McFarland 1987).

It is important to note that species and kin recognition need not be explicit, i.e., the agent need not "know" or "be aware" that the other agent being recognized is kin, as long as its response to it is kin-specific. For example, slime mold bases its behavior on the concentration of slime produced by its kin. It cannot be said that it actively "recognizes" kin but it does act in species-specific ways which result in complex group behavior such as the construction of multi-cellular organisms (Kessin & Campagne 1992). Similarly, ants cannot be presumed to "know" that pheromones they sense are produced by their conspecifics. However, the appropriate responses to those pheromones result in the formation of trails and other complex structures (Franks 1989).

Besides being biologically inspired, the ability to recognize kin is pragmatic as it allows even the simplest of rules to produce purposive collective behavior, as will be demonstrated in the following sections.

### 4.3.3 Mental Models and Theory of Mind

A dominant school of thought in cognitive psychology and AI is based on the premise that social interaction requires a *theory of mind.* Namely, in order to engage in social discourse, agents need to have models of, understand the intentions of, and maintain beliefs about each other. Indeed, an entire field of theory of the mind rests on the necessity of inferring the internal workings of the mind of the agent(s) with whom one is interacting (Read & Miller 1993).

Maintaining a theory of mind is a complex task and requires a high computational and cognitive overhead (Gasser & Huhns 1989, Rosenschein & Genesereth 1985, Axelrod 1984). Further, controversy surrounds its necessity, as work in both developmental psychology and ethology indicates that theory of mind is not necessary for a large repertoire of complex social interactions (Tomasello, Kruger & Rather 1992, Cheney

& Seyfarth 1990, McFarland 1987, Gould 1982, Rosenthal & Zimmerman 1978).

Research in developmental psychology has shown that young children engage in various forms of social interaction even before attaining the sense of self-awareness, a necessary component of constructing a theory of mind. Prior to this stage, occurring around the age of two, children are incapable of separating the internal and external perception of the world (Piaget 1962, Bandura & Walters 1963, Bandura 1971). Even after achieving self-awareness, as determined with the typical dot-and-mirror test (Asendorpf & Baudonniere 1993), around the age of two, children require a number of years before reaching the adult ability to form theories of mind (Bandura 1977, Abravanel & Gingold 1985).

Much research has been aimed at testing whether primates have theories of mind. It has recently been demonstrated that certain species of monkeys, while involved in complex social and cooperative interactions, apparently do not form theories of mind at all (Cheney & Seyfarth 1990, Cheney & Seyfarth 1991). In contrast, chimps appear to have more complex abilities and are indeed able to infer goals of their conspecifics (Cheney & Seyfarth 1990, McFarland 1987). How the internal models are represented and whether they are based on explicit or internal representations, remains open for further study (Gomez 1991).

## An Alternative to the Theory of Mind

Exploring the existence and limits of theory of mind in biology is difficult. The type and amount of knowledge and representation that an animal brings to bear in its social interactions is impossible to circumscribe. In an ideal scenario the experimenter would be able to control for the type and amount of this knowledge and test the resulting behavior, in order to determine what is necessary and what is not.

Computational and robot experiments allow us to do just that. The agents being experimented with are much simpler than those in nature, but it is exactly this

simplicity that allows us to focus on the specific question of internal social models. In order to study the necessity of theory of mind, this work started from the bottom up, by exploring agents which had none at all.

This work studies group behaviors resulting from the simplest interactions among the simplest of agents. The agents have no explicit models of each other, expectations, or intentions. The goal of this approach is to demonstrate what types of complex interactions can be achieved with such simple basic abilities. The results demonstrate that, particularly in homogeneous groups, significant amount of information about an individual's goals is reflected in the observable external state and behavior, and can be obtained with no direct communication. The next section describes the communication constraints.

### 4.3.4 Communication and Cooperation

Communication and cooperation have become popular topics in both abstract and applied multi-agent work (for example see Dudek et al. (1993), Altenburg & Pavicic (1993), and others). Communication is the most common means of interaction among intelligent agents. Any observable behavior and its consequences can be interpreted as a form of communication so for purposes of clarity, some definitions are introduced that will apply to the rest of the thesis.

*Direct communication* is a purely communicative act, one with the sole purpose of transmitting information, such as a speech act, or a transmission of a radio message. Even more specifically, *directed communication* is direct communication aimed at a particular receiver. Directed communication can be one-to-one or one-to-many, but in both cases the receivers are identified.

In contrast, *indirect communication* is based on the observed behavior, not communication, of other agents, and its effects on the environment. This type of communication is referred to as *stigmergic* in biological literature, where it refers to com-

munication based on modifications of the environment rather than direct message passing.

Both direct and indirect communication are practiced by most species in nature. For example, bees use signals, such as the waggle dance, with the sole purpose of transmitting information and recruiting. In contrast, they also use cues, such as the direction of their flight, which transmit hive information as a by-product of their other behaviors (Seeley 1989).

*Cooperation* is a form of interaction, usually based on communication. Certain types of cooperative behavior depend on directed communication. Specifically, any cooperative behaviors that require negotiation between agents depend on directed communication in order to assign particular tasks to the participants.

Analogously to communication, *explicit cooperation* is defined as a set of interactions which involve exchanging information or performing actions in order to benefit another agent. In contrast, *implicit cooperation* consists of actions that are a part of the agent's own goal-achieving behavior repertoire, but have effects in the world that help other agents achieve their goals.

Having defined precise terminology, the communication and the resulting cooperation constraints imposed on the experimental domain can now be described. In order to study the role of communication in a controlled fashion, and to explore how much communication is needed for the group behaviors described here, a minimalist approach was chosen.

No directed, one-to-one communication between the agents was used in any of the experiments. Indirect communication was based on sensing the external state of neighboring agents, as well as sensing their density, and the effects of their actions. Direct communication was undirected, and limited to local broadcast: agents could transmit messages that could be received by others near by. However, the agents did not have the ability to choose the receivers of their messages, and thus to engage in

directed communication.

The undirected communication constraint affects the kinds of communication that can be implemented or can emerge in a multi-agent system. This work focused on implicit cooperation that emerged without explicit task sharing. For an alternative perspective, see Parker (1994).

## 4.4 Summary

This chapter has described the domain constraints that were imposed in order to structure the study of group behavior. This work in the thesis is focused on homogeneous agents using no explicit world models, non-directed communication, and implicit cooperation. All of these constraints were chosen in order to approach the group behavior problem bottom-up and incrementally. This work is concerned with testing the limits of minimal internal modeling and communication in order to find when such simple abilities are sufficient and when more complex representation and communication abilities are necessary.

# Chapter 5

# Selecting and Evaluating Basic Behaviors

This chapter describes how **basic behaviors**, building blocks for control, planning, and learning, are selected, specified, implemented, and evaluated.

The idea of basic behaviors is general: they are the intended as primitives for structuring, synthesizing, and analyzing system behavior. Basic behaviors include natural modes of operation, dynamic attractors, or states of equilibria, and various other terms used to describe stable, repeatable, and primitive behaviors of any system. This work is concerned with finding ways of identifying such behaviors for a specific system, and using them to structure the rest of the system's behavioral repertoire. The power of basic behaviors lies in their individual reliability and in their compositional properties.

This work focuses on basic behaviors for generating intelligent group interactions in multi-agent systems. It is based on the belief that global behavior of such systems results from local interactions, and furthermore, that those interactions are largely governed by simple rules. Basic behaviors present a mechanism for structuring the space of possible local rules into a small basis set.

This chapter will illustrate the process of selecting basic behaviors on concrete ex-

amples of behaviors for a group of agents interacting in physical space. The process of identifying the basic behaviors, formally specifying them, implementing them, testing their properties both theoretically and empirically, and finally combining them, will be carried out. The criteria for selecting basic behaviors for the domain of spatially interacting agents are described first.

## 5.1 Criteria for Selection

*Ensemble, collective or group behavior* is an observer-defined temporal pattern of interactions between multiple agents. Because group behavior is subjectively labeled, of the infinitely many possible such behaviors for a given domain, only a small subset is relevant and an even smaller subset is desirable for achieving the agents' goals.

This work proposes that, for a given domain, a small set of *basic behaviors* can be selected, from which other complex relevant and desirable group behaviors can be generated. Basic behavior should meet the following criteria:

**Necessity:** A behavior within a basic behavior set is necessary if it achieves a goal required for the agent's accomplishment of its task(s), and that goal cannot be achieved with any of the other basic behaviors or their combinations. Thus, a basic behavior cannot be implemented in terms of other behaviors and cannot be reduced to it.

**Sufficiency:** A basic behavior set is sufficient for accomplishing a set of tasks in a given domain if no other behaviors are necessary. The basic behavior set should, under the combination operators, generate all of the desirable higher-level group behaviors.

**Generality:** A behavior is general if it can aid in the achievement of multiple goals.

If such behaviors are designed by hand, as opposed to being observed in an existing system, they should, in addition to the above criteria, also have the following

properties:

1. *Simplicity:* the behavior should be implemented as simply as possible by Occam's Razor,

2. *Locality:* the behavior should be generated by local rules, utilizing locally available sensory information,

3. *Correctness:* within the model in which it is tested, the behavior should provably attain (and in some cases maintain) the goal for which it was intended and within the set of conditions for which it is designed,

4. *Stability:* the behavior should not oscillate under the set of conditions for which it is designed,

5. *Repeatability:* the behavior should perform according to specification in each trial under reasonable conditions and error margins,

6. *Robustness:* the performance of the behavior should not degrade significantly in the presence of specified bounds of sensory and effector error and noise,

7. *Scalability:* the behavior should scale well with increased and decreased group size.

There is no fixed metric for selecting an "optimal" set of behaviors and this work makes no attempt to devise such criteria in any formal sense. Furthermore, this work does not provide theoretical proofs of correctness of the algorithms for the presented behaviors. The proofs are omitted because, while they may be computable for a simple model of the agents and the environment, they become prohibitively difficult for increasingly more realistic models that included sensors, effectors, and dynamics. As an alternative to simplified modeled environments, the behaviors were tested in the fully complex worlds with all of the error, noise, and uncertainly. In order to

make the evaluation more complete, various initial conditions and group sizes were tested, and a large amount of data was obtained for analysis. Behavior evaluation is described in detail in section 5.6.

The next section illustrates the process selecting basic behaviors for the domain of planar mobile agents.

### 5.1.1 Basic Behaviors for Movement in the Plane

The experimental work in this thesis is focused on interactions among mobile agents in two-dimensional space. This domain is realistic in that it has the desired complexity properties: the number of possible collective behaviors is unbounded. Fortunately, the infinite space of possible spatial and temporal patterns can be classified into categories, and thus effectively viewed from a lower level of resolution. The categories are based on task and domain-specific criteria which allow for selecting the (comparatively) few relevant categories and eliminating the rest from consideration. The proposed basic behaviors impose such categories; they define a group behavior, without specifying particular rules that implement it.

Group behaviors in the spatial domain are spatio-temporal patterns of agents' activity. Certain purely spatial fixed organizations of agents are relevant, as are certain spatio-temporal patterns. Purely spatial fixed organizations of agents correspond to goals of attainment while spatio-temporal patterns correspond to goals of maintenance.

In the process of selecting basic behaviors, the designer attempts to decide what behavior set will suffice for a large repertoire of goals. While the dynamical properties of the system provide bottom-up constraints, the goals provide top-down structure. Both of these influences guide the behavior selection process. Energy minimization is a universal goal of powered physical systems. In the planar motion domain this goal translates into minimization of non-goal-driven motion. Such motion is either

| | |
|---|---|
| **Avoidance** | the ability of a group of agents to move around while avoiding collisions with each other and other obstacles. Here, the homogeneous nature of the agents can be used for inter–agent collision avoidance. Thus, two distinct strategies can be devised; one for other agents of the same kind, and another for everything else. |
| **Following** | the ability of two or more agents to move while staying behind each other. |
| **Dispersion** | the ability of a group of agents to spread out over an area in order to establish and maintain some predetermined minimum separation. |
| **Aggregation** | the ability of a group of agents to gather in order to establish and maintain some predetermined maximum separation. This behavior can be implemented as an inverse of dispersion. |
| **Homing** | the ability to reach a goal region or location. |

Table 5.1: A basic behavior set for the spatial domain, intended to cover a variety of spatial interactions and tasks for a group of mobile agents.

generated by poor behavior design, or by interference between agents. Thus, minimizing goal–driven interference means maximizing goal–driven behavior and minimizing unnecessary motion.

Minimizing interference translates directly into an achievement goal of immediate avoidance and a maintenance goal of moving about without collisions. Avoidance in groups can be achieved by dispersion, a behavior that reduces interference locally. It can also serve to minimize interference in classes of tasks that require even space coverage, such as those involving searching and exploration.

In contrast to various goals that minimize interaction by decreasing physical proximity, many goals involve exchange of resources through physical proximity. Consequently, aggregation is a useful primitive. Maintaining aggregation can be translated into following, an interference–minimizing behavior that imposes structured motion

constraints on the group.

The above list constitutes a set of basic behaviors for a flexible repertoire of spatial group interactions, as illustrated in table 5.1. Biology offers numerous justifications for these behaviors. Avoidance is a survival instinct so ubiquitous it obviates discussion. Following, often innate, is seen in numerous species (McFarland 1985). Dispersion is commonplace as well. DeScnutter & Nuyts (1993) show elegant evidence of gulls dynamically rearranging their positions in a field to maintain a fixed distance from each other. Camazine (1993) demonstrates similar gull behavior on a ledge. People maintain similar arrangements in enclosed spaces (Gleitman 1981). Similarly, Floreano (1993) demonstrates that evolved systems of ants use dispersion consistently. Aggregation, as a protective and resource-pooling and sharing behavior, is found in species ranging from the slime mold (Kessin & Campagne 1992) to social animals (McFarland 1987). The combination of *dispersion* and *aggregation* is an effective tool for regulating density. Density regulation is a ubiquitous and generically useful behavior for collecting and distributing objects and information. For instance, army ants stabilize the temperature of their bivouac by aggregating and dispersing according to the local temperature gradient (Franks 1989). Temperature regulation is just one of the many side-effects of density regulation. Finally, homing is a basis of all navigation and is manifested by all mobile species (for biological data on pigeons, bees, rats, ants, salmon, and many others see Gould (1987), Muller & Wehner (1988), Waterman (1989), Foster, Castro & McNaughton (1989), and Matarić (1990*b*)).

Besides the described behavior set, numerous other useful group behaviors exist. For example, biology also suggest surrounding and herding as frequent patterns of group movement, related to a higher level achievement goal, such as capture or migration (McFarland 1987). These and other behaviors can be generated by combining the basic primitives, as will be described and demonstrated in the next chapter. The reminder of this chapter is dedicated to presenting the experimental environment and

data from implementing basic behaviors for the spatial domain.

## 5.2 Basic Behavior Experiments

This chapter covers the experimental part of the group behavior research described in this thesis. The chapter describes the experimental environments, presents the algorithms for implementing the proposed basic behaviors, and evaluates their performance based on a battery of tests and a collection of criteria.

## 5.3 Experimental Environments

Behavior observation is one of the primary methods for validating theories in synthetic AI projects like the one described in this thesis. In order to have conclusive results, it is necessary to try to separate the effects caused by the particular experimental environment from those intrinsic to the theory being tested. In order to get to the heart of group behavior issues rather than the specific dynamics of the test environment, two different environments were used, and the results from the two were compared. The two environments are the Interaction Modeler, and the collection of physical robots.

Another motivation for using both a physical and a modeled environment is the attempt to isolate any observable inconsistencies between the two. In general, it is difficult to determine what features of the real world must be retained in a simulation and what can be abstracted away. By testing systems in the physical world some of the effects that arise as artifacts of simulation can be isolated (Brooks 1991*a*). This is the motivation behind using data from physical robots. By the same token, the current state of the art of physical robot environments imposes many constraints and biases on the types of experiments that can be conducted. Consequently, results from any physical environment must also be validated in an alternative setup. Two

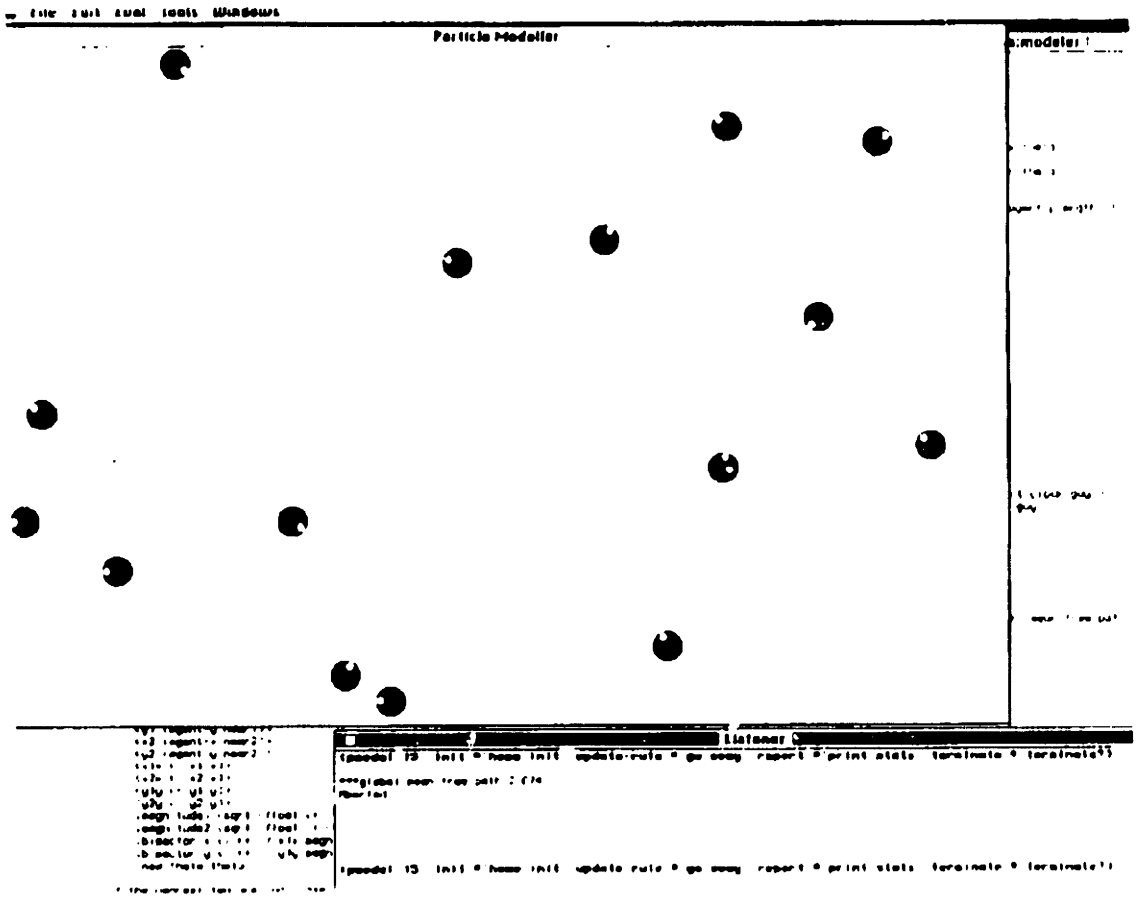


Figure 5-1: The interaction modeler environment. The agents are shown as black circles with white markers indicating their heading. The large rectangle indicates the boundaries of their workspace. The agents are equipped with local sensors and simplified dynamics.

different robot types were experimented on, in order to eliminate system-specific biases.

Since this work is concerned with basic principles of interaction and group behavior rather than a specific domain, it is especially concerned with effects that are common to both the modeled and the physical worlds.

### 5.3.1 The Agent Interaction Modeler

The Interaction Modeler (IM) is a simulator which allows for modeling a simplified

version of the physics of the world and the agent sensors and dynamics (figure 5-1). The Modeler and the control software for the agents are written in Lisp. However, for purposes of realism, the modeler is divided into three distinct components: the simulator, the physics modeler, and the agent specification. The simulator executes the agent specifications and moves the agents according to their control algorithms and their sensory readings. The simulator implements the physics of the sensors, but not the physics of the world. The latter are implemented by the physics modeler that checks the positions and motions computed by the simulator against simplified physical laws, and applies corrections. The IM consists of a simple loop of the simulator and the physics modeler.

The main purpose of the Interaction Modeler is to observe and compare phenomena to those obtained on physical robots. However, the Modeler is also useful for preliminary testing of group behaviors which are then implemented on physical robots. Although it is difficult to directly transfer control strategies from simulations to the physical world, the modeler is useful for eliminating infeasible control strategies at an early stage, as well as for testing vastly larger numbers of agents, performing many more experiments, and varying parameter values.

### 5.3.2 The Mobile Robot Herd

Group behavior experiments are implemented and tested on a collection of 20 physically identical mobile robots affectionately dubbed "The Nerd Herd." Each robot is a 12"-long four-wheeled vehicle, equipped with one piezo-electric bump sensor on each side and two on the rear of the chassis. Each robot has a two-pronged forklift for picking up, carrying, and stacking pucks (figure 5-2). The forklift contains two contact switches, one on each tip of the fork, six infra-red sensors: two pointing forward and used for detecting objects and aligning onto pucks, two break-beam sensors for detecting a puck within the "jaw" and "throat" of the forklift, and

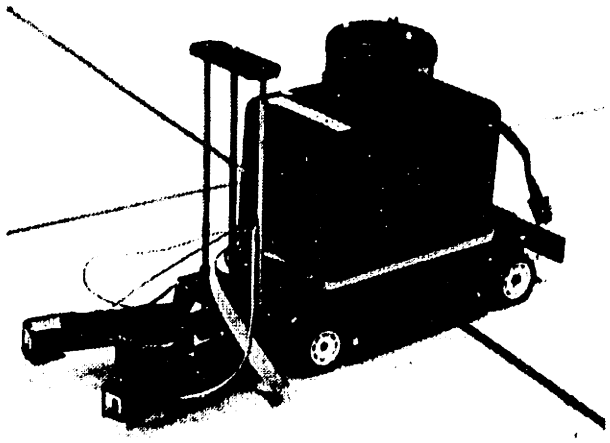


Figure 5-2: Each of the Nerd Herd robots is a 12"-long four-wheeled base equipped with a two-pronged forklift for picking up, carrying, and stacking pucks, and with a radio transmitter and receiver for inter-robot communication and data collection.

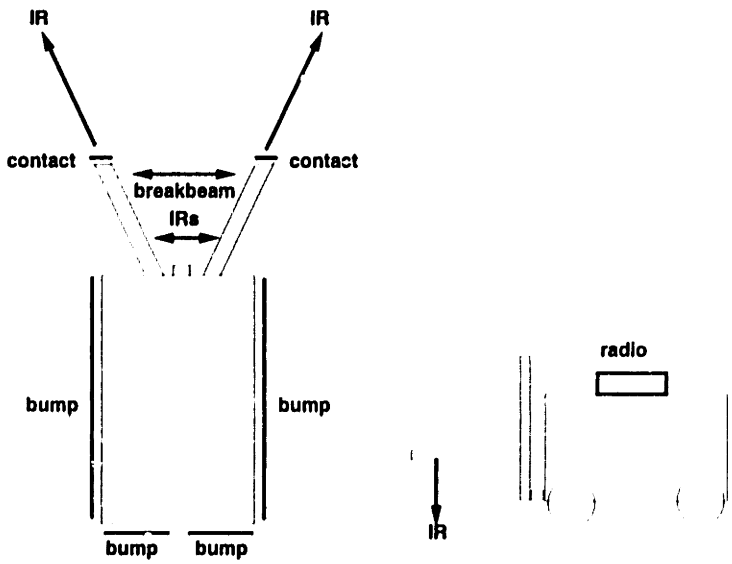



Figure 5-3: Each of the Nerd Herd robots is equipped with contact sensors at the ends of the fork, piezo-electric bump sensors on each side and two on the rear of the chassis, and six infra-red sensors on the fork. Two forward-pointing IRs are located at the ends of the forks, two break beam IRs in the jaw and throat of the fork, and two down-pointing IR for stacking pucks in the middle of each of the fork arms.

two down-pointing sensors for aligning the fork over a stack of pucks and stacking (figure 5-3). The pucks are special-purpose light ferrous metal foam-filled disks, 1.5 inches diameter and between 1.5 and 2.0 inches in height. They are sized to fit into the unactuated fork and be held by the fork magnet.

The robots are equipped with radio transceivers for broadcasting up to one byte of data per robot per second. The system uses two radio base stations to triangulate the robots' positions. The radio system is used for data gathering and for simulating additional sensors. In particular, radios are used to distinguish robots from other objects in the environment, an ability that cannot be implemented with the on-board IR sensors[1].

The mechanical, communication, and sensory capabilities of the robots allow for exploration of the environment, robot detection, and finding, picking up, and carrying pucks. These basic abilities are used to construct various tasks and experiments, in which the robots are run autonomously, with all of the processing and power on board. The processing is performed by a collection of four Motorola 68HC11 microprocessors. Two of the processors are dedicated to handling radio communication, one is used by the operating system, and one is used as the "brain" of the robot, for down-loading the control system used in the experiments. The control systems are programmed in the Behavior Language, a parallel programming language based on the Subsumption Architecture (Brooks 1990*a*).

### 5.3.3 Hardware Limitations

The types of tasks and experiments that can be implemented on the robots are strongly limited by the various sensory, mechanical, and computational limitations of the hardware. In addition to the expected sensory and mechanical error, the robots

[1]The IRs are all the same frequency and mechanically positioned for obstacle detection rather than communication.

suffer from a number of problems.

The mechanical steering system is extremely inaccurate and cannot guarantee commanded direction. This property results in a slow reaction and recovery time, forcing robot speed to be decreased in order to recover from steering errors.

The position triangulation system works sufficiently well when the robots are within the predetermined range of the base stations. However, the exchange of information between the robots, which nominally ought to take place at 1Hz, suffers from extensive loss of data. Consequently, as much as half of the transmitted data is lost or incorrect, and in spite of filtering the data, the robots must be further slowed down in order to minimize errors due to bad radio data.

The infra-red sensors used for stacking have a long range, and vary in sensitivity. Consequently, not only do different robots have different sensing ranges (which cannot be tuned due to hardware restrictions), but the sensitivity between the two sides of the fork on a single robot varies as well.

Given the above limitations, the robots are best viewed as a testbed for a subset of behaviors studied in this work, but are not intended as the exclusive source of data.

Although frustrating, some aspects of physical hardware are beneficial to experimental validity. For instance, hardware variability between robots is reflected in their group behavior. Even when programmed with identical software, the robots behave differently due to their varied sensory and actuator properties. Small differences among individuals become amplified as many robots interact over extended time. As in nature, individual variability creates a demand for more robust and adaptive behavior. The variance in mechanics and the resulting behavior provides a stringent test for all of the experimental behaviors.

### 5.3.4 Experimental Procedure

All robot and modeler programs were archived and all basic behaviors were tested in both domains. All robot implementations of basic and composite behaviors were tested in at least 20 trials[2]. In the case of the Modeler, all behaviors were tested in at least 20 trials, with both identical and random initial conditions. Different strategies for the same group behaviors were tested and compared across the two domains.

Modeler data was gathered by keeping a record of relevant state (such as position, orientation, and gripper state) over time. The same data was gathered in robot experiments through the use of the radio system. The system allowed for recording the robots' position and a few bytes of state over time. For each robot experiment, the robots' IDs and initial positions were recorded. Some of the experiments were conducted with random initial state (i.e., random robot positions), while in others identical initial state was used in order to measure the repeatability of the behaviors. All robot data was also recorded on video tape, for validation and cross referencing.

Throughout this chapter, the Interaction Modeler data is shown in the form of discrete snapshots of the global state of the system at relevant times, including initial state and converged state. The robot data is plotted with the Real Time Viewer (RTV), a special purpose software package designed for recording and analyzing robot data[3]. RTV uses the transmitted radio data to plot, in real-time, the positions of the robots and a time-history of their movements, i.e. a trail, the positions of the previously manipulated pucks, and the position of home. It also allows for replying the data and thus recreating the robot runs.

The robots are shown as black rectangles aligned in the direction of their heading, with their ID numbers in the back, and white arrows indicating the front (see, for example, figure 5-7). In some experiments robot state is also indicated with a symbol

[2]In the case of foraging, most data was obtained with another set of robots, described in section 9.1.

[3]RTV was implemented and maintained by Matthew Marjanović.

or a bounding box. In all data snapshots, the size of the rectangles representing the robots is scaled so as to maintain the correct ratio of the robot/environment surface area, in order to demonstrate the relative proximity of all active robots. The bottom of each plot shows which of the twenty robots are being run. The corner display shows elapsed time for each snapshot of the experiment.

## 5.4 Basic Behavior Specifications

This section gives formal specifications for each behavior in terms of the goal it achieves and maintains.

$\mathcal{R}$ is the set of robots: $\mathcal{R} = \{R_i\}, \ 1 \leq i \leq n$

Basic behaviors in 2D space are specified in terms of positions $p$, distances $d$, and distance thresholds $\delta_{avoid}$, $\delta_{disperse}$, and $\delta_{aggregate}$:

$$p_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \quad p_{home} = \begin{pmatrix} x_{home} \\ y_{home} \end{pmatrix}$$

$$d_{home,i} = \sqrt{(x_{home} - x_i)^2 + (y_{home} - y_i)^2}$$

$$d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

$\mathcal{N}$ is the neighborhood operator which, given a robot $R$ and a distance threshold $\delta$, returns all other robots within that neighborhood:

$$\mathcal{N}(i, \delta) = \{j \in \mathcal{R} \ \ d_{i,j} \leq \delta\}$$

$C$ is the centroid operator which, given a robot $R$ and a distance threshold $\delta$, returns

the local centroid:

$$C(i,\delta) = \frac{\sum_{j \in \mathcal{N}(i,\delta)} p_j}{|\mathcal{N}(i,\delta)|}$$

Using this notation, the basic behavior goals are specified as follows:

Avoidance:

$$\forall (i,j) \;\; d_{i,j} > \delta_{avoid}$$

Following:

$$i = leader, \;\; j = follower$$

$$\frac{dp_j}{dt} \cdot (p_i - p_j) = \| \frac{dp_j}{dt} \| \| (p_i - p_j) \|$$

The goal of following is to achieve and maintain a minimal angle $\theta$ between the position of the leader and the heading of the follower. Thus, the above equality is derived from:

$$cos\theta = \frac{\frac{dp_j}{dt} \cdot (p_i - p_j)}{\| \frac{dp_j}{dt} \| \| (p_i - p_j) \|}$$

$$\theta = 0, \quad cos\theta = 1$$

Dispersion:

$$\forall (i,j) \;\; d_{i,j} > \delta_{disperse} \;\; and \;\; \delta_{disperse} > \delta_{avoid}$$

Aggregation:

$$\forall (i,j) \;\; d_{i,j} > \delta_{avoid} \;\; and \;\; d_{i,j} < \delta_{aggregate}$$

Homing:

$$\forall i,j \;\; d_{i,j} > \delta_{avoid} \;\; and \;\; \frac{dp_j}{dt} \cdot (p_j - p_{home}) < 0$$

## 5.5 Basic Behavior Algorithms

This section presents the algorithms used to implement each of the proposed basic behaviors in the Interaction Modeler and on the robots. All algorithms will be given formally as velocity commands of the form:

$$command(R,\ v)$$

where $R$ is the robot and $v$ is some function of $R$'s position, orientation, and its location relative to the local robot distribution. The formal notation will be translated into algorithmic pseudocode for each of the implemented behaviors.

### 5.5.1 Avoidance

Strategies for avoiding collisions are perhaps the most studied topic in mobile robotics. The work in this thesis was concerned with finding avoidance strategies that performed well in group situations and, in particular, did not decline or oscillate with increasing group sizes. Finding a guaranteed general-purpose collision avoidance strategy for an agent situated in a dynamic world is difficult. In a multi agent world the problem can become intractable. Biological evidence indicates that insects and animals do not have precise avoidance routines (Wehner 1987).

Inspired by such simple strategies, the following general avoidance behavior was used:

$$command(R,\ v \begin{pmatrix} cos(\theta + u) \\ sin(\theta + u) \end{pmatrix})$$

where $u$ is $R$'s orientation and $\theta$ is the incremental turning angle away from the obstacle. The following simple *Avoid-Other-Agents* rule was devised:

```
Avoid-Other-Agents:
If another agent is within d_avoid
   If another agent is on the left
      turn right
      otherwise turn left.
```

The *Avoid-Other-Agents* behavior takes advantage of group homogeneity. Since all agents execute the same strategy, the behavior can rely on and take advantage of the resulting spatial symmetry. If an agent fails to recognize another with its other-agent sensors (in this case radios), it will subsequently detect it with its collision-avoidance sensors (in this case IRs), and treat it as a generic obstacle, using the *Avoid-Everything-Else* behavior:

```
Avoid-Everything-Else:
If an obstacle is within d_avoid
   If an obstacle is on the right only, turn left, go.

   If an obstacle is on the left only, turn right, go.
   After n consecutive identical turns, backup and turn.

   If an obstacle is on both sides, stop and wait.
   If an obstacle persists on both sides,
      turn randomly and back up.
```

A provably correct avoidance strategy for arbitrary configurations of multiple agents is difficult to devise. In order to guarantee convergence, these strategies take advantage of the unavoidable noise and errors in sensing and actuation, which result in naturally stochastic behavior. This stochastic component guarantees that the an avoiding agent will not get stuck in infinite cycles and oscillations. In addition to the implicit stochastic nature of the robots' behavior, *Avoid-Everything-Else* also utilizes an explicit probabilistic strategy by employing a randomized move.

Different variations of the above avoidance algorithm were experimented with,

testing simpler versions and different values of $n$. The various algorithms were compared based on the amount of time the agent spent avoiding relative to the amount of time spent moving about freely. This ratio is an indirect measure of the quality of the avoiding strategy in that the more time the agents spend avoiding the worse the strategy is. Avoiding time is dependent on the agent density, so it was used as a controlled variable in the experiments. The ratio used to evaluate avoidance is an indirect metric; a direct measure of being stuck would be more useful, but the robots did not have the appropriate sensors for determining this state.

Because of the great space of possible configurations of multiple agents, it was difficult to measure "average" performance of an avoiding algorithm. However, no significant performance differences were found among the strategies that were tested.

### 5.5.2 Following

*Following* is defined with respect to two agents, the leader and the follower. It can be achieved with a simple rule that steers the follower to the position of the leader:

$$command(R,\ v(\frac{p_{leader} - p_{follower}}{\| p_{leader} - p_{follower} \|})\ )$$

It can be implemented as a complement of the avoidance behavior:

```
Follow:
If another agent is within d_follow
   If an agent is on the right only, turn right.

   If an agent is on the left only, turn left.

   If an agent is newly in front
      keep going and count time.

   If an agent is in front persistently
      stop for an instant.
```

All but the last rule above are direct complements of the *Avoid-Everything-Else* behavior. The last rule is needed to prevent collisions. In the case of the robots, it converts a binary IR sensor into a range sensor through the use of time. Distance from the object being followed in terms of time–to–collision is estimated from the time an object has continuously remained within the IR sensor range.

Figure 5-4 illustrates following on three robots. Additional data on following will be presented and analyzed in the next section.

Incidentally, this approach to *following* models tropotaxic behavior in biology, in which two sensory organs are stimulated and the difference between the stimuli determines the motion of the insect (McFarland 1987). Ant osmotropotaxis is based on the differential in pheromone intensity perceived by the left and right antennae (Calenbuhr & Deneubourg 1992), while the agents described here use the binary state of the two directional IR sensors.

Under conditions of sufficient density, avoidance and following can produce more complex global behaviors. For instance, osmotropotactic behavior of ants exhibits emergence of unidirectional lanes, i.e., regions in which all ants move in the same direction. The same lane–forming effect could be demonstrated with robots executing following and avoiding behaviors. However, more complex sensors must be used in

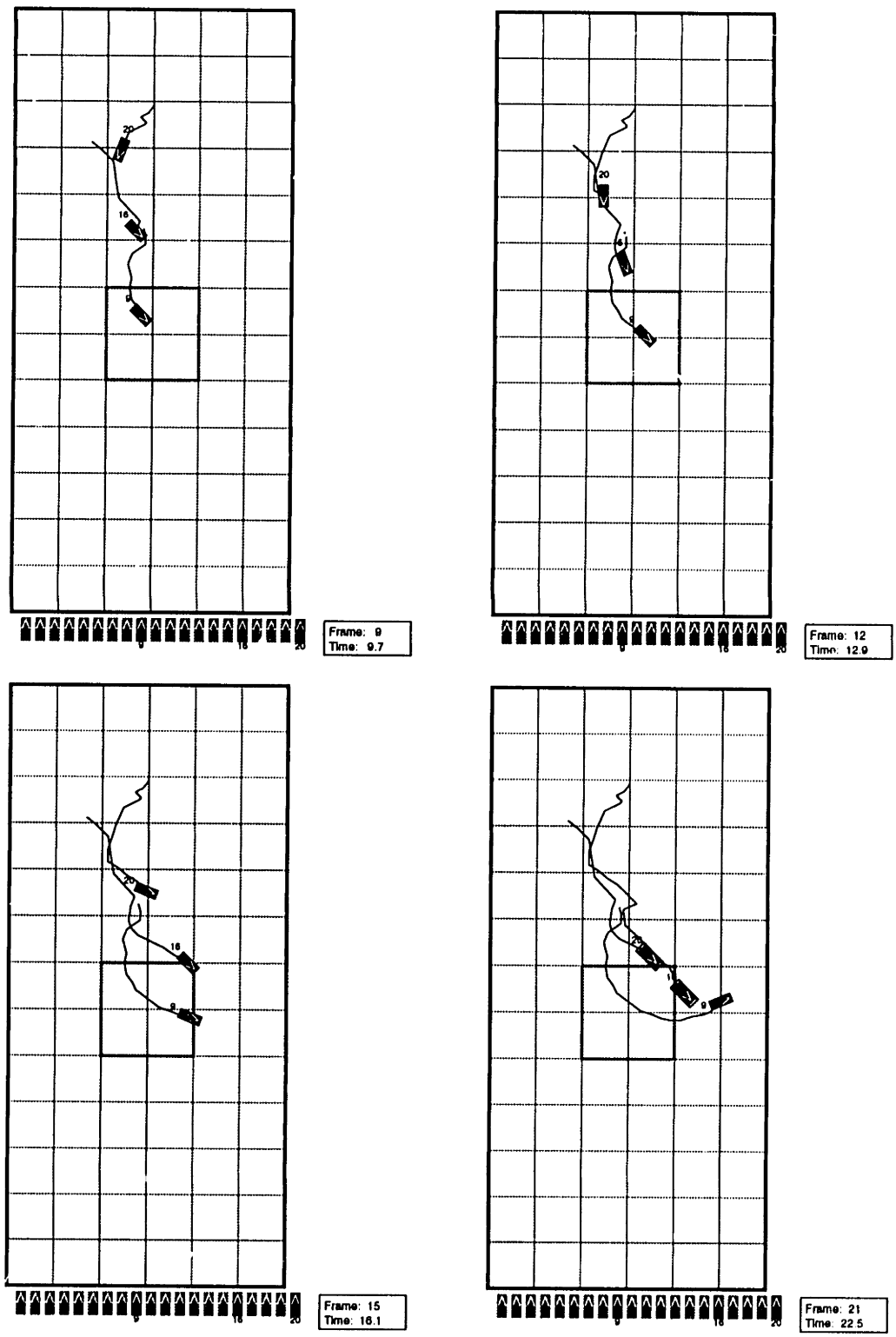



Figure 5-4: An example of following with three robots. Continuous time trails are shown. In spite of deviations in individual paths, the queue is conserved.

order to determine which direction to follow. If using only IRs, the agents cannot distinguish between other agents heading toward and away from them, and are thus unable to select whom to follow.

### 5.5.3 Dispersion

A robust *dispersion* behavior can be designed as an extension of the existing avoidance. While avoidance relies on the presence of a single agent, *dispersion* uses the local distribution of all of the nearby agents (i.e., the locations of other agents within the range of the robot's sensors) in order to decide in which direction to move. The algorithm below computes the local centroid to determine the local density distribution of nearby agents, and moves away from the highest density:

$$command(R,\ -v(\frac{C(R,\delta_{disperse}) - p_R}{\| C(R,\delta_{disperse}) - p_R \|})\ )$$

```
Centroid-Disperse:
 If exactly 1 agent is within d_disperse
    avoid, then go forward.

 If multiple agents are within d_disperse
    move away from the Centroid_disperse, then go forward.

 Otherwise, stop.
```

Under conditions of high density, the system can take a long time to achieve a dispersed state since local interactions propagate far and the motion of an individual can disturb the state of many others. Thus, *dispersion* is best viewed as an ongoing process which maintains a desired distance between the agents while they are performing other tasks.

A number of dispersion algorithms were tested in the modeled environment as well. As in the robot implementation, all of the approaches were based on detecting

the position of the nearest agents. However, the modeler allowed for using more precise information, such as the exact distance and direction of the nearest neighbors. The following algorithm was most successful in terms of speed and reliability:

```
Neighbor-Disperse:
If 1 or 0 neighbors are within d_disperse
   stop.

Find n nearest neighbors within d_disperse
Compute the angle between them,
Compute the negative of the bisector,
align the agent in that direction and go forward.
```

This algorithm was effective in the modeler but did not transfer to the available robots due to their sensing limitations.

Figure 5-5 shows the initial state and the final state of a dispersion experiment using the above centroid-based dispersion rule tested in the Interaction Modeler. Initially crowded in one part of the available free space, the agents apply the simple dispersion rule in order to to establish maximum available inter-agent distance. Figure 5-6 shows the same dispersion algorithm applied to four robots.

Dispersion was also evaluated based on time to convergence. Algorithms using the local centroid, and the nearest two agents, were compared to each other and to a potential field summation approach, in which the scalar distance from each nearby agent was proportional to the magnitude of a repulsive vector associated with it. The vectors of all nearby agents were summed and the agent moved in the direction of the resultant. The performance of the three algorithms was compared using two different initial conditions, random and densely packed. Both were tested in order to normalize for different density distributions through the lifespan of the task. As expected, the random initial position results in faster convergence times than a packed initial condition for all three algorithms. No statistically significant difference was

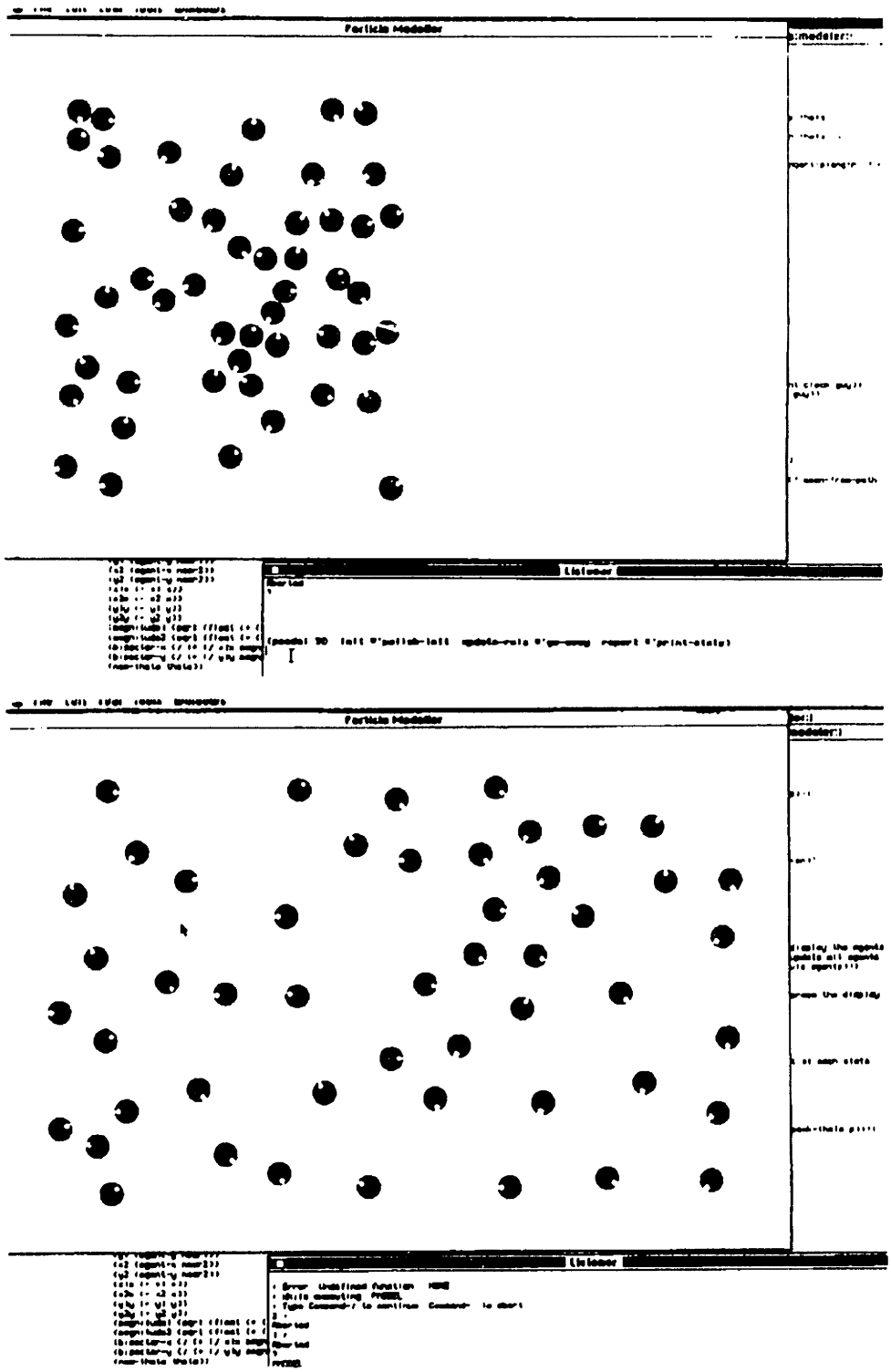


Figure 5-5: An example of dispersion. Fifty agents are initially packed in one half of the workspace. After approximately 20 time steps, they have reached an equilibrium in terms of a maximum available inter-agent distance.

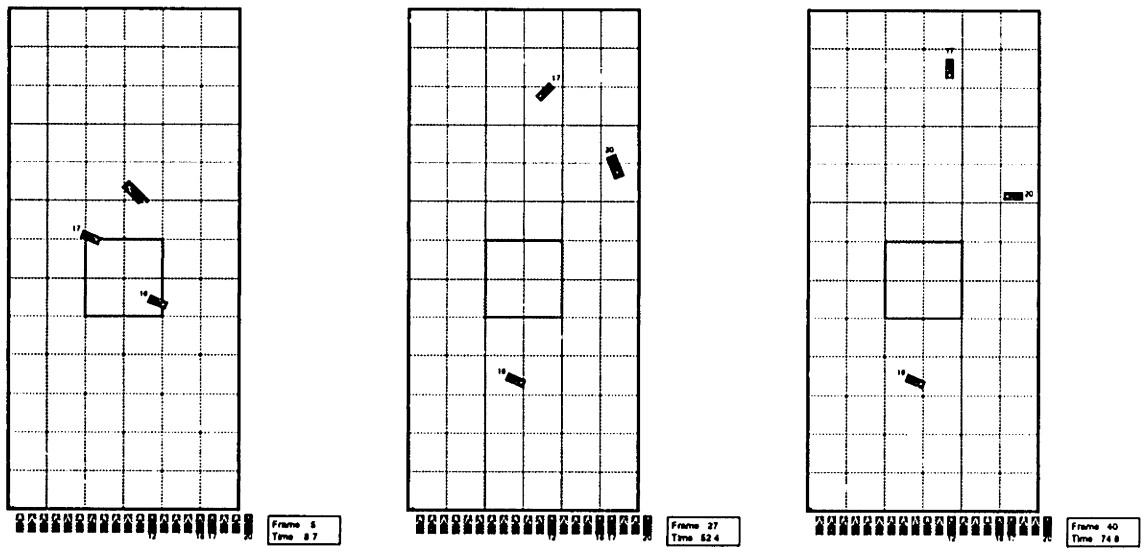


Figure 5-6: Dispersion with three robots, initiated close to each other. The robots found a static dispersed equilibrium state after 74 seconds.

found between the algorithms.

### 5.5.4 Aggregation

*Aggregation* is the inverse of dispersion:

$$command(R,\ +v(\frac{\mathcal{C}(R,\delta_{aggregate}) - p_R}{\parallel \mathcal{C}(R,\delta_{aggregate}) - p_R \parallel})\ )$$

and can be implemented using the centroid operator as well:

```
Aggregate:
If 0 agents are within d_aggregate
   turn toward the local Centroid_aggregate, go.

Otherwise, stop.
```

Aggregation was evaluated using the same criteria used in evaluating dispersion, as well as the same experiments. Analogous algorithms were implemented, using the

local centroid, two nearest neighbors, and potential fields. Instead of varying initial conditions, aggregation algorithms were evaluated using two different terminating conditions. The more difficult terminating condition required that all agents form a single aggregate, whereas the easier of the two conditions required only that they form one or more groups in which all agents are within a fixed distance from their neighbors. As expected, the former terminating condition required more time to be achieved. Aside from that effect, no statistically significant difference was found between the algorithms.

### 5.5.5 Homing

The simplest homing strategy is a greedy local one:

$$command(R,\ v(\frac{p_{home} - p_R}{\| p_{home} - p_R \|})\ )$$

and implemented as a simple pursuit:

```
If at home
  stop.
  otherwise turn toward the goal, go.
```

Figure 5-7 illustrates the homing behavior of five robots using this strategy. The data illustrates that the actual trajectories are far from optimal, due to mechanical and sensory limitations, in particular due to the error in the sensed position. The same algorithm, when tested on the Interaction Modeler, produces more optimal homing trajectories. Figure 5-8 shows another robot run of homing with five robots. In this run the entire time history of the robots' positions are shown, and the positioning errors can be easily seen. Nonetheless, all robots reach home. Figure 5-9 illustrates homing in simulation.

Individual homing is effective as long as the density of agents is low. If enough

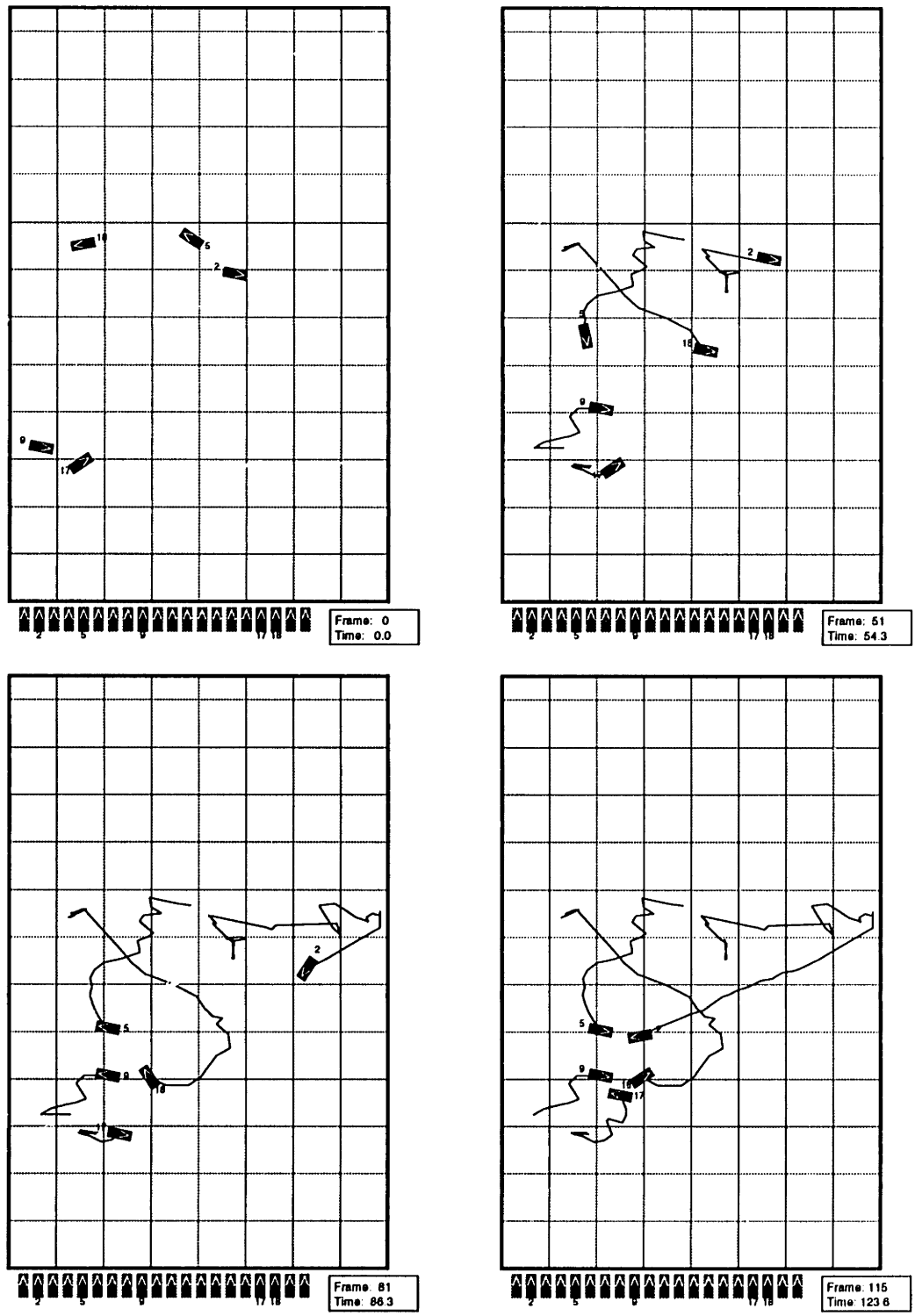



Figure 5-7: Homing behavior of five robots. Started in an arbitrary initial configuration, four of the robots reached the home region within 100 seconds, and the fifth joined them 30 seconds later. The trails reflect errors in position sensing, as well as interference between the robots as approach the home region.

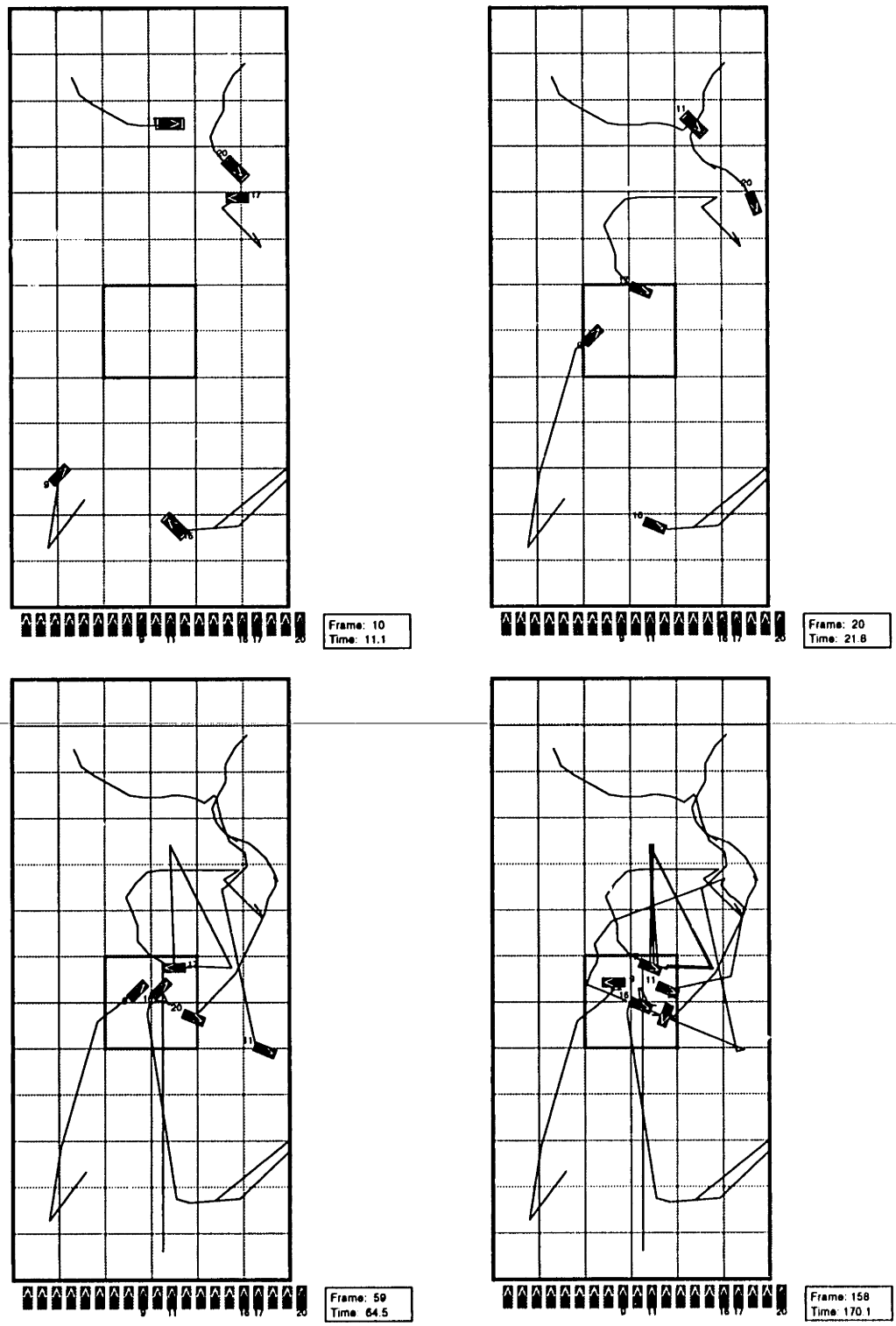



Figure 5-8: Another example of homing behavior of five robots, started in arbitrary initial positions. Trail histories demonstrate drastic errors in positioning, indicated by large jumps in consecutive robot location. In spite of the intermittent position errors, the robots successfully reached home.

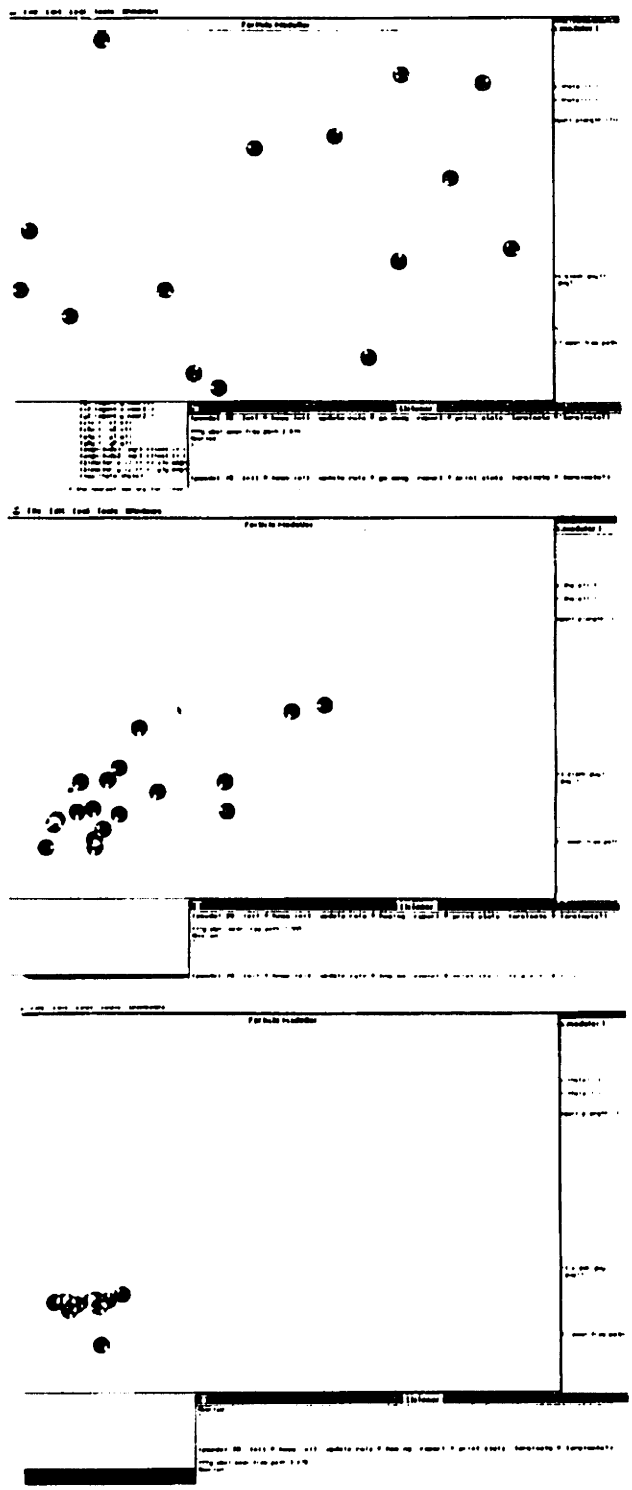


Figure 5-9: Homing behavior of a large group of simulated agents. Increased interference and competition for space is obvious around the goal region.

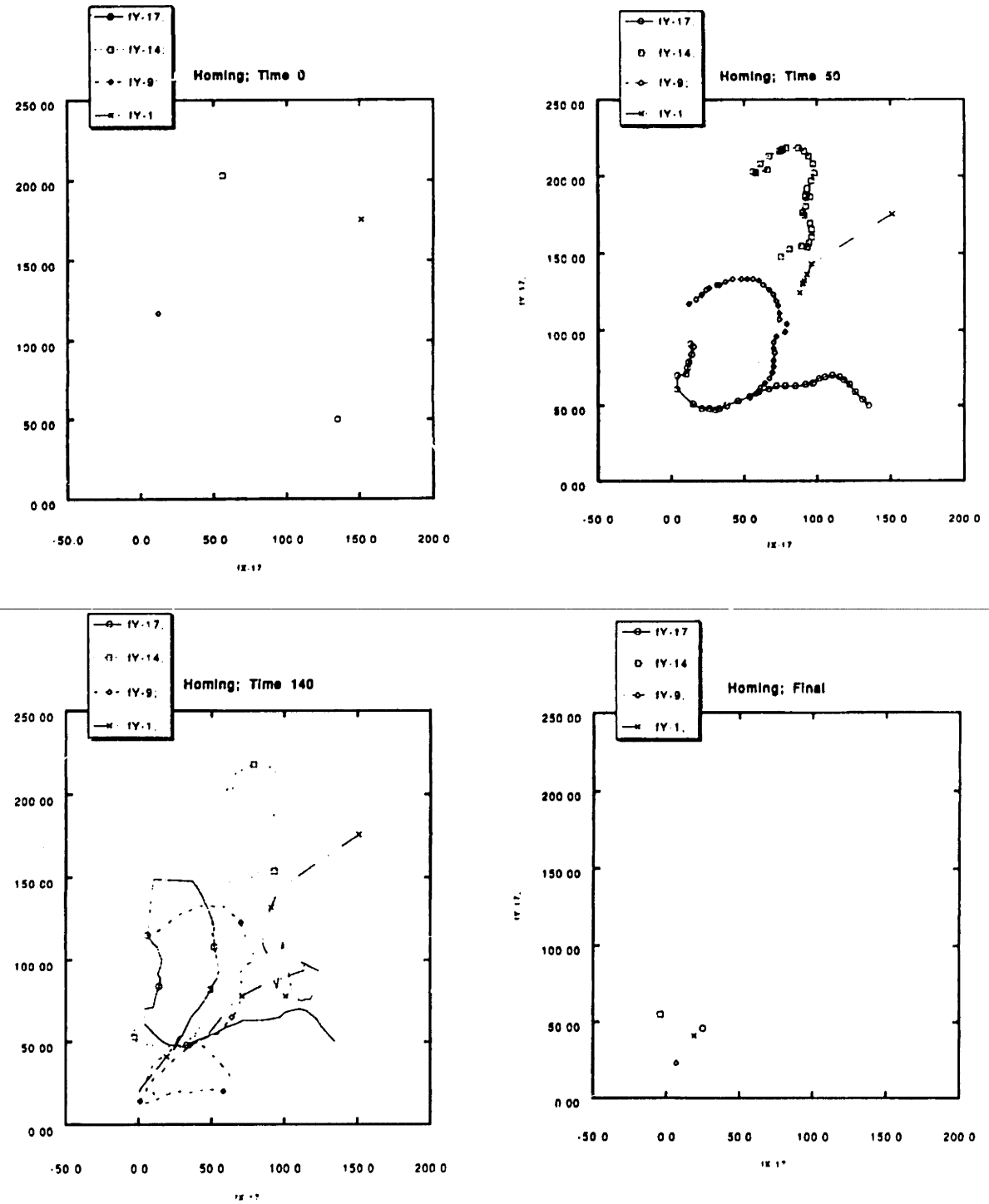



Figure 5-10: Homing behavior of four robots. Trails are marked with different patterns in order to demonstrate the increase in interference with proximity, resulting in circuitous paths.

agents are homing within confined space, they interfere with each other. Figure 5-10 shows the growing interference between robots as they approach the goal region. Entire time-trails are shown to demonstrate how much avoidance is required in order for all of the shown agents to get home.

Simulation and robot experiments described in this work show that interference increases if the agents have non-zero turning radii, unequal velocities, or are subject to sensor and control errors. All of the above conditions are common in situated agents, suggesting the need for some form of group or structured navigation, such as flocking, which will be introduced in an upcoming section.

## 5.6 Basic Behavior Evaluation

### 5.6.1 Empirical Evaluation of Basic Behaviors

Evaluation is one of the most difficult components of research, and it is quite new to the field of AI and Robotics in which much of the work has a show-and-tell style. By nature and by design, the field is based on building artificial computational and/or physical systems in order to study natural phenomena. Synthetic results do not fall cleanly into the well defined set of evaluation criteria designed for natural sciences. Analyzing something one has designed is intrinsically different from analyzing something externally imposed.

Even in synthetic fields, analysis should follow synthesis, as practice should follow theory. The major stumbling block, however, is the lack of standardized evaluation criteria that would be useful but not constraining for a field as young and diverse as AI. Consequently, it is up to the researcher to establish evaluation criteria that are both specifically appropriate for the particular research, so that it can be thoroughly validated, as well as objective and general, so the results can be applied to other projects and domains.

The ideas proposed in this thesis are evaluated in two ways. The first addresses the merit of the general approach and its applicability to various domains. This evaluation is performed in the summary of the thesis, after the entire work has been presented. The second type of evaluation addresses the specific instantiation of the ideas in the spatial domain. This chapter presents the evaluation criteria applied to the implementations and performance of spatial basic behaviors and their composites.

AI and Robotics research in general is exploratory and often prone to phenomenological evaluation. To prevent this, all of the evaluation criteria for the experimental part of the work were established prior to testing and were applied to the performance of each of the behaviors as well as to their combinations. An earlier section on basic behavior selection elaborated the criteria for choosing the basic behavior set and hinted at some evaluation procedures. This section gives a detailed illustration of empirical basic behavior evaluation on the example of following.

Repeatability, robustness, and reliability of following were evaluated based on the average duration of uninterrupted following, which is almost entirely dependent on the accuracy of the front-pointing sensors. The duration of following was dependent on how reliably the front-pointing sensors could detect the "leader". Figure 5-11 illustrates continuous *following* behavior of 3 robots over a four minute period. The robot at the "front" of the queue is moving forward with its wheels slightly turned, thus tracing out a circular path. The other two robots follow their local "leader" according to the presented following algorithm. The path of the first robot is smooth, while the followers oscillate in order to keep the robot ahead of them within IR range. One of the robots diverged after two minutes, while the other two stayed together for the duration of the shown 243.3 second run. Figure 5-12 also illustrates the robustness of following; the robot in the lead moves about randomly and the follower keeps up throughout the duration of the run.

The range of the IR sensors used was directed and short, requiring the agents to

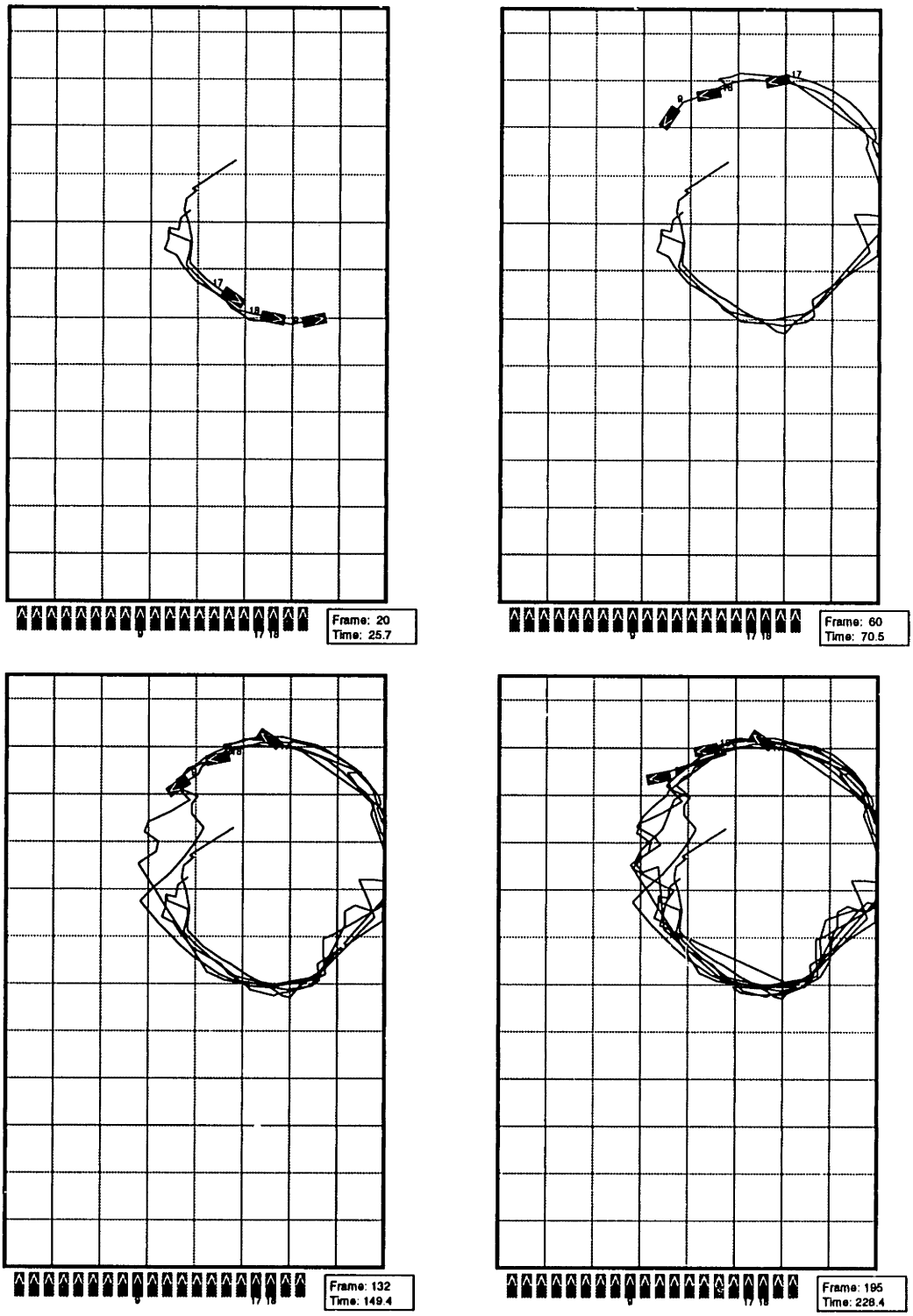



Figure 5-11: Continuous following behavior of 3 robots over 4.8 minutes. In the initial conditions, the wheels of the front robot are turned sideways, resulting in a circular trajectory. The robots reliably maintain a stable queue in spite of individual local variations in control.

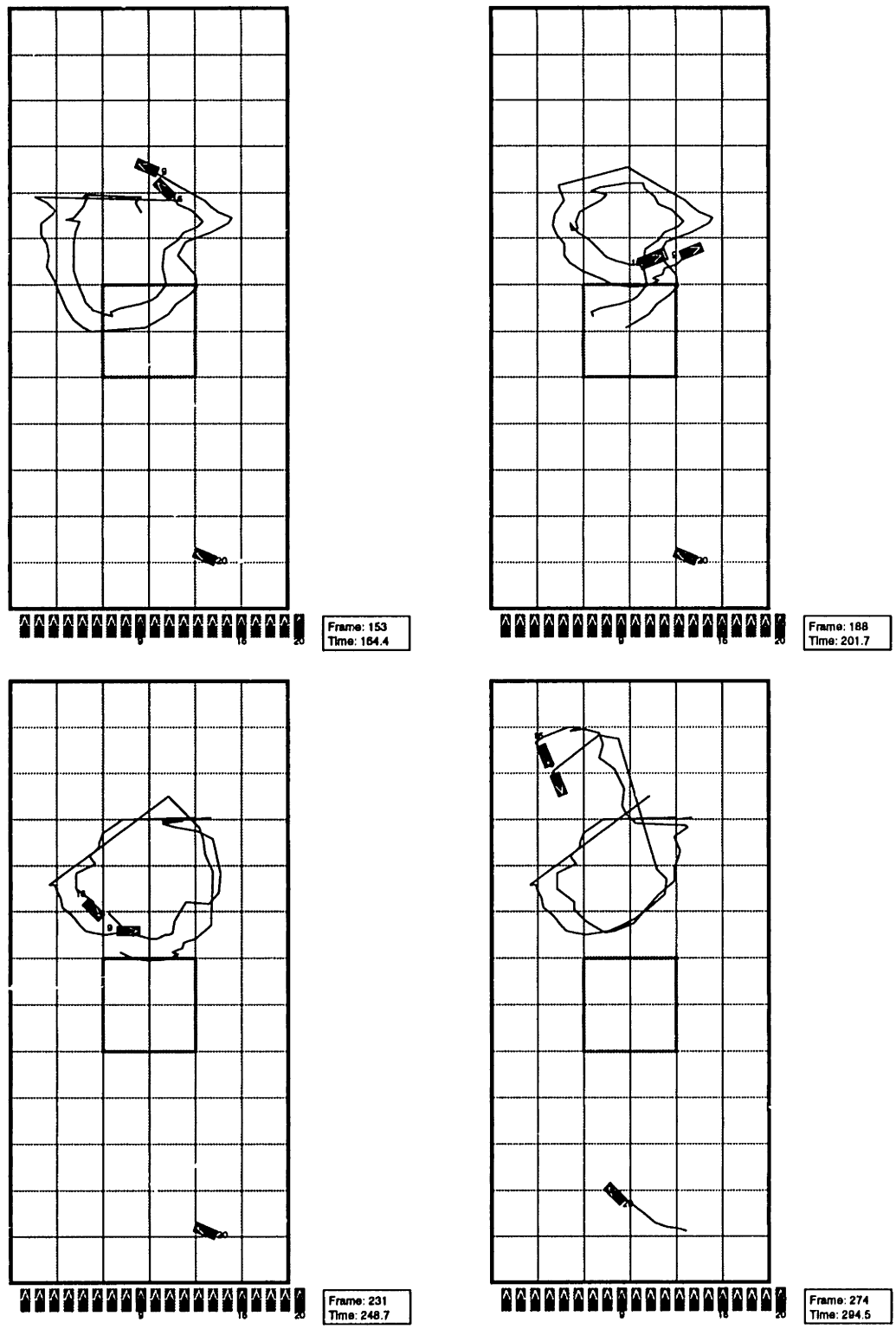



Figure 5-12: Continuous following performance of two robots over 4.9 minutes. The robot in the front moves about randomly, while the follower stays close behind.

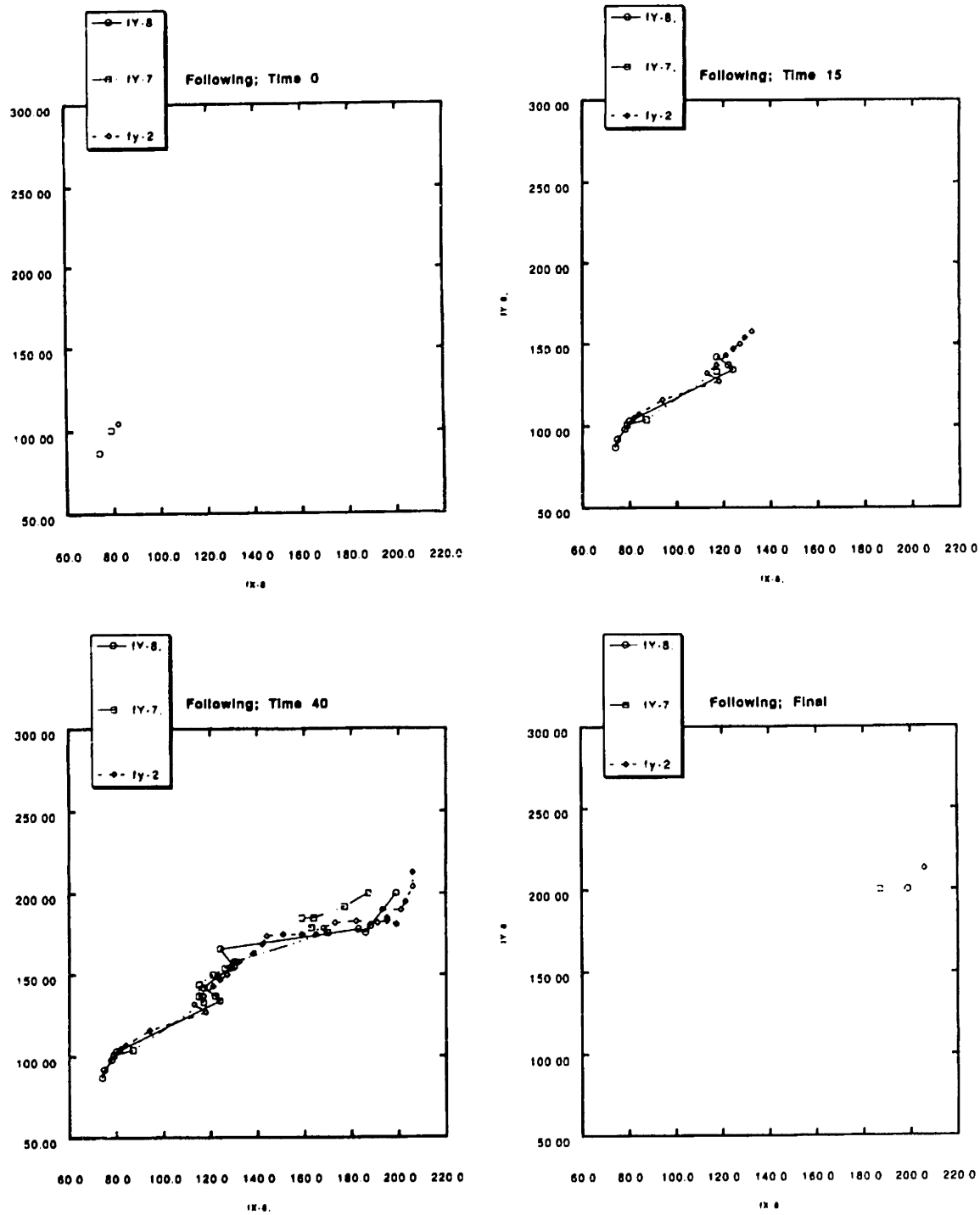



Figure 5-13: Following performance of three robots in the presence of obstacles, sensory, and steering errors that cause the middle robot stall. The third robot passes it and maintains the first robot in its following range. The middle robot senses the now-second robot within its following range, follows it, and the queue is maintained.

stay close together within the queue. Consequently, errors in steering could cause a follower to lose sight of the leader if it failed to turn sufficiently in order to maintain the leader in sight. If the two continued to move in the same direction, as they would during a higher-level task, the follower could catch up with the leader again. If not, they would diverge.

The narrow IR range explains why long queues and trains of agents were physically difficult to maintain. However, queues were locally stable and insensitive to dynamic obstacles and sensory and mechanical irregularities in the form of sensor noise, errors in steering, and perturbations in velocity. Figure 5-13 illustrates the following behavior of three robots in the presence of sensory or effector error. The middle robot stalls due to some error, and the robot behind it stops as well, then turns and follows the leader, as it senses the first robot in its range. The middle robot activates again, senses the second robot within its range, follows it, and the queue is maintained. Figure 5-14 demonstrates following in the presence of static constraints in the environment, such as walls and corners. The robots are able to avoid the walls and maintain the queue.

Following was also evaluated based on scalability in order to test its performance as agents are added and removed. The data above demonstrates the following behavior if an agent stalls, or is removed from the middle of the queue. The next set of data deals with the performance as new agents are added to the queue, the situation that is expected to happen more commonly, since following is a recruiting behavior.

Figure 5-15 demonstrates average following time for two robots in multiple runs. Figure 5-16 plots the data for three robots performing following. Strikingly, the mean following time for two agents is nearly identical as that for three. This is exactly as expected, since following is a completely local behavior between two agents. The failure of any pair is as likely as the failure of any other, and they are mutually independent. Consequently, following is independent of group size, and agents can

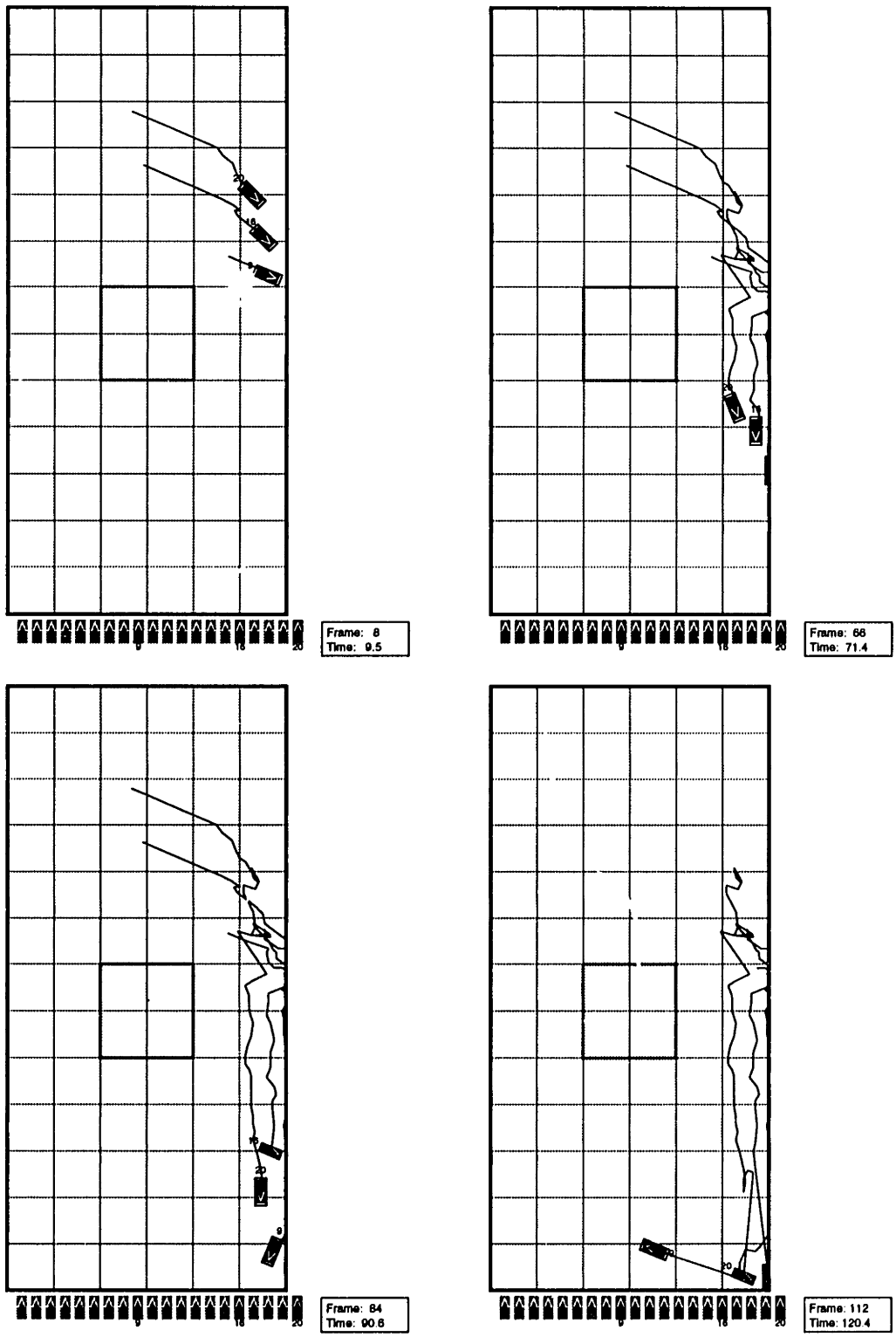



Figure 5-14: Performance of the following behavior of three robots in the presence of external obstacles and constraints. The robots maintain a queue while avoiding the wall and going around a corner.

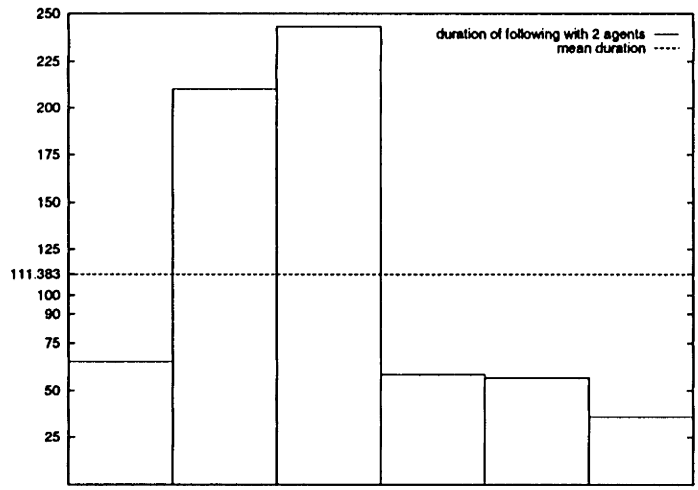



Figure 5-15: Following with 2 robots. The x-axis plots individual trials, the y-axis plots the duration of uninterrupted following. The mean duration is indicated with the dashed line.

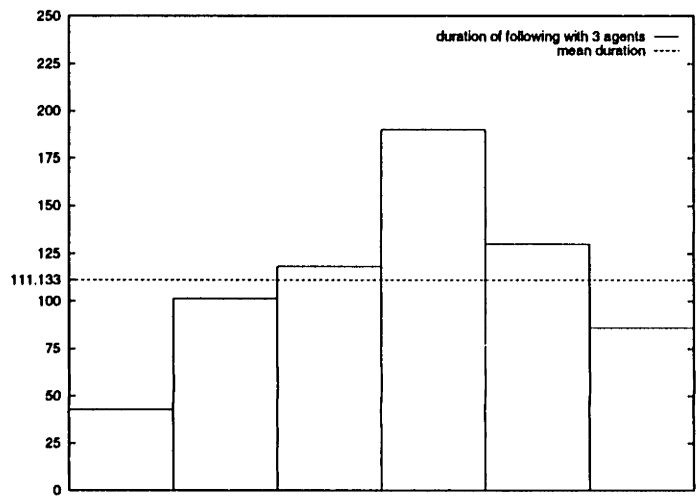



Figure 5-16: Following with 3 robots. The x-axis plots individual trials, the y-axis plots the duration of uninterrupted following. The mean duration is indicated with the dashed line.

be dynamically added and removed from the ends of the queue without affecting the rest.

The previous section compared different distributed algorithms for each of the basic behaviors, and this section has shown what empirical criteria were used to evaluate them by illustrating them on the example of following behavior. The described criteria were systematically applied to all of the other basic behaviors as well.

### 5.6.2 Evaluation of Heterogeneous Groups

An obvious alternative for a fully distributed system of identical agents is a hierarchical distributed system. In order to evaluate the performance of the homogeneous basic behaviors, they were compared to particular hierarchical implementations. This section describes the performance of a hierarchical group of agents on two basic behaviors: aggregation and dispersion. These two behaviors were chosen because they can be stated in terms of achievement goals and, given sufficient space, can reach a static state. The algorithms were evaluated based on the time or the number of steps required to reach that well-defined state.

A version of hierarchical agents was implemented by classifying the agents into a total order, based on a randomly assigned unique ID number, thus simulating an established pecking order in the group (Chase, Bartolomeo & Dugatkin 1994, Chase 1993, Chase 1982, Chase & Rohwer 1987). While in homogeneous algorithms all agents moved simultaneously according to identical local rules, in the hierarchical case the ID number determined which agents were allowed to move while others waited. In all cases, a simple precedence order, a spatially-local hierarchy, was established such that within a small radius the agent with the highest ID got to move. Multiple types of dispersion and aggregation algorithms were tested with such hierarchical agents.

Using the Interaction Monitor, 20 experiments were conducted with each group size (3, 5, 10, 15, and 20 agents) and each of the algorithms. Additionally, the algo-

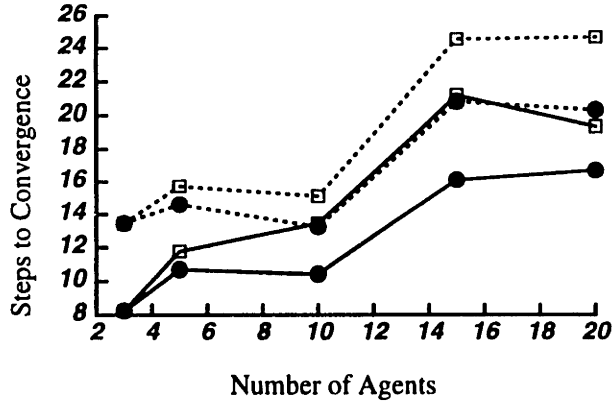



Figure 5-17: The performance of two different aggregation algorithms based on time required to reach static aggregated state. Two termination conditions were tested: a single group (data points shown with boxes) and a few stable groups (data points shown with dots). The performance of hierarchical algorithms is interpolated with solid lines while the homogeneous ones are interpolated with dotted lines.

rithms were tested on two different degrees of task difficulty. Aggregation was tested on two terminating conditions: a single aggregate containing all of the agents, and a small number of stable aggregates. The former terminating condition is more difficult. Similarly, dispersion was tested on two initial conditions: a random distribution of initial positions, and a packed distribution in which all of the agents start out in one half of the available space. The latter condition is more difficult.

It was found that, in the case of aggregation, hierarchical strategies performed somewhat better than totally homogeneous ones. Figure 5-17 plots the average number of moves an agent takes in the aggregation task against the different group sizes and the two different terminating conditions: a single aggregate and a few stable groups. Both hierarchical and homogeneous algorithms behaved as expected, performing better on the simpler of the two terminating conditions. Their performance declined consistently with the growing group size.

Unlike aggregation, in the case of dispersion, homogeneous strategies outperformed hierarchical ones. Figure 5-18 plots the average number of moves an agent makes in the dispersion task for the different group sizes on two different initial con-

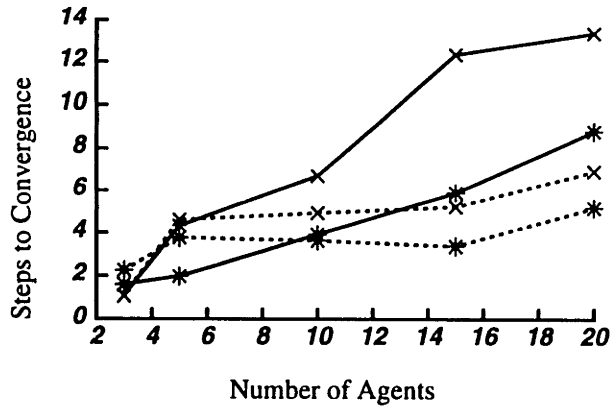



Figure 5-18: The performance of two different dispersion algorithms based on the time required to reach static dispersed state. Two initial states were tested: a random distribution (data points shown with stars) and a packed distribution (data points shown with crosses). The performance of the hierarchical algorithms is interpolated with solid lines while the homogeneous ones are nterpolated with dotted lines.

ditions: a random distribution, and a packed initial state. Again, both hierarchical and homogeneous algorithms improved with the easier initial conditions.

Although the performance difference between the homogeneous and hierarchical algorithms was repeatable and consistent, it was small, and its magnitude barely surpassed the standard deviation among individual trials for each of the algorithms and group sizes. The standard deviation was particularly significant in the case of small (3 and 5) group sizes. Thus, no statistically significant difference was found in global performance of hierarchical and flat algorithms for aggregation and dispersion. Furthermore, the slight differences that were detected between the two strategies would mostly likely be negligible on physical agents, due to sensor uncertainty and effector errors.

The similarity in performance between the homogeneous and simple heterogeneous algorithms is caused by the following:

**Functionally homogeneous agents:** In spite of the linear priority ordering, the agents are fundamentally homogeneous since they are functionally indistinguishable. Thus, the hierarchical relationships between agents are spatially and temporally

independent, since the agents keep no history of their past encounters with each other.

**Simplicity of behavior**: The only behavior being observed is spatial, in the domain where the consequences of actions of identical agents have no time-extended consequences.

**Large group sizes:** In sufficiently large groups of functionally identical agents, any temporary effects are averaged out as fluctuations and noise. This property is crucial for producing reliable global behavior in spite of local perturbations. However, it implies that local differences must be large enough to create strong enough perturbation to affect the global behavior. This is observable in the data, where the general trends in global performance are consistent even if the standard deviation among trials can be quite large.

The experiments comparing simple hierarchical and homogeneous algorithms demonstrate that for the described domain simple hierarchical strategies do not affect the global performance because their impact on the global behavior is negligible. More complex hierarchical strategies could be devised, in order to assure their influence on the global behavior, but would require an increased perceptual and cognitive overhead, such as perhaps keeping a history of past encounters and models of previously encountered agents. This data allows the hypothesis that for simple spatial domains 1) simple homogeneous solutions work well, and 2) more complex strategies requiring individual agents to perform recognition, classification, and representation are required to significantly improve group performance.

### 5.6.3 Evaluating Distributed v. Centralized Algorithms

The beginning of the thesis compared centralized and distributed approaches, and argued that centralized approaches do not scale for the types of systems this thesis has dealt with. For the purposes of comparison, however, a set of special case scenarios was constructed, for which optimal centralized solutions could be computed for the

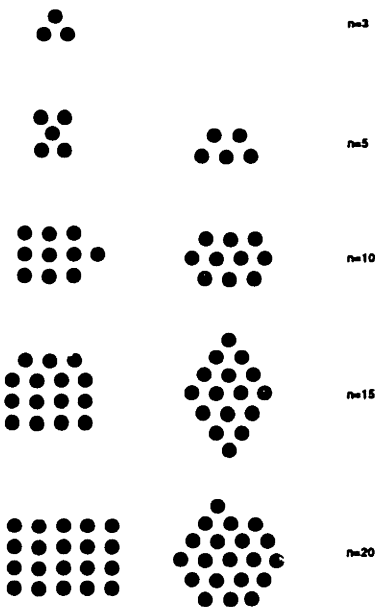



Figure 5-19: The initial conditions used for comparing dispersion algorithms. Maximally packed states for five different group sizes (3, 5, 10, 15, and 20) were tested.

dispersion task. While computing the optimal dispersion strategy for an arbitrary configuration of agents is difficult and, for large group sizes, intractable, the strategy can be computed for special classes of initial positions.

A maximally packed configuration of agents was designed for five group sizes: 3, 5, 10, 15, and 20. Figure 5-19 shows these configurations. These configurations were chosen for two reasons: 1) they presented challenging initial conditions for dispersion, and 2) optimal dispersion solutions could be computed by taking advantage of the symmetry of configurations. The optimal solutions generally consisted of moving the outer agents first until enough space is cleared for the next layer to move, and so on. Using these strategies, the average number of moves per agent for obtaining a dispersed state was computed for each of the group sizes.

The "total knowledge" algorithm was tested along with the existing hierarchical and homogeneous algorithms on the Interaction Modeler. The data for the distributed algorithms was averaged over 20 trials for each group size. Figure 5-20 plots the

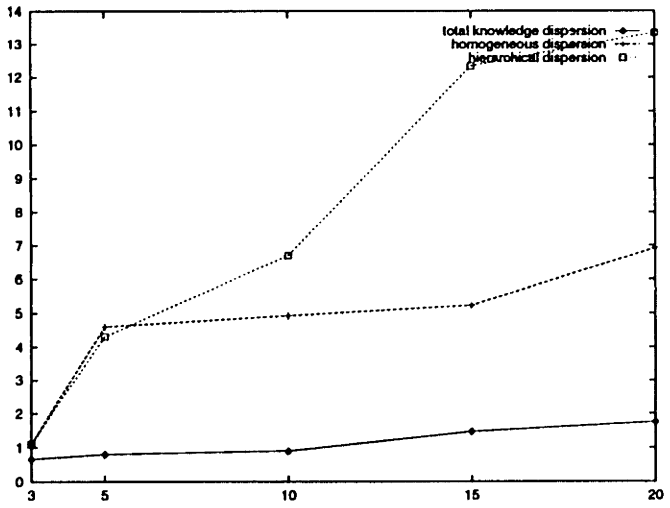



Figure 5-20: The performance of the optimal global "total knowledge" algorithm for dispersion (data points shown with diamonds) compared with the hierarchical and homogeneous dispersion strategies (data points shown with boxes and crosses, respectively).

performance of the three algorithms.

Not surprisingly, the total knowledge algorithm performs the best. However, it is important to note that although its performance declines slower than that of the distributed algorithm, the two are only offset by a constant factor. Given that the performance of the total knowledge algorithm is not practically attainable in real-time, the distributed alternative with minimum computational and sensing overhead presents a useful alternative.

## 5.7 Summary

This chapter has introduced the methodology for selecting basic behaviors and demonstrated it on the spatial domain. A basic behavior set consisting of avoidance, following, dispersion, aggregation, and homing was proposed, implemented in two different experimental environments, and tested in simulation and on physical robots. Experimental data was evaluated using a collection of prespecified criteria. The performance

of the basic behaviors was also tested compared against hierarchical and total knowledge approaches.

The next chapter introduces ways in which the described basic behaviors can combined in order to be achieve a variety of higher-level goals and tasks.

# Chapter 6

# Combining Basic Behaviors

Basic behaviors are designed to be a substrate for a variety of more complex compound group behaviors for a given domain (figure 6-1). Generating compound behaviors requires applying some kind of a combination operator whose properties are well understood and which produces the desired output behavior. This is is considered to be one of the challenges of behavior-based control, i.e., *arbitration*, the problem of coordinating the activity of multiple input behaviors in order to produce desired output behavior.

Depending on the complexity of the system, arbitration can and usually must be performed at multiple points. One level of arbitration can be achieved by designing mutually exclusive behavior conditions (Matarić 1992*c*). Creating a unique one-to-one mapping between conditions and behaviors guarantees a mutually exclusive set of condition--action couplings. In contrast, if the mapping is one-to-many, so that a condition can result in more than one possible behavior, than there is a possibility that the two behaviors may be in conflict, either on a single agent, or between agents.

Mutually exclusive behavior conditions are sufficiently powerful for arbitrating in a system that performs only one behavior at a time. However, in more complex systems multiple behaviors can contribute to the output (Parker 1994, Payton, Keirsey, Kimble, Krozel & Rosenblatt 1992, Ferrell 1993). Consequently, most practical sys-

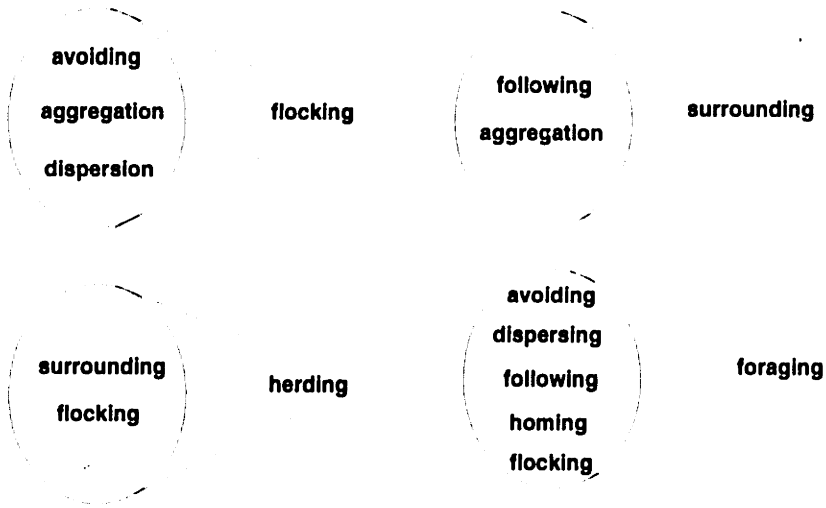



Figure 6-1: Basic behaviors can be combined to generate a variety of more complex behaviors.

tems use mutually exclusive behavior conditions within a coherent layer or submodule of the system dealing with a particular coherent set of tasks. Between modules and layers another level of arbitration is necessary which either implements a type of a sum of or a switch between the inputs. The general form of a behavior–based system involves such combination operators at one or more levels.

The architecture proposed here for combining basic behaviors has the described general form. In order to take advantage of the full expressive combinational power of the basic behaviors, the architecture uses two combination operators: behaviors can be combined directly, by executing multiple behaviors at once, and temporally, by sequencing the behaviors one at a time. Direct combinations allow for multiple concurrently active behaviors to contribute to outputs. Temporal combinations assure a coherent sequence of the outputs. The two types of combination operators, applied to the fixed set of basic behaviors, can generate an unbounded repertoire of collective behaviors (figure 6-2). The following sections describe the operators and demonstrate them with implemented composite behaviors.

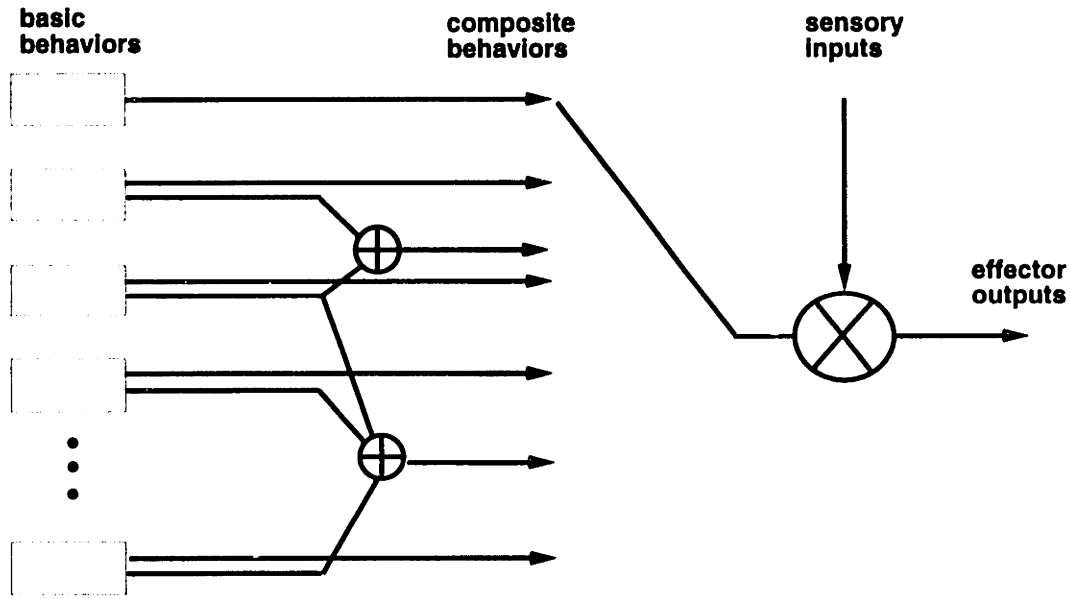



Figure 6-2: The control architecture for generating group behaviors consists of direct and temporal combinations of subsets from a fixed basic behavior set. Direct combinations are marked with $\oplus$, temporal combinations with $\otimes$.

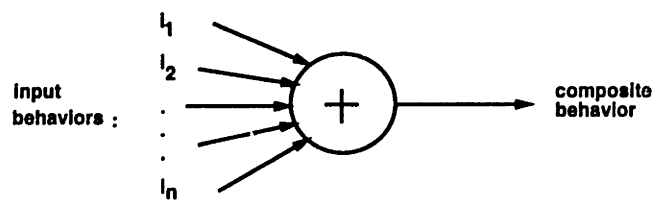



Figure 6-3: The general form of direct behavior combinations. Outputs from two or more behaviors are summed.

### 6.0.1 Direct Combinations of Basic Behaviors

A direct combination of behaviors is some function of the outputs of a subset of the basic behaviors (figure 6-3). In the spatial domain, the outputs of all of the basic behaviors are in the form of direction and velocity vectors, so appropriately weighted sums of such vectors directly produce coherent higher-level behaviors. This method is illustrated by using direct combination to implement flocking.

*Flocking* is defined as collective motion that satisfies the following constraints: all of the agents within sensing range of each other must maintain a fixed range of

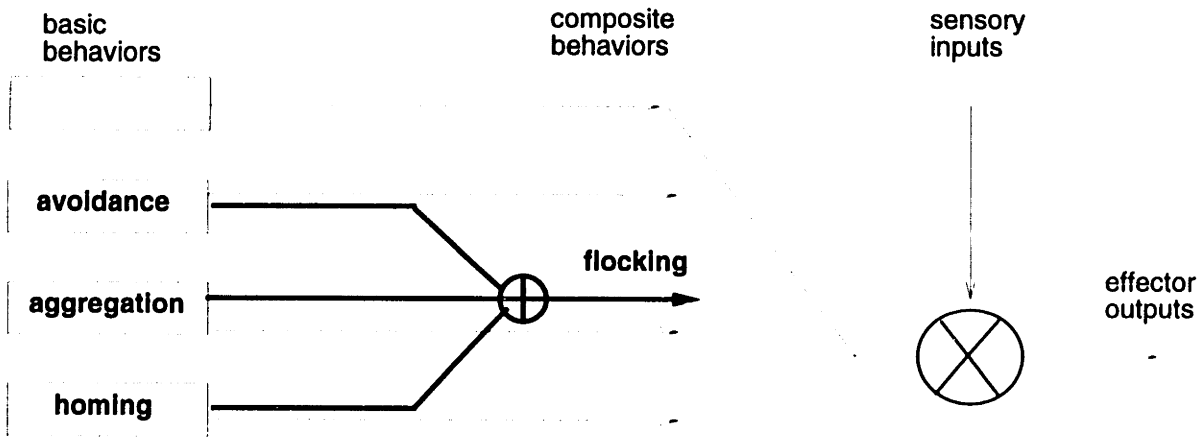



Figure 6-4: The implementation of flocking as a combination of avoidance, aggregation, and homing. Avoidance and aggregation alone produce robust flocking, but homing gives it a goal location and direction to move in.

distances from their neighbors as they move. Formally:

$$\forall(i,j)\ \ d_{i,j} > \delta_{avoid}\ \ and\ \ d_{i,j} < \delta_{flock}$$

$$\frac{dp_i}{dt} \cdot (\ p_i - C(i, \delta_{flock})\ ) < 0$$

Flocking can be implemented by combining the outputs of *avoidance*, *aggregation*, and *dispersion*, such that the specified constraints are satisfied. Intuitively, *aggregation* keeps the robots from getting too far from each other, *dispersion* keeps them from getting too close, and *avoidance* prevents each agent individually, and thus the flock as a whole, from collisions with any non-agent obstacles. Flocking can be further reduced to a combination of just *avoidance* and *aggregation* for a range of values of $\delta_{flock}$, such that $\delta_{flock} < \delta_{aggregate}$, so that *avoidance* also has a dispersing effect.

This version of flocking keeps the robots moving together in whatever direction the front of the flock pursues. Generally, flocking is used as structured motion toward a particular goal. To achieve this, *homing* is added to the sum in order to direct the

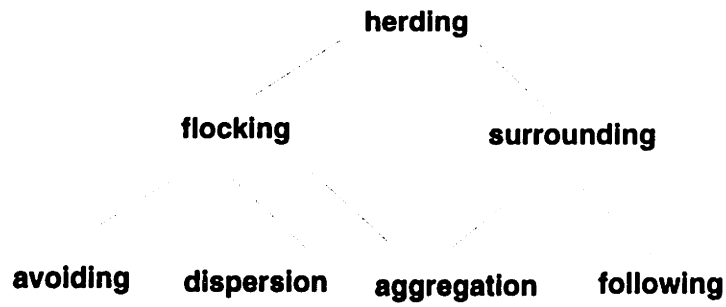



Figure 6-5: An example of direct and temporal basic behavior combination within a higher–level task.

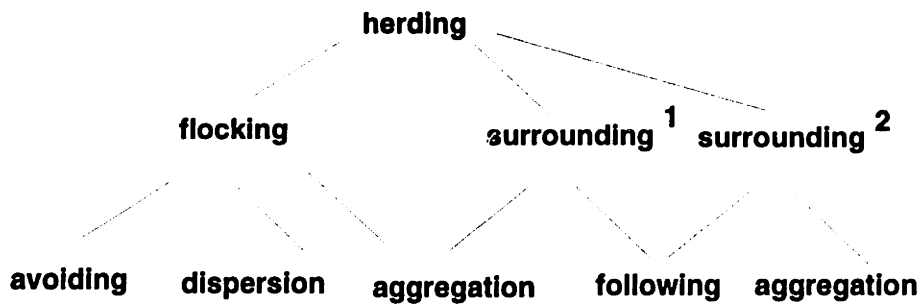



Figure 6-6: Direct behavior combinations use continuous summing functions. Consequently, the same basic behaviors can be reused and recombined repeatedly within a common higher–level goal. In this example, two types of surrounding are used, depending on the sensory conditions.

flock toward a particular goal (figure 6-4).

The given set of basic behaviors allows for many other direct composites, such as *surrounding*, a combination of *aggregation* and *following*, and *herding*, a combination of *surrounding* and *flocking* (figure 6-5).

For any given high–level goal, the structure of direct behavior combination is a directed acyclic graph (DAG) with behaviors as nodes and inheritance relations as arcs. Basic behaviors are the originator nodes of the graph. Except for the final goal node, all other nodes are combinations of originator and other intermediate nodes in the graph. Figure 6-5 illustrates an example of a graph in which *aggregation* is shared by two intermediate nodes: *flocking* and *surrounding*. Since behavior combinations

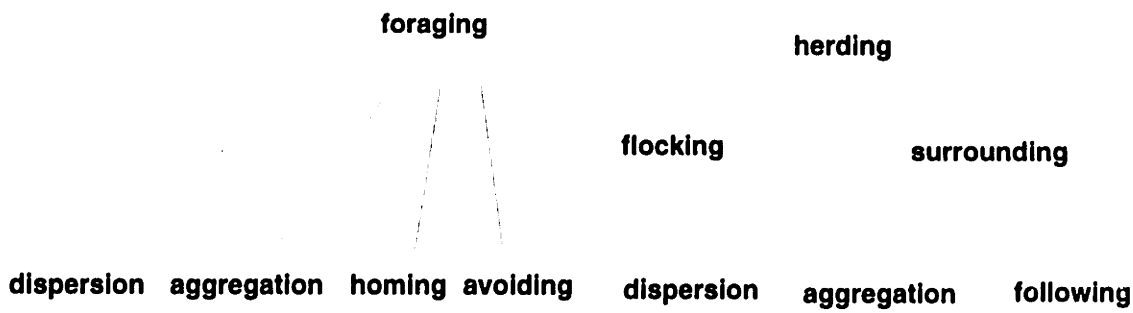



Figure 6-7: An example of applying both direct and temporal combinations to the same basic behaviors to generate various higher-level behaviors. In this case, avoiding is used to generate flocking, and flocking is used in foraging. Similarly, aggregation is used in foraging and in surrounding.

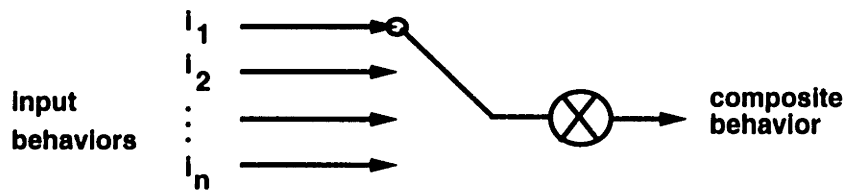



Figure 6-8: The general form of temporal behavior combinations switches between mutually exclusive behaviors. Only one behavior is active at a time, resulting in a behavior sequence triggered by different sensory conditions.

are based on continuous function of the input parameters, the same nodes can be used in multiple combinations. Figure 6-6 illustrates the use of the same basic behaviors (*aggregation* and *following*) to construct two different types of nodes for *surrounding*, and then combining both in *herding*.

Aside from being shared by different direct combination operators, basic behaviors can also be shared between different kinds of operators. Figure 6-7 illustrates how the same inputs, in this case *dispersion* and *avoidance*, can be used in a direct combination, *flocking*, and also in a temporal combination, *foraging*. Temporal combinations are described in the next section.

### 6.0.2 Temporal Combinations of Basic Behaviors

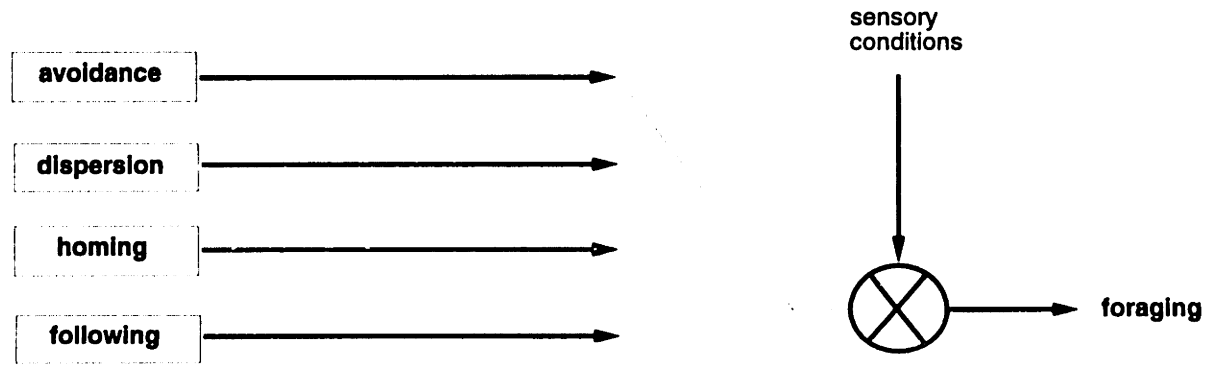



Figure 6-9: The implementation of foraging using a temporal combination of avoidance, dispersion, homing, and following. Each triggered by different sensory conditions, the behaviors collectively result in foraging.

Basic behaviors and their direct combinations achieve and maintain single goals. For example, *dispersion* achieves the goal of establishing a minimum distance between all of the agents while *following* maintains the goal of preserving a queue of moving agents each of which is within a given distance and direction from its neighbors. In order to achieve higher-level tasks defined by multiple sequential goals, basic behaviors must be properly temporally combined.

Such combinations are temporal sequences of basic behaviors, each of which is triggered by appropriate conditions in the environment (figure 6-8). Combining interactions temporally relies on the agents' ability to perceive the state that triggers a behavior change. Given this ability, simple finite state machine controllers can be designed to generate a variety of multi-goal behaviors. This method is illustrated on an implementation of *foraging*, a group task of gathering objects ("food") from the environment (figure 6-9).

In foraging, the high-level achievement goal of the group is to collect objects from the environment and deliver them home. This complex behavior is a prototype for a variety of tasks including harvesting, office cleaning, and clearing toxic spills and mine-fields. For the foraging task, in addition to having the basic behavior

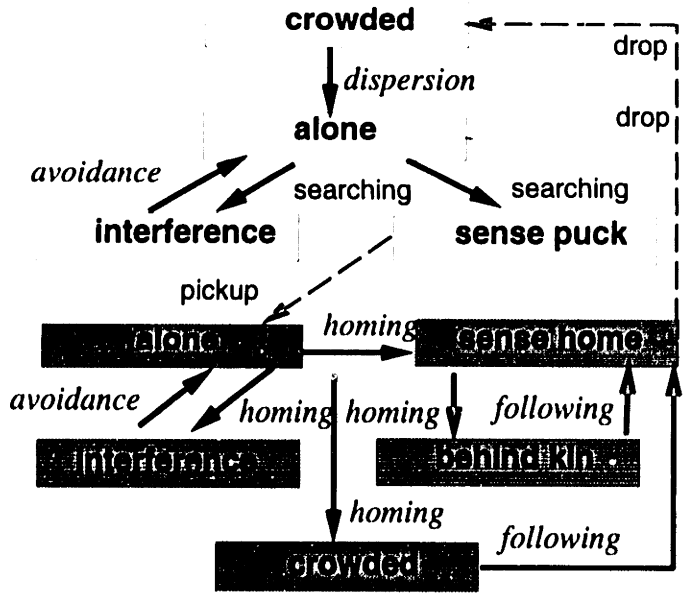



Figure 6-10: The finite-state controller resulting from the temporal interleaving of the four basic behaviors used to implement foraging. This figure shows the conditions as boxed states, i.e., nodes of the graph, and behaviors as labels on the arcs. Shaded states indicate that the robot is carrying a puck.

repertoire, individual agents are also able to search for pucks, pick them up, and drop them. As illustrated in figure 6-10, foraging is initiated by dispersion[1], and then searching. Finding an object triggers homing. Encountering another agent with a different immediate goal, as manifested by its external state, e.g., not carrying a puck [2] induces avoidance. Conversely, encountering kin, another agent with the same external state, such as carrying a puck, or heading toward a pile of pucks, triggers following. Reaching home and depositing the object triggers dispersion or searching. Figure 6-10 gives a finite-state controller view of the task.

Foraging demonstrates how basic behaviors can be temporally combined into a higher-level compound behavior. The combination is simple in that conflicts between two or more interacting agents, each potentially executing a different behavior,

[1]Floreano (1993) shows that evolved systems of ants favor dispersion as the first step in foraging.

[2]Since the robots cannot directly sense each other's external state, this information is broadcast via the radios. A robot that has a puck transmits a limited-radius message about its "puck state."

are resolved uniformly due to agent homogeneity. Since all of the agents share the same goal structure, they will all respond consistently to environmental conditions. For example, if a group of agents is following toward home and it encounters a few agents dispersing, the difference in the agents' external state will either induce following agents of the same kind or avoiding agents of any other type, thus dividing or "specializing" the group again.

Foraging is just one example of a variety of spatial and object manipulation tasks that can be implemented with the described architecture and the given basic behaviors. Other tasks include sorting objects, building structures, surveying and mapping an unknown territory, and many others.

The next section demonstrates robot implementations of composite behaviors.

## 6.1 Implementations of Compound Behaviors

### 6.1.1 Flocking

As described earlier, *flocking* is a form of structured group movement[3] that serves to minimize interference, protect individuals, and enable efficient information exchange. Flocking was implemented with the following simple algorithm:

```
Flock:
Sum outputs from Avoid and Aggregate.
If in front of all sensed robots,
   slow down.
If in back of all sensed robots,
   speed up.
```

The choice of weights on the different behavior outputs was determined by the dynamics and mechanics of the agents, the ranges of the sensors, the agents' turning radii, and their velocity. In the robot implementation, flocking consisted of a com-

[3]A **group** is defined to be a collection of size three or more.

bination of avoidance and aggregation only, by using the appropriate combination of $\delta_{avoid}$ and $\delta_{aggregate}$ thresholds.

The idea that flocking can be generated by simple rules has been popular among many researchers. For example, DeScnutter & Nuyts (1993) and Goss, Deneubourg, Beckers & Henrotte (1993) show a similar approach by demonstrating how simple rules can result in gull flock formation in simulation. Even more directly, Reynolds (1987) presents an elegant graphical simulation of bird flocking. However, the robot implementation[4] required more rules due to the more complex dynamics.

Like following, flocking is a coordinated-motion behavior which is best evaluated by testing its duration, repeatability and robustness. As expected, performance of flocking was dependent on the size of the flock. Small flocks, consisting of four or fewer agents, were unstable, while larger flocks remained stable even if any of the agents failed due to mechanical problems. Figure6-11 demonstrates just such a case, in which one of the agents' position sensors failed and it quickly diverged from the flock.

The utility of flocking can easily be seen by measuring its interference minimizing properties. For instance, it is much more efficient than individualistic homing as the number of homing agents increases. Although flocking involves a compromise between individual and group goals, which may make an individual agent's path locally suboptimal, the collective behavior is more efficient in that all of the agents get to the destination faster than they do, on the average, using individualist greedy homing strategies[5] Homing can be directly added to the combination operator as follows.

Formally:

if $|\mathcal{N}(R, \delta_{flock})| = 0$

[4]Done by Matthew Marjanović.

[5]Roads and highways are human forms of traffic flocking. They impose structure on the collective motion so as to minimize average interference.

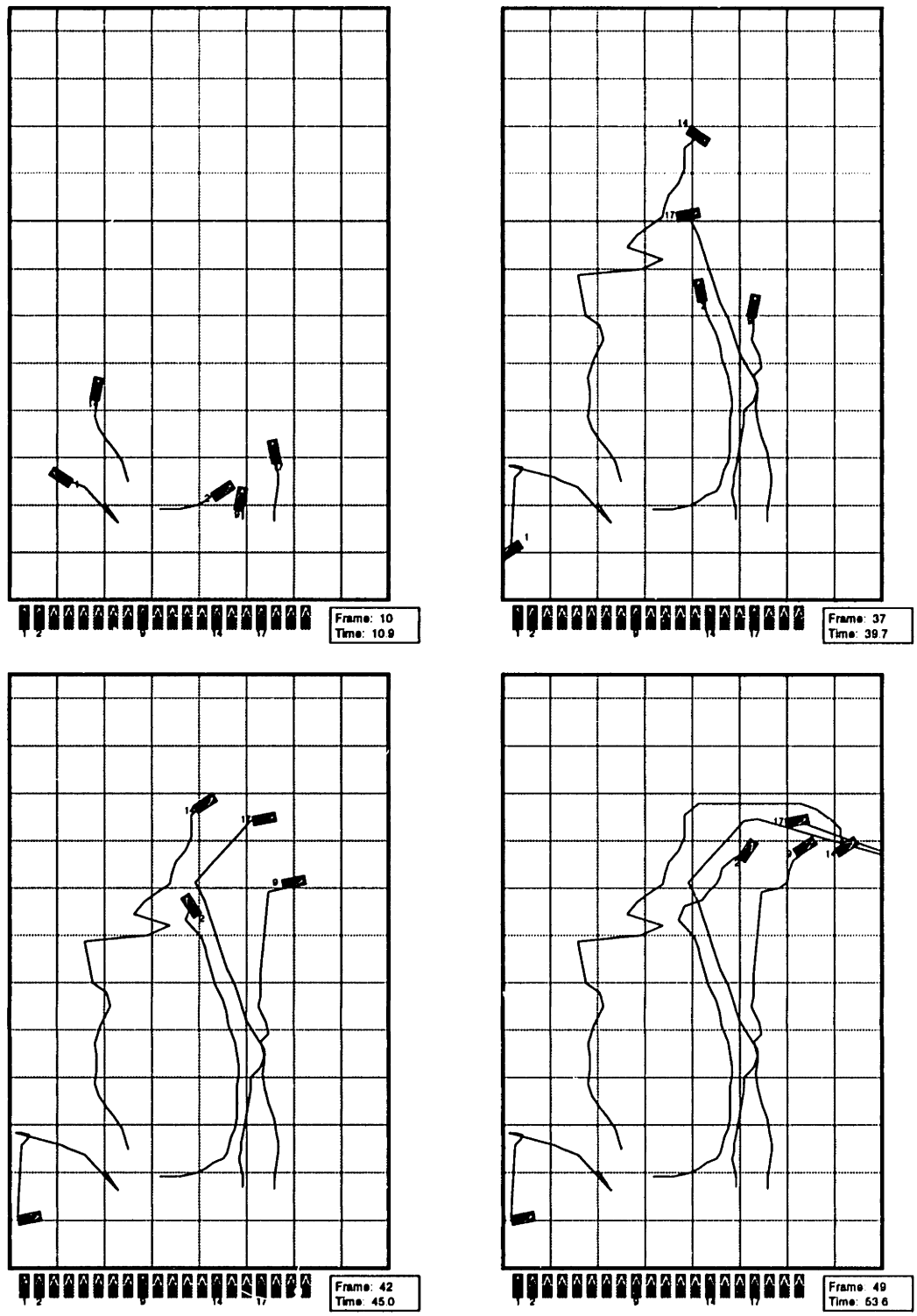



Figure 6-11: Flocking behavior of five robots. One of the robots diverges, without affecting the behavior of the others. Due to stalling, the robot falls behind the flocking zone before it is able to recover and rejoin it. The rest of the robots reorganize within the plot and maintain the global structure.

$command(homing)$

if $|\mathcal{N}(R, \delta_{flock})| \neq 0$

  if $d_{c_{\delta_{flock}},R} > \delta_{aggregate}$

    $command(aggregation)$

Typical flocking behavior is shown in figures 6-12, 6-13, and 6-14. Flocking was also tested in more challenging environments. For example, a barrier roughly the size of two robots was presented in front of the flock as the flock was moving. As expected, the flock split into two groups around the obstacle and rejoined on the other side. Empirical data for this and other experiments is available on video tape.

### 6.1.2 Foraging

Efficient foraging is a validation of the proposed behavior combination strategy. Foraging was tested on two different types of robots and environments, and its performance was repeatable and robust. It is important to note that the shown implementation of foraging did not attempt to directly optimize the amount of time required to collect all of the pucks, although this criterion was indirectly minimized by diminishing interference between agents. In order to evaluate the quality of the foraging strategy itself, it would need to be compared to different foraging algorithms. That, however, was not the purpose of the described experiments.

Instead, foraging was tested to validate that basic behavior sequencing was appropriate and robust, and that the higher-level task of collecting pucks was accomplished effectively rather than optimally. Figures 6-15, 6-16, and 6-17 demonstrate typical foraging performance. They show time slices at different points during the foraging process. Each time slice is marked with a frame number and the elapsed experiment time. Most foraging runs were terminated after 15 minutes, at which time about two thirds of the pucks were collected. The duration of the runs was largely due to the

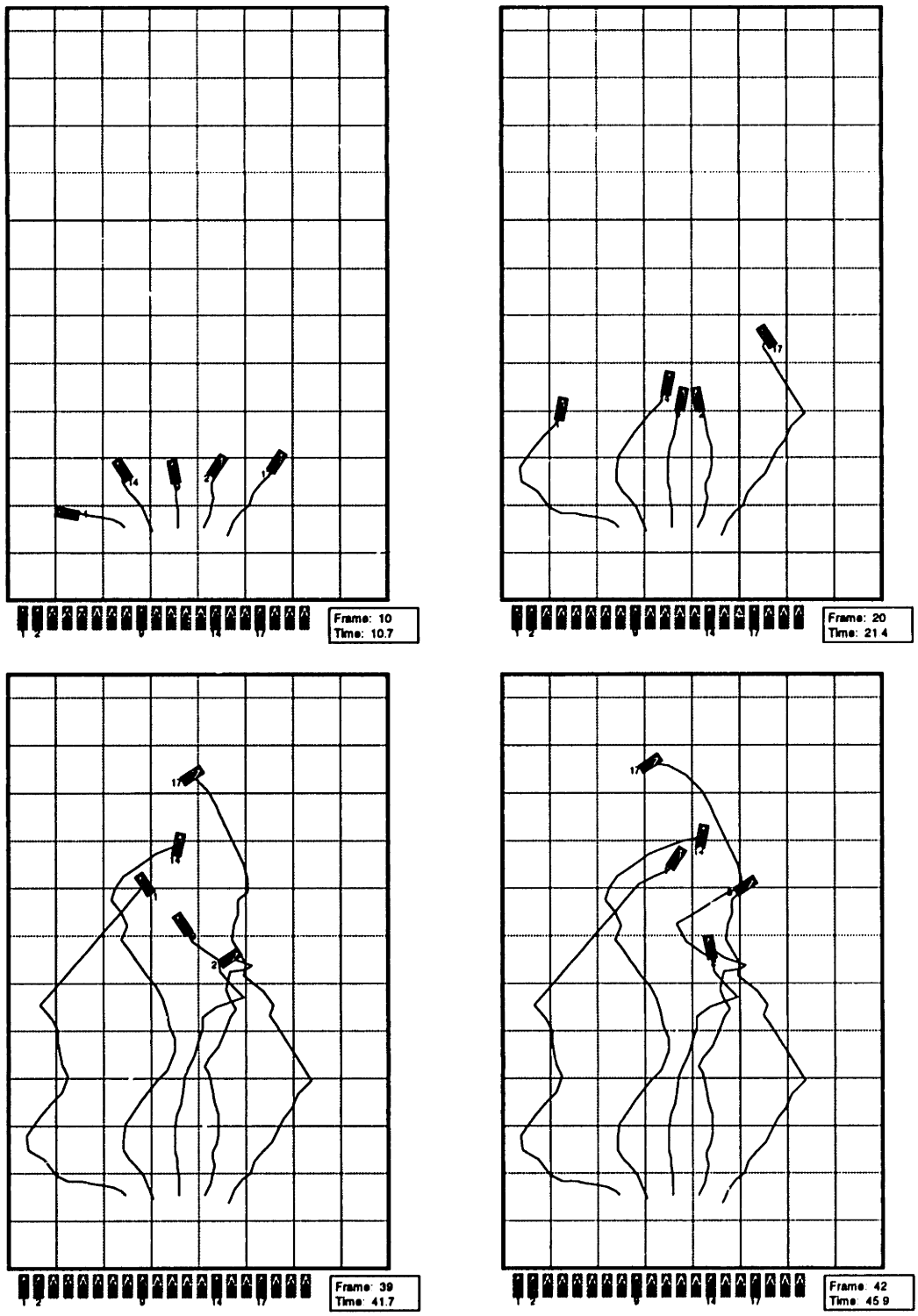



Figure 6-12: Flocking behavior of the same five robots in another trial with random initial positions. The robots maintain a coherent flock, in spite of the often large position errors sensed by individuals. These errors are manifested in the variability in the spacing between the robots as the flock moves.

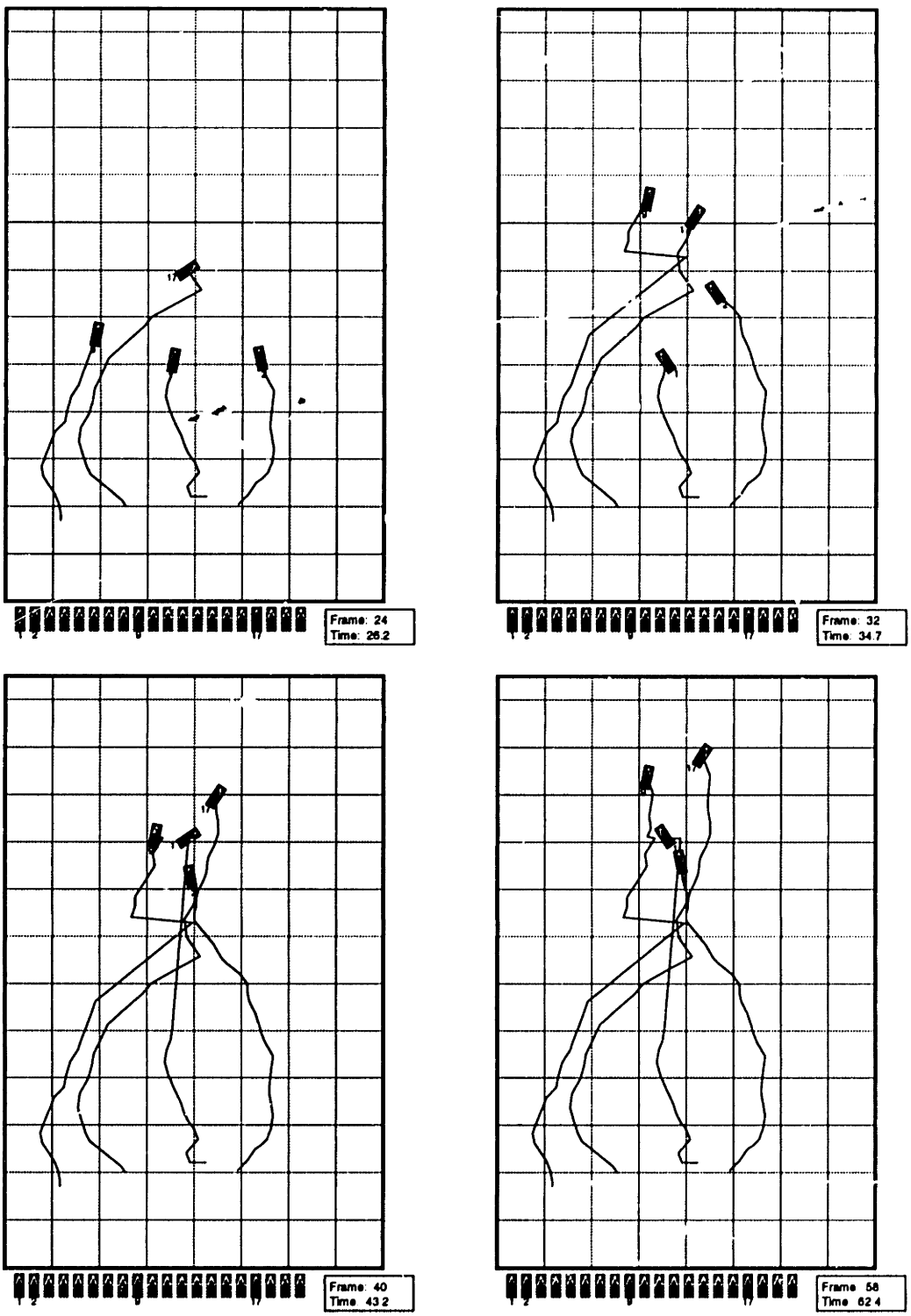



Figure 6-13: Flocking behavior of four robots. The robots are initiated in a line and they quickly move into a flock organization. There are no fixed leaders so robots at the front of the flock occasionally exchange places with others due to velocity and other control variations, all the while maintaining the flock formation.

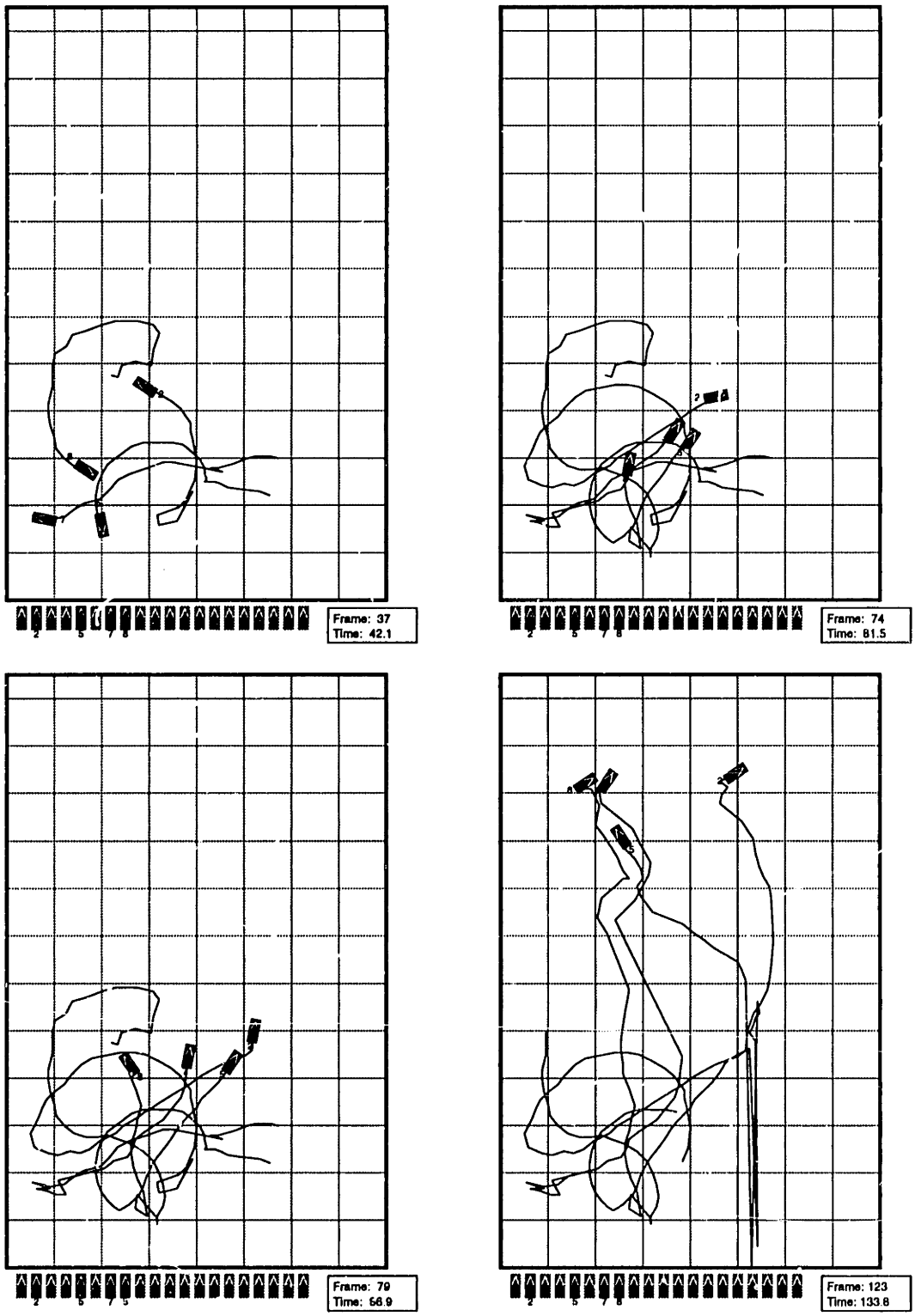



Figure 6-14: Another run of four flocking robots. The robots are started in a difficult initial configuration: facing each other. After an initial reordering they establish a flock and maintain it as they move across the workspace.

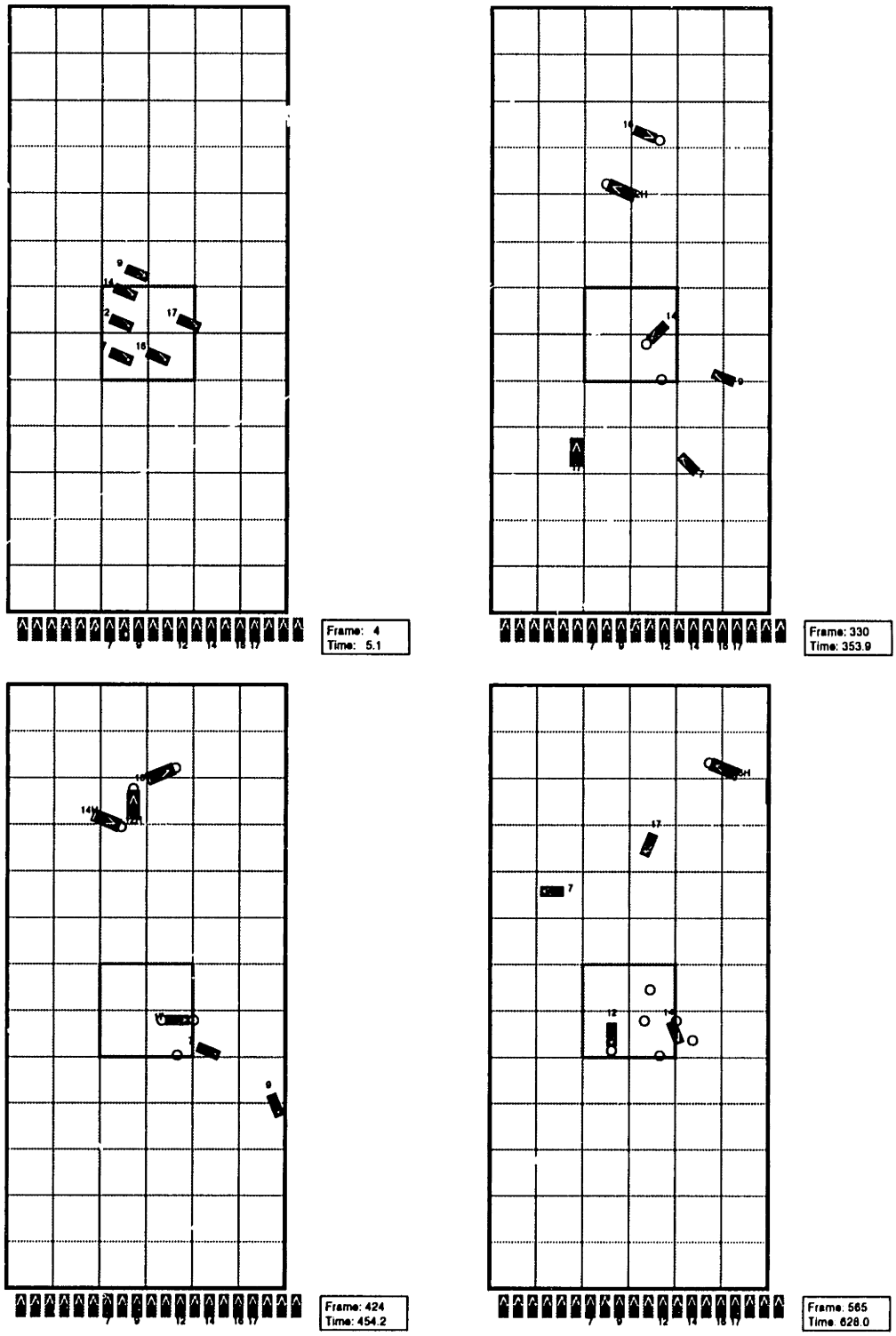



Figure 6-15: Foraging behavior of six robots. The robots are initiated in the home region. After dispersing, they search for pucks, pick them up, and take them home. If they encounter another robot with a puck while they are carrying one, they follow, as shown in the third frame of the data. After some time the pucks accumulate in the home region.

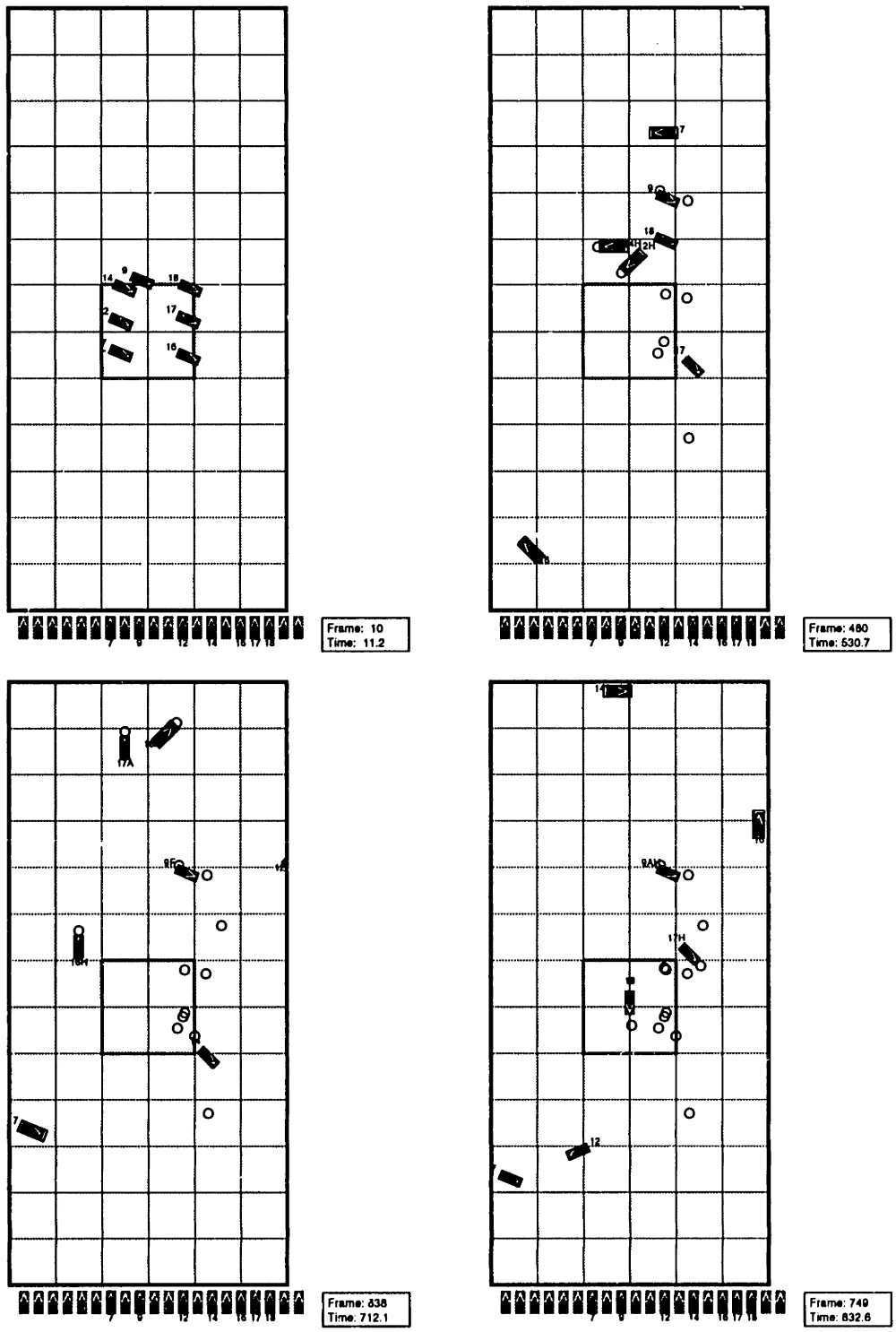



Figure 6-16: Foraging behavior of seven robots. In this experiment more robots effectively manage to transport a larger number of pucks home than the group of six robots shown above. Robots with boxes around them indicate avoiding behavior, such robots 14 and 16 in the last frame of the data, avoiding the walls of the workspace.

inefficient search strategy: the robots did not remember where the pucks were. An improved strategy, in which the robots remembered the location of the pucks and returned to it repeatedly until all of the pucks were transported, was used as a part of the group learning algorithm described in Chapter 9.

Not taking advantage of exact puck location was at least partially justified. Foraging experiments clearly demonstrated a side effect which influenced the global behavior and performance of the system but was not readily predictable. Over the course of each experimental run, as robots traveled to and from the puck site, the pucks were pushed around and gradually dispersed over an expanding area. This, in turn, affected the global behavior of the system, since the robots were now more likely to find picks by random search, and transmitting the location of a found puck would not have been beneficial as the pucks did not remain clustered.

In an environment sufficiently dense with agents and pucks, it is unavoidable that pucks will be moved about and dispersed. This type of side effect is rarely predicted by analytical models, but nonetheless has a large effect on the system's behavior.

It is important to note that foraging could be accomplished by a single agent, so the task itself does not require cooperation. Thus, the goal of the collective solution is to accelerate convergence with the growing size of the group. Arkin et al. (1993) describe simulation results of a similar task with varying numbers of agents and inter-agent communication. Complementary to the results presented here, they find that performance improves with simple communication. They also report an improvement of performance with growing group size up to a fixed point for the particular retrieval and gathering task. This result is in agreement with the results shown here that illustrate the interference effects of larger and thus higher-density groups in confined workspaces. Given the number of pucks to be collected, the collective solutions proposed here always outperformed a single agent, but as the group size grew, so did the importance of behaviors that minimized interference. This relationship will be

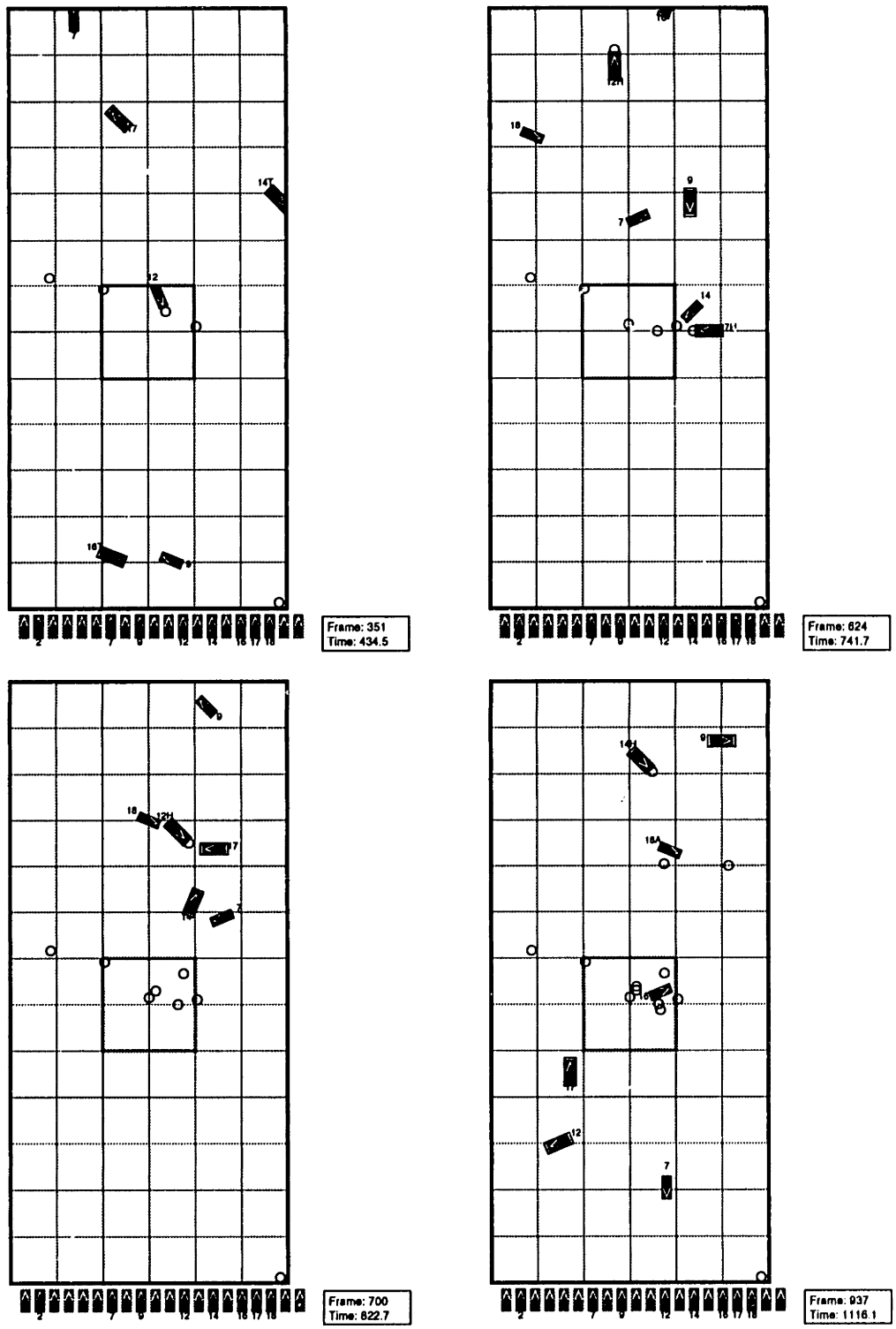



Figure 6-17: Another example of foraging behavior with seven robots. As before, the robots gather around the area with pucks (at the "top" of the workspace), picking them up, and gathering them in the home region. Interference is resolved by avoiding following.

further elaborated on in Chapter 9 which describes the approach to group learning.

## 6.2 Minimizing Inter–Behavior Interference

This chapter has dealt with the issues of combining behaviors and argued that the combination of internally consistent behaviors, based on mutually exclusive conditions, and externally consistent interactions, based on compatible social rules, results in globally coherent group behaviors.

Direct and temporal combination operators both rely on the agents' ability to all respond to external conditions consistently. As long as all of the agents follow consistent social rules, i.e. use compatible social repertoires, conflict is minimized. Homogeneity simplifies the task of combining behaviors, since the concern of conflict between behaviors need only be considered at the level of the individual agent. As long as the behaviors within a single agent do not interfere, they will not do so in a homogeneous group.

Mutually exclusive behavior conditions guarantee simple action selection. However, what happens when two agents with different immediate goals come into contact and their actions interfere with each other? This is in fact analogous to the behavior selection problem at the level of control of a single agent. Not surprisingly, a similar strategy to mutually exclusive conditions applies to conflict resolution at this level as well.

Given a choice of two or more possible behaviors, whether it be at a single agent or multi agent level, which one should be executed? As has been argued previously (Matarić 1992*a*), an unambiguous precedence hierarchy between the competing behaviors or agents is necessary in order to guarantee a globally consistent result. Thus, ensuring minimal higher-order effects and interference in a heterogeneous society can be accomplished by a strict hierarchy of control, the structure of which is determined by the task. When two or more agents come into contact, one must take precedence

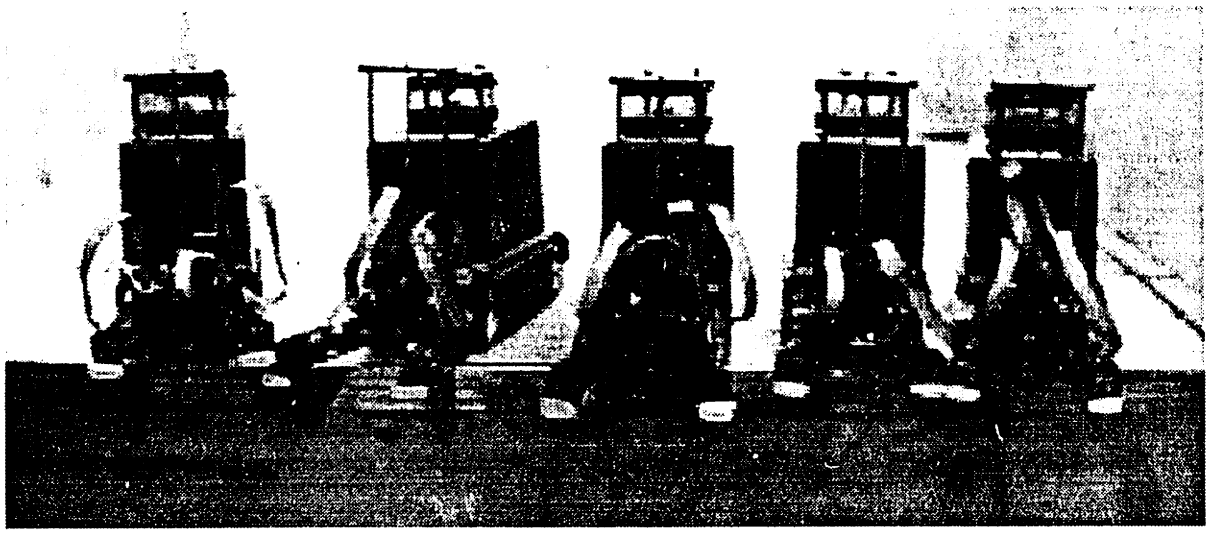


Figure 6-18: Docking behavior, resulting from a combination of three simple rules. See Appendix A for details.

over the others. This strict precedence requirement can be relaxed or even made stochastic, but conflict resolution in the form of some kind of behavior dominance is necessary. This type of social organization appears quite stable and ubiquitous in animal and human societies but requires the agents to maintain identities, distinguishing characteristics, and a history of previous encounters (McFarland 1987, Gould 1982), thus demanding higher cognitive overhead.

## 6.3 Summary

This chapter has addressed methods for minimizing interference between behaviors on a single agent and behaviors between two or more interacting agents. A general architecture was introduced for combining a finite set of basic behaviors into an infinite repertoire of higher level behaviors. The two types of combination operators used by the architecture were demonstrated on composite spatial behaviors of flocking and foraging, implemented and tested on the collection of mobile robots. Additional examples of behavior combinations, such as *docking* and *parking* (figure 6-18) are shown in Appendix A.

The next chapter introduces a methodology for automating the behavior combination process through the use of learning.

# Chapter 7

# Learning in Situated Multi–Agent Systems

The goal of this thesis is to study the problem of scaling up AI, and to introduce new insights and tools for this purpose. The first part of the thesis has dealt with the problem of synthesizing intelligent group behavior, a scaling up of the agents' environment complexity. This part of the thesis addresses scaling up the agents' cognitive complexity, by adding learning into the agents' repertoire of capabilities.

## 7.1 Why Learn?

Learning has two purposes universal across domains. It is useful for:

- adapting to external and internal changes
- simplifying built–in knowledge

The ability to cope with changes in the environment is termed *adaptability.* It allows agents to deal with noise in their internal and external sensors, and with inconsistencies in the behavior of the environment and other agents. Adaptability comes at a phenotypical and cognitive cost, so creatures are adapted only to a specific niche.

| | |
|---|---|
| **problem** | learning in complex situated domains |
| **assertion** | traditional reinforcement learning<br>must be reformulated |
| **approach** | replace states, actions and reinforcement<br>with conditions, behaviors, and heterogeneous reward functions<br>and progress estimators |
| **validation** | implement learning on a group of mobile robots<br>learning to forage |

Table 7.1: A summary of the situated learning problem addressed here, and the structure of the proposed solution.

Consequently, all creatures, natural and otherwise, fail under certain conditions. The purpose of learning is to make the set of such conditions smaller.

Adaptability does not necessitate learning. Many species are genetically equipped with elaborate "knowledge" and abilities, from the very specific, such as the ability to record and utilize celestial maps (Waterman 1989), to the very general, such as plasticity in learning motor control (McFarland 1987) and language (Pinker 1994). But genetic code is finite. In fact, primate and human cortical neural topology is too complicated to fully specify in the available genome, and is instead established by spontaneous synaptic firing *in utero* and in the first decade of life (Vander et al. 1980). In addition to allowing for genetic parsimony, learning is useful for optimizing the agent's existing abilities, and necessary for coping with complex and changeable worlds. It is often argued that societies exist largely for conservation and propagation of behavior strategies too complex to be passed on genetically.

The answer to the built–in versus learned tradeoff varies across species and environments. The work described here addresses this fundamental tradeoff in the domain of situated multi–agent systems.

The rest of the thesis will address the following problem: how can a collection of situated agents learn in a group environment? This problem will be addressed in a nondeterministic, noisy and error-prone domain with stochastic dynamics, in which the agent does not have an *a priori* model of the world.

I propose a formulation of reinforcement learning that uses a different level of description in order to make the state space manageable, and thus make learning possible. Furthermore, I present two methods for shaping reinforcement in order to take advantage of information readily available to the agent in order to make learning more efficient. These ideas are validated by demonstrating an effective learning algorithm on a group of robots learning to forage. Table 7.1 summarizes the problem and the approach.

## 7.2 Review of Relevant Learning Work

There are many things an agent can learn, but not many ways in which it can learn it.[1] According to what is being learned, existing approaches can be classified as follows:

- learning control
- learning new behaviors
- learning to select behaviors/actions
- learning declarative knowledge

**Learning declarative knowledge** is one of the founding areas of AI but also one that is least directly related to the work in this thesis. The only type of declarative knowledge that situated agents have had to deal with to date are maps of the environment. Much of robotics literature deals with the problem of constructing and

[1] In fact, one of the difficulties facing the learning community is the lack of clear structure that delineates the applicability of a given learning methodology.

updating such maps in variety of situated domains (see Matarić (1990*a*) for a review of the literature). Maps and world models are closely tied to action in the world, which is why they are the primary type of declarative knowledge so far used in situated agents[2]. In contrast, this thesis focuses on procedural knowledge which is directly tied to acting and interacting in the world. All of the remaining learning categories are directly tied to action.[3]

**Learning control** is a growing field based on *adaptive control*, a branch of control theory. Problems in adaptive control deal with learning the forward or inverse model of the system, i.e., the plant. Forward models provide predictions about the output expected after performing an action in a given state. Analogously, inverse models provide an action, given the current state and a desired output (Jordan & Rumelhart 1992). Learning control has been applied to a variety of domains and has used a number of different learning methodologies. Connectionist algorithms are most popular, (see Miller, Sutton & Werbos (1990) for a representative collection), but other approaches have also been studied (e.g., Atkeson, Aboar, McIntyre & Reinkensmeyer (1988), Atkeson (1990), Schaal & Atkeson (1994)). Adaptive control problems typically deal with learning a complex dynamical system involved in moving multi-jointed manipulators, moving objects, and systems with non-linearly coupled degrees of freedom.

**Learning new behaviors** deals with the problem of acquiring strategies for achieving particular goals. Because the notion of behavior is not well defined, neither is the behavior learning problem, as discussed in the next section.

[2]Note: not all maps are explicit and declarative. See Matarić (1990*a*) for examples.

[3]Author's bias: declarative learning can be further divided into as many interesting categories, but is not the area pursued here.

### 7.2.1 Learning New Behaviors

Behavior was defined to be a control law with a particular goal, such as *wall-following* or *collision avoidance*. The definition of behavior is necessarily general. It is meant to capture a level of description above basic control, but it does not specify what that level is, since it varies with the domain. Furthermore, the concept of behavior contains informal notions about flexibility and generality, but which are difficult to state precisely without domain-specific grounding.

Consequently, most learning control problems appear to be instances of behavior learning, such as learning to balance a pole (Barto, Sutton & Anderson 1983), to play billiards (Moore 1992), and to juggle (Schaal & Atkeson 1994). Furthermore, work on action selection, deciding what action to make in each state, can be viewed as learning a higher-level behavior as an abstraction on the state-action space. For example, a maze-learning system can be said to learn a specific *maze-solving* behavior.

Genetic learning has also addressed learning behaviors in simulated worlds (Koza 1990). Since learning behaviors requires finding appropriate parameter settings, it can be cast as an optimization problem, for which genetic algorithms are particularly well suited (Goldberg 1989). However, since genetic algorithms operate on an abstract encoding of the learning problem, the encoding requires a good model of the agent and the environment in order to generate useful behaviors. Since the problem of modeling situated worlds is notoriously difficult, few genetic algorithms have produced behaviors that successfully transferred to physical systems (Cliff, Husbands & Harvey 1993, Gallagher & Beer 1993).

However, none of the above learning approaches can be said to learn new behaviors according to the precise definition of the problem. The posed "behavior learning problem" (Brooks & Matarić 1993) requires the agent not only to acquire a new behavior, using its own perceptual and effector systems, but also to assign some semantic label to it, and be able to recognize it as a coherent and independent unit,

separate from the rest of its activities. Behavior learning appears to require bridging the elusive signal–to–symbol[4] gap, and places the burden on the learning system.

Given this definition, no existing work performs behavior learning. Learning control and learning action selection are not strictly instances of behavior learning because in both cases, by definition, only a single behavior is learned and no further abstraction is performed. Similarly, genetic algorithms do not address the stated behavior learning problem either, because in their domain the semantics are also added by the designer.

The signal–to–symbol problem is one of the hallmark challenges in AI. Because it bridges a gap between two already estranged communities, it has not received much attention, although some work has been started on the topic. For example, Thrun & Mitchell (1993) demonstrate a connectionist approach to learning coherent strategies. The separation is not assigned by the designer but is instead selected by the network. Not surprisingly, the result is not semantically meaningful to a human observer, but is nonetheless well suited to the problem at hand. In general it is unlikely that automatically generated "behaviors" and their associated "concepts" and "symbolic representations" will map neatly from the agent's sensorium into the observer's semantic space. Thus it is difficult to work on the problem, as it is important to not bias the learner inappropriately, but to be able to evaluate its performance. The situated domain is particularly well suited for this type of work as it allows for grounding the agents' learning in physical behavior which is observable and thus can be evaluated.

If learning new behaviors is learning *how* to do something, then **learning to select behaviors** is learning *when* to do it. Behavior selection has not been extensively studied so far, largely due to the lack of formalization of "behavior" as building block for control. The work that has been done on the topic (e.g., Maes & Brooks (1990) and Maes (1991)) has used reinforcement learning techniques. Learning behavior selection

[4]Even if the notion of "symbol" is very limited, as described.

is by definition a reinforcement learning problem as it is based on correlating the behaviors the agent performs and the feedback it receives as a result. The remainder of this thesis addresses behavior selection and reinforcement learning.

## 7.3 Reinforcement Learning

Reinforcement learning (RL) is a class of learning methodologies in which the agent learns based on external feedback received from the environment. The feedback is interpreted as scalar (positive or negative) reinforcement. The goal of the learning system is to maximize positive reinforcement (reward) and/or minimize negative reinforcement (punishment) over time. Traditionally, the learner has no explicit built-in knowledge about the task and does not receive direct instruction or answers from the environment. The learner produces a mapping of states to actions called a *policy*.

Reinforcement learning originated in Ivan Pavlov's classical conditioning experiments (see Gleitman (1981)). Embraced by behaviorism, stimulus–response learning became the predominant methodology for studying animal behavior in psychology and biology. Ethology, the study of animals in their natural habitats, developed in response to the tightly controlled laboratory experimental conditions commonly used by behaviorists. In the mean time, computer scientists adopted and adapted reinforcement learning to various machine learning problems.

The first well known application of positive and negative reinforcement to a machine learning task was done by Marvin Minsky, who set up a maze-learning system based on reward and punishment (Minsky 1954). Soon thereafter, Arthur Samuel successfully formulated the problem of learning to play checkers in terms of RL (Samuel 1959). Subsequently, RL was applied to a variety of domains and problems, most notably in the Bucket Brigade algorithm used in Classifier Systems (Holland 1985), and in a class of learning methods based on Temporal Differencing, introduced by Sutton (1988). Reinforcement learning has been implemented with a variety of algo-

rithms ranging from table-lookup to neural networks, and on a broad spectrum of applications, including tuning parameters and playing backgammon.

This section discusses some important issues that arise when traditional models of reinforcement learning, and algorithms applied to those models, are applied to situated domains such as the complex multi-agent domain we are working with.

### 7.3.1 Traditional Models of Reinforcement Learning

Most computational models of reinforcement learning are based on the assumption that the agent-environment interaction can be modeled as a Markov Decision Process (MDP), as defined below:

1. The agent and the environment can be modeled as synchronized finite state automata.

2. The agent and the environment interact in discrete time intervals.

3. The agent can sense the state of the environment and use it to make actions.

4. After the agent acts, the environment makes a transition to a new state.

5. The agent receives a reward after performing an action.

While many interesting learning domains can be modeled as MDPs, situated agents learning in nondeterministic, uncertain environments do not fit this model. The next section describes the reasons why, by addressing each of the model assumptions in turn.

### 7.3.2 State

Most RL models are based on the assumption that the agent and the environment are always in a clearly-defined state that the agent can sense. In situated domains,

however, the world is not readily prelabeled into appropriate states, and the world state is not readily and consistently accessible to the agent.

The state of a situated agent consists of a collection of properties, some of which are discrete (e.g., the inputs from binary sensors), others continuous (e.g., the velocities of the wheels). A monolithic descriptor of all of those properties, even for the simplest of agents, is prohibitively large. Thus, it scales poorly and results in a combinatorial explosion in standard RL. Most models to date have bypassed such continuous state by presuming higher–level sensory operators (e.g., "I see a chair in front of me."), but such assumptions have shown to be unrealistic in systems using physical sensors (Agre & Chapman 1990, Brooks & Matarić 1993). In general, the problem of partitioning a continuous world into discrete states is difficult and unsolved. Furthermore, even if a reasonable partitioning of the world is available, there may be no mapping from the space of sensor readings to this partitioning.

### Perceptual Aliasing

In addition to being error–prone, sensors have limited abilities. Instead of providing descriptions of the world, they return simple properties such as presence of and distance to objects within a fixed sensing region. Consequently, they are unable to distinguish between all potentially relevant world states. The collapse of multiple states into one results in **perceptual aliasing**, a many–to–one mapping between world and internal states. The inability to distinguish different states makes it difficult and often impossible for the learning algorithm to assign appropriate utility to actions associated with such states. Within computational RL, this problem was first addressed by Whitehead & Ballard (1990) and Whitehead (1992), who proposed an approach to adaptive active perception and action that divided the control problem into two stages: a state identification stage and a control stage, and applied appropriate learning methods to each. The approach was demonstrated on a simulated block

stacking task, but is yet to be validated in an embodied domain.

### Generalization

Given the described approaches to encoding state, RL approaches are marred with extremely large state spaces. **Input generalization** refers to the process of collapsing states into functional equivalence classes.

Human programmers perform generalization implicitly whenever they use clever orderings of rules, careful arbitration, and default conditions, in crafting control strategies. They minimize ambiguity and maximize parsimony by taking advantage of their domain knowledge. In contrast, RL approaches are devoid of domain knowledge, and must employ statistical generalization.

Statistically based state generalization has been studied in RL. For example, Chapman & Kaelbling (1991) and Mahadevan & Connell (1991*a*) demonstrate complementary approaches for generating sufficiently differentiated state spaces that eliminate irrelevant state bits (Matarić 1991). Chapman & Kaelbling (1991) start with the most general solution (a single, most general state) and iteratively split it based on statistics accumulated over time. When a bit in a state vector is found to be relevant, the state space is split into two subspaces, one with that bit on, and the other with it off. In contrast, Mahadevan & Connell (1991*a*) start with a fully differentiated, specific set of all states, and consolidate them based on similarity statistics accumulated over time. The goal of both approaches is to produce state space trees which are sufficiently differentiated but smaller than the fully exponential space.

Input generalization is also addressed in Classifier Systems (Holland 1986) which use binary strings as state descriptors or classifiers. The state can contain wild cards (#'s) which allow for clustering two states into one, with the flexible grouping potential of full generality (all #'s) to full specificity (all non-#'s). Generalization results in so-called "default hierarchies" based on the relevance of individual bits which are

changed from #'s to specific values. This process is neatly analogous to that of Chapman & Kaelbling (1991) but is strictly more powerful. Briefly, the process of generating more specific classifiers incrementally implements a power set of the state space, thus never needing to keep around a large number of classifiers, as would a power set implementation of the Chapman & Kaelbling (1991) algorithm which keeps relevance statistics on all of the state bits.

The input generalization problem is also addressed by the connectionist RL literature. Multi-layer networks have been trained on a variety of learning problems in which the hidden layers constructed a generalized, intermediate representation of the inputs (Hinton 1990). For example, such an approach was successfully used by Pomerleau (1992) to train a neural network to drive a vehicle based on generalizing visual snapshots of the road ahead. While all of the RL generalization techniques are non-semantic, the table-based methods and Classifier System approaches are somewhat more readable as their results are a direct consequence of explicit hand-coded criteria. Connectionist approaches, in contrast, utilize potentially complex network dynamics and produce effective but largely inscrutable generalizations.

All of the described generalization techniques are effective but require a large number of trials in order to obtain sufficient statistical information for clustering states. As such, they are an incremental improvement of the overwhelmingly slow exponential learning algorithms. The alternative to purely statistical approaches is to take principled advantage of domain knowledge. Just such an approach will be proposed in an upcoming section.

Paradoxically, the unwieldy, fully-exponential state-action search space of standard RL models gives them one of their main positive properties: asymptotic completeness. While hand coded reactive policies take advantage of the cleverness of the designer, they are almost never provably complete. Most irrelevant input states are easily eliminated, but potentially useful ones can be overlooked. Complete state

spaces, on the other hand, guarantee that the agent will, given sufficient time and sufficiently rich reinforcement, produce a provably complete policy. However, this quality is of little use if the world is dynamic and/or the state space is large.

### 7.3.3 State Transitions

In situated domains, world and agent states change asynchronously, in response to events not all of which are caused by and in control of the agent. Events can have delayed effects, actions can take different amounts of time to execute, and can have different consequences under what may appear to be identical conditions. Thus, situated domains cannot be usefully modeled by deterministic automata.

Some nondeterministic environments have been considered by previous RL work with simulated situated agents (e.g., (Lin 1991)). While these approaches have allowed for nondeterministic state transitions, they have always assumed minimal uncertainty in the sensing and action of the agent. Most models of uncertainty have overly simplified error properties, usually represented with Gaussian noise to each sensed state and each commanded action. In contrast, uncertainty in situated domains is not random, but results from structured and repeatable particular dynamics of interaction of the system and the environment which play an important role in the overall behavior of the system.

As an example, consider the properties of realistic proximity and distance sensors. The accuracy of ultrasound sensors is largely dependent on the incident angle of the sonar beam and the surface, as well as on the surface materials, both of which are difficult and tedious to model accurately. Infra- red and vision sensors also have similarly detailed yet entirely different properties, none of which are accurately represented with simple models. Simple noise models are nonetheless tempting, but they produce artificial dynamics that fail to capture the complexity of realistic physical systems. Consequently, many elegant results of simple simulations have not been

successfully repeated on more complex agents and environments.

Given the challenges of realistic uncertainty modeling, it is difficult to obtain transition probabilities for nondeterministic models of situated domains. Models for such domains are not readily available, and must in general be obtained empirically. However, the process of such world exploration is analogous to that employed in learning, and it is not in general easy to tell if obtaining a world model would require any less time than learning a policy for a given set of goals. Consequently, insightful work on learning world models for more intelligent exploration (Sutton 1990, Kaelbling 1990) is yet to be made applicable to complex situated domains.

The multi--robot domain addressed here is complex and model–free. Consequently, this work will focus on the problem of learning a policy directly. The next section describes the general form of RL algorithms that have been used on learning problems without world models.

## 7.4 Reinforcement Learning Algorithms

Reinforcement learning algorithms have the following general form (Kaelbling 1990):

1. Initialize the learner's internal state $I$ to $I_0$.
2. Do Forever:
    a. Observe the current world state $s$.
    b. Choose an action $a = F(I, s)$
       using the evaluation function $F$.
    c. Execute action $a$.
    d. Let $r$ be the immediate reward for
       executing $a$ in world state $s$.
    e. Update the internal state $I = U(I, s, a, r)$
       using the update function $U$.

The internal state $I$ encodes the information the learning algorithm saves about the world, usually in the form of a table maintaining state and action data. The update function $U$ adjusts the current state based on the received reinforcement, and maps the current internal state, input, action, and reinforcement into a new internal state. The evaluation function $F$ maps an internal state and an input into an action based on the information stored in the internal state. The different RL algorithms vary in their definitions of $U$ and $F$.

Watkins (1989) and Sutton (1988) present thorough reviews of reinforcement learning as applied to well-behaved learning problems. Kaelbling (1990) and Whitehead (1992) address learning in situated, also called "embedded", agents and propose algorithms with improved performance, largely applied to simulated domains.

The predominant methodology used in RL is based on a class of *temporal differencing* techniques introduced by Sutton (1988). All of these methods deal with assigning credit or blame to past actions by attempting to predict long-term consequences of each action in each state. Sutton's original formalization of temporal differencing (TD($\lambda$)) deals with such predictions in Markovian environments, and covers a large class of learning approaches. For example, Bucket Brigade, the delayed reinforcement learning method used in Classifier Systems, is an instance of TD (Matarić 1991). The most commonly known and used TD algorithm is Q-learning, introduced by Watkins (1989). Q is defined and explained in Appendix B, as background for subsequent comparison.

Performance properties of various forms of TD applied to Markovian environments have been studied extensively by Watkins & Dayan (1992), Barto, Bradtke & Singh (1993), Jaakkola & Jordan (1993) and others. The convergence properties are generally the result of a dynamic programming basis of the algorithms, which has certain undesirable properties, discussed next.

### 7.4.1 Learning Trials

Proofs of convergence of TD and related learning strategies based on dynamic programming is asymptotic and requires infinite trials (Watkins 1989). Generating a complete policy, however incorrect, requires time exponential in the size of the state space, and the optimality of that policy converges in the limit as the number of trials approaches infinity. In practice, this translates into hundreds of thousands of trials for up to ten-bit states. Thus, even in ideal Markovian worlds the number of trials required for learning is prohibitive for all but the smallest state spaces.

The situated learning problem is even more difficult, however. Assuming an appropriately minimized state space, a learner may still fail to converge, due to insufficient reinforcement. The next section describes the source of this problem.

### 7.4.2 Reinforcement

Delayed reinforcement is considered to be one of the most important problems in reinforcement learning. The problem of assigning delayed reward or punishment is termed *temporal credit assignment* [5]. Temporal credit is assigned by propagating the reward back to the appropriate previous state-action pairs. Temporal differencing methods are based on predicting the expected value of future rewards for a given state-action pair, and assigning credit locally based on the difference between successive predictions (Sutton 1988).

Reward functions determine how credit is assigned. The design of reward functions is not often discussed, although it is perhaps the most difficult aspect of setting up a reinforcement learning algorithm. The more delayed the reward the more trials the learning algorithm requires and thus the longer it takes to converge. Algorithms using immediate reinforcement naturally learn the fastest. For example, a variation

[5]The first statement of the problem is due to Samuel (1959), whose checkers-learning program learned to reward moves which eventually lead to "a triple jump."

of RL with immediate reinforcement has been successfully applied in the domain of six-legged walking (Maes & Brooks 1990). The approach was appropriate given the small size of the search space and the immediate and accurate reinforcement. More delayed reinforcement was used by Mahadevan & Connell (1991*a*) in a box-pushing task, in which subgoals had to be introduced to provide more immediate reward for making the task learnable.

Most reinforcement learning work to date has used one of the following two types of reward: immediate, and very delayed. Situated domains, however, tend to fall in between the two popular extremes, providing some immediate rewards, plenty of intermittent ones, and a few very delayed ones. Although delayed reinforcement, and particularly impulse reinforcement (delivered only at the single goal), eliminates the possibility for biasing the learning, it usually makes it prohibitively difficult. Most situated learning problems do not resemble mazes in which the reward is only found at the end. Instead, some estimate of progress can be made along the way. This estimate can be intermittent, internally biased, inconsistent, and occasionally incorrect, but if used appropriately, it can significantly speed up learning. The approach presented in the next chapter takes advantage of such intermediate estimates to shape reinforcement and accelerate learning.

### 7.4.3 Multiple Goals

Situated agents typically have multiple goals, some of which are maintained concurrently, while others are achieved sequentially. For example, in the foraging task, an agent maintains a continuous low-level goal of collision avoidance, also keeps a minimal distance from other agent in order to minimize interference, may attempt to remain in a flock, and may be heading home with a puck.

Most RL models require that the learning problem be phrased as a search for a single-goal optimal policy, so that it can be specified with a global reward function.

Not surprisingly, if the world or the goal changes, a new policy must be learned, using a new reward function. The existing policy will conflict with the new learning and will need to be "forgotten."

In order to enable learning of a multi-goal policy, the goals must be formulated as subgoals of a higher-level single optimal policy. Therefore they must be sequential and consistent. Singh (1991) addresses such an approach to multiple goals in a simplified navigation problem in which a simulated agent must reach three special states in a particular order. In order to enforce the specific sequence, the state space must explicitly encode what goals have been reached so far, so extra bits are added to the state vector. Although a natural extension of the RL framework, this method does not address concurrent goals and does not scale well.

Sequences of goals fail to capture the dynamics of complex situated worlds and agents that may have one or more high-level goals of achievement, and also a number of maintenance goals, the interaction of which has important effects on the agents' behavior.

A more general solution to multiple goals within the traditional framework is to use separate state spaces and reinforcement functions for each of the goals. Whitehead, Karlsson & Tenenberg (1993) describe such an approach and discuss methods for merging existing policies for each of the goals. The work focuses on comparing merging strategies, assuming that the necessary information for utility evaluation is available. However, this may not be a reasonable assumption in situated domains. This approach scales better than the last and is more general, but pushes off the multiple-goal problem to a higher level of control.

## 7.5 Related Work in Learning

Compared to the volume of existing work on learning in simulation, the number of learning strategies that have been applied to physical situated systems is relatively

small.

As described earlier, Maes & Brooks (1990) applied a statistical learning reinforcement technique using immediate reward and punishment in order to learn behavior selection for walking on a six-legged robot. Kaelbling (1990) used a simple mobile robot to validate several RL algorithms using immediate and delayed reinforcement to learn obstacle avoidance. Mahadevan & Connell (1991*b*) experimented with Q-learning using monolithic and partitioned goal functions for learning box-pushing on a mobile robot.

Aside from traditional unsupervised reinforcement learning methods described above, other techniques have also been explored. Pomerleau (1992) used a supervised connectionist learning approach to train steering control in an autonomous vehicle. Parker (1994) implemented a parameter-learning scheme for adjusting activation thresholds used to perform task allocation in a multi-robot system. By far the largest effort in learning on mobile robots has been in the field of learning maps of the environment and using them for more efficient navigation (for a review see Matarić (1990*a*)).

The work presented here is, to the best of the author's knowledge, the first attempt at applying reinforcement learning to a collection of physical robots learning a complex task consisting of multiple goals. Tan (1993) has applied traditional RL to a simulated multi-agent domain. Due to the simplicity of the simulated environment, the work has relied on an MDP model which was not applicable to this domain. Furthermore, Tan (1993) and other work that uses communication between agents relies on the assumption that agents can correctly exchange learned information. This does not hold true on physical systems whose noise and uncertainty properties extend to the communication channels.

## 7.6 Summary

This chapter has overviewed the key properties of reinforcement learning strategies based on Markov Decision Process models, and their implications on learning in situated domains. Learning algorithms based on dynamic programming programming and traditionally applied to such Markovian domains were also discussed. Finally, related robot learning and reinforcement learning work was reviewed.

Two main problems arise when the standard MDP formulation is applied to our multi-agent domain: the state space is prohibitively large, and delayed reinforcement is insufficient for learning the foraging task. The next chapter introduces a way of reformulating the learning problem in order to make learning both possible and efficient in the complex domain used in this work.

# Chapter 8

# The Learning Approach

This chapter describes a formulation of the proposed reinforcement learning problem in order to make learning possible and efficient in the complex situated domain at hand, as well as in situated domains in general.

**A model for situated learning should:**

1. minimize state space
2. maximize learning at each trial

This section will discuss the following ways of minimizing the state space in order to make the learning problem tractable, and of shaping reinforcement in order to make learning more efficient:

1. states & actions $\longrightarrow$ conditions & behaviors
2. reinforcement $\longrightarrow$ multi-modal feedback

## 8.1 Enabling Learning

Traditional state-action models used by many RL approaches employ a a level of description which is inappropriate for complex situated domains. Their representations

abstract away important control details, but still must search excessively large state spaces. However, a large state space is not so much a sign of a difficult problem as it is that of a poorly formulated one. Learning in noisy and inconsistent worlds relies on an appropriate representation of the problem. I propose the following reformulation:

*Reinforcement learning in situated domains can be formulated as learning the conditions necessary and sufficient for activating each of the behaviors in the repertoire such that the agent's behavior over time maximizes received reward.*

This formulation accomplishes the desired goal of diminishing the state space by using conditions and behaviors instead of states and actions, with the effect of elevating the level of description of the learning problem.

The first part of the thesis has argued that behaviors are an intuitive and effective level of description for control, and described a methodology for selecting and combining basic behaviors for a given domain and set of goals. **Behaviors** were shown to be goal-driven control laws that hide the details of control. The same reasons that made behaviors a useful abstraction in control make them an appropriate and efficient basis for learning. Behaviors are more general than actions because they are not tied to specific detailed states but instead triggered by a set of general conditions. For example, a *wall-following* behavior applies to any environment and any wall that the agent can sense, and is not dependent on the agent's exact state including such information as its $(x, y)$ position, whether it is carrying a puck, and what is in front or behind it. Thus, the use of behaviors allows for pruning the state space by only considering that part of it that defines the behavior conditions.

**Conditions** are a proper subset of the state space. Each condition is the part of the state that is necessary and sufficient for activating a particular behavior. For example, the necessary and sufficient conditions for picking up a puck are that a puck is between the fingers of the robot. The space of conditions is usually much smaller than the complete state space of the agent resulting in a smaller space for the

learning algorithm. Furthermore, the fewer state elements need to be sensed the less the system will suffer from error and uncertainty.

The described reformulation of states and actions into conditions and behaviors effectively reduces the state space to a manageable size, and makes learning possible. Given this formulation, the value of any condition–behavior pair $(c, b)$ in the learning system is the sum of the reinforcement $R$ received up to that point:

$$A(c, b) = \sum_{t=1}^{T} R(c, t)$$

The amount and quality of the reinforcement determines how quickly the agent will learn. In nondeterministic, uncertain worlds with multiple agents, multiple goals, and bounded time, learning requires shaping of the reinforcement in order to take advantage of as much, rather than as little, information as is available to the agent.

## 8.2 Reinforcement for Accelerated Learning

In general, reinforcement learning can be accelerated in two ways: 1) by building-in more information, and 2) by providing more reinforcement. The reward function implicitly encodes domain knowledge and thus biases what the agent can learn. Simplifying and minimizing reinforcement, as practiced by some early RL algorithms (Sutton 1990), does diminish this bias, but it also greatly handicaps the learner, and, in situated domains, completely debilitates it.

Domain knowledge can be embedded through a reward-rich and complex reinforcement function. This approach is effective, but the process of embedding semantics about the world into the reward function is usually *ad hoc*. In the ideal case, reinforcement is both immediate and meaningful. Immediate error signals, for example, which provide not only the sign but also the magnitude of the error, result in fastest learning. Aside from attempting actions itself, the agent is nearly supervised in that,

after its trial, it is provided with the correct answer. In learning control (Jordan & Rumelhart 1992, Atkeson 1990, Schaal & Atkeson 1994), such error signals are critical as the learning problem is usually finding a complex mapping between a collection of input parameters and the desired output. Immediate reinforcement in RL is typically a weak version of an error signal, reduced to only the sign of the error but not the magnitude or the direction.

In the following sections two ways of shaping the reward function will be proposed, both based on principal embedding of domain knowledge (or "common sense") in order to accelerate learning.

### 8.2.1 Heterogeneous Reward Functions

Monolithic reward function with a single high-level goal, when applied to situated domains, require a large amount of intermediate reinforcement in order to aid the agent in learning. Intuitively, the more subgoals are used the more frequently reinforcement can be applied, and the faster the learner will converge. This thesis has already argued that situated agents maintain multiple concurrent goals, and that such goals can be achieved and maintained by using behaviors as the basic unit of control and learning. Thus, a task in a situated domain can be represented with a collection of such concurrent goal-achieving behaviors. Reaching each of the goals generates an event that provides primary reinforcement to the learner. The following is the general form of such event-driven reinforcement functions:

$$E(c,t) = \begin{cases} e & \text{if the event V occurs} \\ 0 & \text{otherwise} \end{cases} \qquad e \neq 0$$

Event-driven reinforcement for any event E is a function of conditions $c$ and time $t$. The received reinforcement $e$ may be positive or negative.

If appropriate information about the task and the appropriate sensors are avail-

able, each of the goals may be further broken down into one, or more, subgoals, with associated secondary reinforcement. In general, the specification of a high-level behavior provides a collection of subgoals that need to be achieved and maintained. If the achievement of a subgoal can be detected, it can be directly translated into a reinforcement function.

A general heterogeneous reward function has the following form:

$$E(c,t) = \begin{cases} r_{e1} & \text{if event e1 occurs} \\ r_{e2} & \text{if event e2 occurs} \\ . & . \\ . & . \\ . & . \\ r_{en} & \text{if event en occurs} \\ 0 & \text{otherwise} \end{cases}$$

The complete reward function is a collection of inputs from the individual event-driven functions. Each of these functions provides a part of the structure of the learning task, and thus speeds up the learning.

### 8.2.2 Progress Estimators

The previous section has associated reinforcement functions with the achievement of goals and subgoals, each reached by appropriate behaviors. This section describes a mechanism that again uses domain knowledge to trigger principled termination of behaviors in order to ensure exploration, minimize thrashing, and thus further accelerate learning.

Behaviors reach and maintain goals, and are initiated and terminated by events. Not all goals and tasks have immediately available measures of progress. However, few goals need to be defined as long sequences of actions without any feedback. Feedback need not come from the primary goal but can also be derived from a higher

or lower-level related goal. Consider the following example:

A robot is carrying a puck and needs to learn to take it home. It can wait until it has stumbled there to receive a reward. Alternatively, it can use a related subgoal, such as getting away from the food/puck pile, for feedback. The longer the robot with a puck stays near food the more negative reinforcement it receives. This strategy will encourage attempting behaviors that take the robot away from the food, one of which is homing.

While immediate reinforcement is not available in many domains, intermittent reinforcement can be provided by estimating the agent's progress relative to its current (sub)goal and weighting the reward accordingly. Measures of progress relative to a particular goal can be estimated with standard sensors, and furthermore feedback is available from different sensory modalities.

The following are the two general forms of progress estimator functions.

$$P(c,t) = \begin{cases} m & \text{if } c \in C' \wedge \text{progress is made} \\ n & \text{if } c \in C' \wedge \text{no progress} \end{cases} \qquad m > 0, \quad n < 0, \quad C' \subset C$$

$$S(c,t) = \begin{cases} i & \text{if } c \in C' \wedge \text{progress is made} \\ j & \text{if } c \in C' \wedge \text{regress is made} \\ 0 & \text{otherwise} \end{cases} \qquad i > 0, \quad j < 0, \quad C' \subset C$$

$C$ is the set of all conditions, and $C'$ is the set of conditions associated with the given progress estimator, i.e. those conditions for which the given progress estimator is active.

The use of progress estimates minimizes the brittleness of the learning algorithm, which is particularly important in noisy situated domains. Progress estimators have the following desirable properties:

- decrease sensitivity to noise

- encourage exploration
- decrease fortuitous rewards

Each is described in turn.

### Decreasing Sensitivity to Noise

Progress estimators provide implicit domain knowledge to the learner. Since they serve as a source of reinforcement, they strengthen appropriate correlations. In contrast, noise–induced events are not supported by progress estimator credit, and thus have less impact on the learner. Consider the following example:

- An agent A is executing behavior $B$ in condition $c$ and is aided by a progress estimator $P$.
- A receives negative reinforcement as a result of an event induced by a sensor error.

The impact of the negative reinforcement is diminished by the continuous reinforcement received by $P$ throughout the execution of $B$. Thus, the domain knowledge behind progress estimators provides a continuous source of reinforcement to counter intermittent and potentially incorrect credit.

### Encouraging Exploration

Exploration versus exploitation is one of the critical tradeoffs in machine learning. The agent must do enough exploration to discover new and potentially more efficient condition–behavior combinations, but must also optimize its performance by using the best known pairings. Ineffective exploration results in thrashing, repeated execution of one or more inappropriate behaviors.

Since situated environments are event-driven, any given behavior may persist for a potentially long period of time. An agent has no impetus for terminating a behavior and attempting alternatives, since any behavior may eventually produce a reward.[1]. The learning algorithm must use some principled strategy for terminating behaviors in order to explore the condition-behavior space effectively. Progress estimators provide such a method: if a behavior fails to make progress relative to the current goal, it is terminated and another one is tried. By using domain knowledge to judge progress, progress estimators induce exploration by terminating behaviors according to common sense, rather than according to some arbitrary internal clock or an ad hoc heuristic.

### Decreasing Fortuitous Rewards

A *fortuitous reward* is one received for an inappropriate behavior that happened to achieve the desired goal in the particular situation, but would not have that effect in general. Consider the following scenario:

Agent A has a puck and is attempting various behaviors.

While executing *avoidance*, A fortuitously enters the home region.

Without a progress estimator, agent A will receive a reward for reaching home, and will thus positively associate the avoiding behavior with the goal of getting home. It will require repeated trials in order to discover, implicitly, that the correlation is based on the direction it is moving rather than on *avoidance*.

Suppose a progress estimator $H$ is added to the above scenario. $H$ rewards the agent when it decreases its distance to home. If it fails to do so in a given time interval, the behavior is terminated. Now the continuous reward for approaching home will have a discounting effect on any fortuitous rewards the agent receives. In short, $H$ biases the agent toward behaviors that effectively decrease the distance to home. In general, the only way to eliminate fortuitous rewards is to know the

[1]This is similar to the halting problem.

relevance of context *a priori*. Progress estimates achieve this effect incrementally, because behaviors have some measurable duration which allows progress estimators to contribute useful reinforcement.

## 8.3 Summary

This chapter has introduced a formulation for learning using conditions, behaviors, and shaped reinforcement in order to 1) make learning possible and 2) make learning efficient, in complex situated domains. Elevating the system description, control, and learning level to conditions and behaviors instead of states and actions greatly diminishes the state space and makes learning tractable. The use of heterogeneous reward functions and progress estimators builds in domain knowledge and contextual information thus making learning more efficient.

The proposed reformulation forms a better foundation for situated learning, but does not impose any constraints on the kind of learning algorithm that can be applied. The next chapter demonstrates how this formulation was applied to the task of learning foraging in a situated, multi–robot domain using a simple learning algorithm.

# Chapter 9

# Learning Experiments

This chapter describes the learning experiments conducted to test the presented approach to enabling and accelerating reinforcement learning in situated domains.

## 9.1 The Robots

The learning experiments were conducted on a group of up to four fully autonomous R2 mobile robots with on–board power and sensing (figure 9-1). Each robot consists of a differentially steerable wheeled base and a gripper for grasping and lifting objects. The robots' sensory capabilities include piezo–electric bump sensors for detecting contact–collisions and monitoring the grasping force on the gripper, and a set of infra–red (IR) sensors for obstacle avoidance and grasping (see Figure 9-2).

Finally, the robots are equipped with radio transceivers, used for determining absolute position and for inter–robot communication. Position information is obtained by triangulating the distance computed from synchronized ultrasound pulses from two fixed beacons. Inter–robot communication consists of locally broadcasting 6-byte messages at the rate of 1 Hz. In the experiments described here, the radios are used to determine the presence of other robots near by.

As in the first set of robot experiments, the robots are programmed in the Behavior

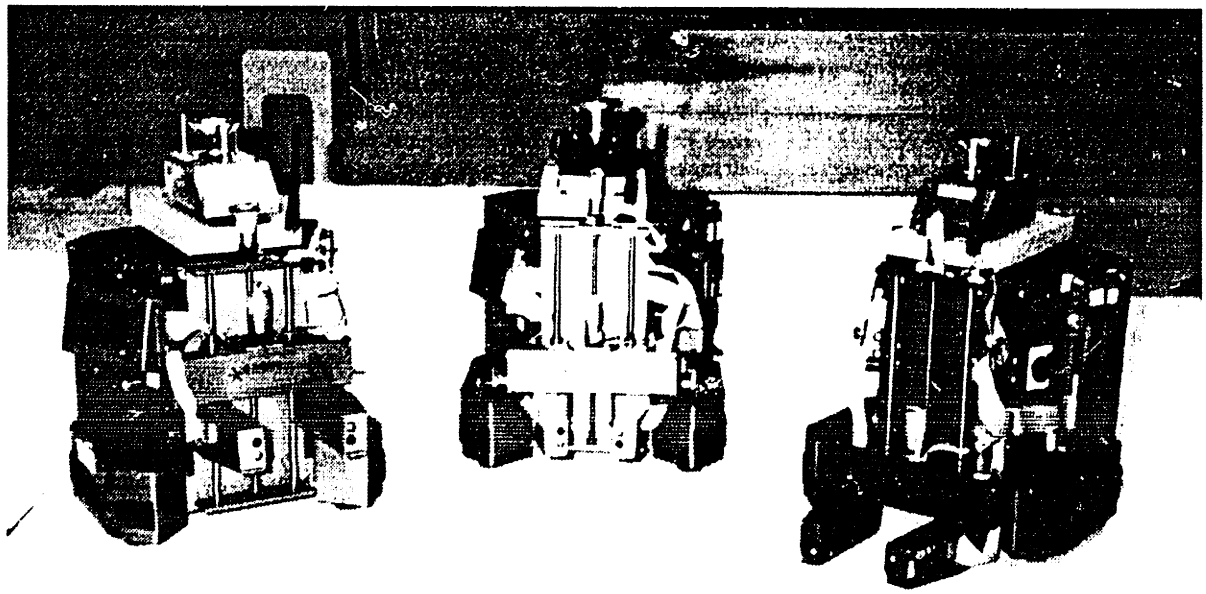


Figure 9-1: Three of the four robots used in the learning experiments. These robots demonstrated learning to forage by selecting among a basic behavior repertoire under appropriate sensory conditions.

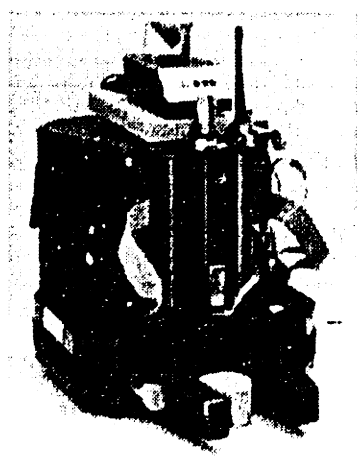


Figure 9-2: Each of the learning robots consists of a differentially steerable wheeled base and a gripper for grasping and lifting objects. The robot's sensory capabilities include piezo electric bump and gripper sensors, infra red sensors for collision avoidance, and a radio transmitter for absolute positioning.

Language, a parallel programming language based on the Subsumption Architecture (Brooks 1990*a*). Their control systems are collections of parallel, concurrently active behaviors, some of which gather sensory information, some that drive effectors, and some that monitor progress and contribute reinforcement.

## 9.2 The Learning Task

The learning task consists of finding a mapping of all conditions and behaviors into the most efficient policy for group foraging. Individually, each robot learns to select the best behavior for each condition, in order to find and take home the most pucks. Foraging was chosen because it is a complex and biologically inspired task, and because the author's previous group behavior work (Matarić 1992*b*, Matarić 1993) provided the basic behavior repertoire from which to learn behavior selection. As was described in an earlier chapter, foraging can be achieved from a small basic behavior set. Such a set, consisting of the fixed behavior repertoire was given to the robots *a priori*:

- *avoidance*
- *dispersion*
- *searching*
- *homing*
- *resting*

Utility behaviors for grasping and dropping objects were also included in the robots' capabilities, but since their conditions were not learned, they are therefore not included in the above basis set. The *resting* behavior was added to expand the agents' behavior space, as well as to introduce an internal clock that can trigger internally-generated events. The internal clock imposed a cyclic "circadian" schedule consisting of periods of "day time" and shorter periods of "night time". The resting

behavior could be used as a part of a regular recharging cycle, or as a chance for the robots to aggregate and exchange information.[1]

Given the behavior repertoire, the robots were expected to learn the appropriate conditions for triggering each of the behaviors. By considering only the space of conditions necessary and sufficient for triggering the behavior set, the state space is reduced to the cross-product of the following state variables:

- *have-puck?*
- *at-home?*
- *near-intruder?*
- *night-time?*

The conditions for *grasping*, *dropping*, and *avoiding* were built-in. As soon as a robot detects a puck between its fingers, it grasps it. Similarly, as soon as a robot reaches the home region, it drops the puck if it is carrying one. Finally, whenever a robot is too near an obstacle, it avoids. The three behaviors were deemed to be "instinctive" because learning them has a high cost. Learning to avoid has a potentially prohibitive damaging cost for the robot, and is an un-intuitive learning task, as it appears to be innate in nature, and can be easily programmed on most systems. Puck manipulation requires a fast and accurate response from the gripper motors, and, like the other basic behaviors, is best suited for parameter learning.

As described, the foraging task may appear quite simple, since its state space has been appropriately minimized. In theory, an agent should be able to quickly explore it and learn the optimal policy. In practice, however, such quick and uniform exploration is not possible. Even the relatively small state space presents a challenge to the learner situated in a nondeterministic, noisy and uncertain world. In its reformulated version

[1] Neither of these options were used in the shown generation of robots.

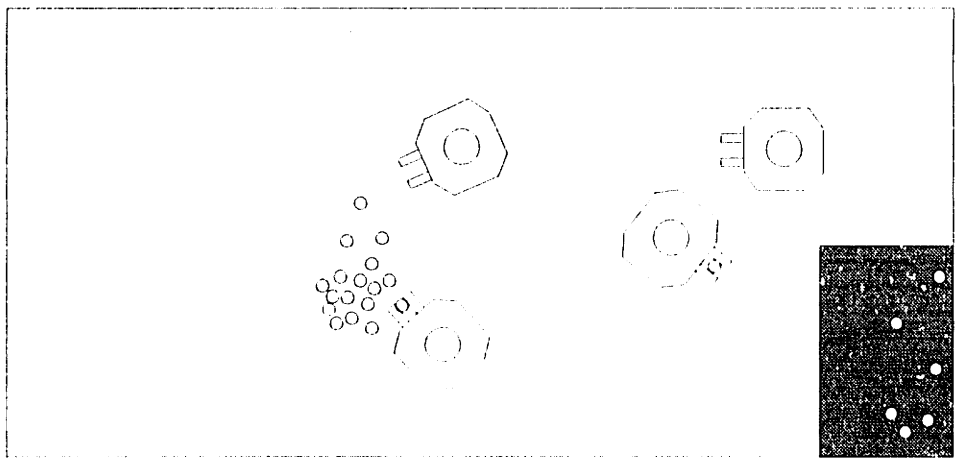


Figure 9-3: A top view of the experimental area in which the learning experiments were conducted. The workspace is small enough to result in frequent interaction and interference between the robots. The home region is shaded.

using the minimized state space, this problem still poses a challenge for traditional RL methodologies using delayed reward, and thus also necessitates the proposed reformulation of reinforcement.

The described learning problem is made more difficult by the fact that the learner is not provided with a model of the world. As discussed earlier, such a model is difficult to obtain, and cannot be assumed to be available. Without it, the agent is faced with implicitly deducing the structure of a dynamic environment including other agents. The behavior of others occasionally facilitates but largely interferes with the individual learning process (see figures 9-3 and 9-4). Thus, the shown scenario poses a difficult challenge for the reinforcement learning paradigm. The next section shows a solution to the challenge.

## 9.3 The Learning Algorithm

The learning algorithm produces and maintains a total order on the appropriateness of behaviors associated with every situation, expressed as a matrix $A(c, b)$. The values in the matrix fluctuate over time based on received reinforcement, and are

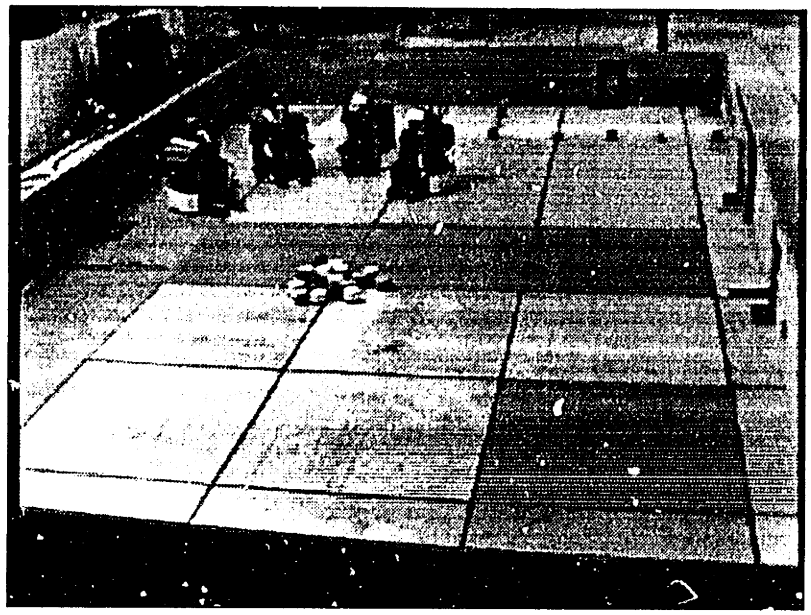


Figure 9-4: The camera's view of the experimental environment used for learning. The boundary of the home region is indicated with a row of pucks for the purposes of the photo. The pile of pucks is also marked.

updated asynchronously, with any received learning signal.

The following events produce immediate positive reinforcement:

1. grasped-puck ($p$)

2. dropped-puck-at-home ($gd$)

3. woke-up-at-home ($gw$).

The following events result in immediate negative reinforcement:

1. dropped-puck-away-from-home ($bd$)

2. woke-up-away-from-home ($bw$)

The events are combined into the following heterogeneous reinforcement function:

$$E(c) = \begin{cases} p & \text{if new puck grasped} \\ gd & \text{if puck dropped at home} \\ bd & \text{if puck dropped not at home} \\ gw & \text{if woke up at home} \\ bw & \text{if woke up away from home} \\ 0 & \text{otherwise} \end{cases}$$

$$p, gd, gw > 0, \quad bd, bw < 0$$

Two progress estimating functions, $I$ and $H$, are used. $I$ is associated with minimizing interference and is triggered whenever an agent is close to another agent. If the behavior being executed has the effect of increasing the physical distance to the other agent, the agent receives positive reinforcement. Conversely, lack of progress away from the other agent is punished, and after a fixed time period of no progress, the current behavior is terminated.

Formally, $I$ is the intruder avoidance progress function such that:

$$I(c,t) = \begin{cases} m & \text{distance to intruder decreased} \\ n & \text{otherwise} \end{cases}$$

$$near\_intruder \in c, \quad m > 0, \quad n < 0$$

The other progress estimator, $H$, is associated with *homing*, and is initiated whenever a puck is grasped. If the distance to home is decreased while $H$ is active, the agent receives positive reinforcement, *status quo* delivers no reinforcement, and movement away from home is punished.

Formally, $H$ is the homing progress function such that:

$$H(c,t) = \begin{cases} i & \text{closer to home} \\ j & \text{farther from home} \\ 0 & \text{otherwise} \end{cases}$$

$$have\_puck \in c, \quad i > 0, \quad j < 0$$

For the sake of a bottom-up methodology, the simplest learning algorithm that uses the above reinforcement functions was implemented and tested. The algorithm is sum of the reinforcement over time:

$$A(c,b) = \sum_{t=1}^{T} R(c,t)$$

$$R(c,t) = E(c,t) + I(c,t) + H(c,t)$$

The influence of the different types of feedback was weighted by the values of the feedback constants. This is equivalent to the alternative of weighting their contributions to the sum:

$$R(c,t) = uE(c,t) + vI(c,t) + wH(c,t)$$

$$u,\ v,\ w \geq 0, \quad (u+v+w) = 1$$

Binary-valued and real-valued $E$, $H$, and $I$ functions were tested. The results showed that differentially weighted reinforcement does not result in faster or more stable learning. This is not surprising, since the subgoals in the foraging task are independent and thus their learning speed is uncorrelated.

### 9.3.1 The Control Algorithm

The following is the complete control algorithm used for learning foraging. Behavior selection is induced by events, each of which is a change in the conditions. Events can be triggered:

1. **externally:** e.g., a robot gets in the way of another,
2. **internally:** e.g., the internal clock indicates night time,
3. **by progress estimators:**, e.g., the interference estimator detects a lack of progress and terminates the current behavior.

Whenever an event is detected, the following control sequence is executed:

1. appropriate reinforcement is delivered for the current condition–behavior pair
2. the current behavior is terminated
3. another behavior is selected

Behaviors are selected according to the following rule:

1. choose an untried behavior if one is available,
2. otherwise choose the best behavior.

Learning is continuous and incremental over the lifetime of the agent, thus ensuring that the agent remains responsive to changes in the environment (e.g., no more pucks are left at a particular location) and internal changes in function (e.g., dying battery slows motion down, or a broken sensor affects the perception of conditions).

Snapshots of a learning experiment are shown to illustrate the progression of a typical of experiment. Figure 9-5 shows the typical initial conditions, figure 9-6 demonstrates a stage during the course of learning, and figure 9-7 shows the environment toward the end of the experiment, when most of the pucks have been collected. Figure 9-8 illustrates the *resting* behavior.

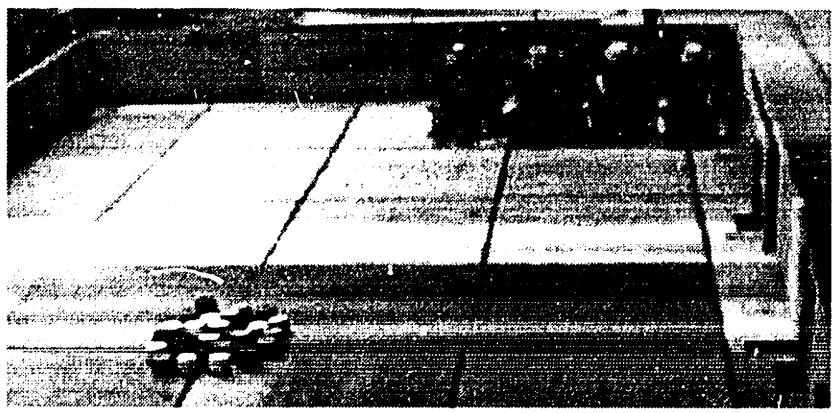


Figure 9-5: Typical initial conditions for learning trials. The robots are initiated either in the home region or in random positions around the workspace.

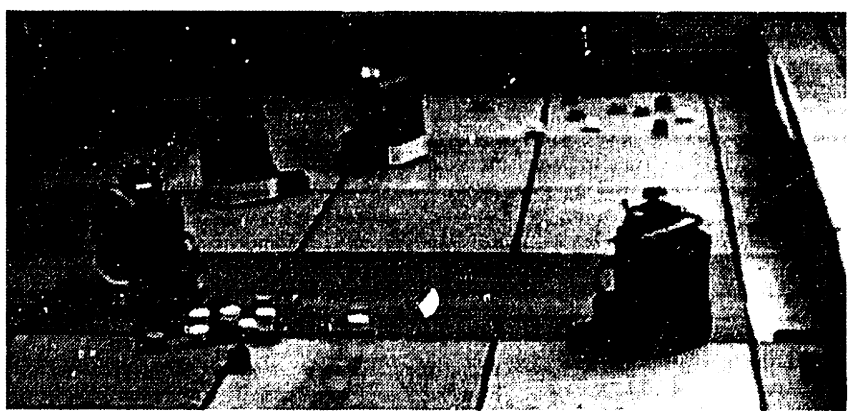


Figure 9-6: A typical environment state during the course of a learning experiment. Since they are learning independently, the robots have likely acquired different parts of the policy. Through interactions with objects in the world and each other they accumulate learning trials in order to complete their learning.

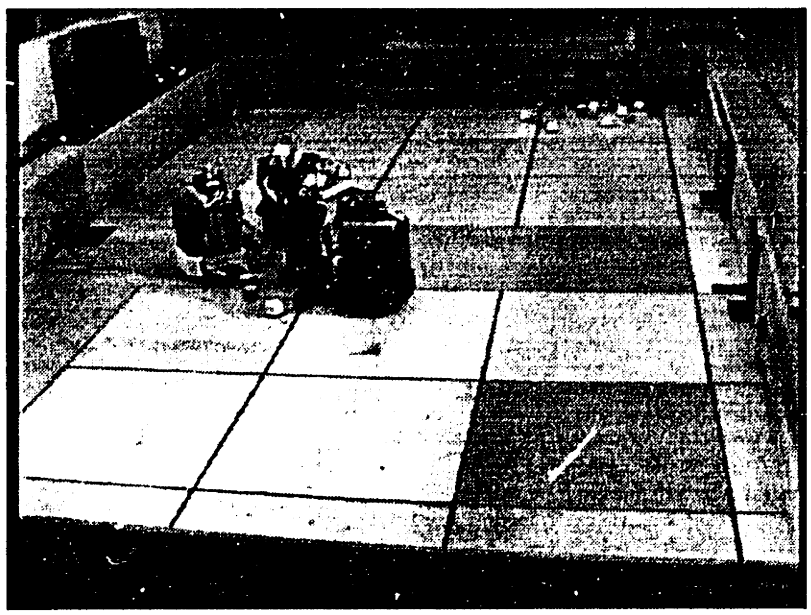


Figure 9-7: A typical environment state after learning. Most pucks have been collected and brought to the home region. The robots have all learned when and where to go get the pucks, and are thus competing for those remaining to be moved.

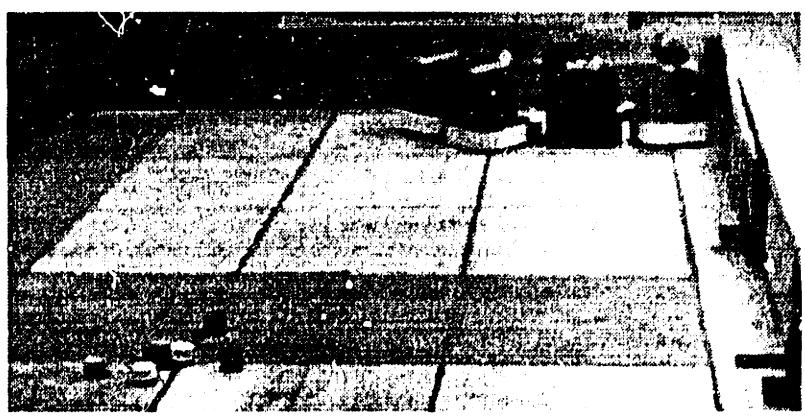


Figure 9-8: An example of the resting (or recharging) behavior of four robots, triggered by their internal clocks. In this case, the robots have all learned to go home to rest, as this photo illustrates a late stage in the learning, as demonstrated by the small number of remaining pucks.

Experimental results are described next.

## 9.4 Experimental Results

The effectiveness of the proposed reinforcement functions was evaluated by testing three different types of reinforcement. The following three approaches were compared:

1. a monolithic single-goal (puck delivery to the home region) reward function using Q-learning, $R(t) = P(t)$, and using the Q-learning algorithm,

2. a heterogeneous reward function using multiple goals: $R(t) = E(t)$, and using the reinforcement summation algorithm,

3. a heterogeneous reward function using multiple goals, and two progress estimator functions: $R(t) = E(t)+I(t)+H(t)$, and using the reinforcement summation algorithm.

Data from twenty trials of each of the three strategies was collected and averaged. The experiments were run on four different robots, and no significant machine-specific differences were found. Data from runs in which persistent sensor failures occurred were discarded. Values of $A(c, b)$ were collected twice per minute during each learning experiment and the final learning values after a 15 minute run were collated. The 15 minute threshold was chosen empirically, since the majority of the learning trials reached a steady state after about 10 minutes, except for a small number of rare conditions, as discussed below.

Evaluating performance of situated systems is notoriously difficult. Standard metrics for evaluating learning mechanisms, such as time-to-convergence, do not directly apply. The amount of time required for a robot to discover the correct policy depends on the frequency of external events that trigger different states in its learning space. Additionally, noise and error make certain parts of the policy fluctuate so waiting

for a specific point of absolute convergence is not feasible. Instead, convergence is defined as a relative ordering of condition–behavior pairs.

| Condition | Behavior |
|---|---|
| no-intruder, no-puck, not-home, not-night | searching |
| no-intruder, no-puck, not-home, night | homing |
| no-intruder, no-puck, home, not-night | searching |
| no-intruder, no-puck, home, night | resting |
| no-intruder, puck, not-home, not-night | homing |
| no-intruder, puck, not-home, night | homing |
| no-intruder, puck, home, not-night | homing |
| no-intruder, puck, home, night | resting |
| intruder, no-puck, not-home, not-night | avoidance |
| intruder, no-puck, not-home, night | avoidance |
| intruder, no-puck, home, not-night | dispersion |
| intruder, no-puck, home, night | resting |
| intruder, puck, not-home, not-night | homing |
| intruder, puck, not-home, night | homing |
| intruder, puck, home, not-night | searching |
| intruder, puck, home, night | resting |

Table 9.1: The optimal foraging policy. Only the top–ranked behavior is shown for each condition. The full table has a total numerical ordering of five behaviors for each condition, a total of 80 entries.

In the described learning task, the optimal policy was derived empirically by the author from the foraging experiments described in the first part of the thesis. This policy is shown in figure 9.1. The performance of the desired policy was tested independently and compared to alternative solutions in order to establish its superiority relative to the imposed evaluation criteria.

The performance of the three approaches is compared in figure 9-9. The x-axis shows the three reinforcement strategies. The y-axis maps the percent of the correct policy the agents learned, averaged over twenty trials, i.e., the ratio of correct condition–behavior pairings according to the optimal policy.

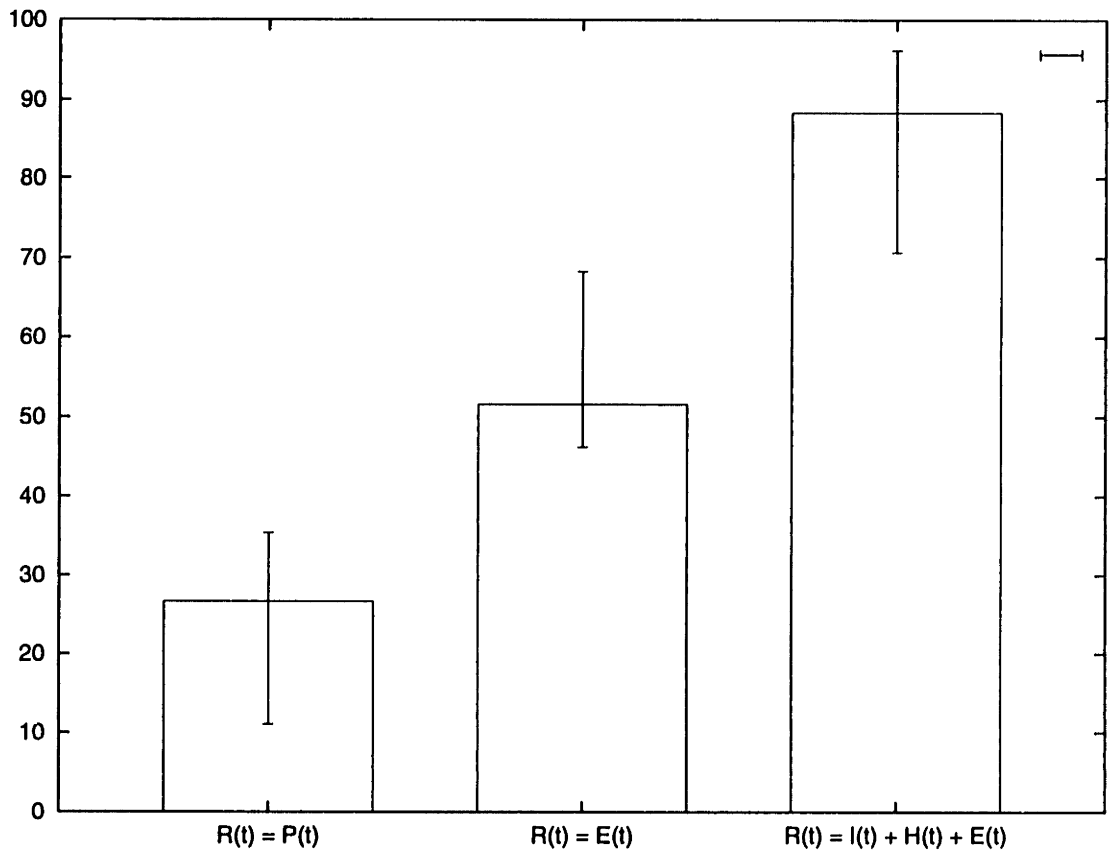



Figure 9-9: The performance of the three reinforcement strategies on learning to forage. The x-axis shows the three reinforcement strategies. The y-axis maps the percent of the correct policy the agents learned, averaged over twenty trials.

| Reinforcement | Effect |
|---|---|
| $R(t) = P(t)$ | converges for at most 1/3 of the policy |
| $R(t) = E(t)$ | converges for at least 1/2 of the policy |
| $R(t) = E(t) + I(t) + H(t)$ | converges for at least 2/3 of the policy |

Table 9.2: A qualitative summary of the performance for the three types of reinforcement used on the foraging task.

Q-learning was tested on the reduced state space using the enumerated conditions and behaviors. The algorithm is functionally equivalent to a simplified version of the second algorithm, using only positive reinforcement for the single goal of dropping a puck in the home region. Given the nondeterminism of the world, and the uncertainty in sensing and state transitions, the single goal provides insufficient feedback for learning all aspects of foraging, in particular those that rely on accurate delayed credit assignment. The performance of Q-learning was vulnerable to interference from other robots, and declined most rapidly of the three approaches when tested on increased group sizes.

As shown in table 9.2, Q performs poorly, but the partial policy it discovers is not random. Rather, it is consistent over all trials and consists of the few condition behavior pairings that receive immediate and reliable reinforcement. Thus, the performance of Q indicates the difficulty of the learning task at least to the extent of demonstrating the immediately reinforced parts as the only parts it is capable of learning.

As shown in figure 9-9, Q does not perform better than chance. However, the partial policy it discovers is not random. It is consistent over all trials and consists of the few condition–behavior pairings that receive immediate and reliable reinforcement. Thus, the performance of Q indicates the difficulty of the learning task at least to the extent of demonstrating the immediately reinforced parts as the only parts it is capable of learning.

The second learning strategy, utilizing reinforcement from multiple goals, outperforms Q because it detects the achievement of the subgoals on the way of the top-level goal of depositing pucks at home. However, it also suffers from the credit assignment problem in the cases of delayed reinforcement, since the nondeterministic environment with other other agents does not guarantee consistency of rewards over time.

Furthermore, this strategy does not prevent thrashing, so certain behaviors are active unnecessarily long. For example, *searching* and grasping are pursued persistently, at the expense of behaviors with delayed reinforcement, such as *homing*. The performance of heterogeneous reinforcement gives us another evaluation of the difficulty of the proposed learning task. With around 60% average performance, it demonstrates that additional structure is necessary to aid the learner. This structure is provided by progress estimators.

The complete heterogeneous reinforcement and progress estimator approach outperforms the other two approaches because it maximizes the use of all available information for every condition and behavior. As predicted, thrashing is eliminated both in the case of learning the conditions for *dispersing* and *homing* because the progress estimator functions enforce better exploration. Furthermore, fortuitous rewards have less impact than in the alternative algorithms. The implicit domain knowledge is effectively spread over the reinforcement in order to guide the learning process continually, thus minimizing the utility of each of the learning trials and consequently speeding up the learning.

### 9.4.1 Evaluation

Table 9.2 shows a coarse performance ordering of the three approaches. Although intuitive, this ordering is not particularly informative. A better way to analyze the approaches is to evaluate each part of the policy separately, thus measuring when and

what each robot was learning. To capture the dynamics of the learning process, each condition–behavior pair was evaluated according to the following three criteria:

1. number of trials required,

2. correctness,

3. stability.

The number of trials was measured relative to a stable solution, whether the solution was optimal or not. The second criterion sought out any tendencies toward incorrect solutions. Finally, the third criterion focused on unstable policies, looking for those in which the behavior orderings tended to fluctuate.

Based on those criteria, some condition–behavior pairs proved to be much more difficult to learn than others. The most prominent source of difficulty was the delay in reinforcement, which had predictable results clearly demonstrated by the performance differences between the three strategies. Learning the conditions for *searching* was difficult as there was no available progress estimator, and the robot could be executing the correct behavior for a long while before reaching pucks and receiving reward. In the mean time it could be repeatedly interrupted by other activities, such as *avoiding* obstacles and intruders, and *resting*.

Another source of difficulty was rareness of occurrence of some combinations of conditions. For example, the condition consisting of the onset of night time while a robot is carrying a puck and avoiding another robot rarely occurred. Consequently, the correct mapping was difficult to learn since the robots did not get a chance to explore all behavior alternatives. This accounts for the incomplete policy even in the case of the most successful reinforcement strategy.

The presence of positive and negative reinforcement in the continuous learning algorithm guarantees that the learner will not get stuck in local minima, but it also allows for oscillations. Two of the conditions oscillated because the alternatives re-

suited in equally effective solutions. For example, in situations when the robot is not carrying a puck and encounters an intruder, any motion away from the intruder will be beneficial and rewarded by the progress monitor $I$. Consequently, *homing* and *searching* are often as effective as *dispersing*. In contrast, if the robot is carrying a puck, then *dispersing* followed by *homing* is optimal and is manifested in the contributions of the $I$ and $H$ progress estimators. As described earlier, it is the combination of the two estimators that speeds up exploration as well as minimizes fortuitous rewards. Only a specific progress measure that minimizes the travel time to the goal can eliminate this effect. Such optimization is difficult in systems using largely local sensing and control and dealing with interference from other agents. Consequently, the policy the robots found was optimal for the properties of their domain.

### 9.4.2 Effects of Heterogeneous Reinforcement

The design of the foraging task using basic behaviors guarantees that its subgoals are independent and consequently the associated reinforcement functions are mutually consistent, since they all contribute to a common high-level goal. Although in theory the more reinforcement is used the faster the learning should be, in practice noise and error in the different reinforcement sources could have the opposite effect. The experiments described here demonstrated a significant amount of noise and inconsistency in the different reinforcers and progress estimators.

For example, each robot's estimate of its position and the proximity of others was frequently inaccurate due to radio transmission delays. These errors resulted in faulty homing and interference progress estimates. Nonetheless, all condition-behavior pairs that involved carrying a puck converged quickly. Furthermore, their $A(c, b)$ values did not tend to oscillate. The fast rate of convergence for associations with behaviors that involved *dispersion* and *homing* result directly from the effects of the two progress estimators. When the two are removed, as in the second tested

algorithm, the performance declines accordingly.

Conversely, the set of conditions associated with finding pucks uniformly took longer to learn, since they had no direct progress measure to accelerate the learning. Furthermore, the learned values tended to oscillate, since the differences between the behavior alternatives were not great, again due to a lack of intermediate rewards. Consequently, empirical results show that even noise and error-induced inconsistencies in the progress estimators do not significantly diminish the benefit of their use.

#### Transition Models

The learning problem presented here, involving a collection of concurrently learning agents in a noisy and uncertain environment, was purposefully chosen for its complexity. The fact that a state transition model was not available to aid the learner presented one of the major challenges.

As argued earlier, such models are not generally available, but partial models could be constructed empirically, either before or during the learning process. The implemented reinforcement functions take advantage of immediate information from the world to generate reinforcement. Thus, they would have an accelerating effect on any learning domain, regardless of whether a transition model is available. An interesting extension of this work would apply the described reinforcement approach to problems that involve incomplete and approximate state transition models in order to study the effects of combining immediate reinforcement with discounted future rewards commonly applied to RL problems.

### 9.4.3 Scaling

The three reinforcement alternatives were evaluated on groups of three and four robots and it was found that interference was a detriment to all three. In general, the more robots were learning at the same time, the longer it took for each to converge. This

was particularly pronounced for condition–behavior pairs without directly associated progress estimators, such as those involved in the non–puck carrying states.

The only behavior capable of reaping benefits from interference was *dispersion*, which was learned faster and more accurately in crowded situations. I have considered adding a social behavior called *yielding* in order to minimize interference by having only one robot move at a time in crowded situations.

This and other extensions of the described approach are discussed in the next section.

## 9.5 Discussion and Extensions

The described formulation of the multi–agent reinforcement learning task is a direct extension of behavior–based control (Matarić 1992*a*, Brooks 1991*b*, Brooks 1986). The presented heterogeneous reward functions are related to subgoals used by Mahadevan & Connell (1991*a*) as well as subtasks used by Whitehead et al. (1993). However, unlike previous work, which has focused on learning action sequences, this work used a higher level of description. The proposed subgoals are directly tied to behaviors used as the basis of control and learning. Similarly, progress estimators are mapped to one or more behaviors, and expedite learning of the associated goals, unlike a single complete external critic used with a monolithic reinforcement function (Whitehead 1992).

### 9.5.1 Social Rules

A decline in performance of all of the algorithms was observed with the increased group size and the associated increased interference between agents. Although not surprising, this is nonetheless an undesirable effect. In an ideal scenario, the presence of other agents should speed up rather than slow down individual learning. This is

only possible in societies where individuals benefit from each other's experience, and they interact according to mutually beneficial social rules.

My most recent work has addressed the problem of learning such social rules. This is a challenging learning problem since social rules do not necessarily have immediate or even delayed payoff to the individual but may only benefit the individual on average from having a global effect. Consequently, social rules involve some "altruistic" behavior, even at the simplest of levels, such as *yielding* in traffic. Such behavior is difficult to learn with individualist reinforcement learning strategies. I am currently working on an algorithm that utilizes the observation of neighboring agents' behavior and received reinforcement in order to acquire and practice social behaviors (Matarić 1994).

### 9.5.2 Heterogeneous Learning

One of the key advantages of heterogeneous reinforcement is the possibility of learning multiple types of behaviors in parallel. Such concurrent multi-modal learning is biologically and pragmatically inspired, and has been an ongoing challenge in the learning community (Franklin & Selfridge 1990, Brooks & Matarić 1993).

In the example foraging task, the basic behaviors were designed by hand and behavior selection was learned. However, the basic behaviors themselves could be learned or optimized in parallel with learning behavior selection. For example, the agents could use a parameter learning scheme to optimize their grasping behaviors whenever in a puck-carrying state. In order to avoid extending the state space and reverting to the traditional problems of monolithic learners, multi-modal learning would be implemented using multiple correlation mechanisms instead of a monolithic $A(c, b)$ matrix. The described reinforcement techniques can be applied at every learning level. No explicit merging of learned policies is needed since the learning modules would be independent.

The proposed method for shaping reinforcement allows for learning multiple goals at once. How each goal and skill should be learned is addressed next.

### 9.5.3 Structuring Learning

One of the difficulties facing the learning community is the lack of structure that taxonomizes the existing learning methodologies. Consequently, the choice of methodology is often based on passing trends and dogma rather than on objective applicability and performance criteria. methodology broadly characterized as reinforcement learning. By applying reinforcement learning to a novel and more complex domain than has been experimented with to date, I was able to establish its limitations for that domain, and propose a reformulation of the representation and reinforcement that makes learning in that domain both possible and efficient.

However, by appropriately setting up the learning task, effective results were achieved from a single learning methodology. An interesting direction to pursue would be to deal with learning problems complex enough to require more than one learning strategy. This would be a way of relating at least a subset of existing learning techniques. Such a fundamental unifying learning problem is discussed next.

### 9.5.4 Signal–to–Symbol Learning

Signal–to-symbol learning encapsulates the entire learning process from the grounding of the agent's experiences in the world to the resulting comparatively high–level representations. To date, systems that have learned from low–level signals, such as sensory information, have either bypassed a symbolic system all together, or had it built–in. On the other end of the spectrum, symbolic learning has not traditionally concerned itself with grounding in the physical world. However, for situated systems, which must make a connection between direct sensory experiences and high–level cognitive activities, symbol grounding is an important problem that must be addressed

(Harnad 1990).

Most situated work to date has not dealt with what are considered to be highly cognitive tasks. However, even learning of "lower-level" capacities, such as complex motor behaviors, requires intermediate and increasingly abstract representations. The process of relabeling information into forms that can be used by other subsystems for achieving different goals is already a step in the direction of bridging the signal-to-symbol gap.

The learning work presented here has been at a level that could use a simple mapping between conditions and behaviors. Nonetheless, even the process of constructing the presented reusable behavior combinations requires some way of labeling the combinations. As most other work, the learning strategy described here was able to use a built-in mapping to labeled behaviors. A more general solution to the problem is desirable, and I hope to address it in future work.

## 9.6 Summary

The goal of the described learning work has been to bring to light some of the important properties of situated domains, and their impact on reinforcement learning strategies that can be effectively applied. I have described why MDP models of agent-world interactions are not effective in the noisy multi-agent domain, how the traditional notions of state and action present an inappropriately low level of system description for control and learning, and how delayed reinforcement is not sufficient for learning in this domain.

I introduced a higher-level description of the learning system based on conditions and behaviors, which greatly decreases the state space as well as results in more robust control. I also introduced a methodology for shaping reinforcement in order to take advantage of all of the information available to the agent. Shaping was necessary given the complexity of the environment-agent and agent-agent interactions. The

approach consists of two methods: one that partitions the learning task into natural subgoals and reinforces each separately, and one that employs progress estimators to generate more immediate feedback for the agent.

The proposed formulation was shown to be necessary and effective in our complex domain. It is general and compatible with the existing reinforcement learning algorithms, and should thus serve to make learning more efficient in a variety of situated domains and with a variety of methodologies.

# Chapter 10

# Thesis Summary

The aim of this thesis has been to gain insight into intelligent behavior by scaling up the level of complexity of the systems being studied. In contrast to many AI systems that have focused either on complex cognition situated in simple worlds, or vice versa, the work described here has addressed situated, embodied agents coexisting and interacting in complex domains (figure 10-1). I hope that the methodologies and results presented here have extended the understanding of synthesis, analysis, and learning of group behavior.

Selection of the appropriate level of control, planning, and learning is one of the motivating forces behind this work. I have proposed basic behaviors as the appropriate level for synthesizing coherent group behavior, and demonstrated the idea in the domain of planar spatial interactions. I devised a basic behavior set and showed that it meets the defining criteria, including no mutual reducibility and simple combination. I then showed how basic behaviors and their conditions can be used as a substrate for learning. Furthermore, I described a methodology for shaping reinforcement in order to make learning possible and more efficient in dynamic multi-agent domains.

An important question remains: how is the basic behavior set derived for a given domain? Given a particular set I have shown how to test and evaluate it, but deriva-

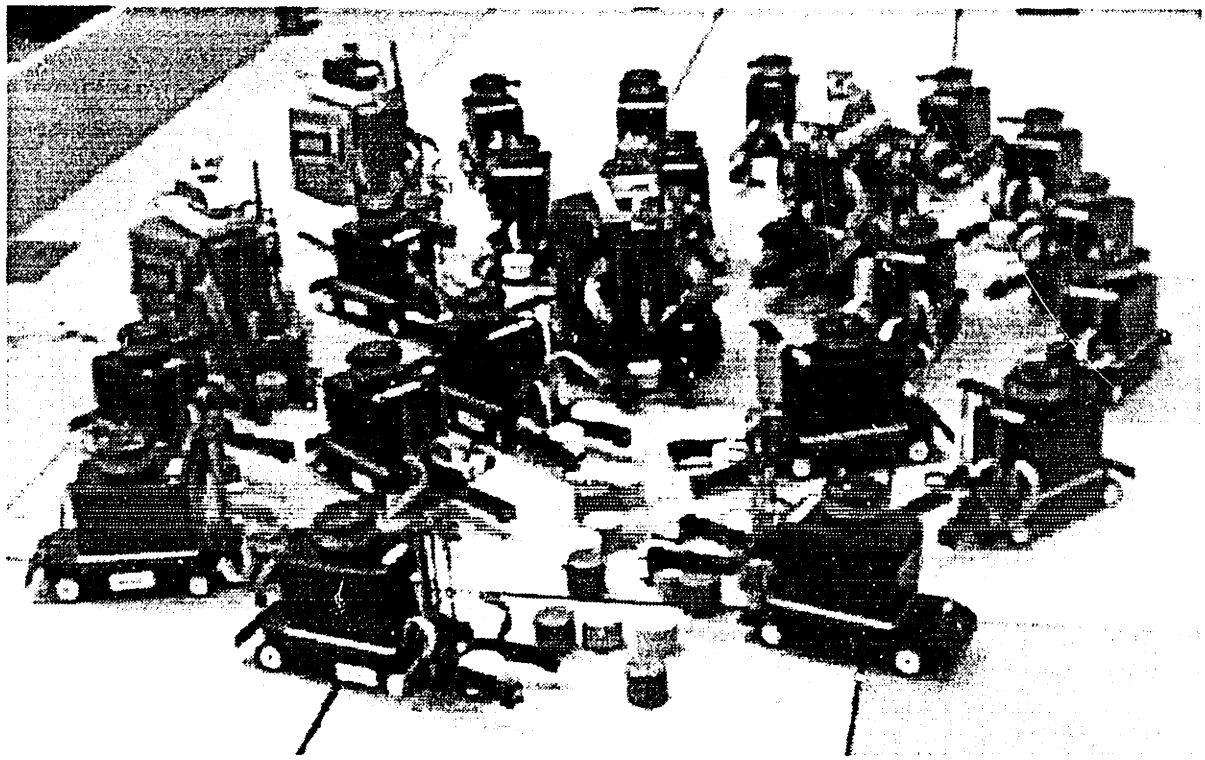


Figure 10-1: A family photo of the physical experimental agents used to demonstrate and verify the group behavior and learning work described in this thesis.

tion, because it is necessarily domain-specific, remains an open question. Some principles are discussed below.

Biology provides us with evidence in support of basic behavior units at a variety of levels. A particularly clean and compelling case can be found in motor control. Controlling a multi-joint manipulator such as a frog leg or a human arm is a complex task, especially if performed at a low level. In order to cut down the complexity, nature imposes an abstraction. Mussa-Ivaldi & Giszter (1992) show that a relatively small set of basis vector fields, found in the frog's spine, generates the frog's entire motor behavior repertoire by applying appropriate combinations of the basis vectors. Bizzi, Mussa-Ivaldi & Giszter (1991) and Bizzi & Mussa-Ivaldi (1990) discuss control of the human arm with a similar approach. The described motor basic behaviors are a result of the types of constraints: the dynamics of the manipulator and the dynamics of the motor tasks. In the case of motor control, the behaviors are designed

for specific optimizations, such as minimizing effort by minimizing jerk, executing straight line trajectories, and using bell-shaped velocity profiles (Atkeson 1989).

Taking the idea from motor control, basic behaviors can be defined as a minimal set (with appropriate composition properties) that takes advantage of the dynamics of the given system by optimizing the relevant parameters. In the case of spatial interactions, the shown behaviors attempt to minimize effort by optimizing paths and minimizing interference.

This work is intended as a foundation in a continuing effort toward studying increasingly more complex behavior, and through it, more complex intelligence. The work on basic behaviors distills a general approach to control, planning, and learning. The work also empirically demonstrates some challenging problems and offers some effective solutions to situated learning. Future work should both analytically tighten and experimentally broaden our understanding of all those issues. The demonstrated results in group behavior and learning are meant as stepping stones toward studying increasingly complex social agents capable of more complex learning, ultimately leading toward human intelligence.

# Appendix A

# Utilizing Dynamics of Interaction: Docking & Parking

Achieving arbitrary agent behaviors can be difficult as there is a often minimal match between the dynamics of the agent and its environment and the human-specified task. We have argued that for each system there exist a collection of behaviors that are easily achieved as they directly result from the dynamics of interaction between the agents and the environment. These behaviors tend to be simple and robust, and if well designed, can be used as a basis for a variety of other, more complex behaviors. We have illustrated this point on group behaviors in the plane. In this section we describe another group behavior that, if programmed top-down, would be difficult to achieve, but can be simply generated by taking advantage of the system dynamics. The behavior is *docking.*

The docking behavior gets the robots to "park" along a boundary of edge. In general, getting a collection of robots to park along a line is difficult. It can be solved by geometric planning, but is intractable for uncertain, dynamic environments with multiple agents. In contrast to a tightly-controlled top down approach, we demonstrate a bottom-up alternative.

Our docking algorithm takes advantage of environmental constraints, i.e. the

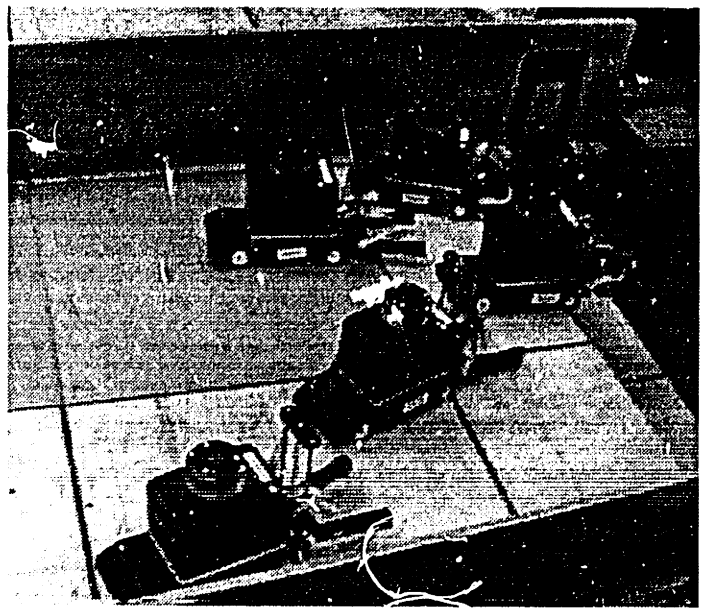


Figure A-1: Docking behavior in progress, based on the constraints of the environment and three rules: avoid, go forward, and don't fall of the edge.

existence of other agents, the boundary, and the walls (see figure A-1). The goal of the experiment is to achieve a state in which all of the robots are parked along the edge of a step, which they can detect using their downward pointing IR sensors.

The algorithms consists of three rules; one keeps the robot moving forward, another avoids collisions, and the third stops the robot from falling off the edge:

```
Docking:
If ground cannot be sensed
   stop.
If another robot is near by
   avoid.
If all is clear
   go forward.
```

The combination of these three rules executed in a confined space, results in a tight docking behavior. Since no position control is used, no specific docking positions are determined *a priori*. The algorithm is insensitive to initial conditions, to the number

Figure A-2: The end result of the docking behavior of five robots.

of robots, and even to the avoidance strategies employed by each of the robots. It has been tested in over 20 trials on groups from one to five robots. In all cases, it quickly resulted in all of the robots lined up along the edge (figure A-2). The environment constraints can be eliminated if the robots use some position control. We only explored the simplest case.

We have also implemented a variant of docking called *parking* which imposes fewer constraints on the structure of the environment. If a queue of robots is moving along and the "leader" stops, the robot behind it will pass it and then stop next to it, forming a line. This process will be repeated by all robots and result in a parking line.

```
Parking:
If behind another agent
   follow.
If agent in front stops
   pass.
If all is clear
   stop.
```

Similar simple behavior and combinations have been used in other behavior

based systems. For instance, Matarić (1990*a*) uses three rules to achieve boundary following on a sonar–based mobile robot. Steels (1994) implements docking onto a charger with two rules: one that approaches a light source above the charger, and the other that avoids obstacles.

# Appendix B

# Q-learning

Watkins (1989) introduced a family of methods he called Q-learning for solving Markov decision problems with incomplete information, i.e. through the use of delayed reinforcement. The simplest version, called one-step Q-learning, is the most commonly used and is thus described below.

Q-learning is based on a temporal differencing strategy that attempts to, at each time step, maximize $Q(s, a)$, the expected discounted reward of taking action $a$ in the input state $s$. The $Q$ values for all state–action pairs are stored in the $Q$ table and updated at each time step. The utility $E$ of a state is the maximum $Q$ value of all actions that can be taken in that state. The $Q$ value of doing an action in a state is defined as the sum of the immediate reward $r$ and the utility $E(s')$ of the next state $s'$ according to the state transition function $T$, discounted by the parameter $\gamma$.

Formally:

$$s' \leftarrow T(s, a)$$

$$E(s) = \max_a Q(s, a)$$

$$Q(s, a) = r + \gamma E(s'),\ 0 \leq \gamma \leq 1$$

Q values are updated by the following rule:

$$Q(s,a) \leftarrow Q(s,a) + \beta(r + \gamma E(s') - Q(s,a))$$

$$0 \leq \beta \leq 1$$

An RL algorithm using Q-learning has the following form:

1. Initialize all $Q(s,a)$; select $s_0$.
2. Do Forever:
    a. Observe the current world state $s$.
    b. Choose an action $a$ that maximizes $Q(s,a)$.
    c. Execute action $a$.
    d. Let $r$ be the immediate reward for executing $a$ in state $s$.
    e. Update $Q(s,a)$ according to the rule above.
    Let the new state be $s' \leftarrow T(s,a)$.

$\beta$ and $\gamma$ are the tunable learning parameters. $\beta$ determines the learning rate. $\beta = 1$ disregards all history accumulated in the current $Q$ value and resets $Q$ to the sum of the received and expected reward at every time step, usually resulting in oscillations.

$\gamma$ is the discount factor for future reward. Ideally, $\gamma$ should be as close to 1 as possible so that the relevance of future reward is maximized. In deterministic worlds $\gamma$ can be set to 1, but in the general case two algorithms with $\gamma = 1$ cannot be compared since, in the limit, the expected future reinforcement of both will go to $\infty$.

The choice of initial Q values can affect the speed of convergence since the farther they are from the optimal policy the longer it takes to converge. If initialized to 0's in a problem whose optimal policy is positive, the algorithm will tend to converge to the first positive value, without exploring alternatives, so random actions must be added to guarantee that the entire action space is explored (Kaelbling 1990). Alternatively, if the optimal policy can be roughly estimated, $Q$ values can be initialized to be

higher and decreased over time. However, Q is sensitive to the coupling between the initial values and the reinforcement function. If the reinforcement function is strictly positive and the Q table is initialized to values exceeding the optimal policy, the system will take longer to converge than if the reinforcement function contains some negative signals.

# Appendix C

# Glossary

**adaptability** the ability to cope with internal and external changes.

**agent** a computational process, physical or otherwise.

**arbitration** the problem of coordinating the activity of multiple input behaviors in order to produce desired output behavior.

**basic behaviors** building blocks for control, planning, and learning.

**basic behavior set** a basis set of behaviors that are directly, or by combination, sufficient for reaching all goals of a system. The elements of the set are not mutually reducible.

**behavior** a control law that achieves and/or maintains some goal.

**behavior conditions** proper subsets of the state space necessary and sufficient for activating a behavior.

**collective behavior** an observer-subjective definition of some spatial and/or temporal pattern of interactions between multiple agents.

**cooperation** a form of interaction, usually based on communication.

**ensemble behavior** observable global behavior of a group or collection of agents

**fortuitous reward** a reward received for an inappropriate behavior that happened to achieve the desired goal.

**group density** the ratio of the sum of the agents' footprints and the size of available interaction space.

**direct communication** an action with the sole purpose of transmitting information.

**directed communication** communication aimed at a particular receiver.

**direct behavior combination** a temporal overlap of two or more behaviors. More than one behavior is active at a time. Implemented with a summation operator.

**embodiment** the state of being embodied, having a body with physical constraints and properties.

**explicit cooperation** a set of interactions which involve exchanging information or performing actions in order to benefit another agent.

**footprint** the sphere of an agent's its influence.

**implicit cooperation** a form of interactions consisting of actions that are a part of the agent's own goal–achieving behavior, but may have effects in the world that help other agents achieve their goals.

**group** a collection of size three or more.

**homogeneity** the property of being situated in the same world, embodied with similar dynamics and executing identical control programs.

**interaction** mutual influence on behavior.

**interference** any influence that partially or completely blocks an agents' goal–driven behavior.

**multi-agent control** producing the desired behavior from a multi–agent system.

**niche** a habitat, a class of environments for which an agent is adapted.

**non-directed communication** indirect and direct communication.

**multi-agent system** a system consisting of at least two agents.

**policy** a mapping of inputs, states, or conditions, to actions or behaviors.

**situatedness** the property of being situated, of existing in some context, in an environment which involves interaction dynamics.

**stigmergic communication** communication based on modifications of the environment rather than direct message passing.

**temporal behavior combination** a temporal sequence of two or more behaviors. Only one behavior is active at a time. Implemented with a switching operator.

# Bibliography


Abraham, R. H. & Shaw, C. D. (1992), *Dynamics: The Geometry of Behavior*, Addison–Wesley, California.

Abravanel, E. & Gingold, H. (1985), 'Learning Via Observation During the Second Year of Life', *Developmental Psychology* **21**(4), 614–623.

Agre, P. E. & Chapman, D. (1987), Pengi: An Implementation of a Theory of Activity, *in* 'Proceedings, AAAI-87', Seattle, WA, pp. 268–272.

Agre, P. E. & Chapman, D. (1990), What Are Plans for?, *in* P. Maes, ed., 'Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back', The MIT Press, pp. 17–34.

Altenburg, K. & Pavicic, M. (1993), Initial Results in the Use of Inter-Robot Communication for a Multiple, Mobile Robotic System, *in* 'Proceedings, IJCAI-93 Workshop on Dynamically Interacting Robots', Chambery, France, pp. 96–100.

Arkin, R. C. (1989), Towards the Unification of Navigational Planning and Reactive Control, *in* 'AAAI Spring Symposium on Robot Navigation', pp. 1–5.

Arkin, R. C. (1992), 'Cooperation without Communication: Multiagent Schema Based Robot Navigation', *Journal of Robotic Systems.*

Arkin, R. C., Balch, T. & Nitz, E. (1993), Communication of Behavioral State in Multi-agent Retrieval Tasks, *in* 'IEEE International Conference on Robotics and Automation', pp. 588–594.

Asendorpf, J. B. & Baudonniere, P.-M. (1993), 'Self-Awareness and Other-Awareness: Mirror Self-Recognition Synchronic Imitation Among Unfamiliar Peers', *Developmental Psychology* **29**(1), 89–95.

Assad, A. & Packard, N. (1992), Emergent Colonization in an Artificial Ecology, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 143–152.

Atkeson, C. G. (1989), 'Learning Arm Kinematics and Dynamics', *Annual Review of Neuroscience* **12**, 157–183.

Atkeson, C. G. (1990), Memory-Based Approaches to Approximating Continuous Functions, *in* 'Proceedings, Sixt Yale Workshop on Adaptive and Learning Systems'.

Atkeson, C. G., Aboar, E. W., McIntyre, J. & Reinkensmeyer, D. J. (1988), Model-Based Robot Learning, Technical Report AIM-1024, MIT.

Axelrod, R. (1984), *The Evolution of Cooperation*, Basic Books, New York.

Bandura, A. (1971), Analysis of Modeling Processes, *in* A. Bandura, ed., 'Psychological Modeling: Conflicting Theories', Aldine–Atherton, pp. 1–62.

Bandura, A. (1977), *Social Learning Theory*, Prentice-Hall, Inc., Englewood Cliffs, N.J.

Bandura, A. & Walters, R. H. (1963), *Social Learning and Personality Development*, Holt, Rinehart and Winston, Inc, New York.

Barman, R. A., Kingdon, J. J., Mackworth, A. K., Pai, D. K., Sahota, M. K., Wilkinson, H. & Zhang, Y. (1993), Dynamite: A Testbed for Multiple Mobile Robots, *in* 'Proceedings, IJCAI-93 Workshop on Dynamically Interacting Robots', Chambery, France, pp. 38–44.

Barto, A. G., Bradtke, S. J. & Singh, S. P. (1993), 'Learning to Act using Real-Time Dynamic Programming', *AI Journal.*

Barto, A. G., Sutton, R. S. & Anderson, C. W. (1983), 'Neuronlike Elements That Can Solve Difficult Learning Control Problems', *IEEE Transactions on Systems, Man, and Cybernetics* **13**, 835–846.

Belić, M. R., Skarka, V., Deneubourg, J. L. & Lax, M. (1986), 'Mathematical Model of Honeycomb Construction', *Mathematical Biology* **24**, 437–449.

Benhamou, S. & Bovet, P. (1990), Modeling and Simulation of Animal's Movements, *in* J. A. Meyer & S. Wilson, eds, 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', The MIT Press.

Beni, G. & Hackwood, S. (1992), The Maximum Entropy Principle and Sensing in Swarm Intelligence, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 153–160.

Bizzi, E. & Mussa-Ivaldi, F. A. (1990), Muscle Properties and the Control of Arm Movement, *in* D. N. Osherson, S. Kosslyn & J. M. Hollerbach, eds, 'Visual Cognitivion and Action, Vol. 2', The MIT Press, pp. 213–242.

Bizzi, E., Mussa-Ivaldi, F. A. & Giszter, S. (1991), 'Computations Underlying the Execution of Movement: A Biological Perspective', *Science* **253**, 287–291.

Bonabeau, E. W. (1993), On the appease and dangers of synthetic reductionism, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 86–10.

Brock, D. L., Montana, D. J. & Ceranowicz, A. Z. (1992), Coordination and Control of Multiple Autonomous Vehicles, *in* 'IEEE International Conference on Robotics and Automation', pp. 2725–2730.

Brooks, R. A. (1986), 'A Robust Layered Control System for a Mobile Robot', *IEEE Journal of Robotics and Automation* **RA-2**, 14–23.

Brooks, R. A. (1990*a*), The Behavior Language; User's Guide, Technical Report AIM-1127, MIT Artificial Intelligence Lab.

Brooks, R. A. (1990*b*), Elephants Don't Play Chess, *in* P. Maes, ed., 'Designing Autonomous Agents', The MIT Press, pp. 3–15.

Brooks, R. A. (1991*a*), Artificial Life and Real Robots, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press.

Brooks, R. A. (1991*b*), Intelligence Without Reason, *in* 'Proceedings, IJCAI-91'.

Brooks, R. A. (1991*c*), 'Intelligence Without Representation', *Artificial Intelligence* **47**, 139–160.

Brooks, R. A. & Connell, J. H. (1986), Asynchronous Distributed Control System for a Mobile Robot, *in* 'SPIE', Cambridge, Massachusetts.

Brooks, R. A. & Matarić, M. J. (1993), Real Robots, Real Learning Problems, *in* 'Robot Learning', Kluwer Academic Press, pp. 193–213.

Brooks, R. A., Maes, P., Matarić, M. J. & Moore, G. (1990), Lunar Base Construction Robots, *in* 'IEEE International Workshop on Intelligent Robots and Systems (IROS-90)', Tokyo, pp. 389–392.

Brown, T. A. & McBurnett, M. (1993), Political Life on a Lattice, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 113–126.

Calenbuhr, V. & Deneubourg, J. L. (1992), 'A Model for Osmotropotactic Orientation (I)', *Journal of Theoretical Biology.*

Caloud, P., Choi, W., Latombe, J., LePape, C. & Yim, M. (1990), Indoor Automation with Many Mobile Robots, *in* 'IROS-90', Tsuchiura, Japan, pp. 67–72.

Camazine, S. (1993), Collective Intelligence in Insect Colonies by Means of Self-Organization, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 158–173.

Canny, J. F. (1988), *The Complexity of Robot Motion Planning*, The MIT Press, Cambridge, Massachusetts.

Chapman, D. (1987), 'Planning for Conjunctive Goals', *Aritifical Intelligence* **32**, 333–377.

Chapman, D. & Kaelbling, L. P. (1991), Input Generalization in Delayed Reinforcement Learning: An Algorithm and Performance Comparisons, *in* 'Proceedings, IJCAI-91', Sydney, Australia.

Chase, I. D. (1982), 'Dynamics of Hierarchy Formation: The Sequential Development of Dominance Relationships', *Behaviour* **80**, 218–240.

Chase, I. D. (1993), Generating Societies: Collective Social Patterns in Humans and Animals, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 174–191.

Chase, I. D. & Rohwer, S. (1987), 'Two Methods for Quantifying the Development of Dominance Heirarchies in Large Groups with Application to Harris' Sparrows', *Animal Behavior.*

Chase, I. D., Bartolomeo, C. & Dugatkin, L. A. (1994), 'Aggressive interactions and inter-contest interval: how long do winners keep winning?', *Animal Behavior, in press.*

Chatila, R. & Laumond, J.-C. (1985), Position Referencing and Consistent World Modeling for Mobile Robots, *in* 'IEEE International Conference on Robotics and Automation'.

Cheney, D. L. & Seyfarth, R. M. (1990), *How Monkeys See the World*, The University of Chicago Press, Chicago.

Cheney, D. L. & Seyfarth, R. M. (1991), Reading Minds or Reading Behaviour?, *in* A. Whiten, ed., 'Natural Theories of Mind', Basil Blackwell.

Chris G. Langton, e. (1989), *Artificial Life*, Addison–Wesley.

Clearwater, S. H., Huberman, B. A. & Hogg, T. (1991), 'Cooperative Solution of Constraint Satisfaction Problems', *Science* **254**, 1181–1183.

Cliff, D., Husbands, P. & Harvey, I. (1993), Evolving Visually Guided Robots, *in* 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', The MIT Press, pp. 374–383.

Colorni, A., Dorigo, M. & Maniezzo, V. (1992), Distributed Optimization by Ant Colonies, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 134–142.

Connell, J. H. (1990), *Minimalist Mobile Robotics: A Colony Architecture for an Artificial Creature*, Academic Press.

Connell, J. H. (1991), SSS: A Hybrid Architecture Applied to Robot Navigation, *in* 'IEEE International Conference on Robotics and Automation', Nice, France, pp. 2719–2724.

Corbara, B., Drogoul, A., Fresneau, D. & Lalande, S. (1993), Simulating the Sociogenesis Process in Ant Colonies with MANTA, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 224–235.

Dallas, J. (1990), Co-operative Search Behavior in a Group of Lego Robots, Master's thesis, University of Edinburgh.

Dario, P. & Rucci, M. (1993), An Approach to Disassembly Problem in Robotics, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 460–468.

Dario, P., Ribechini, F., Genovese, V. & Sandini, G. (1991), Instinctive Behaviors and Personalities in Societies of Cellular Robots, *in* 'IEEE International Conference on Robotics and Automation', pp. 1927–1932.

DeAngelis, D. L., Post, W. M. & Travis, C. C. (1986), 'Positive Feedback in Natural Systems', *Biomathematics.*

Decker, K. & Lesser, V. (1993*a*), A One-shot Dynamics Coordination Algorithm for Distributed Sensor Networks, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 210–216.

Decker, K. & Lesser, V. (1993*b*), Quantitative Modeling of COmplex Computational Task Environments, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 217–224.

Deneubourg, J.-L. & Goss, S. (1989), Collective Patterns and Decision-Making, *in* 'Ethology, Ecology and Evolution 1', pp. 295–311.

Deneubourg, J. L., Aron, S., Goss, S., Pasteels, J. M. & Duernick, G. (1986), 'Random Behaviour, Amplification Processes and Number of Participants: How they Contribute to the Foraging Properties of Ants', *Physica 22D* pp. 176–186.

Deneubourg, J. L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C. & Chretien, L. (1990), The Dynamics of Collective Sorting, *in* 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', The MIT Press, pp. 356–363.

Deneubourg, J. L., Goss, S., Pasteels, J. M., Fresneau, D. & Lachaud, J. P. (1987), 'Self-Organization Mechanisms in Ant Societies, II: Learning in Foraging and Division of Labor', *From Individual to Collective Behavior in Social Insects* **54**, 177–196.

Deneubourg, J. L., Theraulax, G. & Beckers, R. (1992), Swarm-Made Architectures, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 123–133.

DeScnutter, G. & Nuyts, E. (1993), Birds use self-organized social behaviours to regulate their dispersal over wide areas: evidences from gull roosts., *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 285–309.

Donald, B. R., Jennings, J. & Rus, D. (1993), Experimental Information Invariants for Cooperating Autonomous Mobile Robots, *in* 'Proceedings, Internatioonal Symposim on Robotics Research', Hidden valley, PA.

Doyle, J. & Sacks, E. P. (1989), Stochastic Analysis of Qualitative Dynamics, Technical Report LCS-TM-418, MIT Laboratory for Computer Science.

Drogous, A., Ferber, J., Corbara, B. & Fresneau, D. (1992), A Behavioral Simulation Model for the Study of Emergent Social Structures, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 161–170.

Dudek, G., Jenkin, M., Milios, E. & Wilkes, D. (1993), A Taxonomy for Swarm Robotics, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 441–447.

Durfee, E. H., Lee, J. & Gmytrasiewicz, P. J. (1993), Overeager Reciprocal Rationality and Mixed Strategy Equilibria, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 225–230.

Ephrati, E. (1992), Constrained Intelligent Action: Planning Under the Influence of a Master Agent, *in* 'Proceedings, AAAI-92', San Jose, California, pp. 263–268.

Erdmann, M. (1989), On Probabilistic Strategies for Robot Tasks, PhD thesis, MIT.

Erdmann, M & Lozano-Pérez, T. (1987), 'On Multiple Moving Objects', *Algorithmica* **2**, 477–521.

Ferrell, C. (1993), Robust Agent Control of an Autonomous Robot with Many Sensors and Actuators, Technical Report AI-TR-1443, MIT Artificial Intelligence Laboratory.

Fikes, R. E. & Nilsson, N. J. (1971), 'STROPS: A new approach to the application of theorem proving to problem solving', *Artificial Intelligence* **2**, 189–208.

Firby, R. J. (1987), An investigation into reactive planning in complex domains, *in* 'Proceedings, Sixth National Conference on Artifical Intelligence', Seattle, pp. 202–206.

Floreano, D. (1993), Patterns of Interactions in Shared Environments, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 347–366.

Forrest, S. (1989), *Emergent Computation: Self-Organizing, Collective, and Cooperative Phenomena in Natural and Artificial Computing Networks*, North Holland, Amsterdam.

Foster, T. C., Castro, C. A. & McNaughton, B. L. (1989), 'Spatial Selectivity of Rat Hippocampal Neurons: Dependence on Preparedness for Movement', *Science* pp. 1589–1582.

Franklin, J. A. & Selfridge, O. G. (1990), Some New Directions for Adaptive Control Theory in Robotics, *in* W. T. Miller, R. S. Sutton & P. J. Werbos, eds, 'Neural Networks for Control', The MIT Press, pp. 349–360.

Franks, N. R. (1989), 'Army Ants: A Collective Intelligence', *American Scientist* **77**, 139–145.

Fukuda, T., Nadagawa, S., Kawauchi, Y. & Buss, M. (1989), Structure Decision for Self Organizing Robots Based on Cell Structures - CEBOT, *in* 'IEEE International Conference on Robctics and Automation', Scottsdale, Arizona, pp. 695–700.

Fukuda, T., Sekiyama, K., Ueyama, T. & Arai, F. (1993), Efficient Communication Method in the Cellular Robotics System, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 1091–1096.

Gallagher, J. C. & Beer, R. D. (1993), A Qualitative Dynamics Analysis of Evolved Locomotion Controller, *in* 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', The MIT Press, pp. 71–80.

Gasser, L. & Huhns, M. N. (1989), *Distributed Artificial Intelligence*, Pitman, London.

Georgeff, M. P. & Lansky, A. L. (1987), Reactive Reasoning and Planning, *in* 'Proceedings, Sixth National Conference on Artifical Intelligence', Seattle, pp. 677–682.

Giralt, G., Chatila, R. & Vaisset, M. (1983), An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots, *in* M. Brady & R. Paul, eds, 'First International Symposium on Robotics Research', The MIT Press, Cambridge, Massachusetts.

Gleitman, H. (1981), *Psychology*, W. W. Norton & Co., New York.

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA.

Gomez, J. C. (1991), Visual Behavior as a Window for Reading the Mind of Others in Primates, *in* A. Whiten, ed., 'Natural Theories of Mind', Basil Blackwell.

Goss, S., Deneubourg, J. L., Beckers, R. & Henrotte, J. (1993), Recipes for Collective Movement, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 400–410.

Gould, J. L. (1982), *Ethology; The Mechanisms and Evolution of Behavior*, W. W. Norton & Co., New York.

Gould, J. L. (1987), Flower-shape, Landmark, and Locale Memory in Honeybees, *in* R. Menzel & A. Mercer, eds, 'Neurobiology and Behavior of Honeybees', Springer–Verlag.

Gutzwiller, M. (1992), 'Quantum Chaos', *Scientific American.*

Harnad, S. (1990), 'The Symbol Grounding Problem', *Physica D* **42**, 335–346.

Hillis, W. D. (1990), 'Co-evolving Parasites Improve SImulated Evolution as an Optimization Procedure', *Physica D* **42**, 228–234.

Hinton, G. E. (1990), Connectionist Learning Procedures, *in* Kodratoff & Michalski, eds, 'Machine Learning, An Artificial Intelligence Approach, Vol. 3', Morgan Kaufmann, pp. 555–610.

Hogeweg, P. & Hesper, B. (1985), 'Socioinformatic Processes: MIRROR Modelling Methodology', *Journal of Theoretical Biology* **113**, 311–330.

Hogg, T. & Williams, C. P. (1993), Solving the Really Hard Problems with Cooperative Search, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 231–236.

Holland, J. H. (1985), Properties of the bucket brigade algorithm, *in* 'Proceedings, International Conference on genetic Algorithms and Their Applications', Pittsburgh, PA, pp. 1–7.

Holland, J. H. (1986), Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems, *in* R. S. Michalski, J. G. Carbonell & T. M. Mitchell, eds, 'Machine Learning, An Artificial Intelligence Approach, Vol. 2', Morgan Kaufmann, Los Altos, CA.

Huang, Q. & Beni, G. (1993), Stationary Waves in 2–Dimensional Cyclic Swarms, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 433–440.

Huber, M. J. & Durfee, E. H. (1993), Observational Uncertainty in Plan Recognition Among Interacting Robots, *in* 'Proceedings, IJCAI-93 Workshop on Dynamically Interacting Robots', Chambery, France, pp. 68–75.

Huberman, B. A. (1990), 'The Performance of Cooperative Processes', *Physica D* **42**, 38–47.

Jaakkola, T. & Jordan, M. I. (1993), 'On the Convergence of Stochastic Iterative Dynamic Programming Algorithms', *Submitted to Neural Computation.*

Jordan, M. I. & Rumelhart, D. E. (1992), 'Forward Models: Supervised Learning with a Distal Teacher', *Cognitive Science* **16**, 307–354.

Kaelbling, L. P. (1990), Learning in Embedded Systems, PhD thesis, Stanford University.

Kephart, J. O., Hogg, T. & Huberman, B. A. (1990), 'Collective Behavior of Predictive Agents', *Physica D* **42**, 48–65.

Keshet, L. E. (1993), Trail Following as an Adaptable Mechanism for Population Behaviour, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 326–346.

Kessin, R. H. & Campagne, M. M. V. L. (1992), 'The Development of a Social Amoeba', *American Scientist* **80**, 556–565.

Kolen, J. F. & Pollack, J. B. (1993), Apparent Computational Complexity in Physical Systems, *in* 'Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society', Boulder, Colorado, pp. 617–622.

Kosoresow, A. P. (1993), A Fast First-Cut Protocol for Agent Coordination, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 237–242.

Koza, J. R. (1990), Evolution and Co-evolution of Computer Programs to Control Independently-acting Agents, *in* 'Proceedings, Simulation of Adaptive Behavior SAB-90', The MIT Press, Paris, France, pp. 366–375.

Kraus, S. (1993), Agents Contracting Tasks in Non-Collaborative Environments, *in* 'Proceedings, AAAI-93', Washington, DC, pp. 243–248.

Kube, C. R. (1992), Collective Robotic Intelligence: A Control Theory for Robot Populations, Master's thesis, University of Alberta.

Kube, C. R. & HongZhang (1992), Collective Robotic Intelligence, *in* 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', pp. 460–468.

Kube, C. R., Zhang, H. & Wang, X. (1993), Controlling Collective Tasks With an ALN, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 289–293.

Kurosu, K., Furuya, T. & Soeda, M. (1993), Fuzzy Control of Group With a Leader and Their Behaviors, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 1105–1109.

Laird, J. E. & Rosenbloom, P. S. (1990), Integrating, Execution, Planning, and Learning in Soar for External Environments, *in* 'Proceedings, AAAI-90', pp. 1022–1029.

Langton, C. G. (1990), 'Computation at the Edge of Chaos: Phase Transitions and Emergent Computation', *Physica D* **42**, 12–37.

Lin, L.-J. (1991), Self-improvement Based on Reinforcement Learning, Planning and Teaching, *in* 'Proceedings, Eighth International Conference on Machine Learning', Morgan Kaufmann, Evanston, Illinois, pp. 323–327.

Lozano-Pérez, T., Mason, M. T. & Taylor, R. H. (1984), 'Automatic Synthesis of Fine Motion Strategies for Robots', *International Journal of Robotics Research* **3**(1), 3–24.

Lynch, N. A. (1993), Simulation Techniques for Proving Properties of Real-Time Systems, Technical Report MIT–LCS–TM–494, MIT.

Lynch, N. A. & Tuttle, M. R. (1987), Hierarchical Correctness Proofs for Distributed Algorithms, Technical Report MIT–LCS–TR–387, MIT.

MacLennan, B. J. (1990), Evolution of Communication in a Population of Simple Machines, Technical Report Computer Science Department Technical Report CS–90–104, University of Tennessee.

Maes, P. (1989), The Dynamics of Action Selection, *in* 'IJCAI-89', Detroit, MI, pp. 991–997.

Maes, P. (1991), Learning Behavior Networks from Experience, *in* F. Varela & P. Bourgine, eds, 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', The MIT Press, pp. 48–57.

Maes, P. & Brooks, R. A. (1990), Learning to Coordinate Behaviors, *in* 'Proceedings, AAAI-91', Boston, MA, pp. 796–802.

Mahadevan, S. & Connell, J. (1991*a*), Automatic Programming of Behavior-based Robots using Reinforcement Learning, *in* 'Proceedings, AAAI-91', Pittsburgh, PA, pp. 8–14.

Mahadevan, S. & Connell, J. (1991*b*), Scaling Reinforcement Learning to Robotics by Exploiting the Subsumption Architecture, *in* 'Eight International Workshop on Machine Learning', Morgan Kaufmann, pp. 328–337.

Matarić, M. J. (1990*a*), A Distributed Model for Mobile Robot Environment-Learning and Navigation, Technical Report AI-TR-1228, MIT Artificial Intelligence Laboratory.

Matarić, M. J. (1990*b*), Navigating With a Rat Brain: A Neurobiologically-Inspired Model for Robot Spatial Representation, *in* J. A. Meyer & S. Wilson, eds, 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior', The MIT Press, pp. 169–175.

Matarić, M. J. (1991), A Comparative Analysis of Reinforcement Learning Methods, Technical Report AIM-1322, MIT Artificial Intelligence Lab.

Matarić, M. J. (1992*a*), Behavior-Based Systems: Key Properties and Implications, *in* 'IEEE International Conference on Robotics and Automation, Workshop on Architectures for Intelligent Control Systems', Nice, France, pp. 46–54.

Matarić, M. J. (1992*b*), Designing Emergent Behaviors: From Local Interactions to Collective Intelligence, *in* 'From Animals to Animats: International Conference on Simulation of Adaptive Behavior'.

Matarić, M. J. (1992*c*), 'Integration of Representation Into Goal-Driven Behavior-Based Robots', *IEEE Transactions on Robotics and Automation* **8**(3), 304–312.

Matarić, M. J. (1993), Kin Recognition, Similarity, and Group Behavior, *in* 'Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society', Boulder, Colorado, pp. 705–710.

Matarić, M. J. (1994), Learning to Behave Socially, *in* 'sumitted to the Third International Conference on Simulation of Adaptive Behavior'.

McFarland, D. (1985), *Animal Behavior*, Benjamin Cummings.

McFarland, D. (1987), The Oxford Companion to Animal Behavior, *in* 'Oxford, University Press'.

Meier-Koll, A. & Bohl, E. (1993), Time-structure analysis in a village community of Columbian Indians; A mathematically simulated system of ultradian oscillators, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 718–749.

Miceli, M. & Cesta, A. (1993), Strategic Social Planning: Looking for Willingness in Multi-Agent Domains, *in* 'Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society', Boulder, Colorado, pp. 741–746.

Miller, W. T., Sutton, R. S. & Werbos, P. J. (1990), *Neural Networks for Control*, The MIT Press.

Minsky, M. L. (1954), Theory of Neural–Analog Reinforcement Systems and Its Application to the Brain–Model Problem, PhD thesis, Princeton.

Minsky, M. L. (1986), *The Society of Mind*, Simon and Schuster, New York.

Miramontes, O., Sole, R. V. & Goodwin, B. C. (1993), Antichaos in ants: the excitability metaphor at two hierarchical levels, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 790–807.

Moore, A. W. (1992), 'Fast, Robust Adaptive Control by Learning only Forward Models', *Advances in Neural Information Processing 4* pp. 571–579.

Moravec, H. P. & Cho, D. W. (1989), A Bayesian Method for Certainty Grids, *in* 'AAAI Spring Symposium on Robot Navigation', pp. 57–60.

Muller, M. & Wehner, R. (1988), Path Integration in Desert Ants: Cataglyphis Fortis, *in* 'Proceedings of the Natural Academy of Sciences'.

Mussa-Ivaldi, F. A. & Giszter, S. (1992), 'Vector field approximation: a computational paradigm for motor control and learning', *Biological Cybernetics* **67**, 491–500.

Noreils, F. R. (1992), An Architecture for Cooperative and Autonomous Mobile Robots, *in* 'IEEE International Conference on Robotics and Automation', pp. 2703–2710.

Noreils, F. R. (1993), 'Toward a Robot Architecture Integrating Cooperation between Mobile Robots: Application to Indoor Environment', *The International Journal of Robotics Research* **12**(1), 79–98.

Parker, L. E. (1993*a*), An Experiment in Mobile Robotic Cooperation, *in* 'Proceedings, Robotics for Challenging Environment', Albuquerque, New Mexico.

Parker, L. E. (1993*b*), Learning in Cooperative Robot Teams, *in* 'Proceedings, IJCAI-93 Workshop on Dynamically Interacting Robots', Chambery, France, pp. 12–23.

Parker, L. E. (1994), Heterogeneous Multi-Robot Cooperation, PhD thesis, MIT.

Payton, D. (1990), Internalized Plans: a representation for action resources, *in* P. Maes, ed., 'Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back', The MIT Press.

Payton, D., Keirsey, D., Kimble, D., Krozel, J. & Rosenblatt, K. (1992), 'Do Whatever Works: A robust approach to fault-tolerant autonomous control', *Jo rnal of Applied Intelligence* **3**, 226–249.

Piaget, J. (1962), *Play, Dreams and Imitation in Children*, W. W. Norton & Co., New York.

Pinker, S. (1994), *The Language Instinct*, William Morrow and Company, Inc., New York.

Pomerleau, D. A. (1992), Neural Network Perception for Mobile Robotic Guidance, PhD thesis, Carnegie Mellon University, School of Computer Science.

Read, S. J. & Miller, J. C. (1993), Explanatory coherence in the construction of mental models of others, *in* 'Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society', Boulder, Colorado, pp. 836–841.

Resnick, M. (1992), Beyond the Centralized Mindset; Exploration in Masssively-Parallel Microworlds, PhD thesis, MIT.

Reynolds, C. W. (1987), 'Flocks, Herds, and Schools: A Distributed Behavioral Model', *Computer Graphics* **21**(4), 25–34.

Rosenschein, G. & Genesereth, M. R. (1985), Deals Among Rational Agents, *in* 'IJCAI-85', pp. 91–99.

Rosenschein, J. S. (1993), Consenting Agents: Negotiation Mechanisms for Multi-Agent Systems, *in* 'IJCAI-93', pp. 792–799.

Rosenschein, S. J. & Kaelbling, L. P. (1986), The Synthesis of Machines with Provable Epistemic Properties, *in* J. Halpern, ed., 'Theoretical Aspects of Reasoning About Knowledge', Morgan Kaufmann, Los Altos, CA, pp. 83–98.

Rosenthal, T. L. & Zimmerman, B. J. (1978), *Social Learning and Cognition*, Academic Press, New York.

Samuel, A. L. (1959), 'Some studies in machine learning using the game of checkers', *IBM Journal of Research and Development* **3**, 211–229.

Sandini, G., Lucarini, G. & Varoli, M. (1993), Gradient Driven Self-Organizing Systems, *in* 'IEEE/TSJ International Conference on Intelligent Robots and Systems', Yokohama, Japan, pp. 429–432.

Schaal, S. & Atkeson, C. G. (1994), 'Robot Juggling: An Implementation of Memory-Based Learning', *Control Systems Magazine.*

Schmieder, R. W. (1993), A Knowledge-Tracking Algorithm for Generating Collective Behavior in Individual-Based Populations, *in* 'Toward A Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life', pp. 980–989.

Schoppers, M. J. (1987), Universal Plans for Reactive Robots in Unpredictable Domains, *in* 'IJCAI-87', Menlo Park, pp. 1039–1046.

Seeley, T. D. (1989), 'The Honey Bee Colony as a Superorganism', *American Scientist* **77**, 546–553.

Shoham, Y. & Tennenholtz, M. (1992), On the synthesis of useful social laws for artificial agent societies, *in* 'Proceedings, AAAI-92', San Jose, California, pp. 276–281.

Simon, H. (1969), *The Sciences of the Artificial*, The MIT Press.

Singh, S. P. (1991), Transfer of Leanring Across Compositions of Sequential Tasks, *in* 'Proceedings, Eighth International Conference on Machine Learning', Morgan Kaufmann, Evanston, Illinois, pp. 348–352.

Sismondo, E. (1990), 'Synchoronous, Alternating, and Phase–Locked Stridulation by a Tropical Katydid', *Science* **249**, 55–58.

Steels, L. (1989), Cooperation Between Distributed Agents Through Self-Organization, *in* 'Workshop on Multi-Agent Cooperation', North Holland, Cambridge, UK.

Steels, L. (1994), 'The Artificial Life Roots of Artificial Intelligence', *to appear in The Artificial Life Journal.*

Sussman, G. J. & McDermott, D. V. (1972), From PLANNER to CONNIVER–A genetic approach, *in* 'Proceedings, Fall Joint Computer Conference', pp. 1171–1179.

Sutton, R. (1988), 'Learning to Predict by Method of Temporal Differences', *The Journal of Machine Learning* **3**(1), 9–44.

Sutton, R. S. (1990), Integrated Architectures for Learning, Planning and Reacting Based on Approximating Dynamic Programming, *in* 'Proceedings, Seventh International Conference on Machine Learning', Austin, Texas.

Tan, M. (1993), Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents, *in* 'Proceedings, Tenth International Conference on Machine Learning', Amherst, MA, pp. 330–337.

Thrun, S. B. & Mitchell, T. M. (1993), Integrating Indictive Neural Network Learning and Explanation–Based Learning, *in* 'Proceedings, IJCAI-93', Chambery, France.

Tomasello, M., Kruger, A. C. & Rather, H. H. (1992), 'Cultural Learning', *to appear in The Journal of Brain and Behavior Sciences.*

Travers, M. (1988), Animal Construction Kits, *in* C. Langton, ed., 'Artificial Life', Addison–Wesley.

Vander, A. J., Sherman, J. H. & Luciano, D. S. (1980), *Human Physiology*, McGraw-Hill Book Company, New York.

Waterman, T. H. (1989), *Animal Navigation*, Scientific American Library, New York.

Watkins, C. J. C. H. (1989), Learning from Delayed Rewards, PhD thesis, King's College, Cambridge.

Watkins, C. J. C. H. & Dayan, P. (1992), 'Q-Learning', *Machine Learning* **8**, 279–292.

Wehner, R. (1987), 'Matched Filters – Neural Models of the External World', *Journal of Computational Physiology* **A**(161), 511–531.

Weisbuch, G. (1991), Complex System Dynamics, *in* 'Lecture Notes Vol. II, Santa Fe Institute Studies in the Sciences of Complexity', Addison–Wesley, New York.

Werner, G. M. & Dyer, M. G. (1990), Evolution of Communication in Artificial Organisms, Technical Report UCLA–AI–90–06, University of California, Los Angeles.

Whitehead, S. D. (1992), Reinforcement Learning for the Adaptive Control of Perception and Action, PhD thesis, University of Rochester.

Whitehead, S. D. & Ballard, D. H. (1990), Active Perception and Reinforcement Learning, *in* 'Proceedings, Seventh International Conference on Machine Learning', Austin, Texas.

Whitehead, S. D., Karlsson, J. & Tenenberg, J. (1993), Learning Multiple Goal Behavior via Task Decomposition and Dynamic Policy Merging, *in* J. H. Connell & S. Mahadevan, eds, 'Robot Learning', Kluwer Academic Publishers, pp. 45–78.

Wiggins, S. (1990), *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, Springer–Verlag, New York.