1997

Learning and Co-operation in Mobile Multi-Robot Systems

Kirke, Alexis John

Copyright Statement

Learning and Co-operation

in Mobile Multi-Robot Systems

by

Alexis John Kirke

A thesis submitted to the University of Plymouth

in partial fulfillment for the degree of

Doctor of Philosophy

School of Computing

Faculty of Technology

December 1997

Learning and Co-operation in Mobile Multi-Robot Systems

Alexis John Kirke

Abstract

This thesis addresses the problem of setting the balance between exploration and exploitation in teams of learning robots who exchange information. Specifically it looks at groups of robots whose tasks include moving between salient points in the environment. To deal with unknown and dynamic environments, such robots need to be able to discover and learn the routes between these points themselves. A natural extension of this scenario is to allow the robots to exchange learned routes so that only one robot needs to learn a route for the whole team to use that route. One contribution of this thesis is to identify a dilemma created by this extension: that once one robot has learned a route between two points, all other robots will follow that route without looking for shorter versions. This trade-off will be labeled the Distributed Exploration vs. Exploitation Dilemma, since increasing distributed exploitation (allowing robots to exchange more routes) means decreasing distributed exploration (reducing robots ability to learn new versions of routes), and vice-versa. At different times, teams may be required with different balances of exploitation and exploration. The main contribution of this thesis is to present a system for setting the balance between exploration and exploitation in a group of robots. This system is demonstrated through experiments involving simulated robot teams. The experiments show that increasing and decreasing the value of a parameter of the novel system will lead to a significant increase and decrease respectively in average exploitation (and an equivalent decrease and increase in average exploration) over a series of team missions. A further set of experiments show that this holds true for a range of team sizes and numbers of goals.

List of Contents

List of Tables

List of Figures

Acknowledgments

I would like to thank Mike Denham for his support, direction and enthusiasm; Guido Bugmann for keeping me aware of the passing of time, and the need for focus; Sue McCabe for being a model Ph.D. student for me to try to emulate; Raju Bapi for his greatly-valued friendship and good humor; Jack and Lynda Boitano, whose visit to the research group was a high-point of my time there; and finally Charles Garcia-Tobin for his friendship in my final year.

On a more personal note I would like to thank my Mother Margaret for putting me up - and putting up with me - for my first two years; my Father Nick for his pride, encouragement and reviving visits to Prague; and both my Mother and Father for their support, tolerance and love. I would also like to thank Nick, Jnr. for being a good friend and a good brother; my sister Jackie for her support and for putting up with my monologues on the phone; my life-long friend Wayne and the rest of the "Roads" from the top of North Hill; and William Bayliss for his companionship and enthusiasm during my weekend sessions.

AUTHOR'S DECLARATION

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award.

This study was financed with the aid of a studentship from the Engineering and Physcial Sciences Research Council.

Signed .....

Date .....18/8/98.....

Chapter 1

Introduction

1.0 Multi-Robot Systems

The field of robotics has evolved successively over the last half-century. Initially robots have been controlled by humans. Later robots which can control themselves have been introduced; and in recent years the idea of the robot controlling/interacting with other robots, or a Multi-Robot System (MRS), has begun to receive a great deal of attention. Self-controlled robots introduce the obvious benefits of autonomy, and Multi-robot systems also introduce a number of benefits. Some of these are given in [Arkin and Balch 1997]:

- Distributed Action: Many robots can be in many places at the same time

- Inherent Parallelism: Many robots can do many, perhaps different, things at the same time

- Divide and Conquer: Certain problems are well suited for decomposition and allocation among many robots

- Simpler is better: Often each agent in a team of robots can be simpler than a more comprehensive single robot solution

Another three reasons to study multi-robot systems can be added, from [Parker 1996], [Beckers et al 1995] and [Dautenhahn 1995]:

- Increased Robustness: Multiple robots increase reliability, since if one robot breaks down the other robots may still be able to finish the task

- Emergence of ordered behaviour: The social insect model is an existence proof that many simple agents can exhibit complex group behaviour, i.e. perform a complex task together

- Social robotics: The population of autonomous mobile robots will grow rapidly in the near future, thus implying a necessity for studies of interaction and co-ordination strategies amongst them

A number of potential applications have been suggested which can utilise the advantages of multi-robot systems. These applications include mine sweeping, multi-satellite defence systems, maintenance work and decommissioning in nuclear power plants, planetary exploration, lunar base construction, janitorial work, transportation of heavy or difficult loads, robot "platoons" (both on land and underwater), clean-up of toxic waste, search and rescue missions, and security, surveillance, and reconnaissance tasks.

One further reason to study multi-robot systems is to facilitate the search for the emergence of human-like intelligence in individual robots. "Social" Intelligence is the intelligence required for an animal to deal efficiently with other members of its species. The Social Intelligence Hypothesis [Dautenhahn 1997] is a hypothesis from primate psychology that states that primate intelligence "originally evolved to solve social problems and was only later extended to problems outside the social domain." This implies

that social intelligence may be a pre-requisite for human-like intelligence. Thus, if we hope to ever be able to produce truly autonomous robots with human-like intelligence, the studies of multiple robot interaction should run in parallel with single-robot research.

1.1 Mobile Multi-Robot systems

It will be noticed that the majority of applications listed in the last section required the use of *mobile* robots. As a result the vast majority of studies in multi-robot systems have concentrated on mobile robots. This thesis will also concentrate on mobile multi-robot systems. It will now be seen that there are two fundamental tasks a mobile robot can perform.

1.1.1 Fundamental Abilities For Mobile Robots

There are two fundamental types of abilities for mobile robots [Yoshimura et al 1996]. The first is *point-to-point motion* - moving from position A in the space to position B. This would be needed in a transportation goal ("go to A and get a load to take to B"), or for a remote stationary goal ("go to B and press the red button"). The second fundamental task is *sweeping* - moving so as to cover as much of the space as possible. This would be needed in mine sweeping, mapping unknown environments, or waste clearing.

There have been approximately equal amounts of work done on multi-robot sweeping, e.g. [Goldberg and Mataric 1997][Ichikawa and Hara 1996][Yamaguchi and Beni 1996] [Fontan and Mataric 1996][Beckers et al 1995][Werger and Mataric 1996][Unsal and Bay 1994][Arkin 1992][Sugihara and Suzuki 1990], and on multi-robot point-to-point motion, e.g. [Alami et al 1997][Premvuti and Yuta 1996][Li 1994][Wang and Premvuti 1994][Noborio 1994][Caloud et al 1990][Buckley 1989][Herman and Albus 1988][Erdmann and Lorano-Perez 1987]. This thesis will continue the trend of past work in

separating the two types of task and choosing to concentrate on one of them. Specifically it will look at the point-to-point task. (Though it will be seen later that sweeping tasks are needed when *learning* point-to-point tasks.) It will now be seen that there are 5 basic types of point-to-point tasks.

1.1.2 The Five Types of Interaction in Point-to-Point Tasks

Looking at the past work above it is possible to divide the potential interactions of robots in point-to-point tasks into five varieties:

1. *Task Assignment*: Distributing point-to-point goals around the team so they are achieved efficiently e.g. [Brummit and Stentz 1996][Ohko et al 1993][Li 1994][Asama et al 1992][Caloud et al 1990][Herman and Albus 1988][Elgimez and Kim 1990]

2. *Environment Resource Sharing*: Sharing the space resource so as to avoid collisions or "traffic jams" while multiple robots perform separate point-to-point tasks, e.g. [Alami et al 1997][Premvuti and Yuta 1996][Aguilar et al 1995][Wang and Premvuti 1994b][Li 1994][Ota et al 1994][Noborio 1994][Wang 1993][Caloud et al 1990][Buckley 1989][Erdmann and Lozano-Perez 1987]

3. *Environment Information Sharing*: Sharing by communication the map information that robots have about how to move between points e.g. [López de Màntaras et al 1997][Asama et al 1992][Herman and Albus 1988]

4. *Direct Imitation*: Imitating each others' routes through physical following e.g. [Billiard and Dautenhahn 1997][Demiris and Hayes 1996][Bakker and Kuniyoshi 1996][Dautenhahn 1995][Hayes and Demiris 1994]

5. *Robot Cues*: Robots becoming stationary "beacons" which other robots can move towards, e.g. [Bison and Trainito 1996][Vainio et al 1996]

It can be seen that more work has been done on the first and second types of interaction than on the other three. The last three types are actually conceptually related. Environment Information-Sharing is a form of "remote" imitation, just as imitation is a form of physically-grounded information-sharing. Also a robot becoming a beacon causes other robots to "imitate" it in the sense that robots move towards it, i.e. move to the same point as the beacon robot has moved. The beacon robot is also providing information by communicating that "this is the place to be".

1.1.3 Distributed Environment Information Sharing

This thesis will be concerned with the third type of point-to-point task: environment information sharing by communication. In particular it will look at *distributed* systems for map sharing. The only past work involving distributed systems for map sharing that I am aware of, [Asama et al 1992], does not detail or analyse the system involved. [López de Màntaras et al 1997][Herman and Albus 1988] give more detail about their sharing systems, but they are not distributed systems since they are based around a central agent. This concentration on distributed systems will be because they are more robust than centralised systems.

This disadvantage of centralised systems has been widely discussed and these discussions will be reviewed in the next chapter. However, the disadvantages of distributed map-sharing in map-learning MRS have not been highlighted or examined.

1.2 Potential Problems of Map Sharing in Learning MRS

As far as I am aware, no work has been done at all on the potential disadvantages of introducing map information sharing to distributed map-learning systems. [López de Màntaras et al 1997][Herman and Albus 1988] involve systems for map-sharing in a learning MRS, but do not address any of the potential problems of map-sharing in the learning context.

The primary original contribution of this thesis will involve highlighting a potential problem of map-sharing in learning MRS. Such a problem needs to be addressed because of the desirability of continuing to use learning and map-sharing MRS. So before describing the potential problem, the advantages of learning/map-sharing MRS over non-learning MRS will be discussed.

1.2.1 The Advantages of Learning in Mobile MRS

An important use of mobile robot systems is their deployment in environments too hazardous or inaccessible to humans[Barnes et al 1997][Parker 1994], for example planetary exploration, nuclear plant decommissioning, disaster recovery, etc. If the robots are navigating autonomously, and the environment is too hazardous to have been mapped accurately by humans (or has become too hazardous to re-map since the last mapping by humans - leading to an uncertainty as to its current state) then the robots will need to develop their initial model through learning. In this case, a learning multi-robot system with environment information sharing has a clear speed advantage over a single learning

robot. It will take much longer for a single robot to learn a complete model of the environment than for a number of robots to learn a distributed model which they can share to achieve goals (since multi-robot systems are more efficient at sweeping tasks than a single robot).

1.2.2 Learning in Dynamic Environments

Another reason for the importance of learning is that the world is constantly changing; therefore for many tasks a mobile multi-robot system will not be able to cope unless it can deal with a dynamic environment. The robots' model of the environment must be constantly updateable through learning. This idea suggests that robots need to be able to adapt to the environment at the same time as achieving goals. So learning and goal achievement need to occur simultaneously. This is known as *On-line Learning*. The name derives from the idea of not having to take an agent "off-line" from its task achievement when it needs to learn new information. Recognition of these advantages has led to most learning MRS involving on-line learning - e.g. [Sen 1996][Ota et al 1994][Parker 1996][Ueyama et al 1994][Mataric 1994].

1.2.3 The Exploration vs. Exploitation Dilemma

An important area of study in on-line learning algorithms is the Exploration vs. Exploitation Dilemma[Kaebling and Moore 1996][Thrun 1992]. This dilemma will be described in some detail, as it is directly related to the potential problems of learning in map-sharing MRS.

The Exploration vs. Exploitation Dilemma concerns the trade-off between learning how best to achieve goals and achieving goals. Suppose an on-line learning mobile robot is introduced to a new environment with a goal to get to a salient point, say A. Moving

randomly it finds and learns a route R from its introduction point, H, to A. Suppose that at a later time it is at H and is given the goal A again, should it immediately follow the known route R to its goal? Or should it experiment with a different route in case it finds a shorter one? If there is a much shorter route and the robot sticks with R, then it is doing the task less efficiently than it could. But if it tries to find a shorter route r and fails, then it will have increased the time it takes to get to A, thus reducing efficiency. This trade-off is unavoidable, hence the label "dilemma". This dilemma implies that the relative importance of exploration and exploitation of knowledge for a particular set of tasks must be decided.

1.3 The *Distributed* Exploration vs. Exploitation Dilemma

Having addressed above the advantages of learning, particularly on-line learning, in multi-robot systems, the potential problem mentioned in Section 1.2 will now be introduced. This problem is a multi-robot version of the Exploration vs. Exploitation Dilemma discussed in Section 1.2.2 above. This problem will be introduced through an example:

Robots 1,2 and 3 are put in an unknown environment with goals to get to salient points A,B,C and D. They start at a home position H (say a re-charger). When they are given the goals they will try to find them - but since they have no knowledge of the space they will initially have to move randomly, keeping a lookout for goals, and learning as they go. Suppose Robot 1 finds A and then finds B; then it will have a map with routes H->A and A->B. Similarly, suppose Robot 2 finds the routes B->C and C->D. Since 1 and 2 can share their maps with each other and with 3, all the robots can now achieve their goals without learning any more routes and return home until they are given more goals. This will include Robot 3, which can achieve its goals without contributing any information to

the distributed map. In particular, because these routes are now available through communication, no robot will bother trying to learn them by experience, thus reducing the chance of another robot finding a shorter more efficient version.

The only way to make robot 3 try to learn alternative shorter versions of a route is to withhold transmitting older versions of the route to it; but this will slow down goal achievement - thus leading to a dilemma. So the exploration vs. exploitation dilemma in multi-robot systems has two aspects:

- The standard exploration vs. exploitation: how do individual robots balance their exploration of the environment with their exploitation of their knowledge?

- The *distributed* exploration vs. exploitation problem: How do robots balance the quick achievement of goals through map-sharing with the learning of shorter routes by distributed learning?

In the same way that the owner of a robot must decide on how to set its balance in the exploration vs. exploitation dilemma, so the owner of a robot team must decide how to set the balance in the distributed exploration vs. exploitation dilemma.

1.4 The Original Contribution of this Thesis

As a result of the introduction of on-line learning into mobile map-sharing multi-robot systems there is a new issue to deal with - the distributed exploration vs. exploitation dilemma. The identification of this issue is in itself an original contribution. However, the

main original contribution of this thesis is to address this problem, proposing and testing a method for the team owner to set the balance between exploration and exploitation.

As far as I am aware this problem has not been addressed in past MRS work. The closest problem that has been studied previously is the Coverage/Interference trade-off in foraging tasks. By increasing the number of robots the coverage of the environment is increased, thus decreasing the time taken to find items to be foraged. However, studies have shown that increasing the number of robots too far can lead to too much interference between the robots, thus slowing down the foraging process[Arkin 1992][Beckers et al 1995]. Another related piece of work is [Mataric 1994] which touches on the *single* robot exploration vs. exploitation dilemma in a multi-robot sweeping-based task.

1.5 Thesis Overview

In this chapter the evolution of robotics towards multi-robot systems and the reasons to study such systems have been discussed. The thesis has focused on mobile MRS and listed 5 basic tasks for a mobile MRS, picking out the point-to-point map sharing task as a preferred area of study. The advantages of learning in such point-to-point tasks have been discussed, as has the importance of on-line learning in real world dynamic environments. The key contribution of the thesis was then introduced: a system for setting the trade-off between learning and task-achievement in learning MRS.

This contribution is significant because learning, and particularly on-line learning, provides great advantages for MRS, and therefore the setting of the trade-offs in such learning is important. It was shown to be original because such trade-offs have never been discussed in the context of multi-robot systems - only the Coverage/Interference trade-off and single-robot exploration vs. exploitation problem has been addressed.

The next chapter (chapter 2 - A Review of Multi-Robot Systems) will look at the major properties of mobile multi-robot systems and classify systems into sub-areas. It will go on to look at some examples of mobile MRS, and related work in Multi-Agent Systems.

Chapter 3 (Method) analyses the problem of the distributed exploration vs. exploitation dilemma and proposes a solution for setting the balance between distributed exploration and exploitation. It then introduces a simple robot model which this solution will be implemented on to be tested.

Chapter 4 (Results and Discussions) details the experiments used to test the proposed solution, presents the results, and discusses the degree of success the solution shows.

Chapter 5 (Conclusions and Future Work) reviews the thesis, and discusses possibilities for future work.

Chapter 2

A Review of Mobile Multi-robot Systems

This chapter consists of a review of mobile multi-robot systems and a brief review of related learning multi-agent systems (MAS). The review of MRS will first present an outline of Multi-Robot Systems, from simple "collective" systems to full communicating environment-learning peer-evaluating systems. It will then look at examples of these systems and how they relate to the system studied in this thesis. The review of Multi-Agent Systems will concentrate on two systems involving peer-evaluation.

Multi-Robot Systems can be divided up into 2 main areas: Collective Robot Systems, and Model-based Systems. First Collective Systems will be examined.

2.1 Collective Robot Systems

The inspiration for "Collective" robot systems comes from two areas: behaviour-based robotics and differentiative societies. The first of these areas deals with the design of simple but effective robots, and the second with the fact that many simple agents can perform a complex group task.

2.1.1 Behaviour-based Robotics

In the Behaviour-based paradigm[Brooks 1990] a robot has a group of simple reactive behaviours. Each behaviour is triggered by a simple sense stimulus, and there is a hierarchy of behaviours - certain behaviours over-ride others (for this reason the architecture is called the *Subsumption* Architecture). One example is Brooks' robots "Tom and Jerry" which have three behaviours: Wandering, Following and Obstacle avoidance.

When no other behaviour is active, the robots wander around randomly. If an obstacle is encountered, the obstacle avoidance behaviour over-rides the wandering, and the robot avoids the obstacle. Then the wandering behaviour takes over again. If a moving object is detected, the robot follows the object, with the obstacle avoiding behaviour taking over as necessary. Such behaviour-based systems have exhibited limited intelligence, even though they have no model of the environment. In fact, it was a reaction against the use of explicit environment-models that caused the popularity of behaviour-based systems.

2.1.2 The Social Insect Model

The second inspiration for Collective robotics comes from the Differentiative Societies of nature[Parker 1994]. Ant, Termite and some Bee and Wasp societies are Differentiative[McFarland 1993]. In these societies an individual insect has the best chance of propagating its own genes by ensuring the survival of particular relatives (e.g. a queen). Thus all insects work together for the good of the colony: "the individual exists for the good of society, and is totally dependent on that society for its existence."[Parker 1994] This 'working together' is based on very simple, genetically hard-wired, local interactions. When many insects interact - even in this simple way - structured and "intelligent" behaviour can emerge. Examples are termite nest building, ant foraging and body collection[Beckers et al 1995], and bees searching out a food source[Kelly 1995].

It will now be seen how the social-insect model and behaviour-based robotics are combined to inspire Collective Multi-Robot Systems, or Collective Robotics.

2.1.3 The Aim of Collective Robotics

The aim of Collective robotics is to develop complex and useful behaviour in groups of behaviour-based and reactive robots, taking the stigmergy of differentiative insects as an existence proof that complex behaviour can emerge from such simple robot systems[Beckers et al 1995]. Such systems have the advantage that a number of behaviour-based robots can be built for the price of a single more complex intelligent robot[Brooks and Flynn 1989]. They are also directly testable using current robot hardware technology[Parker 1994].

2.1.4 Collective Robotics and Communication

The simplest forms of Collective robot systems do not have communication between the robots. A more relevant (and general) concept in Collective robotics than communication is *interaction*. Robot A inter/acts with Robot B if Robot A's actions change (directly or indirectly) the results of Robot B's actions. Positive/helpful interaction is obviously fundamental to useful MRS - though many Collective robot systems are concerned with the minimising of negative interaction (also called "interference"), for example avoiding collisions and "bottle necks".

Communication is a special case of interaction between a sender and a receiver, where the sender passes on information intentionally or by design. Without communication, robot B can interact with robot A in two ways:

1. Directly: through Robot B's actions being affected by its awareness of robot A's actions (e.g. A's position or current action). [Parker 1994]

2. Indirectly: through Robot B being affected by the effect of robot A's actions *on the environment* [Beckers et al 1995] (this includes Robot B being directly physically acted upon by Robot A).

The first of these is quite a complex problem, as without communication such deductions would require accurate sensory recognition systems. So the more common approach in non-communicating collective systems is approach 2.

Non-communicating collective systems include [Goldberg and Mataric 1997][Werger et al 1996][Yamaguchi et al 1996][Fontan et al 1996][Beckers and Mataric 1995][Arkin 1992][Sugihara and Suzuki 1990]. Communicating collective systems include [Ünsal and Bay 1994][Zhi-Dong et al 1994][Arkin et al 1993].

Having examined Collective Robot Systems, the second class of multi-robot systems will now be introduced: Model-based Systems.

2.2 Robot Groups with Environment/Peer Models

The approach of behaviour-based robotics is to not allow the robots explicit environment models as "the environment is its own best model"[Brooks 1990]. A robot's "understanding" of the environment is hardwired by the designer through reactive rules. In Collective Robotics this translates into "the environment, peer and non-peer, is its own best model". So Collective robots, as well as having no explicit model of their environment, have no explicit peer-model, i.e. no model of the *other robots* in their "collective".

The inspiration for explicit-model based systems is that they allow the robot more "control" over its own behaviour, leading to more flexibility. Rather than depending on a limited number of reactive behaviours, robots can select (and develop) behaviours based on more complex internal models. A significant proportion of model-based multi-robot

systems also involve some sort of peer-model to aid in co-ordinating planning and behaviour within the robot group.

2.2.1 Integrative Societies and Robot Groups with Peer Models

Peer-models are the basis of co-operation in the "Integrative" societies of nature[Dautenhahn 1995]. Integrative societies are the opposite of the Differential societies discussed at the beginning of the collective robotics section[Parker 1994]. Examples of Integrative societies are humans, wolves, and the breeding colonies of many species of birds[Parker 1994]. Unlike animals in Differentiative societies, animals in Integrative societies have the best chance of passing on their genes through their own survival[McFarland 1993]. Thus their motivations are "selfish": the group exists for the good of the individual (rather than the individual for the good of the group, as in ant societies). Therefore the basis of direct group interaction is *Co-operation*: accepting a short-term loss for the sake of another group member, but only because you are expecting a resultant long-term gain for yourself. Such co-operation requires the ability to interact intelligently, that is it requires *social* intelligence. This social intelligence includes the ability to remember how successful past co-operation has been with different members of the group: in other words peer-modelling is vital in Integrative societies. Peer-modelling is not encountered in Collective Robotics, but is a significant aspect of model-based multi-robot systems.

Despite this observation, Integrative societies cannot be described as the explicit inspiration of model-based multi-robotics, since not all model-based systems have peer-models, and those that do have peer-models do not necessarily interact in a truly *co-operative* way by the precise definition of the word[McFarland 1993].

2.2.2 Integrative Societies and Co-operation

It is important to note that the word "co-operation" is used imprecisely in most multi-robot research. This is probably due to the relatively new nature of the field, and its lack of conventions. Many use it when in fact they merely mean "positive interaction" (for example the title "*Co-operation* without Communication"[Arkin 1992]). This is particularly ironic in stigmergically-inspired/collective systems, where it would imply that these artificial-differentiative societies are actually "co-operating". True co-operative behaviour is put into a robotic context in [McFarland 1993] by setting up a "closed robot ecology". At some point autonomous robots will be sent out to accomplish tasks in the real world, and therefore their ecological situation will be similar to our ecological situation. Hence it is argued in [McFarland 1993] that to study robot co-operation in the laboratory, a closed robot ecology must be produced.

The "closed robot ecology" used in [McFarland 1993] is a system where robots which do not accept a short-term loss will die from lack of energy. The environment used for the "robot ethology" studies contains lights, switches that temporarily dim the lights, and recharging points for the behaviour-based robots. The lights actually drain power from the recharging points. Hence robots must continually dim the lights to ensure they can recharge. The system is set up so that one robot could not possibly accomplish the task. Thus the robots must interact positively, helping each other get to, and allowing each other access to, recharging points. When a robot wants access to a recharging point it will send a signal to the robot already on the charging point. The key aspect of this system is that the robots *must* co-operate to survive. This form of interaction fulfils the ethological definition of co-operation: "helping another and experiencing a short-term loss, but eventually achieving a long term gain as a result". The robots experience a short-term loss by moving

off the recharger for other robots. However, they have the long-term gain of the other robots helping them to "farm" the lights.

This thesis will stick to the more helpfully precise convention of co-operation being "short-term loss followed by long-term gain", unless quoting other work.

2.2.4 Categorising Model-Based MRS

In the same way that Collective systems can be categorised into the communicating and non-communicating, so model-based systems can be categorised according to their approach to *Co-ordination* and *Centralisation*.

2.2.5 MRS Co-ordination

Looking first at multi-robot co-ordination, two types can be discerned from past work:

1. The co-ordination of the sharing of the environment resource (e.g. movement space and tools)
2. The co-ordination of the robots' personal resources (e.g. learned knowledge and effectors)

The first of these has been particularly prevalent in studies of mobile model-based MRS. (In fact, even the negative-interference reducing systems of *collective* robotics, introduced later, can be seen as a way of co-ordinating the sharing of the environment resource.) It is also worth noting that there has been some work on more general environmental resource sharing[Wang 1993] (i.e. not necessarily "space", but actual *items* that are part of the environment, like tools[Premvuti and Yuta 1996]). However the sharing of "space" has

been much more widely studied as it is "the basic resource of a mobile robot."[Premvuti and Yuta 1996].

2.2.6 MRS Centralisation

Another division in model-based mobile robot work is that between multi-robot groups with some sort of central-station or "administrating agent" and those without. There is a lot of debate as to the advantages and disadvantages of central agents. The advantages include:

- Global view point for planning of interaction between robots (e.g. avoiding negative interaction[Alami et al 1997], assigning task to robot nearest to task site[Caloud et al 1990]).
- Reduction of individual robot complexity [Leuth 1994]
- Simplification of communication network [Zhi-Dong et al 1994]
- Easier synchronisation [Li 1994]

However such a system tends to be dependent on a single agent, i.e. the central station. This leads to a number of disadvantages of central-stations:

- Lack of robustness, if station fails, the system may go down [Zhi-Dong et al 1994]
- Bottle-neck at the station: it requires sufficient processing-power for multiple robots[Zhi-Dong et al 1994]
- Dependence on the station - robots must stay in contact with it and within its range
- Robots can only join the team if they can communicate with the controller

A large number of researchers have considered these disadvantages to be so compelling that there is actually a field of study called "Distributed Autonomous Robot Systems"(DARS) [Asama et al 1996][Asama et al 1994] which concentrates on decentralised-systems. The multi-robot system presented in this thesis is also a DARS system.

2.2.7 A Categorisation of Model-based MRS by Co-ordination and Centralisation

Table 1.0 overleaf is a table which positions model-based work according to centralisation and co-ordination.

2.3 Research Review in relation to the Thesis

The above partitioning of mobile MRS into Collective and Model-Based systems is inspired by [McFarland 1993] (which refers to Collective and Co-operative systems) and [Parker 1994] (which partitions into "Swarm" and "Intentional" systems). However, the further partitions of communicating/non-communicating and co-ordination/centralisation are a contribution of the thesis. The only previous piece of work which has produced such a fine partition is [Cao et al 1997] but their partition is based on more abstract future research directions (or "axes") whereas the one in this thesis is a taxonomy of existing work.

In Chapter 1 the concept of the distributed map-sharing mobile multi-robot system was introduced. Such a system would need five basic abilities:

1. *Random sweeping* - to explore the environment
2. *Point-to-point motion* - to achieve particular goals in the environment
3. *Environment learning* - to learn the environment as it is explored

4. *Peer-models* - to co-ordinate the transmission of map information

5. *Communication* - to transmit map information

	Centralised	**Decentralised**
Environment-resource Co-ordination	[Alami et al 1997] etc. [Aguilar et al 1995] [Li 1994] [Buckley 1989] [Erdmann and Lozano-Perez 1987]	[Wang and Premvuti 1994] [Noborio and Yoshioka 1994] [Ota et al 1994] [Ueyama et al 1994] [McFarland 1993] [Wang 1993] [Premvuti and Yuta 1990]
Robot-resource Co-ordination	[Caloud et al 1990] [Herman and Albus 1988] [Brummit and Stentz 1996] [Nof 1991] [Elgimez and Kim 1990] [Asama et al 1991] [López de Màntaras et al 1997]	[Stone and Veloso 1997] [Mataric 1997] etc. [Demiris and Hayes 1996][Hayes and Demiris 1994] [Parker 1996] etc. [Dautenhahn 1995] [Asada et al 1995] [Suzuki et al 1994] [Ohko et al 1993]

Table 1.0

The thesis will now look at the past examples of work in multi-robot systems and divide them up into these 5 areas of ability.

A piece of past work will be positioned depending upon which of these 5 areas it concentrates: sweeping, point-to-point motion, environment learning, peer models, or communication (to be found in Sections 2.4, 2.5, 2.6, 2.7 and 2.8 respectively). Obviously some MRS will be relevant to more than one area, but they will categorised according to their main thrust or contribution.

2.4 Multi-robot Sweeping

The first of the 5 areas to be examined is Multi-robot Sweeping. In a multi-robot map learning system robots learn from experience by performing a group random sweeping task. However, much of the studies in sweeping behaviour are to be found in collective robotics. Examples of sweeping task systems in collective robotics include [Ichikawa and Hara 1994] where simulated robots are provided with wandering and obstacle/robot avoiding behaviours, and put into an enclosed space. The robot-avoiding behaviour leads to interaction amongst the robots which forces them to distribute themselves around the space and not collide. Robot A affects Robot B's environment by *being part of the environment in a particular position*, and forcing Robot B into obstacle avoidance. The ability of the robot team to cover the whole of the space in a search is measured using the "Space Coverage Ratio" (SCR). The space is divided up into small sections, and the SCR at a certain time is defined as the ratio between the number of element sections passed over by the robots and the total number of sections in the space. Experiments showed that the SCR increased as the number of the members of the robot team increased up to 10. This thesis is not really concerned with the efficiency of sweeping *per se*, more the efficient use of the

results of sweeping abilities - i.e. of robots maps. So robot avoidance behaviours are not implemented, and the robots simulated in this thesis can "move through" each other.

More complex forms of coverage, involving robot groups forming into patterns, have been studied by [Sugihara and Suzuki 1990], [Yamaguchi and Beni 1996] and [Ünsal and Bay 1994]. The robots have behaviours which enable them to form patterns such as circles, polygons, parabolae and sinusoidals. The fundamental contribution of the three systems is their use of distributed algorithms to form global patterns. They have no central-station monitoring the production of the pattern and controlling individual robots to produce the final desired pattern. Similarly, the system in this thesis uses a distributed algorithm. Such measures will be seen to be taken *locally* in an attempt to cause a *global* improvement.

The lack of a central-station is very common in collective robot systems as the major central-station advantage is its global view, and collective systems usually use (like insects) *local* behaviour control. (A good discussion on the advantages and disadvantages of local and global control can be found in [Parker 1992]).

2.4.1 Multi-Robot Sweeping with Foraging

The robots in this thesis actually utilise multi-robot coverage to find specific goals and salient points. There is a large area of study in collective robots which looks at a related idea: the "foraging" task. In a foraging task, robots move out from a home position and sweep the space, attempting to find goals ("food"). These are actually moveable objects (usually pucks) which the robot must then return to its home position.

[Arkin 1992] demonstrates that simulated robots can interact in foraging without inter-robot communication. He uses the concept of "Recruitment" from studies in ant behaviour; recruitment being "communication that brings nestmates to some point in space

where work is required." Arkin claims that recruitment-like behaviour is possible in the absence of direct communication. He demonstrates this using simulated foraging. The "food" is actually an attractor for the robots (i.e. as they move closer to the "food", their motion will be more forced in the direction of the "food".) When a robot reaches the attractor, it picks it up and carries it back home. However, the carried food remains an attractor to other robots. So these other robots are attracted to the carrying robot, and they are recruited into helping the carrying robot to transport the "food" more quickly. This system is also demonstrated to improve foraging time with *multiple* distributed "food" items. The positive interaction in these experiments is through many robots increasing the sweep coverage, and through robots changing the environment by affecting the position of the attractor (by carry it) thus causing other robots to move towards it and help carry it.

2.4.2 Interference in Multi-Robot Foraging

The foraging task is further studied in [Fontan and Mataric 1996]. As was mentioned earlier, interaction is not always a good thing. The skill in collective robotics is to design behaviours to maximise positive interaction. Negative interaction, or *interference*, could cause robots to slow down a foraging task by always being in the way of each other (as in [Beckers et al 1995] or [Arkin and Hobbs 1992] where too many robots is shown to cause over-interaction). [Fontan and Mataric 1996] attempts to reduce this interference by an ethologically-inspired "Territoriality": having different robots forage different parts of the space, thus keeping them out of each others way. In this physically-implemented experiment, robots collect pucks and deposit them on the edge of the territory nearest to "home".

2.4.3 Interference Reduction through Arbitration

[Goldberg and Mataric 1997] introduce two other ethologically-inspired methods on top of territoriality for reducing negative interaction: pack arbitration and caste arbitration. Pack arbitration involves robots "giving way" to each other in potentially negative interaction situations. Which robot gives way is decided by some form of "dominance hierarchy", for example the robot with the highest serial number.

Caste arbitration involves dividing up the tasks between robots in such a way that they do not interfere with each others tasks. [Goldberg and Mataric 1997] perform a physically-implemented foraging experiment and find the main cause of negative interaction is robots getting in each others way when trying to return pucks home. They find that the most efficient form of arbitration in this case is caste arbitration. One robot is set the task of remaining near home, and the other robots collect pucks and bring them back to this robot who returns them to home.

2.4.4 Foraging Without a Home Location

A slightly different foraging task (which doesn't have a set "home" location) is solved in a very efficient way by [Beckers et al 1995]. They physically-implement a non-communicating collective system. It consists of a number of robots with IR sensors for obstacle avoidance, and a scoop which they use to push pucks along in front of them. The grippers have a sensor which tells them when they have caught 3 pucks. The robots are given three basic behaviours, with only one active at any time (so they are reactive robots, rather than subsumption architecture robots).

When no sensor is active, the robot moves in a straight line (default behaviour) until an obstacle is detected, or three or more pucks have been picked up. If an obstacle is

detected, the robot activates obstacle avoidance behaviour (pucks are not obstacles). If 3 pucks are detected, the robot releases the pucks from its scoop (puck-dropping behaviour). At the beginning of the experiment pucks are distributed randomly about the space. The fascinating result of these simple behaviours is that the robots collect all the pucks up into one neat pile. The optimum number of robots was found to be three. The robots in [Beckers et al 1995] interact with each other mainly by affecting the state of the environment by moving the pucks around. The authors make the interesting observation that Stigmergy is best regarded as an exploitation of the environment as an external memory resource.

2.4.5 Foraging using Robot "Food Chains"

[Werger and Mataric 1996] have produced another physically implementation of the environment exploitation idea by producing Robot "Food" Chains for foraging tasks. [Arkin 1992] assumes that the robot can find its "food", and can find the home it must return the "food" to. In reality, these are complex recognition tasks. [Werger and Mataric 1996] implement foraging by having a group of robots organise themselves into a chain stretching out from the home position. This organisation is produced using basic behaviours. Then foraging robots use the chain to navigate out from home, making circular foraging trips away from the chain to collect "food". They can then use the chain to navigate back to home and drop the "food". There is no communication involved here, the robots move out along the chain by touch, and the chain maintains itself by touch and beam-breaking.

2.4.6 Foraging Conclusion

It can be seen that a large number of studies into multi-robot sweeping have been in collective robotics. In fact the vast of majority of such studies have involved collective

robotics. The behaviour does not seem to have been explicitly studied in model-based multi-robot systems. The main concern in model-based MRS has been in point-to-point motion. This is the second of the five key areas mentioned in Section 2.4, and will now be addressed.

2.5 Multi-robot Point-to-Point Motion

The robots studied in this thesis randomly sweep the space so they can learn point-to-point routes. The actual error-free following of these point-to-point routes, although non-trivial in reality, is taken as given. The important aspect for this thesis is the exchanging of map information between robots, an aspect not studied in detail in model-based, or collective, MRS.

The robots studied in this thesis have "route planning" that is limited to finding a route from their (or another robot's) memory which leads to their goal, and following that route. If these robots encounter any other robots while following a route it is *assumed* that there is some sort of algorithm for avoiding collisions while keeping true to the route. The effect of this is simulated in this thesis by having the robots "pass through" each other.

However, if this system was implemented in hardware robots, more sophisticated route planning may be needed for point-to-point motion. There are two main problems to be addressed in multi-robot point-to-point motion:

1. How do robots navigate around a space containing dynamic obstacles - i.e. other robots?
2. How do robots deal with the situation where they discover midway that a given route is now blocked, or a goal has moved?

The interference-reducing work of collective robotics can be seen as a simple form of study into part of the first problem, i.e. collision avoidance. But full studies of the first, and second problem, have only been undertaken in model-based MRS. (This is for the simple reason that planning requires a model for prediction.) In fact the majority of model-based studies address the first problem of multi-robot point-to-point motion, since this problem is unique to MRS. However later on an example is given of a system addressing the second problem. First of all, collision avoidance with a central planner will be examined.

2.5.1 Collision Avoidance with a Central Planner

[Erdmann and Lozano-Perez 1987] examine a central-station based motion-planning problem for multiple robots in spaces with obstacles, by simulating the robots as moving 2D polygons. The planning method uses the most basic method of centralised multi-robot movement planning: robots are assigned priority, and motion-plans are assigned to robots in priority order (the system assumes the priorities are given). So if robot A has priority over robot B then Robot A's plan is made first, and then robot B's plan is made treating robot A as a moving obstacle.

[Buckley 1989] introduces a system of setting prioritizations in spaces sparsely occupied by obstacles, i.e. where the main obstacles are other robots. This prioritisation system attempts to maximise the number of robots which can move in a straight line from their start point to their goal by adjusting their speeds, thereby minimising the number of robots for whom time-consuming collision-avoidance planning is necessary. The default is Robot A has priority over Robot B if Robot A was following a plan first. However, this may not work if Robot A's line goes through the start point of robot B's plan just before robot B is due to leave there. In this case robot B should be given priority (as it is about to

get out of robot A's way anyway). Similarly if Robot A's line goes through the *goal* point of robot B when robot B is due to be there, robot A must have priority (so B doesn't jam up the route by sitting at its goal). This enforcing of straight line motion reduces the planning time, though not necessarily motion time.

2.5.2 Reducing Central Planner Dependence

[Alami et al 1997][Alami et al 1997b][Aguilar et al 1995] all discuss a mobile robot space-sharing system called the "Plan-Merging Paradigm". This paradigm is interesting as it results in behaviour similar to that in a centralised-planning system, but tries to do the planning in a distributed way (though is still needs a central station for "emergencies"). The robots are moving in an environment of corridors, intersections and rooms. Having individually made their own plans, robots use the plan-merging paradigm: they compare movement plans and refine collisions out of them. This paradigm is therefore exhibiting global control without the robots using the central station. However, the plan-merging paradigm will not always be successful, and when this happens the central station has to step in and do the planning itself.

Another system which has a central station but tries to avoid using it is [Li 1994]. In this system robots plan their routes and define "safety zones". These indicate parts of the route leading to possible conflict (e.g. the part of the route that goes through an intersection). Robots can then broadcast safety zones and, if necessary, negotiate to avoid conflict using fixed rules. If they cannot resolve the conflict then a central station is called in to deal with the problem. The central station actually has another job: that of task allocation, but Li does not go into that aspect in any detail.

2.5.3 Blackboard Systems

[Wang 1993] is the system which takes another step away from a central-station dependence. He introduces a system which although not centrally-dependent still uses a form of *Blackboard*. The Blackboard is a concept from Distributed Artificial Intelligence[Bond and Gasser 1988]: it is a single virtual board where agents can "pin up" messages which can be read by *all* other agents.

Wang's system introduces a twist by giving each robot a "sign-board" which can only be read by a subset of agents. Each robot can only post to its own sign-board, and this sign-board can only be read by robots *within a certain distance* of the posting robot. Wang's robots are assigned priorities for environmental resources according to their serial numbers and when they last used the resource. Robots who have used the resource recently are given lower priority than those who haven't. Algorithms are given for dealing with a robot "road-way" intersection, and for dealing with corridors of length M robots which are only wide enough for 1 robot. There is also an algorithm for sharing a more general "Resource of Capacity M", i.e. a resource which only M robots can use at the same time. (This is an example of the "non-space" environment resource sharing mentioned at the beginning of this chapter.)

2.5.4 Deadlock in Point-to-Point Motion

An important point about the algorithms in [Wang 1993] above is they are guaranteed to be "deadlock" free. Deadlock is an important concept in multi-robot systems, though not one that this thesis concentrates on. It is particularly important in distributed systems, which often have no global view of the environment or of other robots' positions in the environment (i.e. robots use only "local" information). Deadlock occurs when one or more robots get stuck in a state or a loop of states. An example may be two robots

following fixed paths. If their paths cross and they cannot get out of each others way, they will go into deadlock: both continuously trying to take the next move in their route but unable to. A similar problem could occur at an intersection where the algorithm is "move on if you are the first robot to arrive". Such an algorithm will enter deadlock if two robots arrive at the intersection simultaneously.

[Wang 1993] is extended in [Wang and Premvuti 1994] to a full traffic regulation and control system for mobile robots, enabling them to share the environment resource in a more complex environment. The environment can consist of corridors, rooms and intersections. Algorithms are presented for local co-ordination, and deadlock detection and resolution.

[Noborio and Yoshioka 1994] introduce a system for decentralised path-planning in an environment where the *only* obstacles are other robots. The robot plans a straight line to its goal. If two robots meet on the way, they can avoid collision using an algorithm based on circular movements around each other. The robots infer each others behaviour, and choose their own behaviour as a result. This may be circling, or moving backwards or forwards. The behaviour is decided by a selection table, which has been carefully designed to avoid deadlock. This deadlock free characteristic is proved.

[Premvuti and Yuta 1990] introduce a system called "modest co-operation" which they implement on robots approaching a "road" intersection. The idea here is that the robots enter "modest co-operation" mode when they get close enough to the intersection. They slow down, run close to the left side of the road and exchange a high volume of information with other robots around the intersection. They can then plan their routes through the intersection without collision.

2.5.5 Task Assignment for Point-to-Point Motion

A more advanced point-to-point system is the GOPHER system[Caloud et al 1990]. The robots in GOPHER each have a fixed global model (a two-way "road map") of the environment and are set tasks by a Central Task Planning and Scheduling System (CTSP). The CTSP has a global view of the tasks to be performed in the environment and the availability of robots to perform these tasks. (So in a sense the CTSP has simple "models" of the robots, though the robots themselves do not.) The CTSP assigns tasks in a way similar to the Contract Net Protocol[Bond and Gasser 1988] (CNP) from Distributed Artificial Intelligence. The robots bid to do available tasks, and the CTSP chooses the best bid according to its information and the information provided in the bid (e.g. proposed plan and estimated execution time). It is not strictly a Contract Net Protocol system, as the Contract Net Protocol is distributed and needs no central station - different robots take on the roles of bidders and "auctioneers" at different times. An augmented version of the proper CNP will be reviewed in the section on multi-robot peer-modelling.

The GOPHER system is the archetypal central station system. If the GOPHER central agent is removed, the whole system will cease to function. On the other hand, with a distributed (DARS) system, like the one in this thesis, any robot can be removed and the whole system will continue to function. Another fundamental difference between the GOPHER system and the system in this thesis is that the GOPHER system is primarily concerned with the Task Assignment Problem, an issue studied much in traditional Distributed AI. This thesis is not concerned with the task assignment problem, but rather the task *learning* problem.

2.5.6 Point-to-Point Motion Re-planning

All the route-planning systems described so far are designed to deal with a dynamic peer-environment (i.e. moving robots) but a fixed non-peer environment (the non-peer environment being non-robotic obstacles). They are limited in their adaptability as they have no way of *re-planning* when the non-peer environment changes in the middle of a task.

[Herman and Albus 1988] sketch out a proposed replanner to solve this problem for MRS, but do not give any details or demonstration. [Brummit and Stentz 1996], however, actually demonstrate a simulated system for dynamic mission planning. If two robots have goals in an environment with obstacles, a central planner can plan the shortest non-colliding routes for these robots. However, if there is an environment change while executing the plan, the plan may no longer be the quickest route. The central station has a global view of the environment, and the [Brummit and Stentz 1996] dynamic mission planner allows robots to feedback any environment changes while following a plan. The planner can then re-plan the robots paths from their current position to optimise the overall mission time.

This system learns about changes in the environment and adapts its plans accordingly, but it is not strictly a environment modelling system. An environment modelling system not only adapts itself to changes in the environment, but also learns important aspects of its initial environment itself. Such systems are the concern of this thesis. In MRS there are few other such systems, but examples will now be given as the third of the five key abilities is now examined: Multi-robot Environment Learning.

2.6 Multi-robot Environment Learning

The work in this area which is of greatest relevance to this thesis is [López de Màntaras et al 1997]. [López de Màntaras et al 1997] involves physically implementing a model-based MRS with a central-station where robots separately produce partial co-ordinate maps of the parts of the space they sweep (like the system in this thesis, they use a random sweep). The robots then return to the central station and upload their maps. The maps are then combined using fuzzy logic to produce a single central map. The idea is that individual robot map errors will be ironed out in the combination. Robots also upload information to *each other* when they meet. This means that if a robot cannot make it back to the central-station, its learned data may have been passed on to another robot it bumped into that does make it back.

In the system studied in this thesis, data is never centrally combined - it stays in a distributed form. The main advantage of central combination is the removal of errors and uncertainty due to robot motor/sensory inaccuracies, an area which this thesis is not studying. However, the main difference between the [López de Màntaras et al 1997] system and the system studied in this thesis is that [López de Màntaras et al 1997] is concerned with mapping for the sake of mapping - i.e. without any point-to-point task achievement. Therefore it uses off-line learning, and hence the distributed exploration vs. exploitation dilemma is not a relevant issue for it.

2.6.1 Semi-Distributed Models

A further environment learning system which uses a form of topographical map, but that has a *semi*-distributed map, is [Herman and Albus 1988]. It proposes a group of Multiple Autonomous Underwater Vehicles (MAUV). The MAUV robots are based on a hierarchical system: at the *mission* level the task for the whole robot group is converted

into commands for a number of robot sub-groups. At the group level, these commands are then converted into tasks for individual robots. At the robot level, the tasks are converted into servo-commands etc.

All robots in a group have environment models, and in small groups robots can help each other update their environment models, i.e. exchange map information. But in larger groups, to lower communication problems, there will be a group leader who has an environment model and does all the planning for other robots. Since all robots have some sort of (possibly inaccurate) environment model, inability to communicate with the central planning robot (due to destruction, or stealth) is not as important. Thus, although the MAUV system essentially has a central agent with environment model, it is more robust to the removal of that central agent and model. However, the MAUV system does not explicitly detail its algorithms for implementing multi-robot environment modelling, nor does it examine any issues relating to information exchange in learning robots.

2.6.2 The ACTRESS System

Another system which involves multi-robot environment learning but does not give details is ACTRESS[Suzuki et al 1994][Asama et al 1992][Asama et al 1991]. It is possibly the most ambitious multi-robot project to date, though quite how successfully it has been *implemented* is not known to this author.

The ACTRESS system is made up of multiple robotic agents called *robotors*. Each robotor can make decisions for achieving a given goal, execute tasks, and communicate with other robotors. The emphasis here is on heterogeneous agents, i.e. some robotors are mobile robots, some are cranes, some are computers, etc. (The model used to test the idea in this thesis uses homogeneous robots, but should work with a heterogeneous team given the appropriate common communication protocol.)

[Asama et al 1991] discusses the use of an environment manager in the ACTRESS system. Such a manager does not *control* robotors, but stores information about the environment which robotors send it during their tasks. Thus other robotors can use this information, so the robotors are using the central station as a medium for map sharing. This is similar to [López de Màntaras et al 1997] but [Asama et al 1991] does not detail how the maps are merged or transmitted.

Other studies of ACTRESS include [Suzuki et al 1994] which concentrates on distributed task-assignment algorithms. A robot can send a request for physical collaboration in a task. It can also send a request to other robots for environmental information. Thus it is a map sharing system. However, no learning is involved, and the precise nature of the information transfer protocol is not discussed. The system is demonstrated on multiple box pushing, where robots can request help with pushing, and tell the helping robot how to get to the box to be pushed.

2.6.3 Game Theory for Environment Learning

Game theory is an area of mathematics much concerned with multi-agent interaction. A general study of possible applications of game theory to Multi-Robot Systems can be found in [Nof 1991] (and a study of the relationship between the Prisoner's Dilemma Game and MRS will be presented in the section on multi-agent systems.) However [Nof 1991] does not supply any concrete approaches or tests for its ideas.

[Elgimez and Kim 1990] looks at a more specific area of game theory and attempts to apply it to a specific problem by using environment-modelling. The [Elgimez and Kim 1990] system is based in an environment of rooms and wide corridors, divided into two halves connected by a lift. A game theoretic concept called the "Shapley Value solution to an N-person game" is extended and used to develop deployment rules for the robots. The

robots are deployed to go and "deal with" events occurring in different parts of the environment. The environment is divided up into areas, and the areas "play a game" amongst each other to find which provides the most coverage for events; i.e. which area provides robots with a starting point that allows them to deal with as many events as possible, in as little time as possible. The areas are then ranked and deployment is decided using this ranking.

The environment learning here is not in terms of learning the way through the corridors, rooms and doors, but learning the relationship between the environment and the task assignment problem. The system learns which part of the environments should be assigned to which robots. A closely related process is learning which robots to assign to which parts of the environment. To do this, a system needs models of robots as well as of the environment. Specifically, if a robot is assigning tasks to other robots in its team, then it needs models of its peers to make well-informed assignment decisions. This is the fourth of the five key abilities given earlier in Section 2.3: Multi-Robot Peer-Models.

2.7 Multi-robot Peer-models

Peer-modelling will be used in the robots in this thesis to allow a robot team owner to set the balance between distributed exploration and exploitation. The precise details of how this is done will be discussed in the next chapter. In the meantime it suffices to say that the robots have very simple peer-models, consisting only of a single number representing each of their peers.

2.7.1 Combining Peer and Environment Models

An even simpler form of peer-modelling involves learning about the environment and peer-environment with no ability to distinguish between the two. This is the approach

used in [Ota et al 1994] which introduces a strategy-making method that they apply to mobile-robot path selection. The environment consists of two areas, A and B, with 2 straight paths between the areas, 1 and 2. Fifty robots have to go from A to B and back again using the paths. They try to learn the best non-colliding strategy which allows them to do this. The robots' strategies consist of a number of *tactics* together with a probability that a tactic will be selected. The idea is that the robot tries out different tactics to move between A and B, and rewards the tactic according to how short the route it produces is. This "rewarding" involves increasing the probability that this tactic will be selected. (Such a learning system is called "reinforcement" learning[Kaebling and Moore 1996]). So the robots are developing models of which actions work best in the two path environment, given the behaviour of the rest of the group.

2.7.2 Learning a Peer-model to Avoid Interference

The system used in [Mataric 1997][Mataric 1996][Mataric 1994] also implements reinforcement learning to allow a team of robots to develop a foraging task while avoiding negative interaction with each other. The peer learning here is implicit in that robots have to learn about each others' behaviour in trying to minimise negative interaction. Mataric gives the robots a repertoire of simply implementable basic behaviours: safe-wandering (robots wander avoiding collisions), following, aggregation (gathers the robots), dispersion (spreads out the robots) and homing. Mataric demonstrates that these "atomic" behaviours can be combined to produce more complex behaviours, such as flocking, foraging and docking (getting a group of robots to park along a boundary). The learning robots jointly model their foraging task, their group dynamics and their general environment by choosing different weightings of these behaviours.

These papers have two similarities with the system studied in this thesis. Firstly, the robots are designed for on-line learning so they can go on continuously improving their performance. Secondly, the robots implement a system that addresses part of the exploration vs. exploitation problem: the *single*-robot exploration vs. exploitation problem (the papers do not need to address the distributed exploration vs. exploitation problem since the robots do not directly aid each other). The robots use a system of *progress estimators*: "if a behaviour fails to make progress relative to the current goal, it is terminated and another one is tried".

2.7.3 Separating Peer and Environment Models

The next step in artificial peer-modelling is for robots to be able to distinguish between their peers and with the rest of the environment, so they can learn separate behaviours for dealing with them. [Ueyama et al 1994] is such a system for collision avoidance amongst multiple robots moving through a space of obstacles towards their goals. The robots model their peers by developing "behaviour priorities". The robots have 3 behaviours: "go to goal", "move away from obstacle", and "if robot ahead turn right" (they can tell the difference between robots and obstacles). The priorities of the behaviours is learned by a robot according to how successfully it gets to the goal without colliding. When two robots are placed in a space - each with a goal - and with single obstacle which blocks both their paths, it is found that before the robots have learned the correct behaviour priorities, they may be forced into each other (if the "turn right" behaviour is too low priority) or into an obstacle (if the "move away from obstacle" behaviour has too low priority). But after a few learning trials, they succeed in non-colliding navigation.

2.7.4 Peer-Models and Robustness

A simple, but very practical, approach to peer-modelling is taken in the ALLIANCE system [Parker 1996][Parker 1996b][Parker 1995][Parker 1994][Parker 1994b]. Robots in the system are endowed with "impatience": if Robot A does not get a task done quickly enough, then Robot B's impatience will increase to the point where it will take over Robot A's task. These levels of impatience are the robots' temporary dynamic models of each others' current state. This behaviour has the effect of increasing fault-tolerance/robustness in the system. Since an attractive application of multi-robot systems is their deployment in dangerous environments, robustness is a critical concept: systems must deal with robot malfunctions, and maybe even robot destruction.

In ALLIANCE robots broadcast when they are starting a task. This allows other robots to avoid duplicating the task. Furthermore if other robots are not informed of task completion after a certain time (possibly due to malfunction), then they will take over the task. The impatience factor of other robots is matched by an acquiescence factor of the failing robot. If it spends too long doing a task, the robot will actually give the task up.

ALLIANCE has been physically implemented on a simplified nuclear waste clearance problem, and a box pushing problem. It has also been demonstrated in simulation in a multi-robot janitorial team - robots that dust, empty bins, and clean the floor. Parker later extends the ALLIANCE system to facilitate the task assignment problem. In L-ALLIANCE the robots develop more permanent models of their peers (and themselves) by remembering how long their peers (and themselves) take to do different tasks. They use these models to select which the tasks they are best at, to allow their peers to do the tasks they are not so good at.

2.7.5 Peer Models and Artificial Social Intelligence

[Dautenhahn 1995] studies peer modelling as a part of artificial social intelligence. Dautenhahn suggests that artificial social intelligence is not only necessary for truly co-operative behaviour amongst robots, but also for interaction between robots and humans. For robots to gain social acceptance they need to have the intelligence to fit in comfortably with society. Dautenhahn also introduces the idea that artificial social intelligence is a prerequisite for more general artificial intelligence. She then discusses the use of imitative learning between robots and proposes an experimental environment for studying social intelligence called the "Huegellandschaft" - a hilly landscape that robots try to take the path of least energy through. Robots can imitate other robots movement behaviours while trying to learn how to save energy. They will need to learn to recognise which robots provide them with suitably energy-saving behaviours. This will be seen to be similar to the system in this thesis, where robots only positively interact with robots that have been helpful in the past.

[Dautenhahn 1997] implements a similar system where a "child" robot follows a transmitting "mother" robot around. The mother robot transmits different signals depending upon what "she" is seeing. The child robot then learns to associate these signals with its own sensory input, and thus learns the mother robot's "vocabulary". In the future, when the child wants to find the mother it observes what the mother is currently transmitting, and searches for the mother by trying to match its sensory input to that which it associates with such a transmission.

Mobile-robot imitation behaviour has also been studied by [Demiris and Hayes 1996][Hayes and Demiris 1994] in physical experiments. Robot imitation can be thought of as peer-modelling: the imitating robot is building an implicit model of the robot it is imitating which it can later be used to dictate its own behaviour. In [Demiris and Hayes

1996][Hayes and Demiris 1994] a robot follows a teacher robot through a maze, thus learning to navigate the maze. The robot learns to associate the actions it copies with what it is perceiving.

2.7.6 Social Learning

[Mataric 1994b] is inspired by social learning ideas from ethology. She concentrates on a physically-implemented robot group learning "social rules": i.e. behaviours that do not immediately benefit the robot, but do benefit the group as a whole. The robots learn to achieve the task of collecting pucks from a pile and bringing them home. They have two possible social behaviours: "yielding" and "communicating puck location". The first makes a robot stop obstructing another robot, the second tells robots where the puck pile is so they do not have to search for it.

Neither of these behaviours directly benefit the yielding or communicating robot, thus there is no reason why it should ever learn them. Mataric overcomes this by using a form of reinforcement learning where rewards are shared. For example, when a robot gets a puck home it gets a "reward" which reinforces all the behaviours that enabled it to get that puck home. This "reward" is "shared" amongst the other robots, who therefore will reinforce any social behaviours they used to help the puck-returning robot. Thus the robots develop an implicit model of the environment and their peer-environment through their learned behaviours.

2.7.7 Competitive Learning

[Asada et al 1995] discusses social learning and imitative learning in relation to *competitive* learning. Mobile robots can learn to efficiently perform tasks through competition with other robots. This highlights the point that seemingly negative interaction

can have positive future results for learning robots. It is important to draw a distinction between competitive learning (which Asada examines) and competing to be allowed to learn (which will be used later in this thesis to allow the setting of the balance between distributed exploration and exploitation). Competitive learning is learning that actually uses the dynamics of competition to mould a robot's models of the environment and other robots.

A standard testbed for competitive learning has been proposed: robot soccer. [Asada et al 1995] discuss a multi-robot system which has been developed in stages: first a single robot learning to shoot a ball into a goal, and then a single robot learning to shoot the ball past a robot goal-keeper. They also introduce a method for stationary obstacle avoidance and robot collision avoidance. Thus the robot can avoid being blocked off by other robots. They use a system called LEM (Learning from Easy Missions) which allows the robot learning to develop in incrementally more competitive situations (i.e. faster opponents).

Robot soccer as a model for competitive learning is also discussed in [Stone and Veloso 1997]. However, the paper does not directly implement competitive learning. Rather it argues that certain basic collaborative team behaviours must first be developed. They simulate a neural network solution to the problem of one robot ("passer") accelerating the ball to another robot (the "shooter") who then "kicks" it into an open goal.

[Balch 1997][Balch 1997b] actually implement a full learning robot soccer game. The behaviours consist of two possible perceptual states: "behind ball" and "not behind ball", and three possible actions to follow these states: "move to ball", "move to backfield" and "get behind ball". Individual robots use reinforcement learning to develop playing skills based on which actions to take in which sensory states. Balch also looks at measuring behavioural diversity in the team using a measure called "Social Entropy". The greater the Social Entropy in the team, the greater the diversity of learned behaviours.

2.7.8 Peer Models and The Contract Net Protocol

Another multi-robot system which uses competition, but for task distribution rather than learning is the LEMMING system[Ohko et al 1993]. Reference was made earlier to the Contract Net Protocol(CNP). In the LEMMING simulation the CNP is used in a simulated office environment. Robots' tasks consist of taking an item (e.g. a pencil or calculator) from one desk to another. A task to be done is broadcast by a robot and the other robots make bids to take on the task. These bids include the estimated time it will take a robot to do the task. Initially the task broadcasting robot will select the robot with the best bid (i.e. the smallest time estimate.)

When the chosen robot has done its task, it reports this to the task providing robot. The task provider then has information about the task, robots' bids on the task, and the actual performance of one of the robots in that task. It can use this information to decide which robot to award the task to in the future. It can also use it to limit its task offer: only directing it at those who are capable of doing the task. Thus the robots develop models of their peers abilities to do different tasks.

A similarity measure is also employed, so that if a robot wants to offer a task which it has not offered before, it can estimate which robot(s) will be best at it. For example if Robot A has efficiently taken a Book from Desk 1 to Desk 2, and the new task is to take a Book from Desk 3 to Desk 2, then Robot A has a higher chance of being awarded the task, due to a book being involved. The LEMMING system is differentiated from the system studied in this thesis by the fact it is using peer-learning to solve a task assignment (rather than task learning) problem.

[Asama et al 1992] also uses a form of CNP for task assignment. It discusses the use of negotiation amongst robotors in the ACTRESS system for producing collaborating

teams. The agent requiring collaboration uses a negotiation system similar to the Contract Net Protocol: it broadcasts a request for collaboration to all agents. All agents that can help will make a bid. Those bidders that succeed by giving the best bids are used to make up the team.

2.7.9 Peer Models and Robot Co-operation

[Sen 1996] looks at the use of *Utility* measures in physical interaction - i.e. not task assignment or task learning, but actual task *doing*. Sen is concerned with examining the emergence of co-operation - i.e. how can selfish agents learn to help each other to their own advantage?

[Sen 1996] introduces a mail-delivery system in which robots deliver mail picked up from a centre. Obviously if robot 1 is about to deliver some mail to a destination close to where robot 2 wants to deliver mail, then robot 2 can ask robot 1 to deliver its mail as well, thus saving system energy and allowing robot 1 to get on with another task. Then, in the future robot 2 may be heading for a destination where robot 1 wants to make a delivery, so robot 1 can ask robot 2 to reciprocate and make its delivery for it. Sen thus uses the word "co-operation" in the correct sense - short-term loss leading to long-term gain.

In this system it is obviously key that robots reciprocate so that the group is used at full efficiency. To enforce reciprocity, Sen's robots use past helpfulness as a measure of future helpfulness - i.e. if a robot has been consistently helpful to me in the past then it is assumed that the robot may be helpful to me in the future. The robots co-operate in a probabilistical sense - i.e. if robot 2 has been helpful to robot 1 in the past then all we can say is that robot 1 is more *likely* to help robot 2. So robots are never guaranteed help or refusal, there is always a possibility of either. An equation for the precise probability of this help happening is provided by Sen.

The equation that robot 1 uses to judge whether or not to help another robot 2 contains terms that decrease the probability of help the greater the energy cost of a task, and terms for checking that the energy robot 1 has used in *helping* robot 2 is approximately equal to the energy saving robot 1 has made from *being helped by* robot 2. The less helpful robot 2 has been to robot 1 (relative to how helpful robot 1 has been to robot 2) the lower the probability. This probability is a measure of future utility of robot 2 to robot 1 based on past experience.

Sen uses a probabilistic decision mechanism because if robot 1 will absolutely not help robot 2 until robot 2 has helped robot 1, and vice-versa, then the robots are in a deterministic deadlock - neither will to ever help the other. However, the stochastic approach ensures that there is a chance of help even when robots have not previously interacted. Thus, as long as there are cases where robot 1 can do robot 2's task more cheaply than robot 2 (and visa-versa), co-operation will emerge as a result of this probabilistic reciprocity.

[Sen 1996] has some similarities to the solution presented for the distributed exploration vs. exploitation dilemma in this thesis, but this relevance cannot be explained until the contribution has been introduced in the next chapter. In the meantime it is observed that Sen uses physical robot-resource sharing as opposed to informational robot-resource sharing co-operation studied in this thesis.

2.8 Multi-robot Communication

The fifth, and final, of the five key areas for distributed map-sharing mobile multi-robot systems is Communication. There have been few systems analysing on the effect of communication in MRS.

[Arkin et al 1993] studies the improvements in efficiency in the simulated foraging task introduced in [Arkin 1992] when communication of behavioural-state is allowed between robots. Initially all robots search for "food". When a robot has found and is heading towards "food", or is carrying the "food" home, it broadcasts to all other robots that it is in this state. Other robots will move towards the closest robot to them which is broadcasting one of these states and will help it carry the "food" back home. The communication was found to speed up retrieval when compared to the co-operation without communication in [Arkin et al 1993]. An analysis was also done of negative interaction problems which produced a graph showing the optimal number of robots given the number of pieces of "food".

Other work concerned with communication is [Dautenhahn 1997] (mentioned earlier) which develops a communication vocabulary between mother and child robots, [Ichikawa and Hara 1994] which studies the formation of a robot communication chain by sweeping, and [Wang and Premvuti 1994] which introduces a broadcast system for multi-robot communication. It is this sort of broadcast system ([Wang and Premvuti 1994]) which is envisaged as being used in the system in this thesis.

2.9 Related Work in Learning Multi-Agent Systems

This chapter has so far reviewed topics from Multi-Robot Systems, and has related past MRS work to the type of system studied in this thesis: Environment/Peer-modelling, robot-resource sharing DARS. MRS is actually a subset of an area of study called *Multi-Agent Systems* (MAS). MAS is a more general area of study than Multi-Robot Systems. A robot is itself an agent, but so is an organism, an adaptive email program, a computer virus, or an autonomous web search engine. Multi-Agent Systems is in fact a sub-area of

Distributed AI, whose other main area of study is DPS (Distributed Problem Solving) [Bond and Gasser 1988].

Much of MAS is not directly relevant to this thesis. However, two areas of *Learning* MAS are relevant, and they will be examined in the next 2 sections. Learning in multi-agent systems has only been studied more recently, with the only books so far published being [Weiss 1997] and [Weiss and Sen 1996]. The two areas of research in learning multi-agent systems which are of particular relevance to this thesis are Collaborative Interface Agents, and the Prisoner's Dilemma. Collaborative Interface Agents will be examined first.

2.10 Collaborative Interface Agents

[Lashkari et al 1994] introduce the concept of Collaborative Interface Agents using email clients. While the user is manually using the email client, the client/agent tries to analyse what actions a user takes with different types of received emails. For example does he delete them, store them in a certain folder or forward them. So when an email agent receives an email for its user, it checks to see if it can make a good prediction of an action to perform based on the nature of the email (e.g. the *subject* line, *from* line, or keywords), and suggest the action to the user if it can.

If an email agent does not have enough past experience to make a confident prediction then it can ask other more experienced email clients for advice over the Internet. These other clients can tell the first client what action *they* would take (i.e. their user would probably take) if they received that email. The client can take a piece of advice and suggest it to the user. If the user agrees then the client makes a note of which other agent it received that advice from and ups its view of that agent's utility.

So as time goes on all email agents will develop a view of which other agents are most useful to them (i.e. which other agent's users have similar requirements to their users); thus enabling agents to pick which advice to take. The agents can also ask "trusted" agents for advice about which other agents to "trust", and can explore the population for possible new "partners" by making requests for answers to situations they already know the solution to.

One reason that Collaborative Interface Agents are relevant to this thesis is because the "utility" measures which the agents use will be seen to be related to the system introduced in this thesis for setting the balance between distributed exploration and exploitation. A similar point was made about the utility measures in [Sen 1996], however Collaborative Interface Agents are even more relevant as they use *informational* agent-resource sharing (as opposed to the more common physical agent-resource or environment-resource sharing in MRS).

According to [Lashkari et al 1994] agents requests may be refused if "the agent issuing the request may not have been helpful in the past, or may not be important enough" (though no explicit methods are described though for judging when an agent will accept a request for help). A similar line is taken in [Sen 1996], where agents only help if they've been helped in the past. The issue of "when to help" has been studied in the abstract in learning MAS through a game called the "Prisoner's Dilemma"[Axelrod 1984].

2.11 The Prisoner's Dilemma

This relationship between the Prisoners' Dilemma game and the robot systems studied in this thesis will now be analysed, as the Prisoners' Dilemma is a standard benchmark for co-operative behaviour. The analysis will take the form of describing the Prisoner's Dilemma in the context of mobile route-sharing MRS, and is itself a

contribution. The Prisoner's Dilemma is a simple game which captures some fundamentals of co-operative behaviour. It is usually used to study the emergence of co-operation (as in the [Sen 1996] mail delivery agents described earlier). It is so-named because it is usually described using an anecdote about two prisoners. However, a more relevant example will be given which will help to clarify the relationship between this game and this thesis.

Suppose there are two robots R1 and R2 who can learn an environment and transmit their learning. Furthermore assume that each robot must decide independently, before the two start learning, whether it is going to transmit a route to the other robot or not. The robot must then stick to this decision during learning. If it decides to be helpful then it will transmit a route to the other robot during learning.

If the robots' task is for each robot to learn a full map of the environment (either by experience or transmission) then their energy consumption can be divided into 3 scenarios:

1. R1 and R2 decide independently to not share information. So they must both learn the whole environment, but do not use up any energy in transmitting routes. Denote the average amount of energy needed by a robot to learn the whole environment as p.
2. R1 provides R2 with information, but R2 does not provide R1 with information. So R2 does not need to learn the whole space and does not use energy transmitting routes. Denote the amount of energy used by R2 as t. However, R1 needs to learn the whole space *and* use energy transmitting to R2. Denote this average amount of energy that R1 uses as s.
3. Suppose R1 *and* R2 provide each other with routes. Then both need to use transmission energy, but neither need to use the full amount of energy to learn the whole environment. It can be seen that in this situation R1 and R2 will use the same average amount of energy r.

If, for a single robot, the energy used to learn the whole environment is denoted as *L*, the average energy saving made from help by another robot as *Sav*, and the average sum of transmission energy *Trans*, then the following equations apply:

$$p=L$$

$$t=L-Sav$$

$$s=L+Trans$$

$$r=L-Sav+Trans$$

Then since $Sav>Trans$ (since transmission energy is smaller than motor energy) it can be seen that $t<r<p<s$. If the "pay-off" for a robot is defined as the reciprocal of the energy it has to use (i.e. low energy use implies high pay-off), and the pay-off is denoted as the capital letter of the energy consumption, then it can be seen that $T>R>P>S$. This ordering of pay-offs is the main characteristic that makes the interaction between the two robots a game of the Prisoner's Dilemma.

2.11.1 Why The Prisoner's Dilemma is a Dilemma

The Prisoner's Dilemma is a dilemma since robot 1 needs to decide whether to help the robot 2 or not. Suppose that robot 1 thinks that robot 2 is going to help it. Then the highest pay-off it can get is *T*, through not helping robot 2. Alternatively, if it thinks robot 2 is not going to help it, then the highest payoff it can get is *P*, also through not helping robot 2. So the logical thing for robot 1 to not help robot 2. However, by that same logic, robot 2 will not help robot 1. So two robots playing this game will be forced to get the 2nd lowest payoff *P*, which is worse than the payoff *R* if they had both helped each other.

So really they should help each other, but what rational robot would take the risk of getting the lowest payoff *S* in the hope that the other robot would also help? This is the dilemma the robots find themselves in. In fact, to truly ensure this is a dilemma, Axelrod requires another condition along with $T>R>P>S$. It must be ensured that the robots cannot get out of their dilemma by taking turns exploiting each other. This requirement translates as $R>(T+S)/2$. A contribution in the form of a proof of this property for the 2 robot system is given in the appendix.

2.11.2 The Iterated Prisoners' Dilemma

As it stands, the Prisoners' Dilemma scenario is still very different to that presented in this thesis. However, Axelrod introduces the idea of the *Iterated* Prisoner's Dilemma, where multiple games are played by robots, and where the games do not have to be independent. Thus robots can develop models of each other, and try to work towards higher pay-offs. For example if robot 1 finds than robot 2 is optimistic (i.e. robot 2 helps more frequently than average) then robot 1 can become more optimistic itself so as to get more *R* pay-offs.

However, if robot 1 finds that this optimism is irrational (i.e. robot 2 helps *whatever*) then it can go back to refusing to help - thus giving the maximum payoff *T*. However, if robot 1 finds that robot 2 is retaliatory, i.e. that it will not help if it is denied help for too long, then robot 1 will see the sense of sticking to a helpful policy.

Axelrod in fact found that the agent that faired best overall in the IPD only needed a very simple peer-model - just the memory of its opponents last move. The agent's strategy, called TIT-FOR-TAT, was the strategy of never being the first to withhold help, and repeating its opponents response from the previous game. So a TIT-FOR-TAT agent will be helpful if the other agent is helpful, and unhelpful if the other agent is unhelpful.

However, such a policy was found to be inefficient when applied to an actual multi-robot problem, i.e. the mail-delivery robots in [Sen 1996].

2.12 Summary of the Review of Multi-Robot Systems

In this chapter a detailed taxonomy of mobile MRS has been introduced and this thesis has been placed in the context of these as studying Environment/Peer-modelling, robot-resource sharing DARS. This was followed by a look at two areas of Multi-Agent Systems which are of relevance to this thesis.

It was seen that none of these past systems have looked at the negative effects of information-sharing in learning MRS. The principal contribution of this thesis will be seen to have some relationship with certain studies in MRS and MAS involving utility measures. However, these past utility measures have been used to study the emergence of co-operation or the assigning of tasks, whereas this thesis will be using them to address the distributed exploration vs. exploitation dilemma.

Chapter 3

Method

3.0 Proposed Solutions

In this chapter a approach will be proposed for a problem that information-sharing introduces to learning multi-robot systems: the Distributed Exploration vs. Exploitation Dilemma.

3.1 The Distributed Exploration vs. Exploitation Dilemma

The distributed Exploration vs. Exploitation dilemma comes about through robots reducing the amount they learn by experience by achieving their goals too quickly through information sharing. The distributed exploration vs. exploitation problem is concerned with learning the shortest routes, to be specific: the shortest direct routes to goals. If one robot learns a route to a goal, then all other robots in the team will use that route and not bother learning any new, potentially shorter, ones by experience. So to prevent this a robot must sometimes withhold from asking for the route from another robot which has the route to a desired goal. But if a robot never asks for routes it would be pointless having route-sharing robots. So when should a robot ask for routes and when not? The answer to this question depends on the precise balance between exploration and exploitation required in the team. What is needed is some method of simply adjusting this balance. Such a method would be an original contribution to multi-robot systems, and one is now presented.

Robot i keeps a count rg_i of the number of routes ending in found goals (i.e. as opposed to ordinary salient points) that it has learned by experience. Then, given say robot 1, if robot 1's count rg_1 is less than the group average $E(rg_i)$ robot 1 will refrain from

asking for routes to goals. Thus it will be forced to learn new routes to the goals. This solution requires the construction of $E(rg_i)$ through broadcast transmission with other robots. However there is actually a local approximation which requires no transmitted construction of $E(rg_i)$ at all. Such an approximation is obviously desirable for the energy saving and better scaling attributes, and so it will be seen that this approximation is key to the original contribution of this thesis.

The first step to developing this local approximation will be to develop a global approximation to the $E(rg_i)$ method. This is done by observing that the distributed exploration vs. exploitation dilemma only occurs when the robot team have some common goals. When robots have common goals, there will be a significant amount of information interchange, and the number of routes a robot is able to contribute to the team will be related to the number of routes ending in goals it has. Thus the larger rg_i (the number of goal-ending routes learned by robot i) is, the larger the number of routes robot i will contribute to the team. Suppose the number of routes a robot i has been able to transmit to other robots is equal to c_i (i.e. the number of times it has been requested for a route ending in a goal and been able to answer that request). Suppose that the average routes all robots in the team have contributed is $E(c_i)$. If c_1 is greater than or equal to $E(c_i)$ then robot 1 will ask for routes, and if it is less then robot 1 will abstain from asking. Thus if robot 1's contributions of goal-ending routes to the team is equal to or above average, it can ask for the route to a goal. But if it is below average it will not ask for the route, so forcing itself to learn more until it has contributed more routes. It is suggested that this will approximate the $E(rg_i)$ algorithm above. It is further suggested that the following is a local version of this approximation which will require no global construction of $E(c_i)$.

The first step to developing this local version is to make the robots "selfish", i.e. robot 1 will always be happy to request information, and in fact will *request* it whatever the state of its "contributed routes" count. It will be up to other robots to reject robot 1's

request for information. A robot, say robot 2, can keep a count of how many useful routes $c_{2,1}$ it personally has received from robot 1. Robot 2 will then locally calculate $E(c_{2i})$ instead of the global $E(c_i)$ above. For example, Robot 2 will reject robot 1's request for a route if robot 1's count, i.e. $c_{2,1}$, is lower than $E(c_{2i})$. So robot 2 will refuse to provide robot 1 with any more information until Robot 1 has provided it with enough routes ending in goals. The $E(c_{ji})$ "thresholds" can be seen as a distributed approximation of the single $E(c_i)$ spread across the robot team, since if robot 1 does not provide the other robots with enough routes (relative to the number of routes the other robots have received) then the other robots will individually refuse robot 1 information. So the robots not only have a distributed model of the space, but also a distributed approximate model of robot 1's count of goal-ending routes, which is used in a distributed way to ensure robot 1 is contributing routes, i.e. learning new routes ending in goals. Thus it is argued that this local version will on average approximate the global $E(c_i)$ algorithm above, and hence will increase the number of routes to goals robots will learn to goals by experience.

The above algorithm actually turns the robots *interaction* into *co-operation*. This is because a robot in a group using this algorithm will only help another robot if that other robot has helped it in the past. This past helpfulness is usually seen as a predictor for future helpfulness. Hence a robot using the system in this thesis will only help another robot if that robot is likely to help it in the future. In other words a robot will only accept a short-term loss (transmitting information to another robot) if it is likely to receive long-term gains (having information transmitted to it, thus saving searching time). For this reason, the system presented in this thesis will be labelled *Co-operative Learning*.

Co-operative Learning can be used to adjust the balance between distributed exploration and exploitation in the MRS as follows. The criteria for robot 2 refusing robot 1 is $c_{2,1}<E(c_{2i})$. If this is replaced by $kc_{2,1}<E(c_{2i})$, i.e. robot 1's contributions to robot 2 are multiplied by a factor k in the decision inequality, then the larger k is, the harder it will be

for robot 2 to refuse robot 1, hence the more exploitation robot 1 can make of robot 2's information. Furthermore, the lower k is, the more likely it is robot 1 will be refused information about how to achieve its goals, and hence it will have to explore for further routes itself. Thus by increasing k the tendency to exploit is increased, and by decreasing k the tendency to explore is increased.

3.2 Contribution

Co-operative learning, presented in section 3.1, is proposed as a solution to the balance setting in the Distributed Exploration vs. Exploitation Dilemma problem. Results will be given later showing that co-operative learning will allow the setting of desired levels of multi-robot exploration and exploitation.

These results will come from using a simulated multi-robot system which is presented in the next section.

3.3 The Robot Model

This section is a description of the learning mobile robot model which will be used to test the system presented above. The model is a simplified representation of a mobile robot. Such simplification is considered justifiable since it is not individual robot hardware abilities that are being studied here so much as concepts in robot information exchange. It is suggested that the principles introduced in the previous sections can be applied to any mobile multi-robot system where robots want to encourage a more distributed model.

It was mentioned in chapter 2 (Section 2.3) that the robots studied in this thesis have five basic abilities: random sweeping, environment learning, point-to-point motion, communication and peer-modelling. The models of these abilities will be examined one by one.

3.3.1 Random Sweeping

The robot can move forwards and backwards one body-length, and rotate on its axis left or right 90 degrees. It is assumed that the robot can perform these movements perfectly each time; hardware systems are not yet so reliable[Nehmzow 1991].

The robot uses on-line learning and so will only move if it has a goal. A goal is the requirement to find a discriminable sensory cue or salient point. If the robot achieves all its goals - i.e. finds all the required cues - it stops searching and remains still until given another goal. (For convenience, in future the distinction between a goal and a required salient point will be waived. For example, a block marked with a red letter X will be called a "goal", although the actual goal is to *move* to a block marked with red X.) When a robot has a goal it will move randomly according to a distribution. The precise probability distribution is (0.66,0.16,0.16,0.02) for (Forwards, Right 90, Left 90, Backwards). Only a few other distributions were tried in preparing experiments, and this was found to give a satisfactory search of the space. A lot more effort could have been put into finding the optimal distribution for the class of environment used in the robot experiments; but such a study of the relationship between distribution, environment and coverage is beyond the scope of this thesis. Another topic that is beyond the scope of this thesis is robot avoidance in point-to-point motion. As was seen in the review of multi-robot systems in chapter 2, robot avoidance in point-to-point motion is a non-trivial issue, and, since such an issue is beyond the scope of this thesis, robots will simply be allowed to pass through each other like "ghosts" in point-to-point motion. Because of the on-line nature of the learning algorithm, no distinction is made between learning and goal-achieving. Thus the ability of the robots to pass through each other applies to sweeping as much as to point-to-point motion. This connection between the two also means that a full description of the sweeping

process cannot be given until the environment learning system has been described more fully.

3.3.2 Environment Learning

The robot has the ability to develop and recognise discriminable sensory cues, and to detect obstacles. The obstacles are white robot-sized blocks. A robot can detect an obstacle by contact, and will then cease trying to move into the obstacle. Discriminable cues would normally be developed through context[Denham and McCabe 1995]. For example a robot moving in an unknown environment, say a planetary surface, may see an unusual rock and decide to use it as a marker, or discriminable cue. It can then position other places it visits relative to this cue, and it can develop other markers to increase the resolution of its relative positioning system. If, however, the robot comes across another rock which is indistinguishable from the first rock, it will have to widen its context to differentiate the 2 rocks. For example it may see that the second rock is next to a small gully, so it can use "rock plus gully" as a discriminable salient point. In this model all of these abilities are assumed rather than implemented explicitly, and "discriminable salient points" are scattered throughout the environment. They are represented in the environment by coloured capital letters, and will be written in the form [A, red], [C, green] etc. Since a robot's goals are also cues, the goals will also be represented in the form [A, red] or [C, green]. A robot is defined as having achieved a goal once it is *facing* the goal (so a robot may have to rotate, even though it is adjacent to a goal).

A novel learning algorithm is used based on [Denham and McCabe 1995] - the design of this algorithm is motivated by a desire for easy generalisation into multi-robot systems. When this algorithm is put in the context of multi-robot communication in section 3.3.4, its generalisation advantages will become clear. The robot has a memory in the form

of a sequence memory. The sequences are of "Sensory-Action Pairs"[Denham and McCabe 1995] (SAPs): an action taken (i.e. Forwards, Backwards, Right 90, Left 90) paired with the resulting sensory input of the space in front of it. So if a Robot moves forwards one body-length and becomes adjacent to and is now facing the cue [C, red], the appropriate SAP can be written as (forwards, [C,red]). Note that the robot can only see objects directly in front of it. Hence if the Robot had been facing the cue [C,red] but had been 1 step away rather than adjacent, then it would have stored (forwards, 0) where 0 represents an empty robot-sized space.

When a robot recalls an SAP (*action*, *sense*) from its memory it does the following. It looks at the action part of the SAP. If it is the "null" action - i.e. the action part of the SAP is empty - then it randomly generates an action (according to the distribution mentioned earlier), performs this action and stores it in the empty action space. However, if there is an action already stored in the SAP, then it performs that stored action. After taking an action, random or stored, the robot compares its sense input with the stored sense input of the SAP. If the two do not match then it stores the new sense input. In cases where the sense part of the SAP is empty, i.e. it contains the "null" sense, the robot will always detect a mismatch - since no actual sense input will match with the "null" sense.

A robot's memory simply consists of sequences of SAPs. A robot recalling/following a sequence of SAPs will simply perform the behaviour described in the above paragraph on each SAP in the sequence in order. For example, suppose a Robot follows the sequence:

(forward, 0), (right, 0), (forward, 1), (left, [A,red])

(Here 0 represents a empty robot-sized space - i.e. no obstacle or cue; and 1 represents a robot-sized obstacle.) The robot will move forwards and check it sees an empty space. If so

then it turns right, and checks once again it sees an empty space. It then moves forwards again. Suppose an obstacle has been removed, and the Robot now sees a blank space in front of it, then it will detect a mismatch and re-learn (forward,1) as (forward,0). It will then turn left and be facing [A,red], so it can now remove the goal [A,red] from its list of goals to be achieved.

If the robot was following the empty sequence:

(null action, null sense),(null action, null sense),(null action, null sense)

then it would generate actions for each null action, and detect mismatches for each null sense. Thus it would store a sequence of three consecutive randomly generated movements, together with the sensory results of each movement.

The above reaction to a sequence of empty SAPs is actually a random sweeping/goal-searching behaviour. Therefore a robot will *always* be following a sequence in its memory, as long as it has a goal. A robot does not move independently of its memory - its actions are totally bound to its learning. So when given its first goal a robot will try to use its memory to find the goal: it will be following SASs of empty SAPs - i.e. SAPs with null actions and null senses. Thus the robot will move randomly ("motor babble") and store everything it sees (due to mismatch with the null sense).

When the robot is following a pre-learned sequence, it will follow the actions stored in that sequence. However, it also has the ability to detect novelty: i.e. if it takes an action and gets an unexpected sense result - a mismatch - then it knows that the SAS is no longer valid, that the environment has changed in some way (c.f. the obstacle removing example above). It will thus start relearning, ignoring all stored actions from this point - but keeping the earlier actions in the sequence - and just motor babbling.

3.3.2.1 Chunking

A robot's SASs will be referred to as "chunks" since the robot's learning algorithm is inspired by a learning method called "Chunking" [Denham and McCabe 1995]. When given its first goal, a robot will look for a chunk that ends in that goal. It will not have one so it randomly selects a chunk. It will try to follow an empty chunk, thus moving randomly and learning. When the robot meets its first discriminable salient point (say [A,red]), it resets the chunk and sets the chunk's "head" - i.e. its first SAP - to (null-action, [A,red]). It then starts following the chunk from the second (empty) SAP. It will motor-babble until it reaches another discriminable salient point (say [B,red]). This chunk now contains the information necessary for the robot to move from red A to red B, i.e. if the robot is facing red A and follows this stored chunk, it will reach red B (assuming no motor inaccuracy or wheel slip). The robot leaves this chunk stored and, assuming [B,red] wasn't its goal, will follow another empty chunk trying to find its goal. It will set the head of this SAS to (null-action, [B,red]) and start following the empty SAPs (motor-babbling) until it finds and stores another discriminable salient point, say red C, after which it will move to another empty SAS. So it now has chunks for A->B and B->C. As this process continues the robot builds up a map of the environment based on the discriminable cues.

A number of problems may occur during this mapping process. The chunks are limited to a finite size M, hence if a robot doesn't reach a discriminable salient point within M steps it resets (i.e. empties) the chunk and continues moving, learning from where it left off, but storing from the beginning of the chunk. Another problem is circular chunks. To overcome this, if a robot has a chunk red A->red A then it resets this chunk and gives it head (null, [A,red]), following this reset chunk from its second SAP onwards. Finally, if

the robot reaches a discriminable cue C, but the chunk doesn't start with a discriminable cue, then it resets the chunk, putting C at the head, and starting from the second SAP again.

3.3.3 Point-to-point Motion

If a robot has a goal, it uses its memory to find the goal. If it is following an empty chunk, it is essentially using that chunk to achieve its goal by trying to reach the goal through random movement. A robot also uses chunks to reach its goal in a more deterministic way, or at least to increase the probability of reaching a goal.

When a robot is searching for goals and reaches a discriminable salient point, it checks to see if it can reach any of its goals from this point. A robot's goals are held in an ordered list, with goals at the beginning of the list getting first attention. Suppose a robot has a goal list (A,B) - i.e. it has goals A and B. When it reaches, say, C it selects the first goal in the queue, A, and searches its chunks to find if it has the chunk C->A. If it has then it can follow this chunk to A and achieve a goal. If not then it can check if it has a chunk C->B, following this to B if it has.

But suppose the robot has neither of these chunks. It can still attempt to increase its probability of finding A and B by *generating* goals. When a robot is given a new goal it searches for all chunks *ending* in that goal. If such chunks exist, and the robot can find the beginnings of any of these chunks, then it can achieve its goal. So what it does is add the heads of these chunks as goals at the end of the list. For example, suppose the robot has chunks D->A, E->B, F->B, X->D, Z->D and is given goals (A,B). The robot selects the first goal in the list, A, and searches for chunks - it will find D->A and hence update its goals list to (A,B,D) since achieving D will enable it to achieve A. Next the robot selects B and will hence generate the goals E and F, leading to a goal list (A,B,D,E,F). The robot selects the next goal in the queue, D, and searches for chunks ending in that cue, generating

X and Z as goals, and leading to a list (A,B,D,E,F,X,Z). The Robot will then run through the rest of the goals in order - E, F, X and Z - but it can be seen that it has no chunks with these "tails" (i.e. ending in SAPs with these sense parts), so the goal generation will end here until the robot is given another goal.

The aim of this goal generation is to increase the number of discriminable salient points that the robot can reach its original goals from, and thus increase the probability of the robot reaching its original goals. Suppose that the robot manages to find Z after it had done the above goal generation. Then it would have been able to achieve its goal by following Z->D->A. To facilitate this chaining of sequences, each goal has a "root". The root of a goal is its reason for existence - for example D exists as a goal because of A: D "has root A". E exists as a goal because of B: E has root B. A exists for its own sake - hence it is said to have root A, or to be "auto-rooted". Thus the above goal list, including roots, can be written as (A_A,B_B,D_A,E_B,F_B,X_D,Z_D). Roots are used as follows: if the robot reaches cue Z it can remove Z as a goal, but before it removes goal Z, it notes that Z has root D. So when it has removed Z as a goal, it knows it should search its memory for chunks ending in the root D. It will find such a chunk leading straight from its current position Z (it is certain this chunk exists because the production of Z_D by goal generation proves it). The robot will then follow this chunk to D. Then before removing D as a goal, the robot notes that D has root A, and so will search for chunks ending in A. It will find such a chunk with a head at its current position and so will follow this chunk to A, removing A as a goal. The robot's goal list will now be (B_B,E_B,F_B,X_D). In fact, this is not quite true; the goal list will actually be (B_B,E_B,F_B). X_D would have been removed for the following reason: when the robot removes a goal, it also runs through its goal list and checks that all roots are still goals; those goals whose roots are no longer goals (i.e. which are redundant) are deleted. So when the robot achieved goal D_A it would have run through its goal list and removed X_D. This process avoids robots building up redundant goals. (The

robots also need to avoid building up potentially looping structures of goals and roots, e.g. (B_B,E_B,F_B,E_F). To avoid this, a goal cannot be added twice to the list, no matter what its root.) So the robot will now be left with goal list (B_B,E_B,F_B), and it can now move around searching for B, E or F.

It was said earlier that goals were generated whenever a new goal was added. This statement needs to be clarified since goals are being added all the time *during* generation. To be precise goal generation only happens when an *auto-rooted* goal is added. (Or, as will be seen in the next section, when an auto-rooted or non-auto-rooted goal is provided by another robot). Goals whose roots are not themselves are actually subgoals, and if their addition always led to a restarting of goal generation the robot could become deadlocked through infinite recursion.

So the robot generates all subgoals relating to a goal in one go, when that new goal is added. However, when following these subgoals back to the goal, it processes each subgoal (i.e. searches for relevant sequences) independently, rather than using the subgoals to create a supersequence (a "sequence of sequences") and locking itself into following that supersequence. The independent processing ("distributed") approach to sequence-following is designed for use in a system of multiple robots. The reason for its importance in generalisation to multi-robot systems cannot be fully explained until the robot interaction system has been described - but to put it simply, the independent subgoal processing allows a robot to more simply take advantage of new chunks that other robots discover during the time it is following subsequences back to a goal. (There is actually another advantage, not related to multi-robot systems, which has to do with novelty detection. If the robot detects novelty while following a fixed supersequence, it will have to jump out of the process of following the pre-defined set of sequences, and concentrate on the single sequence it is following, trying to re-learn it. Thus this approach would require two robot states: learning and supersequence following. However, the robot model

presented here simplifies this - the robot is always following a sequence, whether it be a subsequence or non-subsequence, and (re-)learning is just a special case of sequence following.)

3.3.4 Communication

The purpose of the robot model design is to facilitate generalisation to a multi-robot system. In this multi-robot system, robots can interact in only one way: remote communication. It is envisioned that robots communicate using broadcast radio communication. A broadcast system is preferred as it scales better than a multi-channel system. It is assumed that a robot can send out a transmission with an identifier such that all the other robots in the receiving area (which is the whole simulation environment) will know who the transmission is from. It is also assumed that robots' broadcasts do not interfere; this can be achieved using the CSMA/CD-W protocol[Wang and Premvuti 1994]. Robots can broadcast Requests, Request Replies, Signals, and SAS's.

An important assumption of the communication (in SAS and non-SAS systems) is that the discriminable sensory cues can be communicated in a form that is recognisable by all Robots, i.e. that two Robots understand what the transmitted sense [A,red] looks like to them. This is a non-trivial assumption and involves the robots developing some sort of common vocabulary. There has been some study into the development of vocabulary[Dautenhahn 1997] which suggests that this assumption may be reasonable, but such problems are beyond the scope of this thesis.

3.3.4.1 Request for Goal Co-operation

It has been mentioned that robots can broadcast Requests, Request Replies, Signals, and SASs. Robots can broadcast one type of request - a Request for Goal Co-operation

(RGC). The purpose of an RGC is to allow robots to share information about the environment. The syntax of an RGC is RGC(r,Q) where r is the robot's call sign, and Q is some discriminable sensory cue. A robot r with goals will send an RGC(r,Q) (Q being the first goal in its goal list) whenever it finds any discriminable salient point. A robot R receiving the broadcast will search its chunks for those ending with Q. It will then broadcast a series of replies rRGC(R,r,Q,q_i) which states that robot R is telling robot r that it has chunks going from q_i to Q. If any of these are actually the cue that robot r is at, then robot r broadcasts a Cue Match Signal CM(R,r,Q,q_i) to tell robot R to broadcast the SAS from q_i to Q. (The system actually involves robot r comparing the lengths of sequences offered by different robots, and randomly selecting from the shortest ones.) Robot r can then learn this sequence and follow it to Q. If robot r is not at a q_i then it just adds the goals q_i (with root Q) to its goal list - generating more goals locally afterwards if appropriate. If it later finds itself at one of the q_i then before removing the goal q_i, it will notice that q_i has root Q and it will broadcast a CM on Q to get the SAS q_i->Q. It can be seen that the RGC system is just a multi-robot version of the single-robot goal generation and subsequence-following system. In fact, a robot will always check if it has the relevant chunks itself before it bothers broadcasting to other robots.

3.3.5 Implementing the Proposed System

The proposed system for dealing with the distributed exploration vs. exploitation dilemma is implemented as follows. In a population of P robots, each robot j has a set of counters c_{ji} (i between 1 and P) which are initially set to 0. To see how this is used to implement the proposed solution, take two robots 1 and 2. Robot 2 increments $c_{2,1}$ whenever robot 1 responds to an RGC positively (i.e. whenever robot 1 has a chunk that robot 2 has requested and broadcasts an rRGC). Then when robot 2 receives an RGC from

robot 1 it calculates $E(c_{2i})$ over all i - i.e. the average number of times robots have responded to its RGCs with subgoals. If $kc_{2,1}<E(c_{2i})$ (where k is set according to the designer-desired balance of exploration vs. exploitation) then robot 2 ignores the RGC, otherwise it tries to answer the RGC. Robot 1 will treat robot 2 in a similar way whenever it receives an RGC from robot 2.

When robot 2 receives a CM from robot 1, it will always respond (and visa-versa). This is because robot 1 will only send a CM to robot 2 if robot 2 has accepted an RGC from robot 1 in the past - i.e. if robot 1 has contributed sufficiently in the past. So the robots use a "my word is my bond" system - once robot 2 has generated a goal for robot 1 it will always provide the chunk for that goal. Without such a system, robots would constantly generating spurious goals.

3.4 Summary of Method

In this chapter various methods of setting the balance between Distributed Exploration and Distributed Exploitation in multi-robot systems have been addressed. It has been suggested that the most efficient approach is Co-operative Learning, due to its use of local rather than global information.

This chapter has also introduced a multi-robot model to be used for testing the effectiveness of the Co-operative Learning system.

Chapter 4

Results and Discussion

4.1 Experimental Testbed

The experimental testbed was a two-dimensional space 93 by 33 robots in size. It consisted of 3 types of objects: obstacles, empty space, and salient points. There was a wall around the space made up of obstacle objects, and various structures were scattered around the space also made up of the obstacle objects. A picture of the test bed is given below:

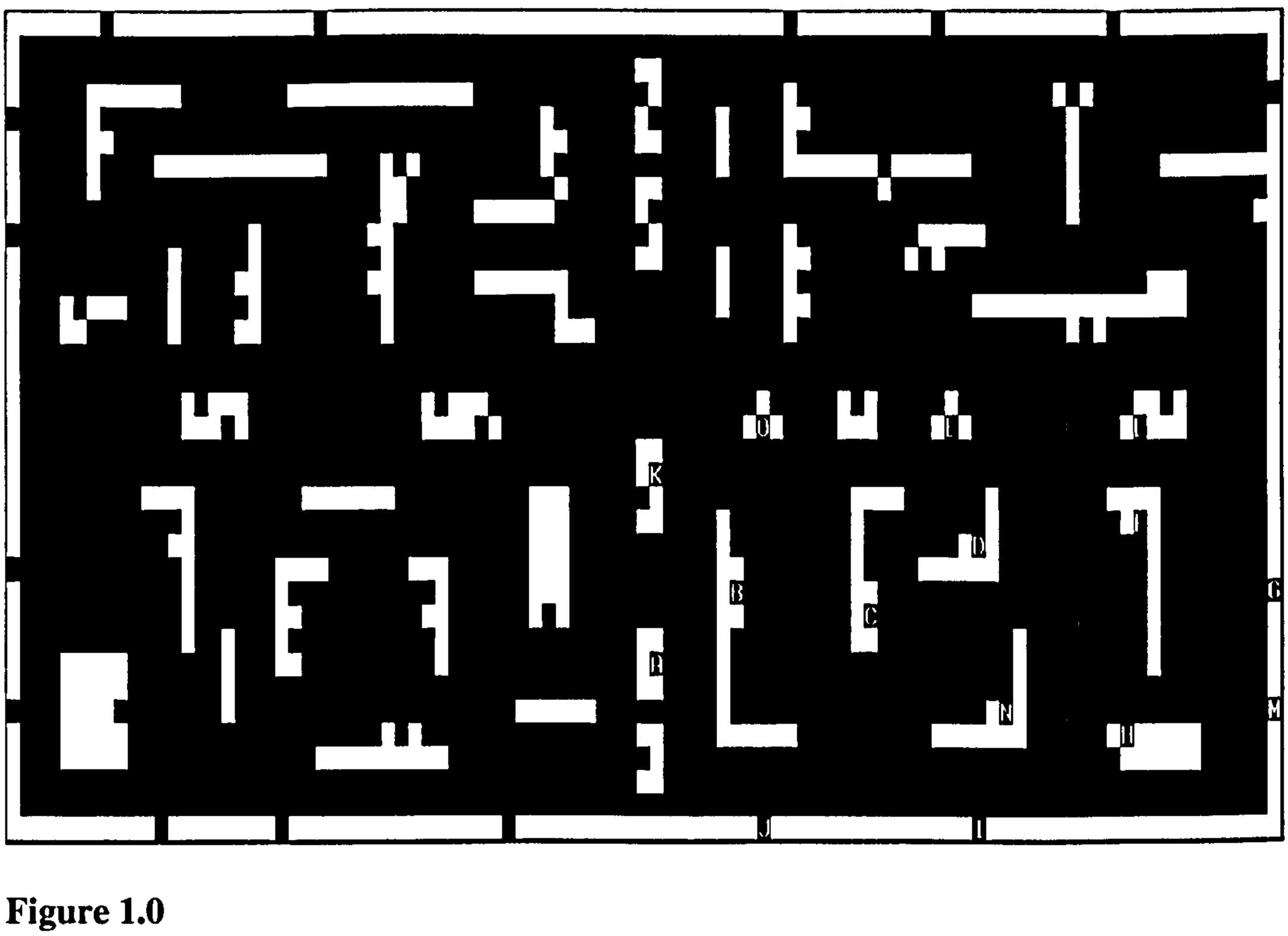

Figure 1.0

The different coloured letters dotted around the space are the discriminable salient points, all 1 robot in size. They are represented by the letters 'A' to 'O' in the colours red, blue,

green and yellow. So there are 15*4=60 discriminable salient points in a space of size 93*53=3069. This gives a salient point/space ratio of 2%.

All experiments started with a 15 robot team in a line across the top of the space. For each occasion, each robot would be given a number of goals, between 8 and 56. Robots would all be given the same goals since this is the worst case for the distributed exploration vs. exploitation problem. If robots have different goals, then they are less likely to learn from each other, whereas if they have common goals, it is possible for a robot to achieve all its goals while learning little itself by experience.

Experiments were terminated when the 15 robots had achieved all their goals. Each result listed here is a mean from 10 trials. Due to the random searching component of exploratory behaviour in unknown environments, each trial would be expected to exhibit different behaviour. Therefore the effects of co-operative learning would be expected to differ from trial to trial. So the most that can be said about the proposed solution is its *average effect* over a number of trials, and its mean deviation from this average. Since 10 trials are done for each point, a t-test will be used to calculate confidence intervals for any average effects.

The level of group exploration is measured by the average number of goal-ending routes robots learned by experience, rather than from other robots. The *Map-Contribution* of a robot to the team map will be defined as the number of goal-ending routes the robot has learned by its own experience. Thus a relatively high level of average map contribution across the team implies a relatively high level of exploration. The level of group *exploitation* is related to the amount of time it takes to achieve a mission (i.e. for all robots in the group to achieve their goals). Because the team uses random searching, the only deterministic influence on their mission time is the amount they are allowed to exchange information. Hence a lower rate of information exchange, i.e. a lower level of exploitation, would on average lead a longer mission time.

So the measure of success for co-operative learning will be if reducing k sufficiently has a 90% chance of increasing average map-contribution (i.e. increasing exploration), and increasing k sufficiently has a 90% chance of decreasing mission time (i.e. increasing exploitation). It is expected that because of the implicit dilemma, increasing exploration will also lead to an increase in mission time, that is a decrease in exploitation; and increasing exploitation will also lead to a decrease in map-contribution, that is a decrease in exploration. The reason for the 90% figure above is that the effects of k will all be examined at the 90% significance level.

4.2 Experiment 1 - Demonstrating the Distributed Exploration vs. Exploitation Dilemma

The distributed exploration vs. exploitation dilemma concerns the fact that once one robot in a map-sharing group learns a route to a goal, all other robots will use that version of the route rather than trying to learn new - potentially better - versions. This effect is shown for a variety of numbers of common goals in figure 1.1, with 99% error bars.

Figure 1.1 shows that the introduction of map sharing (i.e. increased exploitation) reduces the average map-contribution (the average number of new versions of goal-ending routes that are learned by experience per robot). However, it was also found that introducing map sharing also decreases the average route count per robot (a route does not have to end in a goal, just a salient point). Thus a decrease in goal-ending routes learned by experience would be expected anyway. So to confirm the direct effect of map sharing on exploration, the percentage effects on average map contribution are shown in figure 1.2. The effects in figures 1.1 and 1.2 are as a result of maximum exploitation between robots, since robots will always share information if asked. This is the equivalent of using co-

operative learning with an infinite value for *k*. Experiments are now shown examining the effects of finite values of *k* on the robot team.

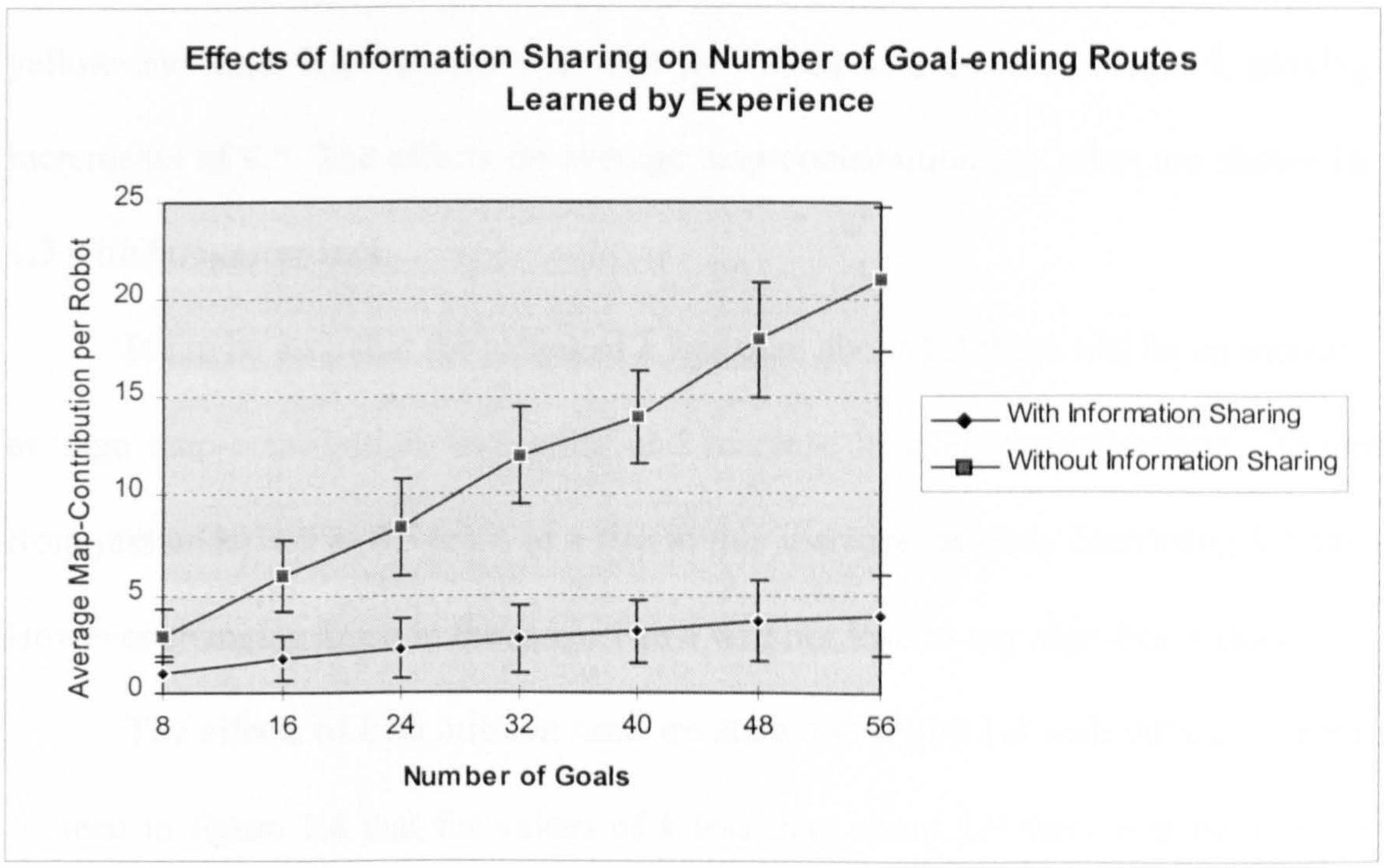

Figure 1.1 Experiment 1

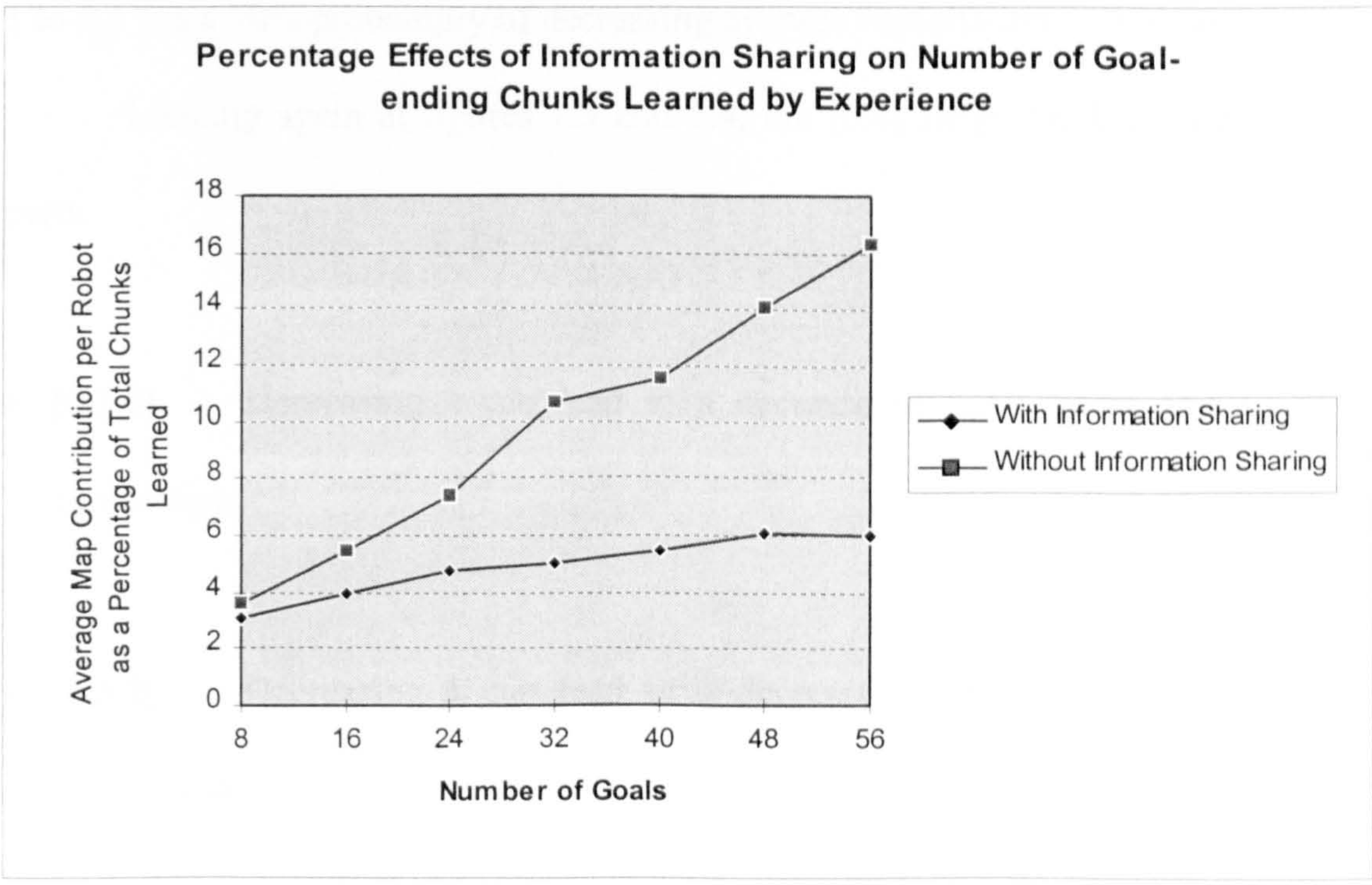

Figure 1.2 Experiment 1

4.2.1 Experiment 2 - The Effects of Altering *k*

Once again the experiments were run with a 15 robot team, each robot being given 56 goals. Each robot's 56 goals were the first 14 letters of the alphabet in red, green, yellow and blue. Experiments were run for values of k between 0 and 4, moving up in increments of 0.5. The effects on average map-contribution per robot are shown in figure 1.3 with 90% error bars.

It can be seen that for values of *k* less than about 1.3 there will be an increase in the average map-contribution, indicating and increase in average exploration. Decreasing *k* from just under 1.5 to 0.5 leads to a rise in this increase, as does decreasing *k* from 1 to 0. However changing *k* just in the range 1 to 4 will not lead to any significant change.

The effects of *k* on mission time are shown in figure 1.4 with 90% error bars. It can be seen in figure 1.4 that for values of *k* less than about 1.9 there will be an increase in average mission time, indicating a decrease in average exploitation. Decreasing *k* from 2 to 1 to 0, and from 1.5 to 0.5, decreases average exploitation at each decrease. However, just adjusting *k* in the range 4 to 1.5 has no significant effect, but decreasing k from 3 to 1.5 to 1 to 0.5 has a 70% probability of decreasing average exploitation each time.

Looking again at figures 1.3 and 1.4, the [0,4] range for *k* can be divided into 3 parts:

- **[0,1.5]** Decreasing *k* can lead to a decrease in exploitation and an increase in exploration.

- **[1.5,2]** Decreasing *k* can lead to a decrease in exploitation but no increase in exploration.

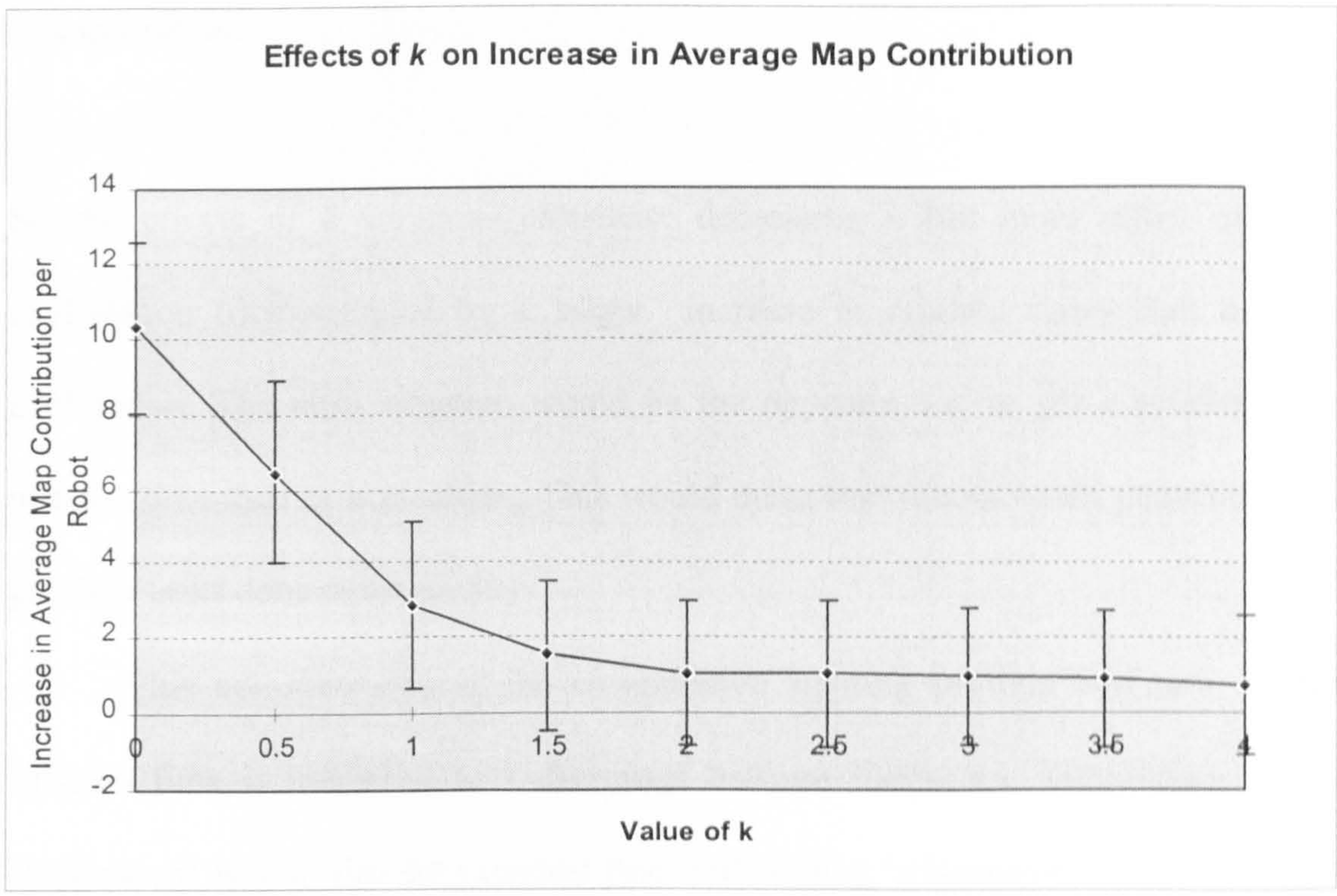

Figure 1.3 Experiment 2

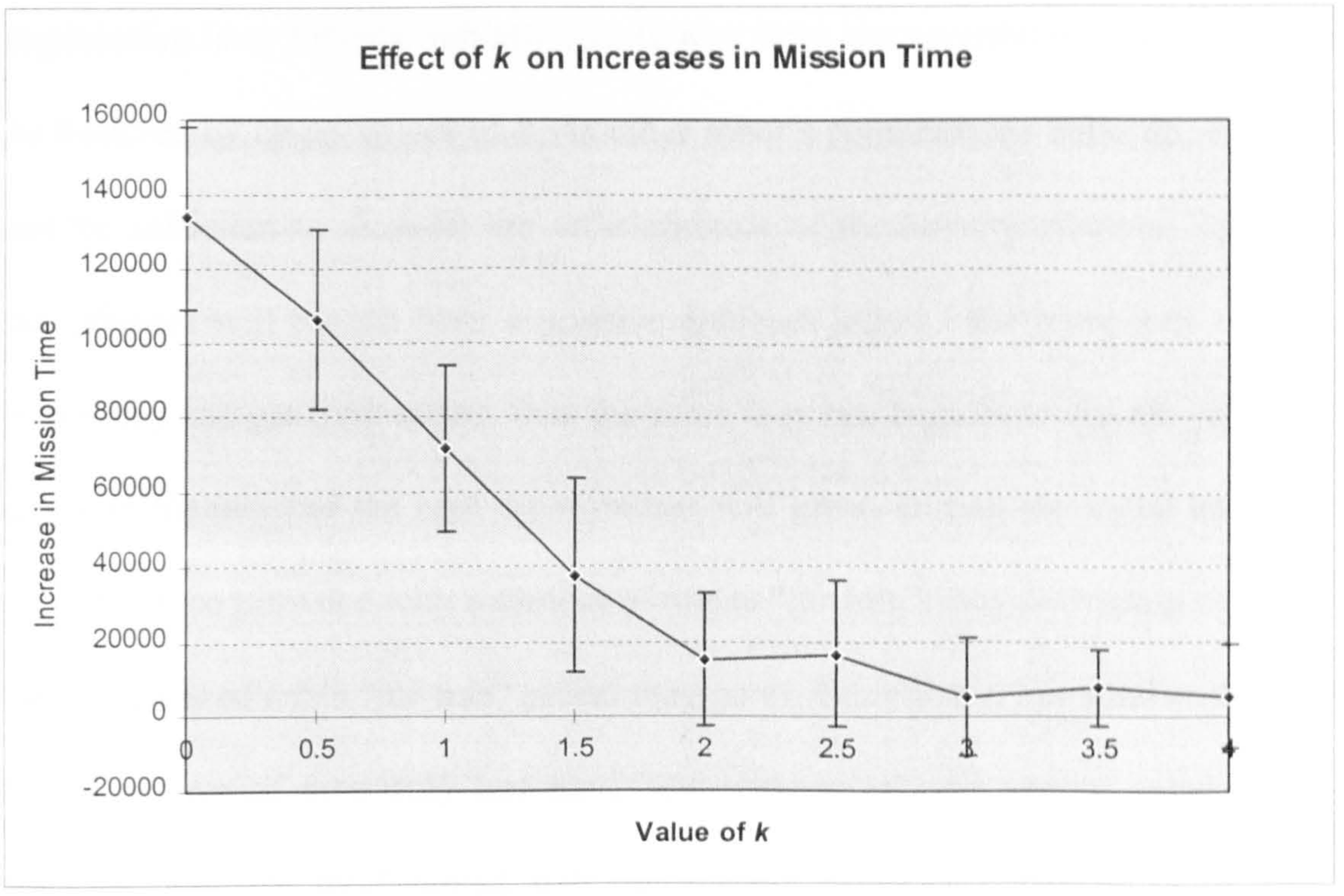

Figure 1.4 Experiment 2

- **[2,4]** Decreasing k leads to no decrease in exploitation and no increase in exploration.

So the effects of k are non-symmetric: decreasing k has more effect on decreasing exploitation (demonstrated by a larger increase in mission time) than on increasing exploration. The ideal situation would be the opposite, i.e. to get a smaller increase in mission time than in exploration. This would mean that robots would contribute more, but get their tasks done more quickly.

This non-symmetry of the co-operative learning solution will now be analysed. Mission time is increased as k decreases because robots are withholding information. However, it would also be expected that withholding information would force robots to learn more by experience, and hence increase map-contribution. For some reason, when k is in [1.5,2] this is not true - decreasing k leads to robots being given information in such a way that their mission time is increased while their map-contribution is not. A possible explanation is as follows. Initially robots who have not contributed much to the team can be buffered by larger values of k. As other robot's contributions build up, this buffer may not be sufficient to disguise the unhelpfulness of the low-contributors. Also the higher contributors will benefit from a positive feedback effect - the more they contribute, the more they can get from others, thus the more they can contribute. So the gap between the low contributors and the high contributors will grow. During the initial buffering period robots will be provided with a number of routes "for free", thus decreasing exploration. For larger values of k this "for free" period may go on indefinitely. For smaller values of k this "for free" period may only last for a few requests. So for certain values of k "in the middle", the "for free" period will occur for a significant number of requests at the beginning of the mission. During the initial part of the mission, robots will not tend to have subgoals, and will hence be making requests for the *initially given* , or auto-rooted, goals.

So the routes received "for free" will tend to be routes to the "main" goals (these routes will begin with future subgoals). Once the "for free" period is over, robots will be spending much of their time searching for and requesting for routes to these subgoals, and to these subgoals' subgoals, the number of which can grow exponentially (since if a goal generates n subgoals, this can lead to 2^n subsubgoals). The initial free period has little effect on the success of robots' request for *sub*goals. Hence a large chunk of the mission time will remain unchanged by the "free-lunch" effect, since it is spent looking for *sub*goals. However, although the robots are learning alot of *sub*goal-ending routes by experience, many of their (main)goal-ending routes will have been learned during the free period *from other robots*. And since the map-contribution is affected mainly by "main" or auto-rooted goal-ending routes *learned by experience*, there will be less increase in map-contribution than in mission time, and hence less increase in exploration than decrease in exploitation. If this analysis is true then it may be that the asymmetry is a property of the environment learning algorithm rather than co-operative learning. Further experiments, involving the measuring the ratio of subgoals to goals, would need to be done to actually test this analysis.

The above discusses the idea of an initial "for free" period for larger values of k. There will actually be a short "for free" period for *all* non-negative values of k. This is due to the initial 0 thresholds of the robots, caused by the fact that all robots will initially have received no routes. When a robot receives its first request, it will be willing to help since, even though its evaluation of the requesting robot will be 0, its threshold is also 0. The robot will continue being automatically helpful until one of its requests is successfully answered. Then its threshold will go above 0 and its "for free" period may be over (depending on the value of k). If k=0 the "for free" period will end as soon as it receives it first route from another robot. But it is interesting to note that this for free period does exist

for k=0. So k=0 does not indicate 0 exploitation. But 0 exploitation could be achieved by any negative value of k.

The 0 threshold effect highlights an implication for low values of k. If k=0.5 and robot 1 has had 8 chunks given to it by separate robots during the initial "for free" period, then its threshold will be 8/14=0.57. However, if any of those 8 robots request a route from robot 1 now, they will not get it since k*1=0.5<0.57. They have effectively "priced themselves out of the market". There are a number of factors here: robot population, robots contributing, and size of k. The maximum number of routes a robot can be given during a "for free" period to avoid this effect is k*(population-1). So if it is required that a robot can receive a route from every robot in the population during the "for free" period , then the value of k must be 1. If it is only required that a third of the population can do this then k=0.33. Thus low values of k can reduce robot interaction in more ways than originally intended.

Further on from this, if when k=0.5 and robot 1 has had 8 chunks given to it by separate robots during the initial "for free" period, and is therefore refusing requests from the 8 contributing robots, these 8 contributing robots will not be developing a true model of robot 1. This is because, even if robot 1 is increasing its map-contribution, the 8 robots will not know since robot 1 will not help them. This highlights an effect of co-operative learning in developing peer models: robot's models of each other only develop through successful interaction. The lower k is, the harder it will be for robots to develop local models of each others' usefulness, since the less they will successfully interact. So decreasing k has two effects: it increases the size of the map contribution needed to get help from the team, and it makes the teams opinions of each other lower and less accurate. The lowness of the opinions will be offset somewhat by the fact that the threshold may be lower as a result. But the inaccuracy of opinions can lead to some robots be overvalued

relative to others, and thus pushing up the threshold and decreasing general interaction. This could contribute to the increasing slope as k decreases.

This extra decrease in exploitation due to inaccurate models is similar to robots getting stuck in a non-optimal payoff D in the Prisoner's Dilemma, and also to the problem in [Sen 1996] of how to get selfish robots co-operating when they have insufficient information about each other. [Axelrod 1984] overcomes this problem using a TIT-FOR-TAT strategy. [Sen 1996] argues that the TIT-FOR-TAT strategy is over-simplistic for multi-robot systems, and his more flexible utility measuring reciprocal algorithm uses a probabilistic co-operation decision to allow robots to develop models of each other, even when they have not been useful to each other in the past. Perhaps such a probabilistic decision mechanism could be used for the co-operative learning in this thesis to temper the rate of decrease in exploitation and to reduce the 0 threshold "for free" period. So in the same way that robots use random searching to explore the non-peer environment, they could use a form of random search in exploring the peer environment more fully, and perhaps reduce the standard deviations of the results above through a more consistent and thorough search.

Sen's algorithm also has parameters for altering the probability distribution of the random search. These, like k, can be thought of as "co-operativity" parameters. The higher they are, the greater the chance of interaction; though of course Sen is not looking at reducing interaction for the sake of learning, since his robots only learn about each other, and the co-operation is *physical* (i.e. robot 1 does a task for robot 2 that it can do more easily than robot 2). Sen is using co-operation to maximise the advantages of the multi-robot scenario (parallel and more efficient achievement of goals) which avoiding the disadvantages (robots not contributing to the team achievement of goals). This is what this thesis is attempting to address, but in the context of multi-robot *learning*.

The is one final factor related to local approximations which may increase exploration. Robot 1 can only develop a model of robot 2 as long as robot 1 is making requests from robot 2. As more robots in the team achieve their goals, they will make less requests from other robots, thus towards the end of the mission time there will be robots who are unable to be seen as "contributing enough routes to the team" simply because a large part of the team does not need any more routes. Hence the stationary members of the team will be less inclined to help those robots still moving, despite the fact that, in a group with common goals, the stationary members of the team may be those with the most relevant knowledge (since they have found all the goals). So the robots left over will have their search time slowed down inordinately by this "I'm alright Jack" syndrome. If robot 2 was judging requests from robot 1 using robot 1's *actual* count of map contribution, then even when robot 2 was stationary, its opinion of robot 2 could still be increasing. But in the local approximation of this count, if exploration is increased by 1 route (i.e. robot 1 on average has to provide 1 more goal-ending chunk learned by experience to the other 14 robots) then if half of those robots stop moving before robot 1 has provided them with 1 extra chunk, then robot 1 has only half the team left to persuade to help him in his goal.

It should be emphasised that all of the above analyses are hypothetical, and that further experiments would need to be done to confirm them.

4.2.2 Summary of the Effects of *k* on Co-operative Learning

In terms of exploration, for values of k less than about 1.3 there will be an increase in average exploration, and decreasing k from just under 1.5 to 0.5 leads to a rise in this increase, as does decreasing k from 1 to 0. In terms of exploitation, for values of k less than about 1.9 there will be a decrease in average exploitation - decreasing k from 1.9 to 1 to 0, and from 1.5 to 0.5, decreases average exploitation at each decrease.

For *k* in the [0,4] range, *k* can be divided into 3 parts: [0,1.5], [1.5,2], and [2,4] in which both exploration and exploitation change, only exploitation changes, and neither change, respectively. So the effects of *k* are non-symmetric: it has more effect on exploitation than on exploration. This may be because for *k* in the range [1.5,2], robots' "for free" time provided by *k* only gives them a chance to gain help about auto-rooted goals as opposed to subgoals. A similar "for free" period occurs initially for all non-negative values of *k* due to robots' initial 0 thresholds.

Possible effects of lower *k* are robots pricing themselves out of market, and an increase in the inaccuracy of deterministic co-operative learning in developing peer models. This is related to the problem in [Sen 1996] and suggests the possibility of using Sen's stochastic learning algorithm to make robot's models of each other more accurate for lower *k*, and possibly make the co-operative learning more stable. Another effect of deterministic co-operative learning is the "I'm alright Jack" syndrome when robots have finished their tasks.

4.2.3 Experiment 3 - The Effect of Number of Goals on Co-operative Learning

In the following experiments, the regions [0,1.5], [1.5,2], [2,4] will be represented by the values of *k* of 0.5, 2 and 3.5.

To test the effect of the different numbers of goals, the number of goals per robot was altered between 8 and 56 (going up in increments of 8). Each robot's *n* goals where the first *n* letters of the alphabet in red, green, yellow and blue. Thus the robots had common goals. Experiments 3a, 3b and 3c - where k=0.5, k=2 and k=3.5 respectively - examine the effect of the number of goals. These results for increases in average map-contribution are shown in figures 1.5, 1.6 and 1.7 respectively, with 90% error bars.

It can be seen that only Figure 1.5 indicates any significant increases in goal-ending routes learned by experience, i.e. when k=0.5. Co-operative learning causes an increase in average map-contribution for 16 goals and above. Also an increase in goals from 8 to 32 and 16 to 56 will lead to an increase in the effects of co-operative learning on average map-contributions. There is, however, no significant change in average map-contributions as goals increase for k=2 and k=3.5. Neither is there any significant increase in map-contributions due to co-operative learning when k=2 and k=3.5.

So why does an increase in goals lead to an increase in effects of co-operative learning when k=0.5? In chapter 3, when developing the local approximation ideas, it was observed that the approximation of the count of a robot's map-contribution depended on a significant amount of interaction due to common goals, with a greater amount of interaction improving the accuracy of the approximation. More common goals means more interaction and hence an improvement in the accuracy of the approximations.

Another reason could be related to the initial 0 thresholds of robots. Given a system

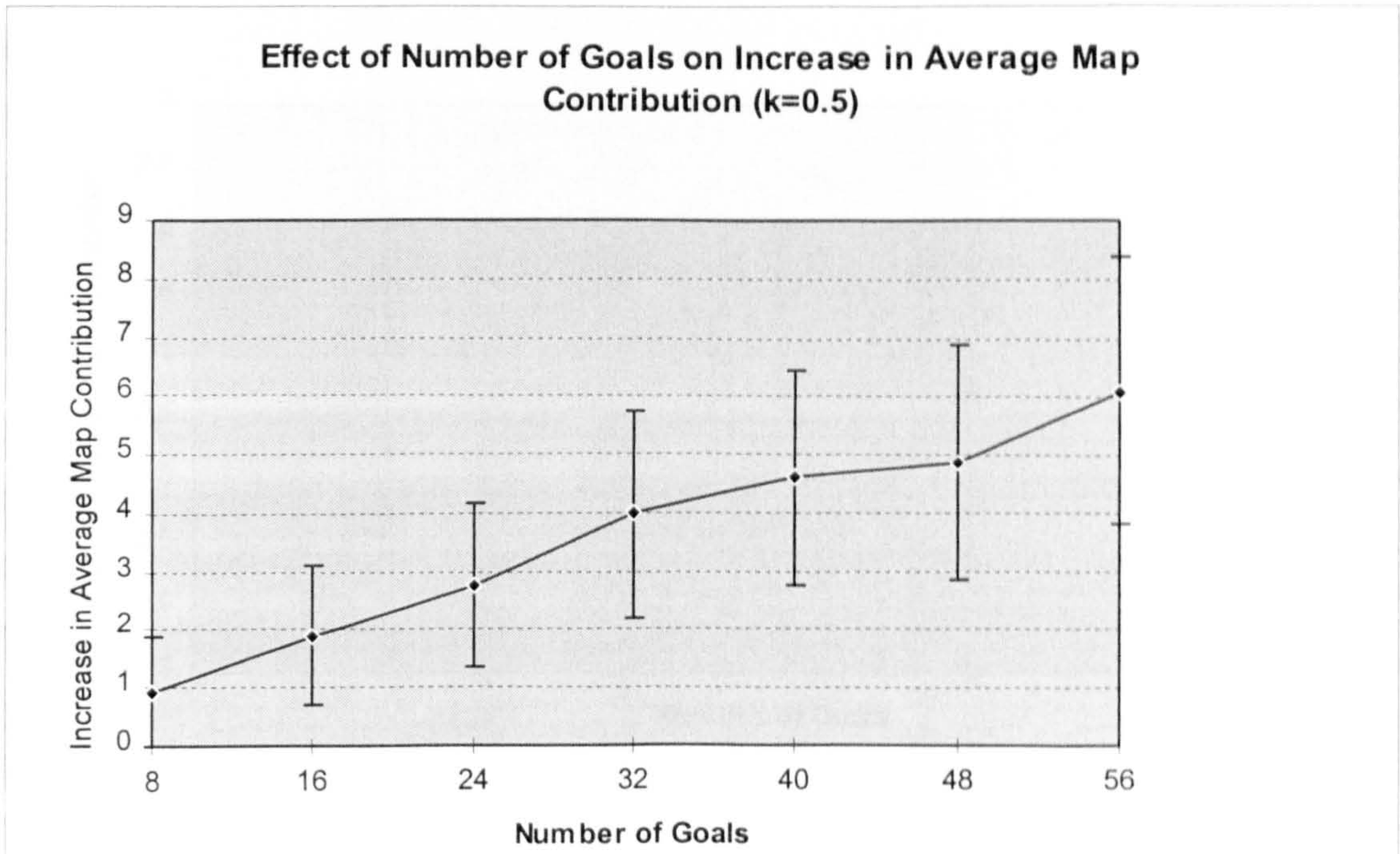

Figure 1.5 Experiment 3a

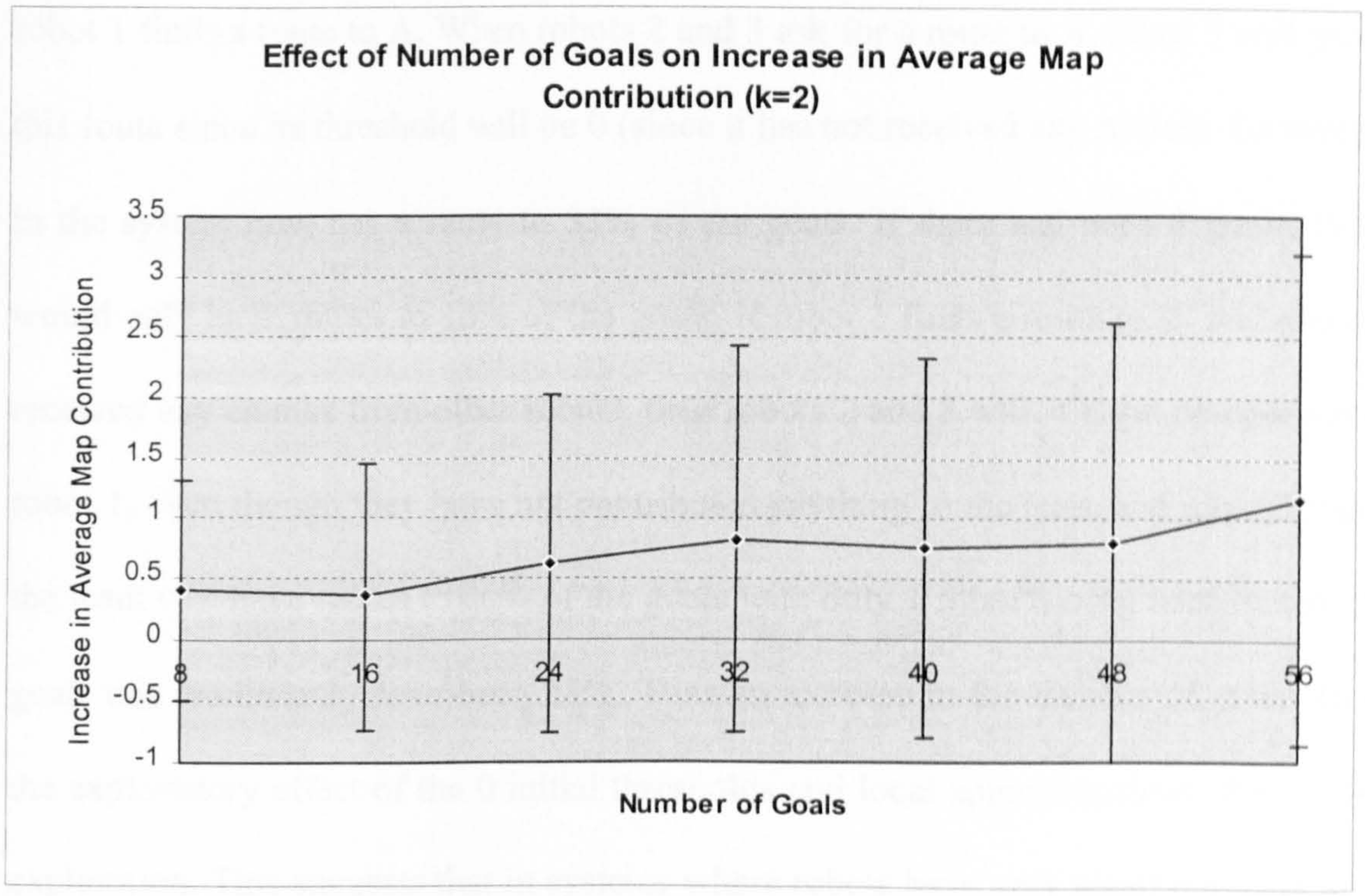

Figure 1.6 Experiment 3b

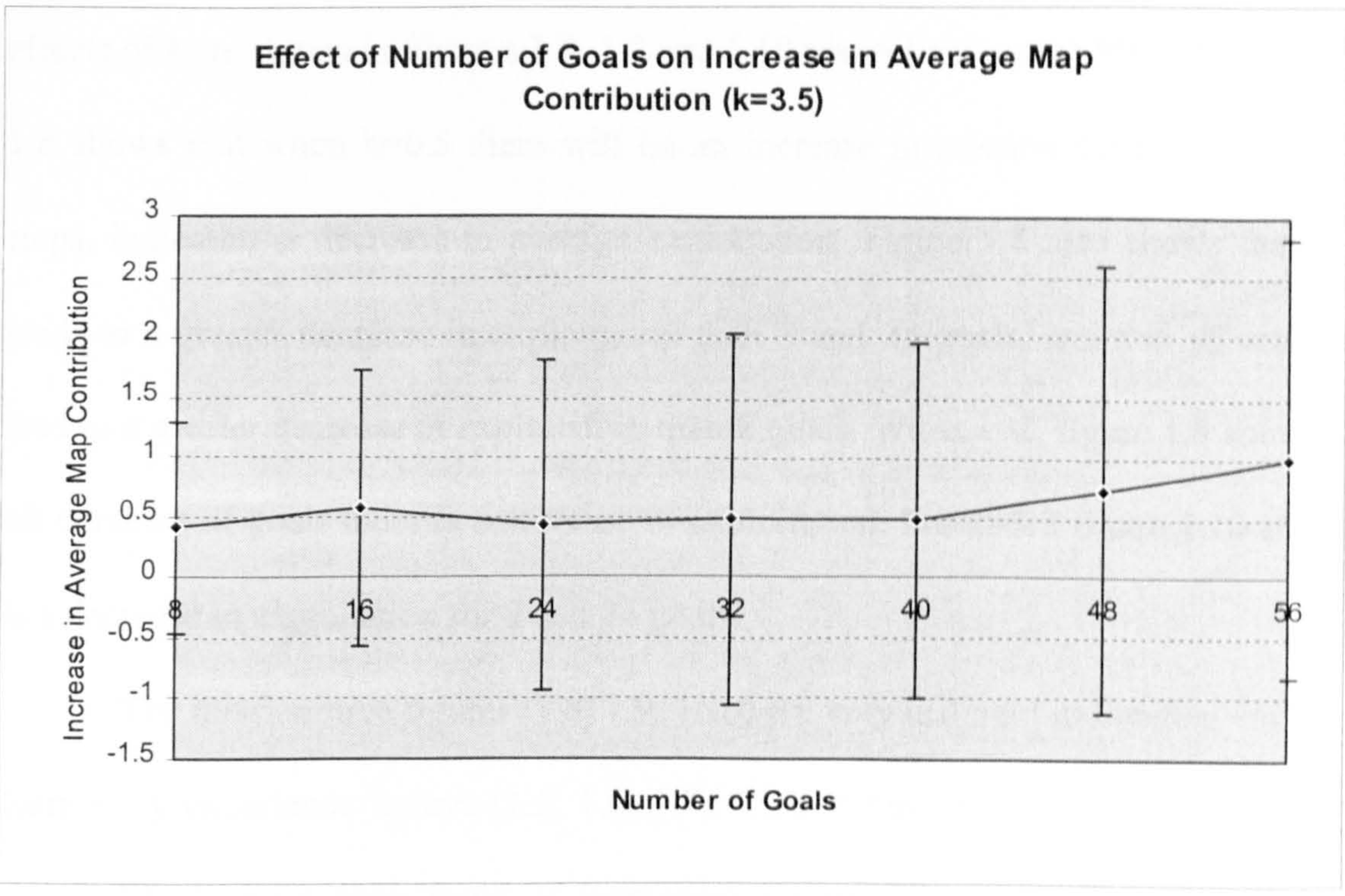

Figure 1.7 Experiment 3c

of 3 robots, each with goals (A,B,C), suppose the robots have not interacted before and robot 1 finds a route to A. When robots 2 and 3 ask for a route to A, robot 3 will give them this route since its threshold will be 0 (since it has not received any routes). So every robot in the system now has a route to 33% of the goals. If there had been 8 goals, then they would only have routes to 13% of the goals. If robot 1 finds a route to B, and still has not received any chunks from other robots, then robots 2 and 3 will still get co-operation from robot 1, even though they have not contributed anything to the team and robot 1 has. Thus the team will have routes to 66% of the goals with only 1 robot having contributed. With 8 goals this would only have been 25%. Thus an increase in the number of goals decreases the exploitatory effect of the 0 initial thresholds and local approximations, thus increasing exploration. This suggests that in systems where robots have very small numbers of goals, it may be necessary to use the full global algorithm described in chapter 3 rather than the local approximation.

Moving now to the effects of the number of goals on the *mission-time* increasing effects of *k* are shown in Figures 1.8, 1.9 and 1.10 respectively, with 90% error bars. Figure 1.8 shows that when k=0.5 there will be an increase in mission time of at least 35,000 steps, indicating a decrease in average exploitation. Figure 1.8 also shows that 32 goals leads to a greater decrease in exploitation than 8 and 24 goals, and that 48 and 56 goals lead to a greater decrease in exploitation than 8 goals. When *k*=2, figure 1.9 shows that for all numbers of goals there is a decrease in exploitation. For k=3.5 figure 1.10 shows there is a decrease in exploitation for 8 and 24 goals.

The mission time figures (1.8, 1.9, 1.10) are very different to the goal-ending routes learned by experience figures (1.5, 1.6, 1.7). This is most noticeable when k=0.5, where figure 1.5 shows a clear trend of increasing ranges in number of goal-ending chunks learned by experience, whereas in figure 1.8 no such trend is detectable.

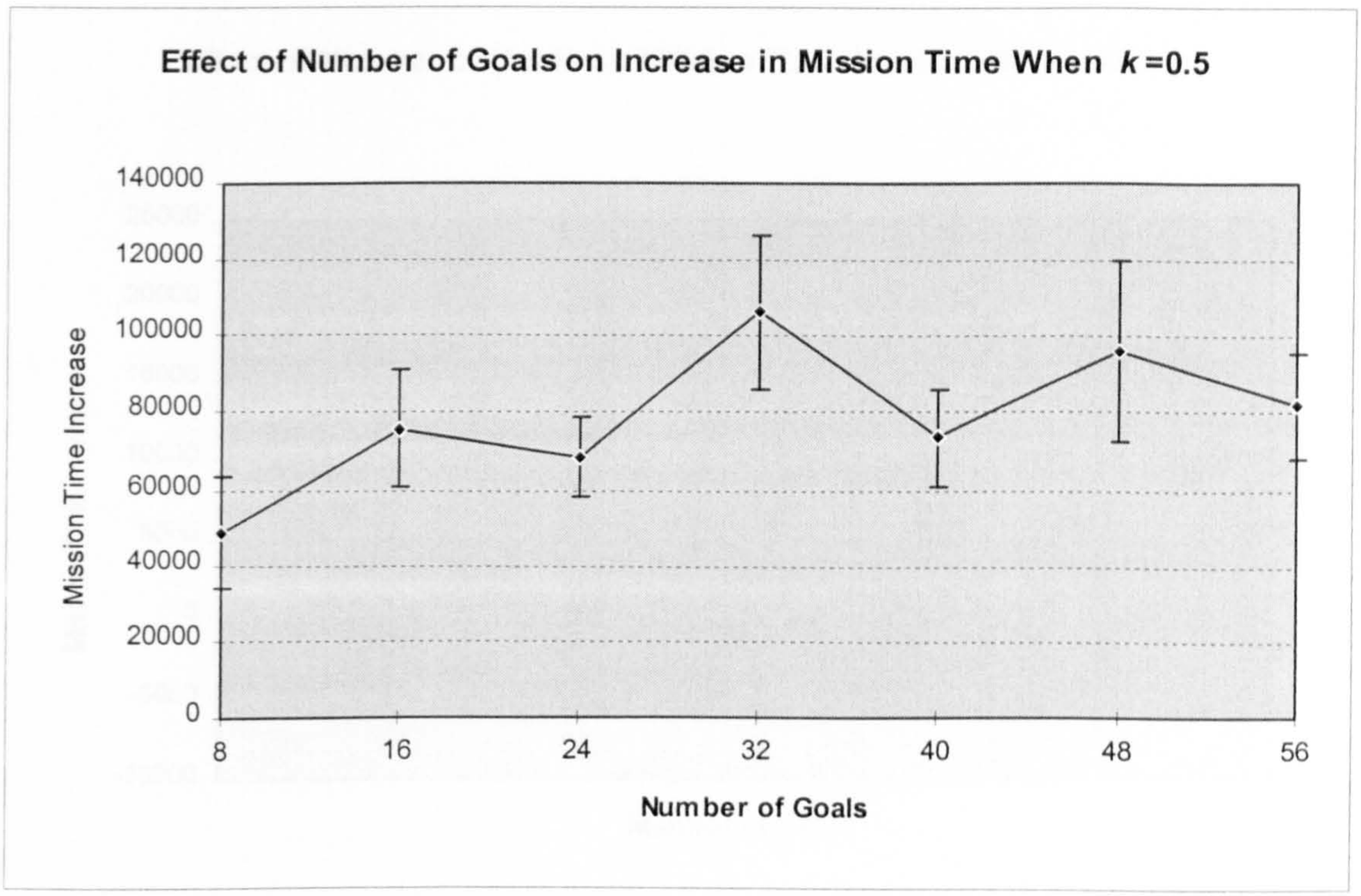

Figure 1.8 Experiment 3a

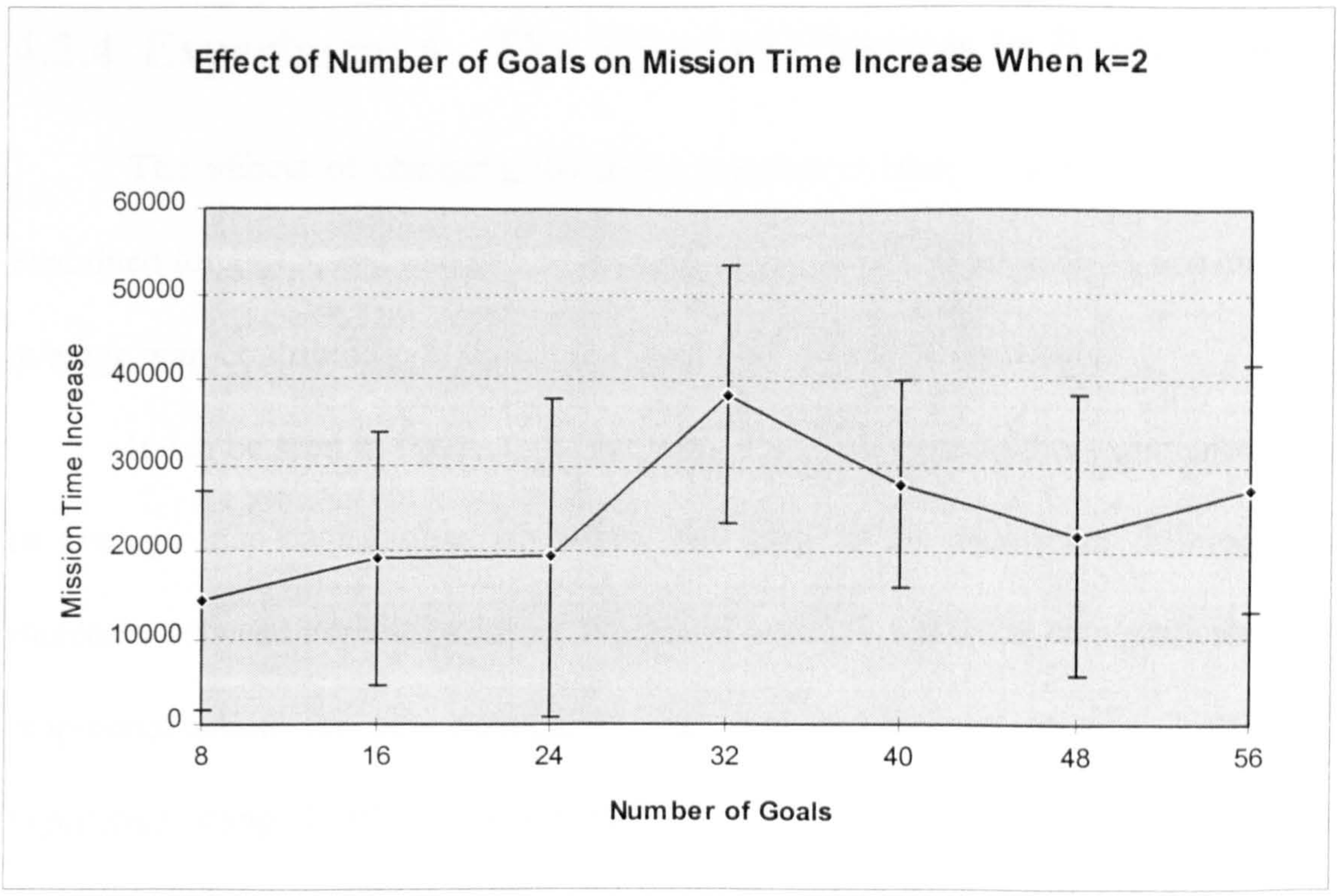

Figure 1.9 Experiment 3b

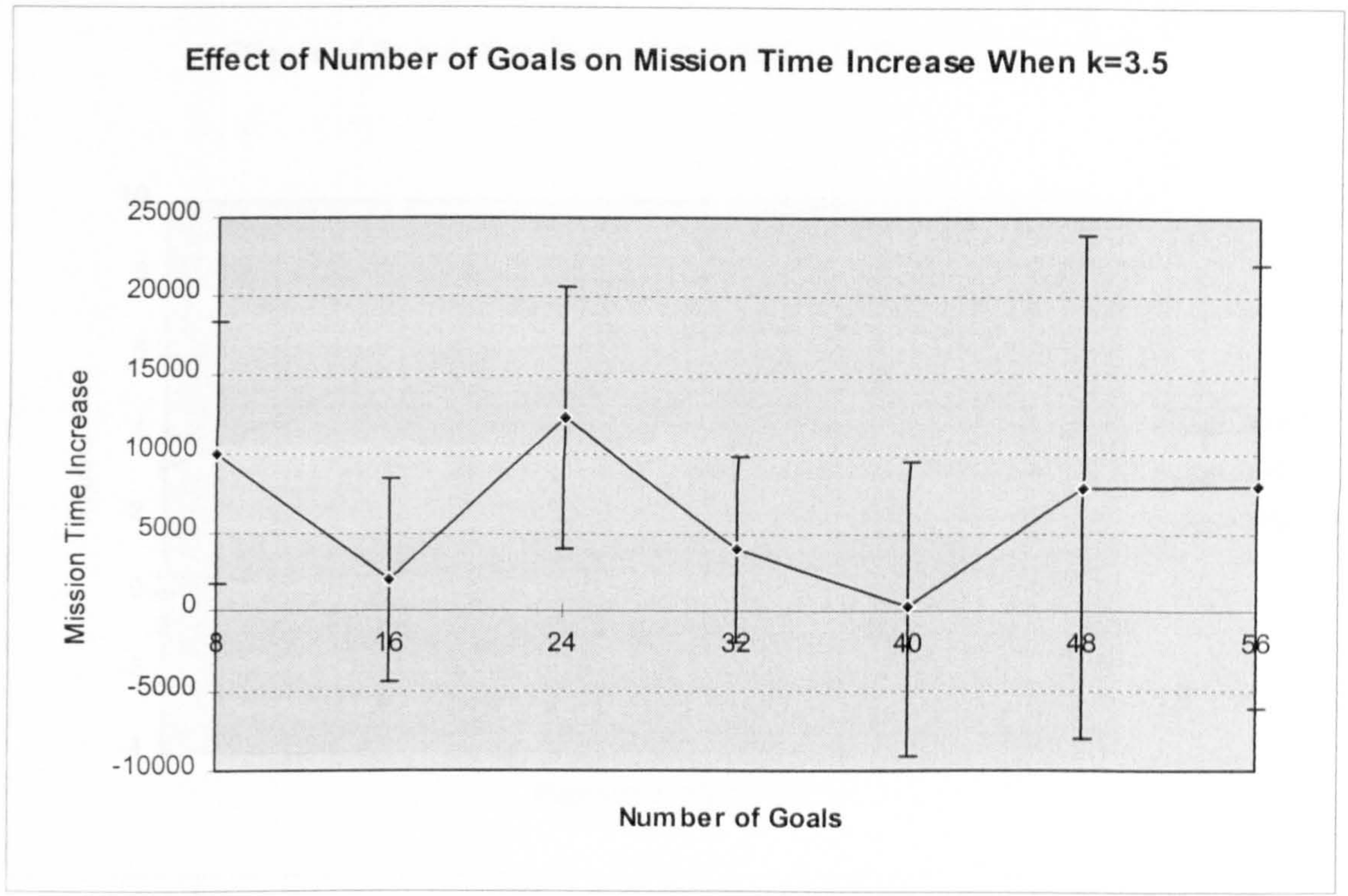

Figure 1.10 Experiment 3c

4.2.4 Experiment 4 - The Effect of Changes in Population Size

The effects of changing the robot population from 3 to 15 (in steps of 3) were examined for k=0.5, k=2 and k=3.5. A graph of the effects of population size on increase in average map-contribution is shown in figure 1.11 with 90% error bars.

It can be seen in figure 1.11 that when k=0.5 all populations experience an increase in average map-contribution per robot, but there is no significant difference in these increases between population sizes. When k=2 and k=3.5 there is no significant increase in map-contribution for any population size, and populations of size 3 experience no significant change in effects for different values of k.

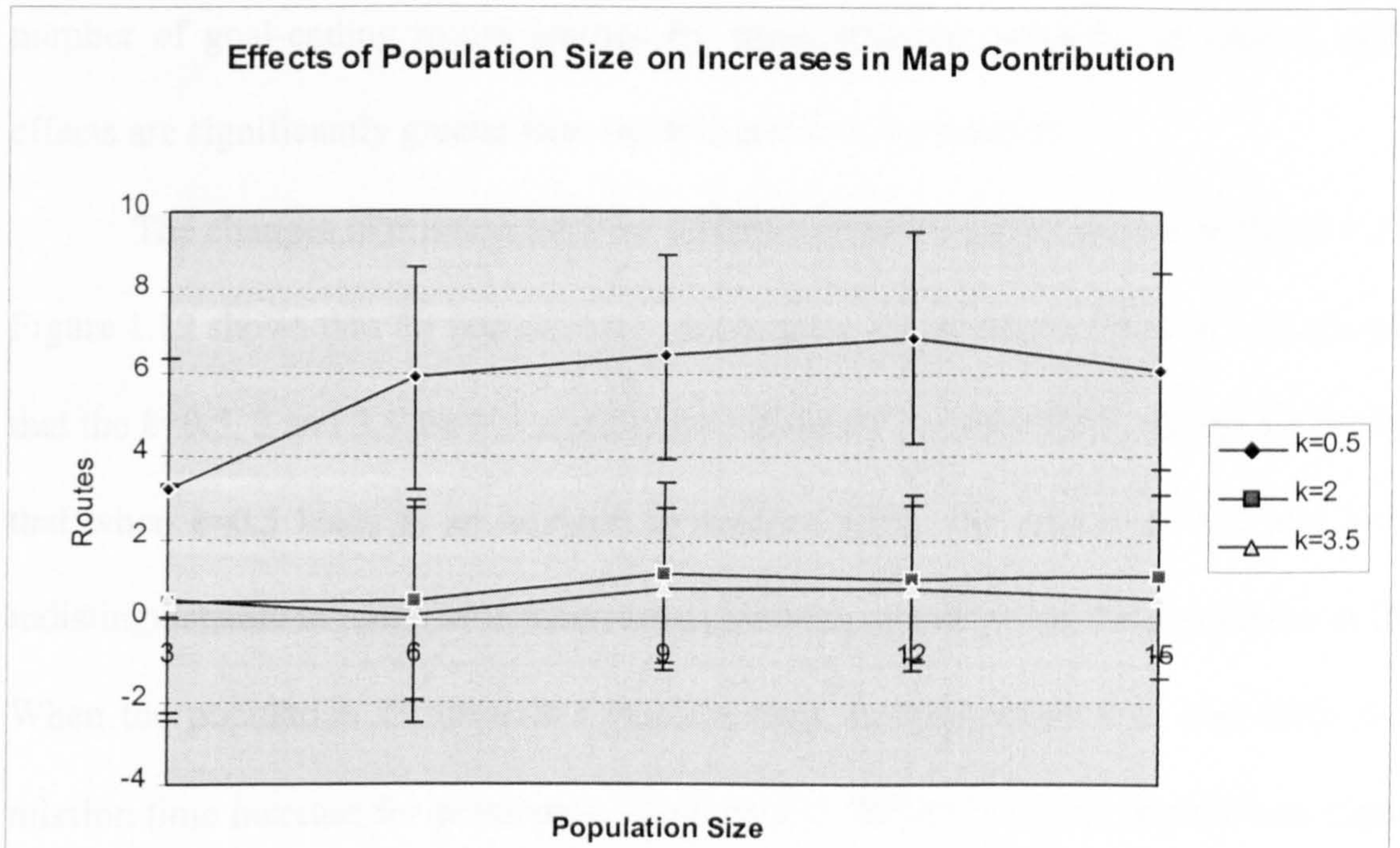

Figure 1.11 Experiment 4

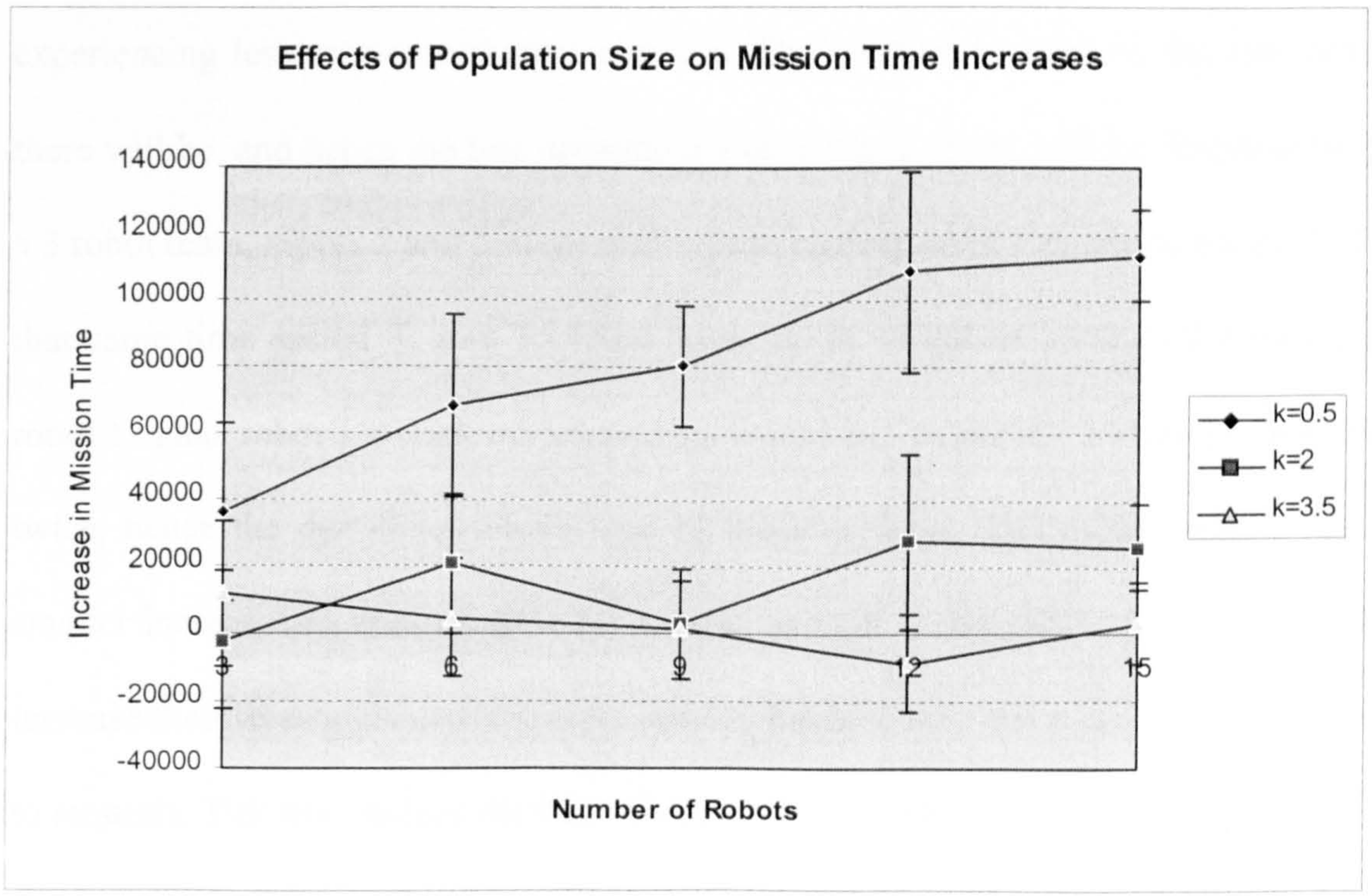

Figure 1.12 Experiment 4

For populations 6 and upwards, co-operative learning with k=0.5 increases the number of goal-ending routes learned by experience per robot by at least 3, and these effects are significantly greater than the k=2 and k=3.5 scenarios.

The changes in mission time for different populations are shown in figure 1.12. Figure 1.12 shows that for population 3, once again, the average effects of k on are such that the k=0.5, 2 and 3.5 are not significantly different in their effect, except for the fact that when k=0.5 leads to an increase in mission time. The effects of k=2 and k=3.5 are indistinguishable in terms of mission time increases, except when the population is 15. When the population 15 there is a mission time increase when k=2, and there is also a mission time increase for population 12 when k=2. When k=0.5 all populations experience an increase in mission time due to co-operative learning, with further increases in mission time moving from populations 3 to 9 to 15.

The reasons why lower populations experience less significant increase in mission time may be related to the reasons given for populations with lower numbers of goals experiencing less map-contribution increase. The smaller the team is, the less interaction there will be, and hence the less accurate the distributed model will be. Suppose that given a 3 robot team, robots 2 and 3 make one request each of robot 1 in a time period T. Then in that same time period T, in a 15 robot team, up to 14 robots could make requests from robot 1. Thus robot 1's (lack of) knowledge would be "explored" 14 times rather than just twice, hence the distributed model can be more accurate. The other possible reason for smaller increases in mission time for smaller groups is that larger groups of robots will increase their thresholds above 0 more quickly because they are more likely to get answers to requests. This may reduce the 0 threshold over-generosity.

As was mentioned at the end of experiment 2, it should be emphasised that the analyses for experiments 3 and 4 are hypothetical, and that further experiments would need to be done to confirm them.

4.3 Contrasting the Effects of Goals and Population

When k=0.5, increasing number of goals shows a clear trend through the increase in average map-contribution (Figure 1.5), but a less clear trend in mission time (Figure 1.8). Conversely, increasing the population shows a clear trend for increase in mission time (Figure 1.12), but no clear trend for the increase in average map contribution (Figure 1.11). So population increases the effects of low *k* on mission time, and goals increase the effects of *k* on map contribution.

This raises the issue of efficiency. Without co-operative learning, the system would be working at maximum exploitation (infinite *k*). The effect of co-operative learning is to allow a reduction in exploitation, giving a consequent increase in exploration. Reducing exploitation increases mission time, and usually the less time a mission takes the better. So an efficient system will get as much exploration as possible for the smallest reduction in exploitation possible. As population increases there are significant increases in mission time, without any significant increases in exploration. Hence co-operative learning is less efficient for larger populations. Furthermore, as the number of goals increases, co-operative learning gives more exploration with little significant increase in mission time. Thus co-operative learning is most efficient for higher numbers of common goals. So the ideal MRS for co-operative learning would involve smaller numbers of robots, with a large number of common goals. However, this is not to say that the system does not work for larger populations, or smaller goal numbers, as it clearly does. This is just an observation of which scenarios use co-operative learning most efficiently.

4.4 Summary of the Effects of Goals and Population Size on Co-operative Learning

When k=0.5, co-operative learning causes an increase in exploration for 16 goals and above, and a decrease in exploitation (leading to an increase in mission time of at least 35,000 steps for all numbers of goals). An increase in goals from 8 to 32 and 16 to 56 will lead to an increase in exploration, and 32, 48 and 56 goals lead to a greater decrease in exploitation than 8 goals. Also 32 goals leads to a greater decrease in exploitation than 24 goals.

When k=2, all numbers of goals experience a decrease in exploitation, but there is no significant change in exploration, nor a trend in exploration as goal numbers increase. For k=3.5 there is a decrease in exploitation for 8 and 24 goals. But once again there is no significant change in exploration, nor any trend as goal numbers increase.

So greater numbers of goals increase exploration - possibly because more common goals leads to more interactions and therefore better models, and because larger numbers of goals lessen the effects of the initial 0 threshold, though these ideas would need to be tested in further experiments.

Moving on to the effects of population size, when k=0.5 all populations experience an increase in exploration, but there is no significant difference in these increases between population sizes. Similarly all populations experience a decrease in exploitation when k=0.5, but with a further decrease in exploitation moving from populations 3 to 12 and 9 to 15. For k=2 and k=3.5 there is no significant increase in exploration for any population size, and populations of size 3 experience no significant change in effects for different values of k. Population 3 also has no significant difference it its effects on exploitation for different values of k, except that when k=0.5 there is a decrease in exploitation. For all populations, co-operative learning with k=0.5 increases the exploration. For k=0.5 with

populations greater than or equal to 6 average map-contribution per robot increases by at least 3, and these effects are significantly greater than the k=2 and k=3.5 scenarios. The effects of k=2 and k=3.5 are indistinguishable in terms of exploitation decreases, except for population 15, and for the fact that when the population is 12 and 15 there is an exploitation decrease when k=2. So low populations experience less significant decreases in exploitation, possibly for reasons similar to those which make populations with smaller numbers of goals experience smaller increases in exploitation. That is, because of less accurate models and a greater impact of the initial 0 threshold, though once again these ideas would need to be tested in continued experiments.

In conclusion, it has been shown that map sharing between robots leads to a reduction in exploration. It has been further shown that co-operative learning allows an increase in, and the setting of, average levels of exploration for a wide range of populations and goals. In practice, looking at figures 1.3 and 1.4, possible settings for k could be:

- k=0.2 Very High Exploration
- k=0.5 High Exploration
- k=1.3 Medium Exploration
- k=3 Low Exploration

These settings are suggested for populations of 15 robots with 56 common goals each. For smaller populations of robots and smaller numbers of goals, the value of k needs to be decreased.

It should be noted that reducing k has more effect on exploitation than exploration thus reducing efficiency. It is also worth noting that co-operative learning is most efficient for smaller numbers of robots with large numbers of common goals.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Multi-robot systems are an extremely useful tool, in particular systems of multiple mobile robots. In the dynamic and complex environment of the real world, these systems can develop and adapt their own models through goal-driven learning. In particular, mobile robots need to have dynamic *route* models. If robots in the team are able to exchange their information, then the speed at which the environment is learned and tasks achieved will be greatly increased. However, it will also lead to the team "locking in" to one version of a route learned by one robot, and not looking for shorter versions. One contribution of this thesis has been to observe, and show the existence of through experimentation, the *Distributed* Exploration vs. Exploitation dilemma, the dilemma of choosing between fast learning, and avoidance of lock-in. This dilemma has not been observed in any other work on multi-robot systems.

The operator of a multi-robot system may wish to increase learning time to encourage exploration so robots experiment with possibly shorter routes to goals. A major aim of this thesis, in addressing the distributed exploration vs. exploitation dilemma, has been to provide a simple method for increasing and decreasing exploration in such circumstances; a method which could run in parallel with the robots normal learning/exchanging/goal-achieving algorithms. It is clear that a controlled reduction of information exchange would achieve this, and it was observed that using the average map-contribution across all robots as a decision threshold as to whether robots interact or not, was a possible control. The major contribution of this thesis came from observing that this

average map-contribution threshold could be approximated locally using Co-operative Learning with a multiplying factor *k*. However, the comparative effects of the local approximation being used, as opposed to the global version discussed at the beginning of chapter 3, were not examined.

A simulation of 15 robots was used to demonstrate that decreasing and increasing *k* within certain ranges would increase and decrease exploration, though not as efficiently as hoped since decreasing *k* decreased exploitation more than it increased exploration. It was further shown that these effects held true for a wide range of goals and populations. However, lower numbers of goals and robots required lower values of *k*, and co-operative learning was shown to be more efficient in cases of lower populations and higher numbers of goals.

The main idea behind this work is that teams of robots can help their owners both by helping each other, and by not helping each other. Help too little and you waste the point of a team, help too much and you waste the powers of an individual. This point has only been addressed in detail in multi-robot systems in [Sen 1996], and Sen does not address its effects on task learning (only on task assignment), nor does he observe that it results in a special case of the exploration vs. exploitation dilemma. This thesis has done both, and demonstrated a concrete and simple method which can be used to set the balance between team and individual.

5.2 Future Work

The key contribution of this thesis was the been the use of a local approximation to model a robot's performance in a group. Therefore an important possible piece of further work would be to look at the effects that the use of a local approximation had on the results. Experiments 2, 3 and 4 in chapter 4 could be run using the full global algorithm described

at the beginning of chapter 3. This would quantify the actual effect of the local approximation. It was suggested after experiment 3 in chapter 4 that the effect of number of goals and population size on distributed exploration and exploitation was due partly to the local approximation being used instead of a global one. The experiments suggested above would help to examine these ideas.

A number of other areas of possible future work were raised in the discussion in Chapter 4. It was suggested that the asymmetric effect of k on exploration and exploitation was due to robots spending the majority of the mission time requesting subgoals. Experiments could be done examining the ratios of goals to subgoals for different values of k. If these experiments support the hypothesis then it may be that the asymmetry is a property of the environment learning algorithm rather than co-operative learning.

Another helpful set of experiments would be to repeat the experiments in chapter 4, but using a probabilistic decision algorithm for when robots interact. Rather than robot 1 always interacting with robot 2 if robot 2's threshold is high enough, an algorithm could be designed such that robot 1 has a probability of interacting with robot 2. The higher robot 2's approximated map-contribution, the greater the chance of robot 1 helping robot 2. This would mean that robots who have finished their goals may still have a chance of helping robot's that are still moving, even if the moving robots are of low value to the finished robots. This may reduce the "I'm alright Jack" syndrome mentioned in Chapter 4. It may also improve robots' models of each other in general, since robots are being given more of a chance to "explore" their peers. The final effect would be that even when robots have 0 threshold, they can be given a chance of refusing other robots, thus reducing the 0 threshold effects.

It would be helpful to test the effectiveness of co-operative learning with more deterministic search algorithms. These would be algorithms which have some knowledge of where goals are likely to be, or which can detect goals are a distance and move towards

them. Such algorithms would give a clearer view of the effects of co-operative learning, instead of the "blurred" view given by a totally random algorithm. It would also be interesting to see if co-operative learning can be used in non-space learning multi-agent systems, for example in concept learning systems, or in non-physical space learning systems, e.g. WWW bots. Any such multi-agent system which can benefit from encouraging independent learning amongst its agents could theoretically benefit from co-operative learning.

The distributed model of a robot across the team can actually be thought of as that robot's "utility" to the team. If the robot has a low utility, the team will not waste energy on it. This could be useful for detecting robot malfunction. If a robot has a slowed-down or broken-down motor or transmitter, then its utility will get lower and lower until the team ignores it. So an alarm system can be set up such that if a robot's average utility goes below a certain value an operator is informed of a possible malfunction. This could be made even more robust by modifying the co-operative learning system as follows. In the current system, robot 2 will increase its map-contribution count of robot 1 if robot 1 replies with a route. However, robot 2 does not distinguish between good and bad routes. If robot 2 finds that the route was incorrect, it could downgrade its route count of robot 1 appropriately. Thus if robot 1 has a damaged memory or sensory system it can be eventually detected due to other robots low utilities for it.

This thesis has concentrated on task learning. However, by viewing a route count as a utility measure, systems with co-operative learning can also use the route count for task assignment purposes. The request in the robot model used in this thesis was RGC, request for goal co-operation. Another possible request is RTC - request for task co-operation. In this case robot 1 actually requests robot 2 to take on one of its goals, say A. If robot 2 accepts then robot 1 views itself as having achieved the goal, and robot 2 views itself as having a new goal. There are 2 ways of viewing RTCs, and these are related to the criteria

of making and accepting requests. The most obvious criteria is for robot 1 only to make a request if it has a high utility for robot 2, i.e. if it thinks robot 2 is good enough to do the job. Of course, robot 2 may still not take on the job unless robot 1 has been helpful to it in the past (either by taking on jobs or by sharing information). This will prevent robots from palming off all their tasks onto other robots. So if robot 1 has a concept of its own utility, and believes that robot 2's utility is higher than its own - i.e. robot 2 can get the job done better or more easily - then it will request robot 2 does it. Robot 1's concept of its own utility may be calculated as a function of how many requests it has successfully answered compared to how many it has successfully requested.

Another way of viewing RTC is in terms of emerging hierarchies. If it is assumed that for robot 2 to accept robot 1's request requires that robot 2's route count for robot 1 is *twice* the average route count, the RTCs will only have an effect if relatively "highly-knowledgeable" robots emerge. These robots may be "highly- knowledgeable" because of their physical or cognitive abilities. They are natural leaders, and will cause a hierarchy to emerge with the most intelligent robots at the top. They can be used by the operator to distribute goals around the team. Should this robot malfunction, its relative utility will degenerate, and another robot (possibly the next most intelligent) will take its place. Thus there is a robust dynamic hierarchy.

Appendix

There now follows a proof that the 2 robot scenario in Section 2.11 in Chapter 2 fulfills the 2nd requirement for being a Prisoners' Dilemma. This proof should be read in conjunction with the equations and inequalities in sections 2.11 and 2.11.1.

The 2nd requirement for a Prisoners' Dilemma is $R>(T+S)/2$, i.e.

$$r < \frac{2}{\frac{1}{t}+\frac{1}{s}}$$

$$L - Sav + Trans < \frac{2}{\frac{1}{L-Sav}+\frac{1}{L+Trans}}$$

$$L - Sav + Trans < \frac{2}{\left(\frac{L+Trans+L-Sav}{(L-Sav)(L+Trans)}\right)}$$

$$L - Sav + Trans < \frac{(L-Sav)(L+Trans)}{2L+Trans-Sav}$$

Since $L > Sav$ we have $2L + Trans > Sav$ and therefore:

$$(L - Sav + Trans)(2L + Trans - Sav) < (L - Sav)(L + Trans)$$

Then, expanding out the brackets and collecting terms, we have:

$$2LSav + 2LTrans - Trans.Sav - 2Sav + Trans^2 < 0$$

$$Sav(2L - Trans - 2) < -(2LTrans + Trans^2)$$

Then, as long as $2L > (Trans + 2)$ we have the condition for the Prisoner's Dilemma:

$$Sav > \frac{2LTrans + Trans^2}{2L - Trans - 2}$$

If we assume $L >> Trans$ and *Trans* is small then the condition becomes:

$$Sav > \frac{2LTrans}{2L - 2}$$

$$Sav > \frac{L}{L - 1} Trans$$

Since $L >> Trans$ we will also have $Sav > Trans$. And if L is large enough, we will have $L \approx L - 1$ which will mean that this Prisoner's Dilemma condition is fulfilled. However, even if L is not large enough so that $L \approx L - 1$, if it is large enough to make $Sav >> Trans$ the condition will still be fulfilled. Thus we can see that for most practical situations, the two robot example fulfills the second requirement for the Prisoner's Dilemma.

References

(Note: where publications were obtained from the World Wide Web, the journal/proceedings reference *and* the URL, current August 1997, is provided.)

[Aguilar et al 1995]

Aguilar, L., Alami, R., Fleury, S., Herrb, M., Ingrand, F., Robert, F., "Ten Autonomous Mobile Robots (and even more) in a Route Network Like Environment", in Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, Pennsylvania, USA 1995

[ftp://ftp.laas.fr/pub/ria/felix/publis/iros95.ps.gz]

[Alami et al 1997]

Alami, R., Fleury, S., Herrb, M., Ingrand, F., Qutub, S., "Operating a Large Fleet of Mobile Robots using the Plan-Merging Paradigm", Proceedings 1997 IEEE International Conference on Robotics and Automation, Albuquerque, USA 1997

[http://www.laas.fr/~felix/publis/ieee-icra97/paper.html]

[Alami et al 1997b]

Alami, R., Fleury, S., Herrb, M., Ingrand, F., Robert, F., "Multi Robot Cooperation in the Martha Project", IEEE Robotics and Automation Magazine (Special Issue on Robotics & Automation in Europe: Projects funded by the Commission of the European Union), 1997

[http://www.laas.fr/~felix/publis/ieee-ram96/paper.html]

[Alami et al 1995]

Alami, R., Robert, F., Ingrand, F., Suzuki, S., "Multi-robot Cooperation through Incremental Plan-Merging", Proc. 1995 IEEE International Conference on Robotics and Automation

[ftp://ftp.laas.fr/pub/ria/felix/publis/ieee95.ps.gz]

[Alami et al 1995b]

Alami, R., Aguilar, L., Bullata, H., Fleury, S., Herrb, M., Ingrand, F., Khatib, M., Robert, F., "A General framework for multi-robot cooperation and its implementation on a set of three Hilare robots", Proceedings ISER 1995, Stanford, CA, USA 1995

[ftp://ftp.laas.fr/pub/ria/felix/publis/iser95.ps.gz]

[Arkin 1994]

Arkin, R.C., "Integration of Reactive and Telerobotic Control in Multi-agent Robotic Systems", in Proceedings 1994 International Conference on Simulation of Adaptive Behavior (From Animals to Animats 3), pp 473-478, 1994

[Arkin and Balch 1997]

Arkin, R.C., Balch, T., "Co-operative Multi-agent Robotic Systems", to appear in "AI-based Mobile Robots: Case Studies of Successful Robot Systems", Kortenkamp, D., Bonasso, R.P., Murphy, R., eds., AAAI Press

[ftp://ftp.cc.gatech.edu/pub/people/arkin/web-papers/coop.ps.Z]

[Arkin et al 1993]

Arkin, R.C. , Balch, T., Nitz, E., "Communication of Behavioral State in Multi-agent Retrieval Tasks", Proceedings 1993 International Conference on Robotics and Automation, pp. 588-594, 1993

[Arkin and Hobbs 1992]

Arkin, R.C., Hobbs, J.D., "Dimensions of Communication and Social Orginisation in Multi-agent Robotic Systems", in Proceedings 1992 International Conference on Simulation of Adaptive Behavior (From Animals to Animats 2), pp 486-493, MIT Press, USA 1992

[Arkin 1992]

Arkin, R.C., "Cooperation without Communication: Multi-agent Schema Based Robot Navigation", Journal of Robotic Systems, Vol. 9, No. 3, pp. 351-364, 1992

[Asada et al 1995]

Asada, M., Uchibe, E., Hosoda, K., "Agents That Learn from Other Competitive Agents", in On-line Proceedings Agents that Learn from Other Agents Workshop, http://www.cs.wisc.edu/~shavlik/ml95w1/, 1995 International Machine Learning Conference, Carnegie-Mellon University, USA 1995
[ftp://ftp.cs.wisc.edu/machine-learning/shavlik-group/ml95w1/asada.ps.gz]

[Asama et al 1996]

Asama, H., Fukuda, T., Arai, T., Endo, I., eds, "Distributed Autonomous Robotic Systems 2", Springer-Verlag, Tokyo 1996

[Asama et al 1994]

Asama, H., Fukuda, T., Arai, T., Endo, I., eds, "Distributed Autonomous Robotic Systems", Springer Verlag, Tokyo 1994

[Asama et al 1992]

Asama, H., Ozaki, K., Matsumoto, A., Ishida, Y., Endo, I., "Development of Task Assignment System Using Communication for Multiple Autonomous Robots", Journal of Robotics and Mechatronics, Vol.4, No. 2, pp 122-127, 1992

[Asama et al 1991]

Asama, H., Ozaki, K., Ishida, Y., Habib, M.K., Matsumoto, A., Endo, I., "Negotiation between Multiple Mobile Robots and an Environment Manager", in Proceedings 1991 Conference on Advanced Robotics, pp 533-538, Italy 1991

[Axelrod 1984]

Axelrod, R., "The Evolution of Cooperation", Basic Books, USA 1984

[Bakker and Kuniyoshi 1996]

Bakker, P., Kuniyoshi, Y., "Robot See, Robot Do: An Overview of Robot Imitation", in Proceedingds AISB Workshop on Learning in Robots and Animals, UK 1996

[http://www.etl.go.jp/etl/robotics/Projects/Humanoid/postscript/agents_97.ps.gz]

[Balch 1997]

Balch, T. "Social Entropy: a New Metric for Learning Multi-robot Teams", Proceedings FLAIRS-97, 1997

[http://www.cc.gatech.edu/grads/b/Tucker.Balch/papers/entropy.ps.Z]

[Balch 1997b]

Balch, T., "Learning Roles: Behavioral Diversity in Robot Teams", in Proceedings 1997 AAAI Workshop on Multi-agent Learning

[http://www.cc.gatech.edu/grads/b/Tucker.Balch/papers/learningroles.ps.Z]

[Balch and Arkin 1995]

Balch, T. ,Arkin, R.C., "Motor Schema-based Formation Control for Multiagent Robot Teams", 1995 International Conference on Multiagent Systems, pp. 10-16, San Fransisco, USA 1995

[http://www.cc.gatech.edu/grads/b/Tucker.Balch/papers/form.ps.Z]

[Balch and Arkin 1994]

Balch, T., Arkin, R.C., "Communication in Reactive Multiagent Robotic Systems", Autonomous Robots, Vol. 1, No. 1, pp. 27-52, 1994

[http://www.cc.gatech.edu/grads/b/Tucker.Balch/papers/comm.ps.Z]

[Barnes et al 1997]

Barnes, D.P., Ghanea-Hercock, R. A. , Aylett, R. S. , Coddington, A. M., "Many Hands Make Light Work? An Investigation into Behaviourally Controlled Co-operant Autonomous Mobile Robots", in on-line Proc. Autonomous Agents 1997, http:// sigart.acm.org:80/ proceedings/agents97/

[http://sigart.acm.org:80/proceedings/agents97/A056/A056.PDF]

[Beckers et al 1995]

Beckers, R., Holland, O.E., Deneubourg, J.L, "From local actions to global tasks: stigmergy and collective robotics", in Proceedings of the Fourth International Conference on the Synthesis and Simulation of Living Systems (Artificial Life 4), pp. 181-189, the MIT Press, USA 1995

[Billiard and Dautenhahn 1997]

Billard, A., Dautenhahn, K. "The social aspect of communication: a case study in the use and usefulness of communication for embodied agents", accepted for presentation at 4th European Conference on Artificial Life, UK 1997

[http://cyber.reading.ac.uk/staff/people/kd/WWW/paperbis.ps]

[Bison and Trainito 1996]

Bison, P., Trainito, G., "A Robot Duo for Co-operative Autonomous Navigation", in Distributed Autonomous Robotic Systems 2, pp. 423-430, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Bond and Gasser 1988]

Bond, A.H., Gasser, L., "An Analysis of Problems and Research in DAI", in Readings in Distributed Artificial Intelligence, pp 3-35, Bond, A.H., Gasser, L., eds., Morgan Kaufmann Publishers, USA 1988

[Brafman and Tennenholtz 1996]

Brafman, R.I., Tennenholtz, M., "On Partially Controlled Multi-Agent Systems", Journal of Artificial Intelligence Research , Vol. 4, pp 477-503, 1996

[http://www.cs.ubc.ca/spider/brafman/jair96.ps]

[Brooks 1990]

Brooks, R.A., "Elephants Don't Play Chess", in Maes, P., ed., Designing Autonomous Agents: Theory and Practice to Engineering and Back, MIT/Elsevier, USA 1990

[Brooks and Flynn 1989]

Brooks, R.A., Flynn, A.M., "Fast, Cheap, and out of Control: A Robot Invasion of the Solar System", Journal of the British Interplanetary Society, Vol. 42, pp. 478-485, 1989

[Brooks et al 1990]

Brooks, R.A., Maes, P., Mataric, M.J., More, G., "Lunar Base Construction Robots", in Proceedings 1990 IEEE International Workshop on Intelligent Robots and Systems, pp389-391, 1990

[Brummit and Stentz 1996]

Brummit, B.L., Stentz, A., "Dynamic Mission Planning for Multiple Mobile Robots", in Proceedings 1996 IEEE International Conference on Robotics and Automation, Minneapolis, MN, USA 1996

[http://www.frc.ri.cmu.edu/~belboz/research/icra96/icra96.4.ps]

[Buckley 1989]

Buckley, S.J., "Fast Motion Planning for Multiple Moving Robots", in Proceedings 1989 IEEE International Conference on Robotics and Automation, pp 322-326, 1989

[Cai et al 1996]

Cai, A., Fukuda, T., Arai, F., "Integration of Distributed Sensoing Information in DARS Based on Evidential Reasoning", in Distributed Autonomous Robotic Systems 2, pp. 268-279, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Caloud et al 1990]

Caloud, P., Choi, W., Latombe, J-C., Le Pape, C., Yim, M., "Indoor Automation with Many Mobile Robots", in Proceedings 1990 IEEE International Workshop on *Intelligent* Robots and Systems, pp 67-72, 1990

[Cao et al 1997]

Cao, Y.U., Fukunaga, Kahng, A.B., Meng, F., "Co-operative Mobile Robotics: Antecedents and Directions", in Weiss, G. ed., "Distributed Artificial Intelligence Metts Machine Learning - Learning in Multi-Agent Environments", Springer-Verlag, Germany 1997

[Crosher 1996]

Crosher, D., "The Evolution of Learning and Exploration through Improved Exploitation of Opponents", in Complexity International (on-line journal), Vol. 2, http://www.csu.edu.au/ci, 1996

[http://www.csu.edu.au/ci/vol2/dtc/dtc.html]

[Dautenhahn 1997]

Dautenhahn, K., "I could be you - the phenomenological dimension of social understanding", to appear in Cybernetics and Systems, Vol. 5, No. 28, Special Issue on Epistemological Aspects of Embodied AI, 1997

[http://cyber.reading.ac.uk/staff/people/kd/WWW/paperbis.ps]

[Dautenhahn 1995]

Dautenhahn, K., "Getting to Know Each Other - Artificial Social Intelligence for Autonomous Robots", Robotics and Autonomous Systems 16, pp 333-356, 1995

[http://cyber.reading.ac.uk/staff/people/kd/WWW/getting.ps]

[Demiris and Hayes 1996]

Demiris, J., Hayes, G., "Imitative Learning Mechanisms in Robots and Humans", Proceedings 1996 European Workshop on Learning Robots, Klingspor, V., ed., Italy 1996

[http://www.dai.ed.ac.uk/students/johnde/ewlr96.ps]

[Denham and McCabe 1995]

Denham, M.J. ,McCabe, S.L., "Robot Control Using Temporal Sequence Learning", in Proceedings 1995 World Conference on Neural networks, Washington DC, USA 1995

[Deneubourg and Goss 1989]

Deneubourg, J.L., Goss, S., "Collective patterns and decision-making", Ethology, Ecology and Evolution 1, pp 295-311, 1989

[Elgimez and Kim 1990]

Elgimez, K., Kim, S.H., "Deployment of Robotic Agents in Uncertain Environments: Game Theoretic Rules and Simulation Studies", AI EDAM 1, Vol. 6, pp 1-17, 1990

[Erdmann and Lozano-Perez 1987]

Erdmann, M., Lozano-Perez, T., "On Multiple Moving Objects", Algorithmica 2, pp 477-521, 1987

[Fontan and Mataric 1996]

Fontan, M.S., Mataric, M.J., "The Role of Critical Mass in Multi-Robot Adaptive Task Division", submitted to IEEE Transactions on Robotics and Automation, also Brandeis University Computer Science Technical Report CS-95-187, October 1996

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/ieeetra96.ps.Z]

[Goldberg and Mataric 1997]

Goldberg, D., Mataric, M.J., "Interference as a Tool for Designing and Evaluating Multi-Robot Controllers", in Proceedings 1997 AAAI Conference, USA1997

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/aaai97-wdani.ps.Z]

[Hakura et al 1996]

Hakura, J., Yokoi, H., Kakazu, Y., "Affordance in Autonomous Robots (Grounding Mechanism of Sensory Inputs)", in Distributed Autonomous Robotic Systems 2, pp. 156-167, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Hayes and Demiris 1994]

Hayes, G., Demiris, J., "A Robot Controller Using Learning by Imitation", in Proc. 1994 International Symposium on Intelligent Robotic Systems, France 1994

[http://www.dai.ed.ac.uk/students/johnde/irs.ps]

[Herman and Albus 1988]

Herman, M., Albus, J.S., "Overview of the Multiple Autonomous Underwater Vehicles (MAUV) Project", in Proceedings 1988 IEEE International Conference on Robotics and Automation, pp 618-620, Philadelphia 1988

[Heyes 1994]

Heyes, C.M., "Social Cognition in Primates", in N.J. Mackintosh, ed., Animal Learning and Cognition, Academic Press: London, UK 1994

[Ichikawa and Hara 1996]

Ichikawa, S., Hara, F., "Experimental Characteristics of Multiple-Robots Behaviors in Communication Network Expansion and Object-fetching", in Distributed Autonomous Robotic Systems 2, pp. 183-194, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Ichikawa and Hara 1994]

Ichikawa, S., Hara, F., "An Experimental Realization of Co-operative Behavior of Multi-Robot System", in Distributed Autonomous Robotic Systems, pp 224-234, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[d'Inverno and Luck 1996]

d'Inverno, M., Luck, M., "Understanding Autonomous Interaction", in Proc. 12th European Conference on Artificial Intelligence, John Wiley & Sons, UK 1996

[Kaebling and Moore 1996]

Kaelbling, L.P., Moore, W.M., "Reinforcement Learning: A Survey" in Journal of Artificial Intelligence Research, Vol. 4, pp. 237-285, 1996

[Kelly 1995]

Kelly, K., "Out of Control: the new biology of machines", Fourth Estate Paperbacks, 1995

[Kuniyoshi 1997]

Kuniyoshi, Y., "Fusing Autonomy and Sociability in Robots", in Proceedings 1997 International Conference Autonomous Agents (Agents97 Invited talk), pp.470-471, USA 1997

[http://www.etl.go.jp/etl/robotics/Projects/Humanoid/postscript/agents_97.ps.gz]

[Lashkari et al 1994]

Lashkari, Y., Metral, M., Maes, P., "Collaborative Interface Agents", in Proceedings 1994 National Conference on Artificial Intelligence, Vol. 1, AAAI Press, Seattle, 1994

[ftp://media.mit.edu/pub/agents/interface-agents/coll-agents.ps]

[Leuth and Laengle 1996]

Leuth, T., Laengle, T., "Managing Different Types of Interactions among Robots", in Distributed Autonomous Robotic Systems 2, pp. 195-206, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Leuth 1994]

Leuth, T., "Processing Complex Tasks with Cooperating Robot Systems", in Proc. Conf. on Solving Complex Problems with Agent Systems, pp. 5-25, Bielefeld, Netherlands 1994

[Li 1994]

Li, S., "Co-ordinating Multiple Mobile Robots through Local Inter-robot Communication", in Distributed Autonomous Robotic Systems, pp 190-198, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[López de Màntaras et al 1997]

López de Màntaras, R., Amat, J., Esteva, F., López, M., Sierra, C., "Generation of Unknown Environment Maps by Cooperative Low-Cost Robots" , in on-line Proc. Autonomous Agents 1997, http://sigart.acm.org:80/proceedings/agents97

[http://sigart.acm.org:80/proceedings/agents97/A009/A009.PDF]

[Mataric 1994]

Mataric, M.J. "Interaction and Intelligent Behavior" , MIT EECS PhD Thesis, May USA 1994, also MIT AI Lab Tech Report AITR-1495, USA 1994

[ftp://publications.ai.mit.edu/ai-publications/1000-1499/AITR-1495.ps.Z]

[Mataric 1995]

Mataric, M.J., "Issues and Approaches in the Design of Collective Autonomous Agents", Robotics and Autonomous Systems, Vol.16, Nos. 2-4, pp. 321-331, USA 1995

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/ras-si.ps.Z]

[Mataric 1993]

Mataric, M.J., "Kin Recognition, Similarity, and Group Behavior", in Proceedings 1993 Annual Cognitive Science Society Conference, pp 705-710, Lawrence Erlbaum Associates, USA 1993

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/cogsci93.ps.Z]

[Mataric 1996]

Mataric, M.J., "Learning in Multi-Robot Systems", in Adaptation and Learning in Multi-Agent Systems, Weiss, G., Sen, S., eds., Lecture Notes In Artificial Intelligence (LNAI), Vol. 1042, pp 152-163, Springer-Verlag, USA 1996

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/ijcai95ws.ps.Z]

[Mataric 1994b]

Mataric, M.J., "Learning to Behave Socially", in Proceedings 1993 International Conference on Simulation of Adaptive Behavior (From Animals to Animats 3), MIT Press, Massachusetts 1993

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/sab94.ps.Z]

[Mataric 1997]

Mataric, M.J., "Reinforcement Learning in the Multi-Robot Domain", Autonomous Robots, Vol. 4, No. 1, pp 73-83, January 1997

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/ar.ps.Z]

[Mataric 1997b]

Mataric, M.J., "Learning Social Behavior", to appear in Robotics and Autonomous Systems, 1997

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/ras-rolf.ps.Z]

[Matsumoto and Nagai 1996]

Matsumoto, A., Nagai, H., "The Modeling of the Team Strategy of Robotic Soccer as a Multi-Agent Robot System", in Distributed Autonomous Robotic Systems 2, pp. 115-126, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[McFarland 1993]

McFarland, D., "Towards Robot Cooperation", in in Proceedings 1993 International Conference on Simulation of Adaptive Behavior (From Animals to Animats 3), pp 440-444, MIT Press, Massachusetts 1993

[McFarland and Bosser 1993]

McFarland, D., Bosser, T., "Intelligent Behavior in Animals and Robots", MIT Press, Massachusetts 1993

[Miller 1990]

Miller, D.P., "Multiple Behavior-Controlled Micro-Robots for Planetary Surface Missions", in Proceedings 1990 Conference in IEEE Systems, Man, Cybernetics, pp 289-292, California 1990

[Ming et al 1996]

Ming, A., Masek, V., Kanamori, C., Kajitani, M., "Co-operative Operation of Two Mobile Robots", in Distributed Autonomous Robotic Systems 2, pp. 339-349, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Nehmzow 1991]

Nehmzow, U., "Location Recognition in a Mobile Robot Using Self-Organising Feature Maps", G. Schmidt, ed., Information Processing in Autonomous Mobile Robots, Springer Verlag 1991

[Noborio and Yoshioka 1994]

Noborio, H., Yoshioka, T., "On a Deadlock-free Characteristic of the On-line and Decentralised Path-planning for Multiple Automata", in Distributed Autonomous Robotic Systems, pp 111-122, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Nof 1991]

Nof, S.Y., "Games Theoretic Models for Planning Co-operative Robotic Work", in Proceedings 1991 NSF Design and Manufacturing Systems Conference, pp 553-556, Texas 1991

[Ohko et al 1993]

Ohko, T., Hiraki, K., Anzai, Y., "LEMMING: A Learning system for Multi-Robot Environment", in Proceedings 1993 International Conference on Intelligent Robots and Systems, pp 1141-1146, IEEE, Japan 1993

[Ota et al 1994]

Ota, J., Arai, T., Yokogawa, Y., "Distributed Strategy-making Method in Multiple Mobile Robot System", in Distributed Autonomous Robotic Systems, pp 123-133, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Ozaki et al 1994]

Ozaki, K., Asama, H., Ishida, Y., Yokota, K., Matsumoto, A., Kaeysu, H., Endo, I., "Negotiation Method for Collaborating Team Organization among Multiple Robots", in Distributed Autonomous Robotic Systems, pp 199-210, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Parker 1992]

Parker, L., "Local versus Global Control Laws for Co-operative Agent Teams", Technical Report (A.I. Memo No. 1357), MIT Artificial Intelligence Laboratory, Massachusetts 1992

[Parker 1994]

Parker, L., "Heterogenous Multi-Robot Cooperation", PhD. Thesis, Department of Electrical Engineering and Computer Science MIT, Massachusetts 1994

[Parker 1996]

Parker, L., "Multi-Robot Team Design for Real-World Applications", Distributed Autonomous Robotic Systems 2, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, pp 91-102, Springer-Verlag, Tokyo, 1996

[http://avalon.epm.ornl.gov/~parkerle/publications/DARS_96.ps.Z]

[Parker 1994b]

Parker, L., "ALLIANCE: An Architecture for Fault Tolerant, Co-operative Control of Heterogeneous Mobile Robots", in Proceedings 1994 IEEE International Conference on Intelligent Robots and Systems, pp 776-783, Germany 1994

[http://avalon.epm.ornl.gov/~parkerle/publications/TM-12920.ps.Z]

[Parker 1992b]

Parker, L., "Adaptive Action Selection for Co-operation Agent Teams", in Proceedings of the Second International Conference on Simulation of Adaptive Behavior (From Animals to Animats 2), pp 442-450, MIT Press, 1992

[Parker 1995]

Parker, L., "The Effect of Action Recognition and Robot Awareness in Co-operative Robotic Teams", in Proceedings 1995 IEEE International Conference on Intelligent Robots and Systems, Vol. 1, pp 212-219, Pittsburgh 1995

[Parker 1996b]

Parker, L., "On the Design of Behavior-Based Multi-Robot Teams", in Journal of Advanced Robotics, Vol. 6, No. 10, 1996

[http://avalon.epm.ornl.gov/~parkerle/publications/Adv_Rob.ps.Z]

[Premvuti and Yuta 1996]

Premvuti, S., Yuta, S., "Consideration on Conceptual Design of Inter Robot Communications Network for Mobile Robot System (Concept of Implicit Communication and how to Realize with Current Technology", in Distributed Autonomous Robotic

Systems 2, pp. 441, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Premvuti and Yuta 1990]

Premvuti, S., Yuta, S., "Consideration on the Co-operation of Multiple Autonomous Mobile Robots", 1990 Proceedings IEEE International Workshop on Intelligent Robots and Systems, pp 219-223, 1990

[Rausch and Levi 1996]

Rausch, W.A., Levi, P., "Asynchronous and Synchronous Co-operation", in Distributed Autonomous Robotic Systems 2, pp. 245-256, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Sanderson et al 1996]

Sanderson, A.C., "Co-operative Navigation among Multiple Mobile Robots", in Distributed Autonomous Robotic Systems 2, pp. 389-400, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Sen 1996]

Sen S., "Reciprocity: a foundational principle for promoting cooperative behavior among self-interested agents", in Proceedings 1996 International Conference on Multiagent Systems, Japan 1996

[http://euler.mcs.utulsa.edu/~sandip/icmas96-reciprocity.ps]

[Sharman et al 1997]

Sharman, K., Ünsal, C., J. S. Bay, J.S., "A Reactive coordination Scheme for a Many-Robot System", to appear in IEEE Transactions on Systems, Man and Cybernetics, August 1997

[ftp://armyant.ee.vt.edu/pub/bay/jp97_smc.ps]

[Stone and Veloso 1997]

Stone, P., Veloso, M., "Towards Collaborative and Adversarial Learning: A Case Study in Robotic Soccer", to appear in Internation Journal of Human-Computer Systems (IJHCS) in 1997

[http://www.cs.cmu.edu/afs/cs/user/pstone/public/papers/96ijhcs.ps.Z]

[Sugawara and Sano 1996]

Sugawara, K., Sano, M., "Co-operative Acceleration of Task Performance: Foraging Behavior of Interacting Multi-Robots System", in Distributed Autonomous Robotic Systems 2, pp. 233-242, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Sugihara and Suzuki 1990]

Sugihara, K., Suzuki, I., "Distributed Motion Coordination of Multiple Mobile Robots", in Proceedings 1990 IEEE International Conference on Robotics and Automation, pp 138-143, Ohio 1990

[Suzuki et al 1994]

Suzuki, T., Yokota, K., Asama, H., Ozaki, K., Ishida, Y., Matsumoto, A., Kaetsu, H., Endo, I., "A Human Interface for Interacting with and Monitoring the Multi-agent Robotic

System", in Distributed Autonomous Robotic Systems, pp 50-61, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Thrun 1992]

Thrun, S.B., "The Role of Exploration in Learning Control", in Handbook of Intelligent Control, pp. 527-559, Van Nostrand Reinhold, New York, 1992

[Ueyama et al 1994]

Ueyama, T., Fukuda, T., Atsushi, A., Fumihito, A., "Hierarchical Control Architecture with Learning and Adaption Ability for Cellular Robotic System" , in Distributed Autonomous Robotic Systems, pp 17-28, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Ünsal and Bay 1994]

Ünsal, C., Bay, J.S., "Spatial Self-Organization in Large Populations of Mobile Robots", 1994 IEEE International Symposium on Intelligent Control, Ohio 1994

[http://www.cs.cmu.edu/~unsal/publications/spatial.html]

[Vainio et al 1996]

Vainio, M., Halme, A., Pekka, A., Pekka, K, Jakubik, P., Schonberg, T., Wang, Y., "An Application Concept of an Underwater Robot Society", in Distributed Autonomous Robotic Systems 2, pp. 103-114, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Wang 1993]

Wang, J., "DRS Operating Primitives Based on Distributed Mutual Exclusion", in Proceedings 1993 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp 1085-1090, Japan 1993

[Wang and Premvuti 1994]

Wang, J., Premvuti, S., "Resource Sharing in Distributed Robotic Systems Based on A Wireless Medium Access Protocol (CSMA/CD-W)", in Distributed Autonomous Robotic Systems, pp 211-223, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Wang and Premvuti 1994b]

Wang, J., Premvuti, S., "Fully Distributed Traffic Regulation and Control for Multiple Autonomous Mobile Robots Operating in Discrete Space", in Distributed Autonomous Robotic Systems, pp 134-161, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994

[Weiss 1997]

Weiss, G. ed., "Distributed Artificial Intelligence Metts Machine Learning - Learning in Multi-Agent Environments", Springer-Verlag, Germany 1997

[Weiss and Sen 1996]

Weiss, G., Sen, S., Eds., "Adaption and Learning in Multi-Agent Systems", Springer-Verlag, Germany 1996

[Werger and Mataric 1996]

Werger, B.B., Mataric, M.J., "Robotic 'Food' Chains: Externalization of State and Program for Minimal-Agent Foraging" in Proceedings 1996 International Conference on Simulation of Adaptive Behavior (From Animals to Animats 4), pp625-634, MIT Press/Bradford Books 1996

[ftp://ftp.cs.brandeis.edu/pub/faculty/maja/sab96-bm.ps.Z]

[Yamaguchi and Beni 1996]

Yamaguchi, H., Beni, G., "Distributed Autonomous Formation Control of Mobile Robot Groups by Swarm-based Pattern Generatation", in Distributed Autonomous Robotic Systems 2, pp. 141-155, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Yoshimura et al 1996]

Yoshimura, Y., Ota, J., Inoue, K., Kurabayashi, D., Arai, T., "Iterative Transportation Planning of Multiple Objects by Co-operative Mobile Robots", in Distributed Autonomous Robotic Systems 2, pp. 171-182, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer-Verlag, Tokyo 1996

[Zhi-Dong et al 1994]

Zhi-Dong, W., Nakano, E., Matsukawa, T., "Co-operating Multiple Behavior-Based Robots for Object Manipulation (System and Co-operation Strategy)", in Distributed Autonomous Robotic Systems, pp 371-382, Asama, H., Fukuda, T., Arai, T., Endo, I., eds, Springer Verlag, Tokyo 1994