MULTI-AGENT EXPLORATION AND RENDEZVOUS

Nicholas Roy

Department of Computer Science
McGill University, Montréal

July 1997

A Thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements for the degree of
Master of Science

0-612-37159-X

ABSTRACT

We consider the problem of rendezvous between two robots exploring an unknown environment. That is, how can two autonomous exploring agents that cannot communicate with one another over long distances meet if they start exploring at different locations in an unknown environment. The intended application is collaborative map exploration.

Ours is the first work to formalize the characteristics of the rendezvous problem, and we approach it by proposing several alternative algorithms that the robots could use in attempting to rendezvous quickly while continuing to explore. The algorithms are based on the assumption that potential rendezvous locations, called landmarks, can be selected by the robots as they explore; these locations are based on a distinctiveness measure computed with an arbitrary sensor.

We consider the performance of our proposed algorithms analytically with respect to both expected- and worst-case behaviour. We then examine their behaviour under a wider set of conditions using numerical analysis. This numerical analysis is confirmed using realistic simulation of multi-agent exploration and rendezvous. We also examine the exploration speed, and show that a multi-robot system can explore an unknown environment faster than a single-agent system, even with the constraints of performing rendezvous to allow communication.

We conclude with a demonstration of rendezvous implemented in the real world.

RÉSUMÉ

Nous traitons le problème du rendez-vous entre deux robots qui explorent un environnement inconnu. Plus précisément, nous étudions comment deux agents d'exploration autonômes, ne pouvant communiquer sur de longues distances, peuvent se rencontrer lors de l'exploration d'un terrain inconnu à partir de positions initiales différentes. L'application visée est celle de l'exploration collaborative.

Après une formalisation des caractèristiques du problème de rendez-vous, nous proposons plusieurs alternatives d'algorithmes utilisables par des robots pour se retrouver rapidement tout en explorant l'environnement. Les algorithmes utilisent l'hypothèse que les sites de rencontre potentiels, appelés ≪landmarks≫, peuvent être déterminés par les robots au cours de l'exploration. Ces sites sont choisis selon une mesure de distinctivité ou de qualité à partir d'un déctecteur arbitraire.

Nous examinons alors analytiquement les algorithmes proposés, selon leurs comportements attendu ou selon le pire cas. Nous etudions ensuite plus largement leur comportement par l'analyse numérique. Cette analyse numérique est vérifiée par la simulation réaliste d'explorations et de rencontres multi-agents. L'aspect de la vitesse d'exploration est également envisagé et nous démontrons qu'un système multi-robots peut explorer un environnement inconnu plus rapidement que dans le cas de l'utilisation d'un seul robot, et cela même si un rendez-vous est nécessaire pour permettre la communication des informations acquises.

Nous terminons avec un démonstration de rendez-vous dans le monde réel.

ACKNOWLEDGEMENTS

My thanks are due in no small measure to my supervisor, Dr. Gregory Dudek, for his guidance and assistance with this thesis, his patience as I chased one hare-brained idea after another, his whole-hearted support of my research, and his willingness to discuss any aspect of my research at any time, or anyone else's research for that matter.

I wish to thank Marc Bolduc and Thierry Baron, without whom this thesis would be entirely theory; these two gentlemen have accepted my ever-increasingly strange requests for new robot configurations and additions to the RoboDaemon without so much as a raised eyebrow. Well, maybe *one* raised eyebrow...

I would like to thank Kristine Lake for listening to me rant about my research for the last 18 months, and especially proof-reading this thesis. A biologist's eye is essential for Computer Science theses...

My western Canadian and occasionally humourous translation of the abstract into a résumé was improved immeasurably by Nicolas Desaulniers, and proof-read by Thierry Baron and Kathleen VanderNoot to whom je voudrais dire merci.

My thanks to Hisham Petry, Robert Sim, Eric Bourque and Yiannis Rekleitis for support, advice and comradeship.

And last but most certainly not least, to Michael Daum for proof-reading, debugging, the ubiquitous occupancy grids, and most of all, for making it all a whole lot more fun.

DEDICATION

This dissertation is dedicated to my parents,

Chunilal and Elizabeth Roy

for their love and support for the last 24 years, but most importantly, for teaching me to be good at what I do.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

With the advent of relatively cheap and robust mobile robots, the possibility of using teams of mobile robots to perform simple tasks has become a reality. In many contexts, multi-agent systems may be more effective, faster, or more desirable than a single, very powerful monolithic robot system. Despite the loss of one agent in a multi-agent system, the remaining agents can potentially continue the task to completion. Manufacturing several small, unsophisticated agents may be much cheaper, even without without sacrificing speed or functionality, than manufacturing a single, powerful robot. Several small, independent agents may be capable of more than any existing monolithic system. Furthermore, there are many tasks that simply require more than one agent, either human or mechanical, working in tandem.

These advantages notwithstanding, there are difficulties associated with multi-agent systems. The problems of task division, synchronisation and coordination are significant; the correct behaviour to follow if a member of team becomes lost, damaged or stuck is not always clear. Furthermore, the correct behaviour to maximise the efficiency of the distributed team is equally difficult to determine. Practical considerations can further contribute to the complexity of a multi-agent system when compared with a single agent system.

One example of such a practical limitation is inter-agent communication. Most existing hardware agents are only capable of communication over short distances. Environmental geometry, wireless transmission technology, power considerations and atmospheric conditions (or water conditions for underwater agents) all contribute to limitations on communication range. In the absence of sophisticated satellite receivers or high power devices, a common constraint for successful communication is maintaining "line-of-sight" between agents, a constraint that is rarely satisfied in the real world. However, existing research indicates that multi-agent robot systems for the majority of real-life applications enjoy substantial speed gains only with some level of communication [**6**], when compared with single-agent systems or multi-agent systems that do not communicate. Many distributed-agent algorithms, for instance dynamic path-planning, assume and rely upon instantaneous, infinite bandwidth communication between agents at all times in order to achieve promised performance levels [**12**].

Another limitation is the ability of the collaborating agents to share information in a manner that allows any advantage over single agent systems at all. Using the example of exploration, the agents must be able to merge the maps generated by the exploration process; if the maps cannot be merged, then each agent must itself explore the environment to completion, and no task speed-up is achieved. However, under many circumstances, the agents must often share a common reference point to be able to merge maps[1] - completely independent maps cannot always be merged reliably. Unless the agents start at exactly the same place in the environment, they must agree on a place to meet *a priori*, and share information. However, choosing a meeting point reliably, especially in an unknown, unconstrained environment is a very difficult problem.

The solution proposed in this work to these problems is termed 'Multi-agent Rendezvous'. A rendezvous is a meeting between two or more agents at an appointed place and time, for example, when two people meet to share notes at a familiar location. This problem is ubiquitous in nature – migratory animals must be able to meet to share information about food. Non-social animals must be able to find each other during mating season. Humans are equally familiar with the problem of rendezvous, as any family whose members become separated at a zoo or mall well knows. Multi-agent robot systems also have an inherent need for the ability of inter-agent rendezvous. The ability to meet facilitates localisation, allows collaborative map exploration and has a plethora of other advantages, but most importantly allows communication.

1.1. Problem Statement

The problem discussed in this thesis is that of how to determine the best strategy for a successful rendezvous between two agents in optimal time. The context that will be used in this thesis for studying the problem of rendezvous is multi-robot exploration of unknown environments. Although multi-agent robotic systems are useful to a number of domains, exploration of unknown environments is an appropriate context for a number of reasons. The task of exploration is one that has been studied previously, and there is a well-established body of work on single robot and multi-robotic exploration methods. The existence of such established algorithms allows this thesis to focus more narrowly on the problem of rendezvous. Furthermore, some of the difficulties with multi-agent

[1]It is sometimes possible to merge maps using their shapes. However, if the agents' sensors are substantially different, or there are spatial ambiguities, the merging process may fail.

systems that are solved by rendezvous, in particular the problems of communication and of map-fusion, are integral to the exploration task. If these problems can be addressed by rendezvous in the context of exploration, then the technique is generalisable to other domains.

In particular, we are interested in multi-robot exploration using video or sonar sensing. In practice, the particular sensing modality has numerous pragmatic implications, a major factor being the range at which the agents can either recognise one another, or any landmarks in the environment. In the context of a general rendezvous strategy, we will initially, however, consider a generic "abstract" sensor that allows the agents to recognise one another when they are sufficiently close together and which allows them to evaluate any point in space as to its suitability as a rendezvous point. We will consider how the rendezvous task can be efficiently accomplished under various assumptions about the environment and the perceptual abilities of the agents involved.

1.2. The Approach

The rendezvous task itself is divided into two sub-problems.

(i) The first sub-problem is how to choose an appropriate rendezvous point, given an unknown environment. The ability of the agents to meet in the environment is a function of their ability to reliably choose appropriate rendezvous points. For instance, mountain tops may be a good outdoor rendezvous point. However, if the agents cannot reliably measure the height of mountains or the agents are indoors, another notion of good rendezvous points may be appropriate.

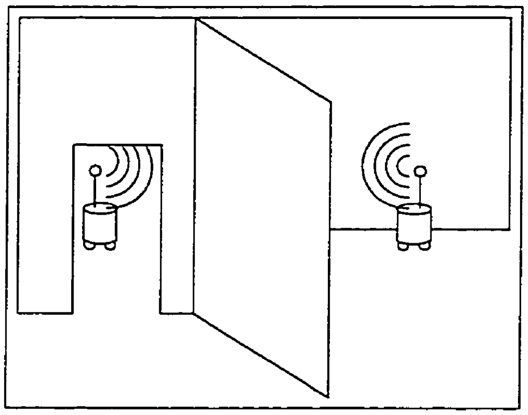


FIGURE 1.1. Two robots, searching an unknown environment for good rendezvous points.

(ii) The second sub-problem is that of dealing with confounding factors to the rendezvous process. One such factor is the ability for agents to agree on the same location for rendezvous.

Sensor noise may cause agents to disagree; agents may not be aware of the same spatial areas, and therefore may choose different points. For instance, if two agents have explored two different mountain ranges, then they will not choose the same peak as the highest mountain peak. An appropriate rendezvous strategy will take into account such asymmetry between the agents' exploration. If the agents agree to meet at noon, and one agent cannot reach the rendezvous point by noon, pre-arranged behaviour must account for such asynchronies, and allow for missed rendezvous attempts.

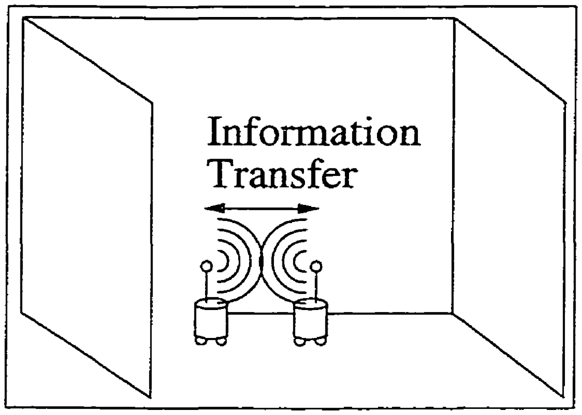


FIGURE 1.2. Two robots, after a successful rendezvous, sharing map information.

In the simplest formulation, the rendezvous will involve the agents searching through the environment for good meeting points, and then travelling to the best meeting point at a pre-arranged time.

1.3. Contribution

This is the first description of this problem in multi-agent mobile robotics. Most multi-agent algorithms rely upon certain assumptions that are not necessarily tenable in practice. Certainly, unlimited communication is not a realistic assumption, yet many of the classical path-planning and task division algorithms assume it. The method of rendezvous is the first proposed as a solution to these problems with practical mobile robotics.

This thesis describes the idealised task of mobile robot rendezvous formally, including several issues that complicate the task in practice.

- We formalise the parameters of the rendezvous problem that necessitate more than one attempt.
- Two classes of solutions are proposed, and analysed both analytically and empirically.

- We simulate the rendezvous problem at two levels; the first level is a purely algorithmic simulation, simply to test the efficacy of the various algorithms under the different conditions we describe.
- Subsequently, we have developed a realistic simulation, using spatial metrics and simulated sensing and motion.
- Finally, we demonstrate the speed-up possible under multi-agent systems by comparing the running-time of the multi-agent system versus the single-robot system.

In order to motivate the rendezvous process, we also develop a novel, simple and robust exploration algorithm, that uses a hybrid of local potential field descent and global optimisation. Using the exploration method, we simulate robots exploring an unknown environment and periodically performing rendezvous.

1.4. Outline

This thesis begins by describing the task of rendezvous, and some of the issues that must be addressed by any system that is attempting multi-agent meetings. The problem of choosing the rendezvous locations is formalised, as are the factors that make rendezvous difficult. In the context of this formalisation, some representative algorithms which illustrate key classes are defined and discussed, first analytically, and then numerically. A simulation of multi-agent exploration with rendezvous is developed, and finally implemented on a physical multi-robot system. The algorithms for rendezvous are the focus of the work, both on a qualitative and quantitative basis.

The outline of this thesis follows:

Chapter 2 discusses related work, both to the problems of multi-agent robotics, and also to the problem of exploration of unknown environments.

Chapter 3 presents a formalised notion of space that will be used both for exploration and for rendezvous. It is in this chapter that we begin to break the rendezvous problem down into multiple components. First, the framework for finding rendezvous points will be described, as well as associated issues. We then present a description of the algorithms, and examples of each. We conclude with an analytical description of the algorithm performances at representative sample points in the problem space.

Chapter 4 presents a numerical analysis of the algorithms under more detailed conditions than the extreme cases analysed in Chapter 3. The experimental framework is first described, and then an examination of the results. This numerical analysis is shown to be a confirmation of the analytical results.

Chapter 5 describes the implementation of the exploration algorithm. The potential field algorithm is detailed, along with the underlying occupancy grid representation that is used for representing the spatial information gathered during exploration. It is this exploration method that is used to motivate the rendezvous problem.

Chapter 6 describes the experimental framework of the spatial simulation, and then presents the results of the spatial simulation. The results are highly illustrative of many of the problems of rendezvous.

Chapter 7 describes an implementation of the exploration and rendezvous tasks using real robots. The experimental context is an indoor laboratory situation.

Finally, we conclude with a summary of the work, and a description of the important results. As this is a new research area, a considerable number of issues remain to be addressed. We discuss some of the unresolved problems of rendezvous, and remaining open questions.

CHAPTER 2

Related Work

The interdisciplinary nature of this thesis results in related work in a number of subject areas. The first such area, and the main thrust of this thesis, is the coordination of multiple robots engaged in some common task.

2.1. Rendezvous

The problem of rendezvous is not a new one; there exists a body of research in the optimisation and operations research community involving search problems. Rendezvous is a particular variant of the search problem, similar to games with mobile hiders, called *princess and monster games* [**2**]. There are many variants of the rendezvous problem itself, involving distinguishable [**3**] or indistinguishable agents [**4**] and collaborating or interfering agents. The environment may have focal points, or may be completely homogeneous.

There are a number of differences between the highly theoretical approach most prior work takes and the approach used in this thesis:

- In the existing literature, the assumption made is that the environment is known *a priori*. In this thesis, the environment is not known, and one of the key problems is to *find* the focal points (what we term *landmarks*) and meet.
- In prior work, the problem of what to do when the initial rendezvous fails has not been addressed, since the theoretical agents have perfect sensing, synchronisation, etc. In this thesis, we are dealing with realisable agents, with the concomitant problems of noise, asynchrony, real-time travel limitations.

However, there are is a key similarity between the prior work and this thesis:

- Communication between agents is prohibited, until the agents are within a pre-determined line-of-sight range. Indeed, the graph-theoretic approach reduces this distance to 0 in many cases.

It is interesting to note that one of the algorithms proposed by Alpern [2] is equivalent to the first deterministic algorithm we propose in Chapter 3.

2.2. Multi-agent robotics

There are two principle approaches to multi-robot collaboration:

(i) The first approach is to examine a particular task that has been studied for uni-robotic systems and extend it to multiple-agent systems. This approach generally yields multi-robot algorithms for a given task.

(ii) The second approach is to examine the properties and abilities of multiple-agent systems, either *qua* a robotic system, or in comparison to uni-robotic systems. This approach often results in behaviour studies of collectives, or *swarms* of robots.

Almost all of the algorithms that have been developed for multiple-agent systems assume either a central controller, or a large degree of communication. Latombe [37] divides the motion planning algorithms for multiple agents into those using centralised planner, and those using decoupled planning. An example of the centralised planner is Schwartz and Sharir's [51] exact cell decomposition method for motion planning of two discs, discussed in a part of their classic "Piano Mover's Problem" series. These systems are barely multi-robotic, as opposed to multi-effector, single-brained robot systems. Clearly, a centralised planner assumes full communication between the planner and each agent. Furthermore, the high complexity of perfect, centralised planning of multiple agents implies a static environment; most centralised planners cannot cope with dynamic environments [5]. With centralised planners, there are the two most common assumptions of both full communication with the agents, and the permanence of the plan, once computed. Neither one of these assumptions is valid for most real-world mobile robot scenarios. As a result, there has been considerable research of late in distributed planners and conflict-resolution strategies.

The decoupled planners, such as prioritised planners [25, 13], path-coordination [44] or dynamic multi-agent planners [12] require no central planner, but require considerable communication, initially and often during the execution of the motion. Within the class of decoupled planners, and in particular, conflict-resolution strategies between plans across agents, Azarm and colleagues [5] further added the types of distributed planners based on master-slave relationships [55], and traffic-rule based dynamic planners [28]. Mutual-exclusion across spatial resources [1] demands static environments similar to centralised planners, whereas sensor-based planners [29] suffer from the same limitations as the uni-robot reactive control methods.

In the context of studying the behaviour of multiple-robot systems, Dudek et al. [23] provided a comprehensive taxonomy of the different types of multiple-agent systems, or *swarms*, including the various types of communication available. In addition, the three possible types of communication were described:

- No communication ("COM-NONE")
- limited communication ("COM-NEAR")
- full communication ("COM-INF")

As the authors point out, COM-INF "is the classical assumption, which is probably impractical if [the number of agents] $\gg 1$." A modest understatement, given that radio communication breaks down in many situations as soon as line-of-sight is lost. This thesis makes the assumption of swarms of range-limited communication, instantiated in this case using a line-of-sight constraint.

There has been considerable work in studying the range of behaviour of multiple-agent systems, especially attempting to maximise efficiency and minimise complexity [41] [26]. Mataric has looked at models of collaborative behaviour between mobile robots [41], and examined the "emergent behaviour" properties that result. She also observed that the form of communication plays an important role in how collaborative actions proceed. Parker [45, 46] has developed control strategies for heterogeneous multiple robot systems, and made clear the need for effective communication.

Arkin and his colleagues give a description of the canonical tasks of multiple-agent robots, and the effect of various levels of communication on speed of task completion, including the difference between explicit (i.e., by radio) and implicit communication. Implicit communication is the information that the robot embeds in the environment as a byproduct of performing a task - such as tire tracks. Implicit communication is also an important issue in multiple-agent robotics, in that it leads to dramatic speed increases for some tasks, without the constraints of explicit communication modalities. Donald et al. also discuss using implicit information in the context of co-ordinated furniture pushing [19]. However, we do not intend to deal with implicit communication in this work; the technique of rendezvous is being applied to solve the limitations of explicit communication.

Balch and Arkin [6] describe several tasks: *consuming*, *foraging* and *grazing*. In this thesis, the task of exploration is an example of a grazing task, in that the entire environment needs to be covered by at least one robot's sensors, in order to acquire a complete representation. Further tasks that have been addressed in the context of multi-agent systems are box-pushing [45, 19, 18], formation holding [7, 9, 52] and exploration and mapping [49, 15]. Like Donald's box-pushing and the grazing task, many of these applications use passive sensing or implicit information to perform their task. There is a dearth of work in real applications that demand full, active communication that have been implemented on real robots. Rekleitis and colleagues' [49] work on using multiple robots for exploration is very much in the spirit of this work, using multiple agents to overcome limitations in the use of a single robot for exploration. While their approach overcomes inherent limits in localisation in an unknown environment, the goal is to increase the precision of the map acquired. This work is focussed on increasing the speed of map acquisition.

Yoshida et al. addressed the problem of how to reduce a global communication network to local communication, in order to minimise information complexity [54]. However, there has not been much research in overcoming communication limitations, except by limiting the scope of the system to some area (such as a factory or a port) where communication between agents can be guaranteed by some global co-ordinator.

Finally, the problem of map generation from co-operative multi-agent exploration was discussed and implemented by Ishioka et al. [30]. Their work is a canonical example of the potential applications of the technique presented in this paper, in which co-operative heterogeneous robots generated maps of unknown environments. They did not discuss the problem of rendezvous, but focussed only on how to merge maps once the rendezvous has occurred. It is worth noting that map fusion is also closely related to the generic image-registration problem.

2.3. Spatial Reasoning

The secondary thrust of this paper is in the area of spatial reasoning, in particular spatial representation and exploration techniques. There exist a plethora of different techniques both for representing space and also moving about in both unknown and known environments. Kuipers [34] lists some of the representations as "Configuration space, Generalised Cones, Voronoi Diagrams, the Grid Model, the Segment Model, the Vertex Model, the Convex Polygon Model, the Graph Model and the Polygonal Region Model", not including his own contribution of the topological model [33]. Perhaps one of the most successful methods of spatial representation was the work of Moravec and Elfes' on occupancy grids, which are used in this thesis [43, 24]. Kadonoff and his colleagues developed a strategy for combining several of these models in the same system [27]. Dudek also developed a strategy for combining different spatial models [20], employing different levels of symbolic and sub-symbolic abstraction for localisation, navigation and long-term planning.

While it is clear from the graph-theoretical work that focal points in the environment are essential to effective rendezvous, it is Kuipers' selection of distinctive locations in a simple 2-D environment (considered previously in the context of map-making [34]), that is the basis for the landmarks in this thesis. The distinctive locations in that work were determined by active hill-climbing over the distinctiveness function, that is, by local gradient ascent over some function of the sensor output. The local maxima in a continuous property of the environment allowed for the conversion a metric environment representation into a graph-like or topological one [**14, 22, 35**]. Similar approaches to this organisation of space have been approached by Levitt, using visual sensing as opposed to sonar [**39**]. Miller divided space into voronoi regions based on the degrees of freedom in the error estimate, creating a semantic hierarchy of space of a different nature [**42**]. Brian Pinette used visual landmarks, as opposed to spatial landmarks, for his work in qualitative homing [**48**].

After developing the visual map making system [**10**], Brooks attempted to abolish internal representations completely [**11**]. Brooks' claim was that the world itself would serve as its own representation, and that internal maps were not important at many lower levels. The experience of the subsumption architecture that implemented these ideas has shown that only the most primitive of behaviours can exist without some higher level of abstraction, however, the "intelligence without representation" is an interesting comment on the ideas of spatial representation.

In this work we ignore dead-reckoning error; in practice, one can use one of several existing techniques to update continuously the robot's position estimate [**8**] [**38**] [**40**] [**20**] [**32**].

2.4. Conclusion

There are some interesting and unusual approaches to multi-agent robotics that are worth mentioning. Russell has developed a method of leaving short-lived heat trails, that allow agents to find one another at a distance by searching and following the trails of heat [**50**]. A similar approach is used by Deveza [**17**] to use chemical odours for navigation. These techniques, motivated by ethology, are very similar to techniques used by animals in the wild. It reflects the very strong human tendency to navigate by modifying the environment by visual signs.

CHAPTER 3

The Rendezvous Problem

This chapter presents a formal description of the rendezvous problem. We have separated it into two separate sub-problems. The first sub-problem is how to select points in the environment for potential rendezvous; we discuss issues of representation and issues of task dependence. The second sub-problem, the major focus of this work, deals with factors that can affect the success of any rendezvous attempt. We formalise these parameters and propose two main classes of algorithms to solve the problem appropriately. We give examples of each class of algorithm and give examples of the individual algorithms in operation.

Finally, we conclude with an analytical examination of the algorithm performances at certain extremes; we will by analysing the time to rendezvous under the parameters that affect performance. This analysis prefaces the numerical analysis in the following chapter which contains a more detailed depiction of the algorithm performances.

3.1. The Rendezvous Algorithm

In the simplest, idealised, noise-free case, each robot should select the location in the environment that is the most distinctive. Each robot should navigate to this location and wait for the other robot(s) to arrive. At such a time, they should fuse their maps and suitably partition any remaining exploration to be done.

(i) Travel throughout environment
(ii) Record points in the environment that are good rendezvous locations
(iii) At the pre-arranged meeting time, choose the best rendezvous location
(iv) Travel to the best location, and share information with the other agents

Unfortunately, this happy scenario rarely occurs outside of simulation, thus making this thesis somewhat longer than it might otherwise be. The problem in practice is that due to sensor variations or disjoint landmark sets, the agents may not agree on where the single, ideal landmark is situated. Therefore, the problem of rendezvous is also one of determining what action to take when a rendezvous attempt fails. We are interested in strategies that would permit a robot to interleave exploration and attempted rendezvous so that if the rendezvous fails, the robot can continue its work. This approach will allow the strategies to remain robust even in the face of a robot's complete inability to find its associates.

3.2. Landmark Selection

The tractability of most problems in computer science, and in many other disciplines for that matter, depends to a great extent of the particular choice of representation used to solve the problem. For the purposes of the rendezvous problem the representation of the environment will have considerable effect on our ability to choose points in the surroundings that may or may not constitute "good" rendezvous locations – these points we refer to as **landmarks**.

3.2.1. Spatial Representations. David Miller [42] gave three purposes for the spatial representation system in a mobile robot:

(i) It provides a framework for incorporating newly discovered information about the robot's environment.

(ii) It provides the necessary information to do route planning.

(iii) It gives information from monitoring the position of the the robot during task execution.

Our ability to choose which point or points to visit at every rendezvous attempt will be dictated in part by our spatial representation. Therefore, the representation of the environment must be considered carefully. There are two principle classes of spatial representations. The first represents space as a metric grid, where each location is either filled with some object, or empty. This representation uses a discretised map, also referred to as an "occupancy grid". The chief advantage of this representation is that it gives a reasonably non-lossy description of space, up to the level of the discretisation. As a consequence, many other forms of spatial representation and other spatial information, such as object shapes, can be derived from the grid. However, the occupancy grid has two major disadvantages. It is tremendously storage intensive, and suffers from discretisation artefacts in many instances that are difficult to eliminate without increasing the resolution (and therefore the storage requirements). Furthermore, deriving useful information from a metric map, while usually possible, is often computationally intensive. One example is path-planning, which is $\mathcal{O}(n)$ in the number of cells in the grid.

An alternative representation of space is the topological map, which represents space as a set of connected nodes. The arcs of the connected graph can be used to represent a number of properties, such as distance, travel time, or control strategy for travelling from one node to the next. The topological map has the advantage of being storage efficient, and depending on the information stored at nodes and arcs, can provide information such as navigation paths in much shorter time-complexities. However, the topological map suffers in that any spatial information not stored explicitly in the nodes and arcs is lost. For example, not every location in a mapped environment is represented in the topological map. Furthermore, the topological map created for one application may not be easily transformed to another. For example, in a world where travelling up sharp inclines requires a disproportionate amount of power, a map created for navigating in optimal time between nodes is not likely to contain the information for navigating with minimal power consumption.

From the discussion of these two spatial representations arises the first question of which to choose for representing good landmarks. Since the problem of rendezvous only requires storing the location of small set of points in a much larger environment, clearly a topological representation is likely sufficient. The further question then is how to choose points in the environment to serve as landmarks, for future rendezvous.

3.2.2. Landmarks. In the context of cultural environments, typical notions of good landmarks generally rely upon some knowledge of the environment to facilitate rendezvous. For instance, humans often rely upon pre-existing structures such as building doors, monuments or hilltops to identify rendezvous locations. In the context of exploration of unknown environments, no such assumptions can be made about the existence of such structures. Therefore, we define the notion of **distinctiveness**, or **landmarkness**, as a value defined *at every point in the environment*, and use this value to find landmarks in the environment. If the distinctiveness is a function of the agent's sensor(s), then there is no issue of environmental dependence on the ability to find landmarks — every location is a potential landmark.

An example of a rendezvous situation would be two people who agree to meet at a large zoo, never having visited this zoo before. The two people agree to meet on the highest hill at the zoo; for these two people, the hilltops are good rendezvous locations, or landmarks.

As an agent travels throughout the environment, every visited location is evaluated by the agent in terms of its distinctiveness. By restricting landmarks to lie along the robot trajectory, we avoid issues of landmark visibility and viewpoint independence. However, it should be emphasised that these landmarks need not be recognisable as such from afar, such as a mountaintop is. We recognise locations as landmarks by actually visiting them.

> *The zoo where the two people are meeting is moderately forested; as a result, it is very difficult to tell from afar where the hilltops are, and how high they are. So, the two people at the zoo travel to each hilitop, to see how high it is compared to the other hilltops.*

The assumption is that distinctive locations (with respect to some sensor-based computation) serve as locations that all robots can independently select as good landmarks. We refer to the scalar measure of suitability as a rendezvous point as distinctiveness: $D(x, y, \theta)$. The position (x, y) and orientation θ of the robot are commonly termed the *pose* of the robot; the pose vector is $\vec{q} = (x, y, \theta)$. Therefore, for a pose vector $\vec{q}$ we can define $D(\vec{q})$.

This is implicitly a function of sensor data $\vec{f}(q)$, so we have a new distinctiveness function $\hat{D}$, such that:

$$D : (x, y, \theta) \rightarrow \mathcal{R} \tag{3.1}$$

$$\vec{f} : (x, y, \theta) \rightarrow \mathcal{S} \tag{3.2}$$

$$\hat{D} : \mathcal{S} \rightarrow \mathcal{R} \tag{3.3}$$

$$D = \hat{D} \circ \vec{f}(\vec{q}) \tag{3.4}$$

Although the agent's sensor may not return scalar values, some scalar suitability measure can be usually be computed from the sensor. Some intuitive examples of environmental attributes that might serve as distinctiveness measures are: spatial symmetry, distance to the nearest obstacle, or altitude (for 3D surfaces – for example humans might select hill tops). If we choose our the distinctiveness function to be orientationally invariant, then

$$D(x, y, \theta) = D(x, y) \tag{3.5}$$

The possible distinctiveness measures are heavily dependent on the types of sensors the robots have at their disposal. Because the robot assigns a value to every point, a good sensing modality is one that allows the distinctiveness to be defined at any location in the environment, and for which there exists some metric that can order the resultant landmarks in the environment in terms of distinctiveness. This ordering allows the landmarks to be ranked in terms of their likelihood to lead to a successful rendezvous.

3.2.3. Distinctiveness Surfaces. Certain generic properties apply to suitable landmarks and the distinctiveness function $D(x, y)$ independent of the sensing modality. If the distinctiveness function is smooth, locally convex, and has few local extrema or inflection points, then it may be possible to define highly stable and mutually agreed-upon landmarks with great ease; these landmarks could be found by having the robots perform gradient ascent over the measured surface. However, although this strategy is attractive in principle, we believe that in many real environments, sensor noise, occlusion and other factors may make the "distinctiveness surfaces" highly non-convex and thus complicate the process.

This function $D(x, y)$ then defines a surface across the $x - y$ plane. The task of choosing landmarks then becomes the task of identifying the local maxima (or minima, if preferred), of the distinctiveness surface. Figure 3.1 shows an example of a bounded environment and its associated distinctiveness surface. Note that although the surface is not particularly smooth, there are a few definite sharp peaks in the surface that would make good rendezvous points. The very flat areas in the surface are regions of the space that are inaccessible to the robot. These areas were computed by generating an occupancy grid from the map, and then performing a breadth-first, connected-space search of the grid. The distinctiveness was measured only in regions accessible to the search through connected space.

The principal measure of distinctiveness[1] used in this work (and used to generate the surface of Figure 3.1) encompasses both a notion of enclosed space and symmetry, by summing the ranges returned from the sonars, and dividing by the difference of opposing sonar pairs:

$$D = \frac{\sum_{i=1}^{n} R(\theta_i)}{\sum_{i=1}^{n} \mid R(\theta_i) - R(\theta_{i+n/2}) \mid} \tag{3.6}$$

This measure is presented here for reference only; it will be explained further in Chapter 6 during the discussion of the actual robot exploration.

3.2.4. Finding Distinctiveness Peaks. One way to identify the potential rendezvous points, or landmarks, is to sample the distinctiveness surface uniformly across the space, and then identify the maxima in the surface off-line. However, this method has the disadvantage of being

[1]Recall that we have assumed the function is rotationally invariant, so that $D(x, y, \theta) = D(x, y)$. The issues of rotational invariance associated with this distinctiveness function will be address in Chapter 6.

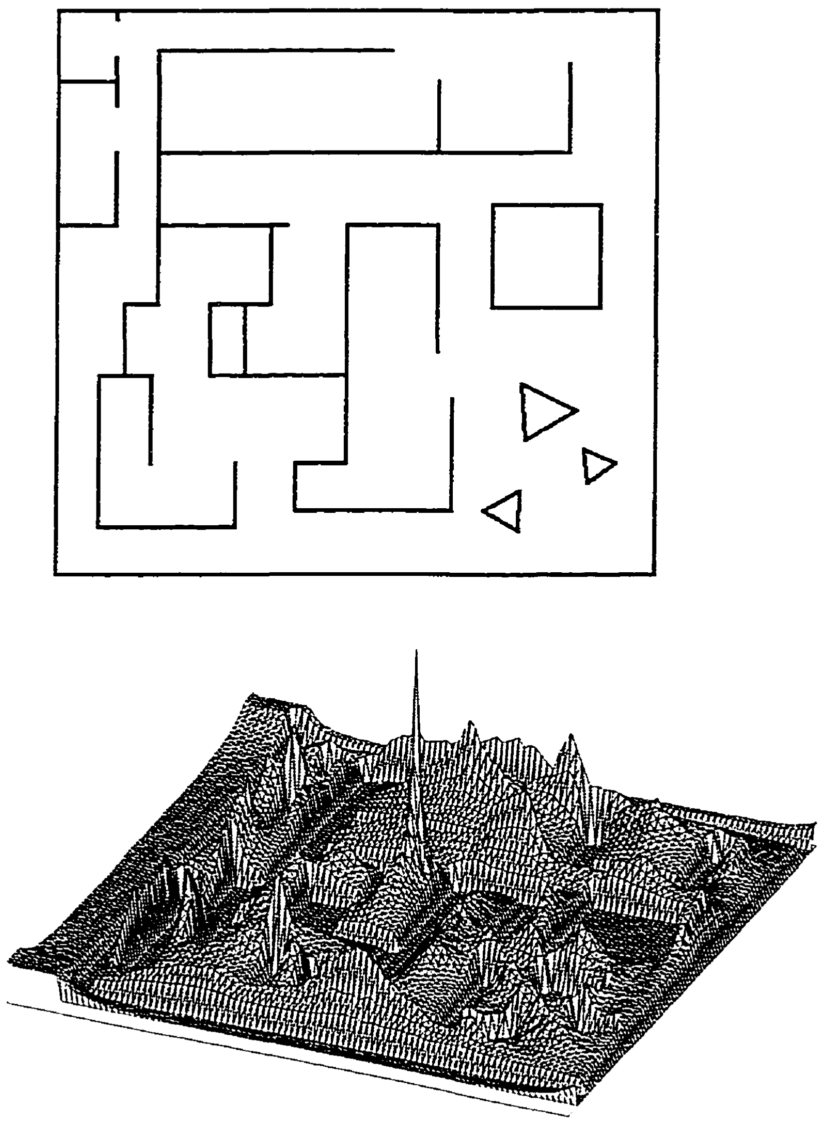


FIGURE 3.1. An example map, and its associated distinctiveness surface.

both storage- and time-intensive. The robot must visit every position in space where the surface will be sampled, and must store the equivalent of an occupancy grid with the distinctiveness for each sample. Since we want only points at the distinctiveness maxima, such an exhaustive sampling would be an unnecessary waste of time and storage.

The notion of a landmark also serves as the basis of the topological mapping strategy proposed by Kuipers [34]. The strategy defined by Kuipers follows contours in the distinctiveness surface until local minima or maxima are perceived, and then performs gradient ascent, hill-climbing to that landmark, which is represented as a node in the graph. Connections between the landmarks are represented as control strategies that allow the agent to follow the distinctiveness contour. However, this strategy has a very tight coupling between the search for landmarks and the actual trajectory of the robot — the trajectory of the robot is dictated by a distinctiveness function at landmarks. The dedication of the agent's resources to the task of identifying landmarks is essential, and Kuipers' agents are constantly actively searching for landmarks without interference from competing behaviour.

The two people at the zoo are primarily interested in seeing the zoo. They therefore limit the hilltops they visit to those hilltops that contain animals they want to see.

However, the task of locating landmarks for rendezvous cannot always dictate the robot trajectory. Although we are developing the technique of multi-agent rendezvous in the context of exploration, we would like to generalise rendezvous to any multi-agent system. As a result, the agents must be able to gather landmarks during the execution of any task. The constraints of some tasks may not allow the agent to suspend execution of the primary algorithm in order to follow the distinctiveness surface, hunting for landmarks.

We therefore must be able to gather landmarks independently of the agent's trajectory, or we must be able to overcome any landmark dependencies on the robot's trajectory through space. We therefore impose two constraints on the distinctiveness function - that it is time-independent and orientation-independent. For example, the "Northern-most" point in the already-explored environment is a poor choice. If the explored area of each robot is circular, then two robots will only have the same "northern-most" point if the environment is highly constrained or if the explored regions are very similar. In Figure 3.2, we see that despite having relatively similar explored regions, the robots will choose quantitatively different landmarks. If the landmarks are separated substantially, either by distance or by some obstacle such as a wall, a rendezvous at these landmarks will fail.

Furthermore, an orientation-dependent function will be equally troublesome. A distinctiveness function that is simply the sum of the front and back sonar ranges will give very different values for a robot looking down a corridor, as opposed to a robot looking at a wall. Unfortunately, most immediately obvious distinctiveness functions are orientation-dependent, especially those that use the sonar rings found on most mobile robots. The solution we have chosen is to sample the distinctiveness function at pre-determined orientations.

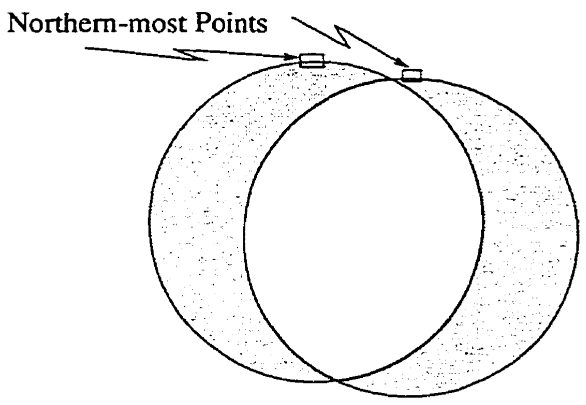


FIGURE 3.2. An example of the effect of choosing a poor measure of distinctiveness. Even though the two robots have relatively similar explored regions, the best landmark each chooses is different enough to cause rendezvous difficulties.

There still remains the issue of spatial sampling — as the agents travel through the environment they must sample the distinctiveness function sufficiently often that they capture the landmarks accurately. Coarse sampling can lead to mis-measuring a peak's height, or even failing to observe a landmark entirely. One possible solution is to be careful to sample the environment as finely as possible. However, since the distinctiveness sampling is a function of the task-dependent trajectory, the problem is independent of the distinctiveness function and must be addressed in some other manner. Possible solutions to this problem will be discussed with rendezvous algorithms.

3.2.5. Inter-agent Differences and Sensor Noise. In addition to using the same distinctiveness function, the agents must compensate for differences in their perceptions of the environment. In order for two robots to agree on a good landmark, they must have similar perceptions of the environment or be able to convert their percepts into a common intermediate form. In the extreme case of two agents with dramatically different sensing modalities, there is essentially no way for them to rendezvous based on the recognition of environmental characteristics. Sensor noise can play a similarly problematic role. We model this aspect of the problem by parameterising the extent to which the two agents can reliably obtain the same measurement of distinctiveness at the same location. The two parameters are systematic differences between agents, and random noise.

We consider the base case, $D(x,y)$, to be "ground truth" with respect to the distinctiveness that should be measured by all the agents. However, $D(x,y)$ is a function of the sensors the agents use:

$$\vec{S}_i(x,y) = \vec{S}(x,y) + \vec{\eta}_i(x,y) + \vec{\lambda}_i \tag{3.7}$$

$$D_i(x,y) = D(\vec{S}_i(x,y)) \tag{3.8}$$

where $\vec{S}(x,y)$ is the ideal perception of the environment by the given sensor, in the absence of any noise. $\vec{S}_1(x,y)$ is Agent 1's perception of the environment at position (x,y) that encapsulates the agent's systematic error $\vec{\eta}(x,y)$ over the measurement at that position; $\vec{\lambda}_i$ is that agent's random sensor noise.

For the purposes of modelling the inter-agent differences, we model λ ad η as scalars, and collapse the random and system errors into one term. With full generality, we can consider one of the agents as the reference perceiver (the arbiter of good taste) with a percept $D_1(x,y) = D(x,y)$ while the other robots obtain a sensor measurement which can be viewed as noisy with respect to that of the first robot:

$$
\begin{aligned}
D_i(x,y) - D_1(x,y) &= \hat{\delta}_i\hat{\eta}_i(x,y) + \hat{\delta}_i D_1(x,y) && (3.9)\\
D_i(x,y) &= (1-\hat{\delta}_i)D_1(x,y) + \hat{\delta}_i\hat{\eta}_i(x,y) && (3.10)\\
& && (3.11)
\end{aligned}
$$

where $\hat{\eta}_i(x,y)$ is all stochastic and systematic noise processes of each robot, and $\hat{\delta}_i$ specifies the extent to which the two robots (D_i and D_1) sense (or perceive) the same thing. If both robots have exactly the same perceptions of the environment we have $\hat{\delta} = 0$. In the context of this formalism, $\eta(\hat{x},y)$ combines both intrinsic sensor noise and any differences in the type of sensor used.

3.2.6. Assumptions. In this research, we have neglected issues of navigation and assume an agent can always accurately reach a desired goal in the environment. While our framework can accommodate navigational error, it is outside the scope of this paper. For concreteness, the reader can imagine a point robot capable of arbitrary motion within free space. We also assume an ideal compass on each robot, to allow perfect orientation. As previously mentioned, we assume an unknown environment, and no shared spatial information or communication between agents except at landmarks.

3.3. Rendezvous Strategies

3.3.1. Formal Parameters of the Rendezvous Problem. Even with this formalism that describes how to gather landmark locations, there remains the issue of how to choose which landmark to visit, and when. In the ideal case, the obvious choice is the "best" landmark, i.e., the point in the environment that has the largest known maximum of the distinctiveness function. However, the ideal case is rarely, if ever, observed in practical mobile robotics. Therefore, strategies must be developed to accommodate the various confounding factors that make the rendezvous problem challenging in practice.

In order to estimate the effectiveness of alternative strategies for rendezvous, we have identified key attributes that must be formalised. Important attributes of the rendezvous problem are:

- Similarities — the reproducibility of the perceptions between agents.
  The similarity of two agents is a function of whether they sense the same environmental attributes, or do they even use the same sensors. This parameter is the δ parameter of Equation 3.10.
- Sensor noise — the distinctiveness measures observed by the two robots are unlikely to be identical. This is expressed by the $\hat{\eta}(x, y)$ term of Equation 3.10, and leads to strategies that must effectively consider a larger number of candidate rendezvous landmarks since the single best candidate may not be determined reliably across agents.
  Note that the for the purposes of modelling differences in sensor measurement across agents, the $\hat{\eta}$ and $\hat{\delta}$ parameters can be treated as a single parameter, δ. Varying the effect of noise on the measurement between agents is akin to varying the noise parameter itself. We therefore will perform all simulations by varying δ alone, and letting $\hat{\eta}$ be a randomly distributed value.

  The people at the zoo cannot measure the height of the hills with a great deal of precision. Therefore, there is some disagreement over which hills are really higher.
- Landmark Commonality — the extent of overlap between the spatial domains of the agents. In the ideal case, the agents will share all landmark knowledge. However, this is clearly not a realistic scenario, and not ideal for exploration. In order for all agents to have all landmarks in common, they would have identical spatial knowledge. This would either demand infinite communication abilities, which is the very limitation we are attempting to overcome, or would prevent the robots from separating, and therefore would prevent any speed-up in any of the multi-agent tasks being attempted.
  More likely is that the robots have explored partially-overlapping areas, and will have some different landmarks that are not in the common region. This is modelled formally as the number d of landmarks out of a total set of n that are not common to the robots. The effect of the non-commonality is that both robots must consider a larger number of candidate landmarks, since any given subset of landmarks selected by one robot may not be known to the other robot. If the agents have no landmarks in common, any rendezvous attempts are

doomed to fail. This is possible any time two agents do not have any part of their trajectories in common, or where the common area contains no landmarks.

> *The two people at the zoo do not have time to visit every hilltop. They therefore miss some hilltops, and do not have the chance to visit every hilltop to see how high they all are.*

- Synchronisation — the level of synchronisation between the agents.

  If the agents cannot agree to meet at the same time, rendezvous becomes a much more difficult task. Further confounding factors may involve the time delays implicit in travelling between landmarks. If an agent fails to make a rendezvous because the path is blocked, or if there is insufficient time to travel to the landmark, appropriate action must be taken by the other agents. The probability that a agreed rendezvous is missed is modelled by the parameter j. This effect leads to a need for strategies that may re-visit the same landmarks repeatedly to compensate for missed meetings. One such behaviour might be to return to very good landmarks more often, while another strategy might dictate actively seeking out alternative landmarks.

  > *The two people at the zoo do not have synchronised watches. Also, one walks a lot slower. Therefore, sometimes they miss each other on their rendezvous attempts.*

- Landmark Cardinality — the number n of points considered for rendezvous by each agent.

  If there is exactly one landmark, then the rendezvous algorithm cannot make any attempt to compensate for variations in the problem parameters, including asynchrony. In this extreme case, the problem is "solved" simply by waiting at (i.e., revisiting) that one landmark until the other robot arrives, or the batteries run down. At the other extreme, if every point visited is considered as a landmark, the algorithm may be swamped, preventing it from exploiting its abilities to find the other agents.

Implicit in the description of these attributes are certain assumptions. It is assumed that if the agent roles are asymmetric, that there is an *a priori* agreement of which agent will play which role. It is assumed that all agents share some notion of synchronisation — that is, all agents can agree on when rendezvous attempts should be made, however, this synchronisation may be noisy. The second assumption is that all agents have the same landmark set cardinality — they all attempt rendezvous over the same number of landmarks (even if they are not using identically the same landmarks in their sets). Finally, it is assumed that all agents are performing the same task, and using the same rendezvous strategies.

3.3.2. Landmark Selection Algorithms. Looking to biology, some simple algorithms are observed. Most animals rely upon well-known common meeting points, such as a beehive or a watering hole. In an unknown environment with mutually unknown starting locations, however, such an absolute reference point is almost impossible to define. A common strategy has one agent (e.g., a child lost at the zoo) wait to be found while other agents (e.g., desperate parents) cover the space, performing search. Another equally naive but much less common strategy has agents moving from landmark to landmark randomly until a rendezvous occurs.

We have developed two main classes of algorithm: deterministic and probabilistic. The deterministic class of algorithm creates a list of all possible combination of landmarks and specifies the order the landmarks should be visited. There is no random aspect to the landmark sequence, and therefore the algorithms will generate the same landmark visits for a given landmark set. The probabilistic class of algorithm does not generate an *a priori* ordering of landmarks, but simply generates probabilities for landmarks being visited.

3.3.2.1. *Deterministic Algorithms.*

Given the same set of landmarks and associated distinctiveness values, these algorithms will always create the same ordering of landmarks.

- **Sequential** – One robot picks a landmark and waits there for the other robot, which visits every landmark in turn. If the second robot has visited every landmark without encountering the first robot, the first robot moves to another landmark it has not yet visited.

For example, given the two sets of landmarks (A, B, C) and (D, E, C) and two robots, one robot assumes the passive role, and the other the active role. They visit the landmarks in the sequence described in the Table 3.1. Note that the actual distinctiveness values of the landmarks are not listed; what is important to the sequential search is not the actual values, but the relative ordering of the landmark values.

The active robot has landmarks (A, B, C, E) where A is the best landmark it has observed so far. The passive robot has landmarks (D, B, E, C) where D is the best landmark observed so far. These sets demonstrate the notions of agent dissimilarity, in that the active agent believes that C is a better landmark than B, whereas the passive agent believes that B is a better landmark than C. Furthermore, the active agent has explored the area with landmark A, which is unknown to the passive agent.

As the visit sequence indicates, the active agent cycles through all its landmarks, before returning to the beginning of the set. The passive agent remains at a landmark for n cycles, where

Robot	Set	Sequence
Active	A, C, B, E	A B C E A C B
Passive	D, B, C, E	D D D D B B B

TABLE 3.1. The sequence of landmarks visited by the sequential algorithm, from the given landmark sets. Note that rendezvous occurs on the 7th iteration.

n is the size of the landmark set, before moving to the next landmark. This generates a list of all pair-wise combinations of landmarks, sorted by distinctiveness.

Although this strategy is asymmetric across agents, the extension from a pair of agents to an arbitrary number of agents can be easily accomplished by evenly dividing the agents into two classes of active and passive agents. Since multi-agent rendezvous, especially for tasks such as map-merging, does not require all agents be present at any given rendezvous, there will be task speed-up from any subset of agents completing rendezvous.

This strategy is simple and relatively immune to noise because it does not rely heavily on the relative rankings of landmarks. However, it is sensitive to asynchrony. If the two robots have the same ordered landmark sets but suffer from synchronisation problems and hence miss meetings at commonly-selected landmark, n iterations must pass before an identical pair of landmarks occurs in the visit sequence. An even greater problem is that low landmark commonality in the landmark sets will cause a delay of up to $d \times n$ iterations before a rendezvous will occur, where d is the number of landmarks unique to an agent. In the above example $d = 1$, since each agent has one unique landmark. Since both of those landmarks were unknown to the other robot and were the best landmark in both sets, we see a delay of $d \times n = 1 \times 4$. For n of four the cost is minimal, but for a much larger landmark set, the penalty could be much worse. The following algorithm is more robust to this type of error.

- **Smart-sequential** – Each pairwise combination of landmarks known to a robot is assigned a "goodness" value. This value is the product of the distinctiveness of the pair. The list of landmark pairs is sorted by this product, and one side of each pair is discarded, leaving an ordered list of n^2 landmarks from a set of n. The robot then visits the landmarks in this order.

The smart-sequential strategy takes into account the fact that the landmarks may be mis-ordered across agents, but takes advantage of sensor data by assuming that the relative mis-orderings are likely to be small. If δ is low, landmark combinations with high values are explored before landmark combinations where one landmark has a very high value and the other has a relatively low value. The lists of landmarks can be thought of as being "perturbed" rather than grossly mis-ordered. This leads to an increased probability of meeting even with substantial asynchrony, or with high-valued landmarks that are unique to one agent. The smart-sequential method is tantamount to guessing where the other robot might be, given relatively similar, but not identical, landmark rankings.

For example, given the two sets of landmarks (A, B, C) and (D, E, C) and two robots, we again have one robot assuming a passive role, and the other the active role. It is possible to preserve the agent symmetry by making all agents take one role, but since the smart-sequential method generates pair-wise combinations, the method is more effective when one agent *a priori* takes one "side" of the list, and the other agent takes the other "side".

They visit the landmarks in the sequence described in the Table 3.2. Note that the distinctivenesses are listed, because of the dependence that the smart-sequential method has on the actual values, as opposed to the relative ordering.

Robot	Set	Sequential Visit	Smart-sequential visit
Agent 1	A=9, C=8, B=7, E=2	A A A A C	A C
Agent 2	B=11, C=10, F=5, E=3	B B B B C	B C

TABLE 3.2. The sequence of landmarks visited by the smart-sequential algorithm, from the given landmark sets. Note that rendezvous occurs on the 2nd iteration.

Each agent generates a list of every pair-wise combination of landmarks. The distinctiveness of the two landmarks is multiplied, to give a value for the pair, and the pairs are then sorted on this product. Table 3.3 shows the combinations, the values and the visit sequence generated from the landmark sets given in Table 3.2. As the table clearly shows, despite unique landmarks (landmarks A and F), and relative mis-orderings between the agents (C & B are reversed between the agents), a rendezvous should still occur on the second iteration — a substantial improvement over the seven iterations necessary for the success of sequential under the same conditions.

Agent 1 Pair	Comb. Value	Visited	Agent 2 Pair	Comb. Value	Visited	Success
A A	99	A	B B	121	B	A-B : No
A C	72	C	B C	110	C	C-C : Yes
C C	64	C	C B	110	B	C-B : No
C A	63	A	C C	100	C	A-C : No
A B	63	B	B F	55	F	B-F : No
B A	63	A	F B	55	B	A-B : No
C B	56	B	C F	50	F	B-F : No
B C	56	C	F C	50	C	C-C : Yes
etc...						

TABLE 3.3. The pair-wise combination of landmarks for the two robots, their values, and the visit sequences generated.

Smart-sequential has its domain of superiority where the agent differences (e.g. noise) are low, but not negligible, or where the landmark sets are not identical. Although it is not a probabilistic strategy as such, it essentially groups landmarks together into types of high probability through low probability, in attempt to "guess" where the other agent(s) might be. Landmarks are allowed to be revisited regularly, but as it moves through the list from high to low distinctiveness factors, the revisit rate for good landmarks drops considerably.

Smart-sequential also does not perform well under conditions of high noise or high asynchrony. It suffers under conditions of high noise, because it relies upon a reasonable, if not 100% accurate knowledge of the distinctiveness surface; as noise destroys the accuracy of that measurement process, the estimates based on that knowledge become poor. This algorithm also suffers under the condition of high asynchrony, because there is no provision for re-visiting a landmark more than n times, in a set of n landmarks. The problems with the distinctiveness function and asynchrony led to the development of probabilistic landmarks.

3.3.2.2. *Probabilistic Algorithms.*

The landmarks are sorted with respect to their distinctiveness and then assigned a likelihood of visitation p_i for landmark i as a function of its rank in the sorted list i.e $p_i = f(i)$. The algorithm probabilistically selects a landmark to visit, using p_i for each landmark.

The probabilistic algorithms use different probability functions to accommodate different parameters of the problem space. For example,

- **Exponential** – The likelihood of visiting the $i-th$ best landmark is $\propto e^i$. This function has the effect of emphasising the relatively highly distinct landmarks, at the cost of landmarks with relative low distinctiveness.
- **Random** – On each attempted visit, each robot selects a landmark at random and goes there.

The particular exponential function used in the simulations was

$$E_i = \frac{10^4 e^{\tau(D_1 - D_i)}}{\rho} \qquad (3.12)$$

$$\tau = \frac{.25log(.001/D_1)}{D_1} \qquad (3.13)$$

$$\rho = \sum_{i=1}^{n} E_i \qquad (3.14)$$

$$P_i = 100\frac{E_i}{\rho} \qquad (3.15)$$

where D_i is the distinctiveness of landmark i, E_i is the output of the exponential function for landmark i, and P_i is the probability of visiting that landmark. The constants in these formulae were chosen empirically. ρ is a normalisation constant to ensure that the probabilities for the landmark set sum to 1.0, and ρ is a user-definable decay constant for tuning the exponential function response.

Robot	Set	Landmark Probabilities	Visit Sequence
Agent 1	A=9, C=8, B=7, E=2	A=39% C=30% B=24% E=7%	A C C
Agent 2	B=11, C=10, F=5, E=3	B=44% C=36% F=12% E=8%	B C C

TABLE 3.4. The sequence of landmarks visited by the smart-sequential algorithm, from the given landmark sets. Note that rendezvous occurs on the 2nd iteration.

Agent 1 τ = -.252 $\rho = 2.549$		Agent 2 τ = -.211 $\rho = 2.270$	
$E_A = 1$	$P_A = 39.2\%$	$E_B = 1$	$P_B = 44.0\%$
$E_C = .776$	$P_C = 30.4\%$	$E_C = .809$	$P_C = 35.6\%$
$E_B = .603$	$P_E = 23.6\%$	$E_F = .281$	$P_F = 12.4\%$
$E_E = .170$	$P_B = 6.6\%$	$E_E = .184$	$P_E = 8.1\%$

TABLE 3.5. The table showing the output of the exponential function for the landmark sets given in Table 3.4. E_i is the output of the exponential function, and P_i is the probability of visiting that landmark.

Table 3.4 shows a set of sample landmarks, and their associated probabilities, as calculated from Equation 3.13. The full expansion of the probability calculation from the landmark values is given in Table 3.5. Once the probabilities have been calculated according to the appropriate formula (exponential, in the example case), a random process can easily generate a sequence of landmarks to visit. In the example given in Table 3.4, the agents made a successful rendezvous in three iterations.

3.4. Analytical Analysis

We can make an analytical assessment of the performance of the deterministic rendezvous algorithms, compared to the random algorithm baseline. If there is no noise, no asynchrony, and 100% landmark commonality, then all of the algorithms which use the distinctiveness measure to sort landmarks will lead to a rendezvous after only one attempt (i.e., both robots will go straight to the mutually agreed upon best landmark.). The random algorithm can never assure a rendezvous but will have a small, equal probability of leading to a rendezvous on every attempt.

More interesting is the performance of the algorithms in the limit of high noise, $\delta = 1$, such that no common ordering between agents of the same landmarks can be reliably determined. The first assessment is the algorithmic time complexity, i.e., the expected time to rendezvous, for the three algorithms in the limit of $\delta = 1$. The expected time to rendezvous is the median number of attempts before a successful rendezvous. For a landmark set of size n, the probability of any single, random rendezvous attempt being unsuccessful is:

$$P_{unsuccessful} = \frac{n-1}{n} \tag{3.16}$$

If the asynchrony rate is accounted for, then the probability of an attempt being unsuccessful rises to

$$P_{unsuccessful} = \frac{n-1}{n} j \tag{3.17}$$

These equations give rise to table 3.6. The first column refers to both robots having the same set of landmarks. The second column considers the scenario where the robots may fail to get to the appointed landmark at the same time (or fail to notice one another). This probability is the asynchrony, j. The third column deals with the case where d of each robot's n landmarks are not in the other robot's landmark set.

Algorithm	Simple	Async.	$<$ 100% Comm.
Random	$\frac{1}{log_2(\frac{n}{n-1})}$	$\frac{1}{log_2(\frac{n}{n-1})+log_2 j}$	$\frac{1}{log_2(\frac{n}{dn-d})}$
Sequential	$n/2$	$\frac{n}{2} + j^{\frac{-1}{\log j}}$	$\frac{n}{2} + \frac{d}{n}^{\frac{-1}{\log \frac{d}{n}}}$
Smart-seq.	n	$n + j^{\frac{-1}{\log j}}$	$n + \frac{d}{n}^{\frac{-1}{\log \frac{d}{n}}}$

TABLE 3.6. Expected case behaviour. The columns denote the ideal case, the case where the asynchrony $j \neq 0$ and the case where the landmark sets are not identical, but each agent has d non-common landmarks.

In the deterministic sequential algorithm, the expected time of the simplest case (identical landmark sets, no asynchrony), is very straightforward. One agent sits at a landmark, and the other agent visits every landmark in turn until they meet — on average $n/2$ landmarks. However, in the presence of asynchrony, additional sweeps of all n landmarks will have to be performed. To find the expected number, k such additional sweeps, we use

$$0.5 = j^k \tag{3.18}$$

noting that each extra sweep i of k will reduce the probability of failure, and k such sweeps must reduce the probability of failure to 50%. Thus, on average $j^{\frac{-1}{\log j}}$ sweeps during the rendezvous will fail due to asynchrony. Similarly, for non-identical landmark sets, additional sweeps of n landmarks will have to be performed on average $\frac{d}{n}^{\frac{-1}{\log \frac{d}{n}}}$ times.

In the worst case, the performance time complexity is much more straightforward. For the probabilistic algorithms, such as the random strategy, or whenever asynchrony is an issue, the worst-case is always $\mathcal{O}(\infty)$, because a meeting can never be guaranteed. Similarly, a rendezvous can never be guaranteed if any asynchrony is present, and so for $j \neq 0$, the worst case for all algorithms is $\mathcal{O}(\infty)$.

However, the deterministic algorithms are guaranteed to terminate when $j = 0$. In the worst case, the two algorithms terminate in n^2 iterations when they share no common landmarks. At this point, both agents can determine that they cannot meet, and continue exploration. If, however, the agents share identical landmark sets, the sequential algorithm has a much lower worst-case complexity than the smart-sequential strategy, because one agent is guaranteed to visit every landmark in the other agent's set in n iterations.

Algorithm	Simple	Async.	$<$ 100% Commonality
Random	∞	∞	∞
Sequential	n	∞	$nd + n$
Smart Seq.	$n^2 - n + 1$	∞	$n^2 - (n - d) + 1$

TABLE 3.7. Worst case behaviour. Columns as in Fig. 3.6.

A full derivation of these results is in Appendix A.

3.5. Conclusion

In this chapter, we have described how to choose points in the environment appropriately, regardless of the underlying task that the agents are performing, independent of rendezvous. We also show how to represent the landmarks without depending on a particular representation of space (e.g. metric or topological). Especially relevant to the problem of rendezvous are certain formal parameters that we have outlined. We describe two principal classes of rendezvous algorithms, *deterministic* and *probabilistic* — these algorithms determine how to choose what landmark to visit, especially when the first attempt fails.

In the next two chapters, we will show that the choice of an appropriate rendezvous strategy depends on the extent to which the robots have found the same set of landmarks, the amount of sensor noise (or, equivalently, the similarity of the sensors) and the reliability of the robots being at a mutually selected rendezvous point and detecting one another.

CHAPTER 4

Numerical Simulation

In Chapter 3, we formalised the parameters of the rendezvous problem and described several algorithms that could deal with the problems of rendezvous in different ways. We gave an analytical description of the algorithms, but neither the expected case estimates nor the worst-case bounds in the limit of high noise provided a realistic picture of the the performance of the algorithms, as this limit will rarely, if ever, occur.

The sensor differences, $\vec{\lambda}$ will likely not be extremal. Therefore, more useful than the analysis in the limit of high noise in determining the performance of the algorithms is the performance of the algorithm under conditions of worsening noise, especially under different conditions of disjoint landmark sets and asynchrony.

One way to determine the ability of a particular algorithm to allow agents to rendezvous quickly would be to allow the physical robots to explore an environment and actually rendezvous. However, while this method has the advantage of providing convincing results, it has considerable practical difficulties, in that a number of effects may be introduced (this is in fact the case, as we shall see in later chapters) by the nature of the exploration task, the physical robots, the spatial domain in which the robots move, etc. Before attempting to run the rendezvous algorithms on the actual robot, we would like to analyse of the rendezvous algorithm alone, independent of any of the practical considerations of using robots that will introduce new constraints on the problem.

This chapter presents a numerical analysis of the behaviour of the various algorithms under different conditions, and especially performance with increasing noise. First, we discuss the method of simulating numerically the landmark acquisition and rendezvous process. Subsequently, we demonstrate the results obtained, and some of the interesting features of these results.

4.1. Experimental Method

Rather than simulating an actual exploration, two agents were modelled as having already explored an unknown area, and having collected a set of n landmarks.

(i) The distinctiveness values of the ordered landmarks were generated with a function, $f(x)$ where x was the landmark index.

(ii) Random noise δ as developed in equation 3.10 was then applied to the two sets.

(iii) The appropriate rendezvous strategy was then used to generate a sequence of landmarks for the two agents, with a maximum length of n^2.

e.g. $\{l_1, l_2, l_3, l_4, \ldots, l_n\}$, $\{m_1, m_2, m_3, m_4, \ldots, m_n\}$

(iv) The sequences were terminated at the first position with the same landmark, and the running time was considered to be the length of the sequences.

e.g. $\{l_1, l_2, l_3, l_4, \ldots, l_i\}$, $\{m_1, m_2, m_3, m_4, \ldots, m_i\}$ where $l_i = m_i$.

It should be noted that the parameter space of this problem is substantial, and therefore not all aspects of the problem were explored. Only the more relevant and interesting aspects are presented here.

4.1.1. Generating Simulated Landmark Sets.

The first step in our simulation is to generate the distinctiveness values for the landmarks. A number of functions were considered for generating typical landmark sets, e.g., random, linear, and exponential functions with varying slopes and decay constants.

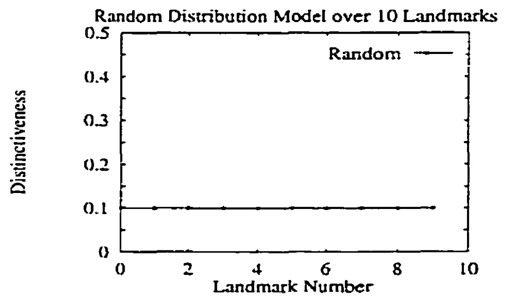


FIGURE 4.1. Example random distribution function of landmark distinctiveness values. All 10 landmarks are equally distinctive in this case.

Figure 4.1 shows a typical distribution function for landmark distinctiveness values. Clearly, this is a less than ideal scenario, since the point of this exercise is to differentiate positions in space by their distinctiveness values. If all points in space have the same value, then they are in fact indistinguishable on basis of distinctiveness. A sensing modality that causes such a uniform distribution of distinctiveness would be a poor choice; we would indeed be wise to choose either a different sensing modality, a different distinctiveness function, or a different line of work. Such a distribution is also unrealistic in that no real sensor will generate such a uniform distribution — noise alone will generate different distinctiveness values.

The linear function was used for generating the most realistic set of landmark values, and for the numerical analysis presented here. Figure 4.2 show a typical linear distribution.

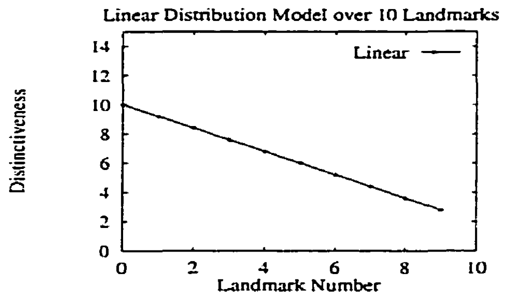


FIGURE 4.2. Example linear distribution function of landmark distinctiveness values.

4.1.1.1. *Modelling Noise.* The landmark set is generated by a distinctiveness distribution model $F(i)$, with a range of values, $[0, max(F(i)]$. The noise was then modelled as a percentage δ of full scale:

$$D_i = F(i) + Random(0 : \delta \cdot \max F(i)) \tag{4.1}$$

The random function was a uniform random function in the range $[0 : \delta \cdot \max F(i)]$, with mean $\frac{1}{2}\delta \cdot \max F(i)$.

The distinctiveness values D_i were then re-normalised into the range $max(F(i))$. Figure 4.3 shows typical sets of landmark values for two agents, using the linear function for generating values, and a δ of 0.1 applied to the resultant values. At $\delta = 0$, the two ordered sets are identical. The x axis shows the landmark number, and the y axis is the distinctiveness of that landmark. Clearly, the two agents share roughly the same perception of the environment, but there are enough differences that the relative ordering of any two landmarks between the agents can be quite different.

Sorting the landmarks by distinctiveness value gives rise to Table 4.1, where the different relative orderings becomes much clearer. The numbers represent the indices of the landmarks, x, used by the linear distribution function, $f(x)$. Note that in the limit of no noise, $\delta = 0$, the landmark index is the same as its position in the sorted list.

4.1.2. Simulating Rendezvous. To simulate rendezvous, we simulated visiting landmarks by selecting landmarks based on their distinctiveness values. One of the previously defined strategies

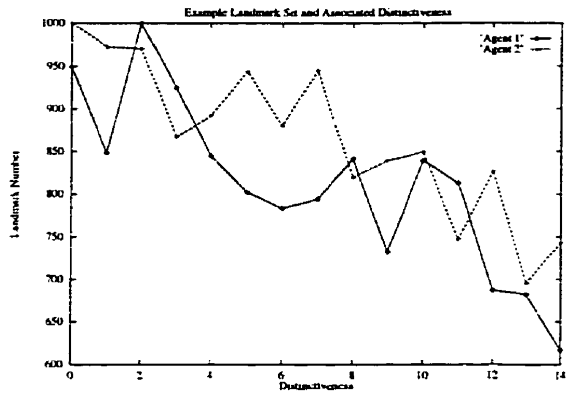


FIGURE 4.3. Example values of 2 landmarks sets with $\delta = .1$. Due to noise, the agents may not agree on a preference ordering of the landmark. Note the relative mis-ordering between landmarks 2 and 3.

Agent 1	2 0 3 1 4 8 10 11 5 7 6 9 12 13 14
Agent 2	0 1 2 7 5 4 6 3 10 9 12 8 11 14 13

TABLE 4.1. The relative ordering given by the distinctiveness values of the landmarks given in Figure 4.3. The numbers are the landmark indices used by the distribution function $f(x)$.

was used to create a (potentially infinite) sequence of landmarks for each agent to visit. The sequence was terminated at the first instance where both agents had a landmark at the same position in the sequence, corresponding to a successful rendezvous.

Table 4.2 shows example sequences generated by the sequential and smart-sequential algorithm. The smart-sequential sequence is terminated at the second position (the rest of the sequence that would have been generated is shown in italics), because the 2nd element of both sequences is identical.

Agent	Sequential			Smart-Sequential		
Agent 1	2 2	$\boxed{2}$	*2 2 2 2 2*	2	$\boxed{0}$	*2 3 2 0 3 0*
Agent 2	0 1	$\boxed{2}$	*7 5 4 6 3*	0	$\boxed{0}$	*1 0 2 0 1 1*

TABLE 4.2. Two possible sequences of landmarks to be visited, generated by Sequential and Smart-Sequential, from the landmark sets given in Table 4.1. Again, the numbers correspond to the landmark index used by the distribution function, $f(x)$.

We use the time to successful rendezvous as a measure of the algorithm's success. The length of the sub-sequences until rendezvous is used as a measure of time until successful rendezvous. Again, without noise, the deterministic algorithms (sequential and smart-sequential) are guaranteed to generate sequences of length 1, that is, meet on the first try. By generating a sequence for each algorithm under different conditions, (varying δ, the asynchrony j, and the landmark set commonality d), we can measure the time to rendezvous under the various conditions.

4.1.3. Modelling Landmark Commonality. Recall from Chapter 3 that we explicitly parameterise the extent of overlap between the spatial domains of the agents as d landmarks out of the total set of n that are unique to one robot. The effect is that both robots must search through more landmarks, as any subset of landmarks may not contain, or even be known to the other robot. Note that in this case, existing algorithms (discussed in Chapter 2, section 2.1 in the literature may fail, unless the the rendezvous is successful on the first attempt.

For this numerical analysis, each agent chooses a set of n random landmarks out of a possible $n+d$, thus giving $n-d$ common landmarks, and d unique landmarks. Which landmarks are unique is generated randomly. These n landmarks are then used by the algorithms to generate the appropriate visit sequences. However, there is a reduced chance that the sequences will match at any given index, since each sequence is now populated with landmarks that have no match in the other sequence. Table 4.3 shows an example landmark set where the commonality is fairly low, $d = 5$ out of $n = 15$ landmarks are unique.

Agent 1	**2**	0	**17**	**1**	4	**19**	10	11	**15**	7	6	9	12	13	14
Agent 2	0	7	**5**	4	6	**3**	10	9	12	**8**	**16**	11	14	13	**18**

TABLE 4.3. The relative ordering given by the distinctiveness values of a linear distribution function, with $\delta = .1$, and $d = 5$, $n = 15$. The numbers are the landmark indices used by the distribution function $f(x)$. The emphasised numbers are unique to each agent.

4.1.4. Modelling Asynchrony. Again, recall from Chapter 3 that we explicitly parameterise asynchrony, j, as the likelihood that any particular rendezvous attempt will fail. This effect leads to a need for strategies that may re-visit the same landmarks repeatedly to compensate for missed meetings.

For this numerical analysis, each position in the visit sequences has a probability assigned to it. If that probability is greater than j, only then is it examined to see whether the sequences match at the position. If the probability of a particular sequence position is less than j, it is skipped.

Table 4.4 shows an example run with asynchrony. Each pair of landmarks in the sequences has an associated probability, listed in the bottom row. If the probability is less than $j = .5$, then it cannot be used and must be skipped. The emphasised numbers are those that have an insufficient probability to be used. Note that the sequences do terminate after three iterations, since the sequences match at the third position, and the probability at $i = 3$ is greater than j. Had this probability been lower, then the sequences would have been at least n landmarks longer, given the sequential algorithm.

Agent	Sequential							
Agent 1	2	2	2	2	2	2	2	2
Agent 2	0	1	2	7	5	4	6	3
Probability	.4	.3	.6	.5	.3	.8	.7	.7

TABLE 4.4. The visit sequences generated by the Sequential algorithm, with the associated probabilities. The asynchrony parameter, j is set to 0.5. Again, the numbers correspond to the landmark index used by the distribution function, $f(x)$.

4.2. Experimental Results

Using the foregoing models, a set of experiments was conducted to simulate the rendezvous process. Each measurement determined the number of rendezvous attempts, or iterations, needed to achieve rendezvous under different conditions. Measurements were taken at 14 values of δ, where each measurement was made 1000 times; these 1000 trials gave a mean number of iterations to rendezvous for a particular algorithm and a particular value of δ.

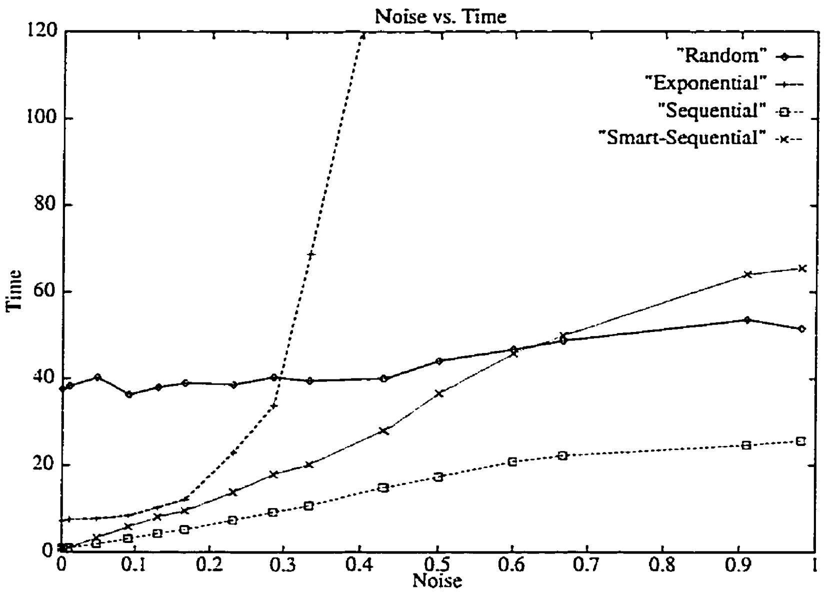


FIGURE 4.4. Baseline Performance - Time to Rendezvous as a function of Noise-level δ

4.2.1. Base case: Time as a function of Noise. The baseline simulation shows the performance of four algorithms in the face of increasing noise. The size of the landmark set is 50 landmarks, asynchrony j is 0, and the landmark sets have 100% commonality. The four algorithms are the deterministic sequential and smart-sequential algorithms, and weighted probabilistic distributions with exponential and linear probability functions. Recall that the exponential probabilistic function, for example, would have an exponentially higher probability of visiting the best landmark on any given rendezvous attempt over any other landmark. Figure 4.4 shows that the sequential algorithm is the best performer, especially in the face of high noise (i.e., $\delta > 0.2$) , which concurs with the analytical result. Clearly, exponential is a very fragile function, failing catastrophically with noise, $\delta > 0.2$.

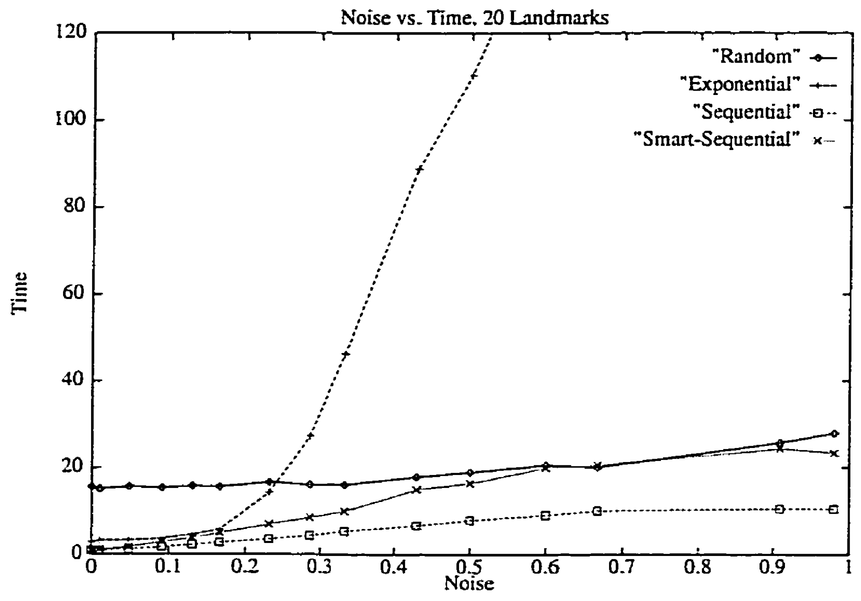


FIGURE 4.5. Performance with 20 Landmarks, ideal case

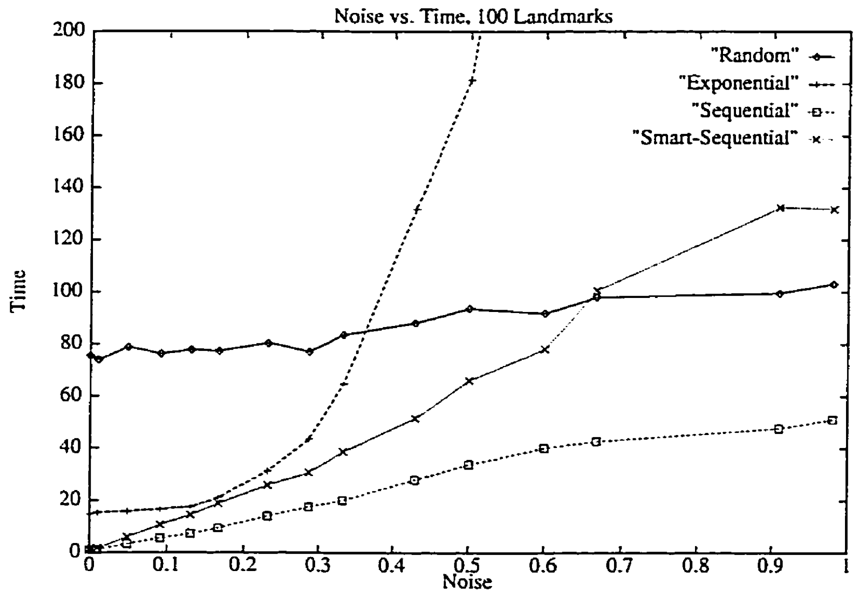


FIGURE 4.6. Performance with 100 Landmarks, ideal case

4.2.2. Different Landmark Set Sizes. Figures 4.5 and 4.6 show the performance of the algorithms with different landmark set cardinalities. Unsurprisingly, the performance of the algorithms scales with landmark set size.

4.2.3. 50 % Asynchrony. In the face of asynchrony, however, the algorithms exhibit less intuitive behaviour. Asynchrony, again, is the probability that a particular rendezvous at a mutually agreed place and time actually occurs. The simulation (which creates landmark sequences) implemented asynchrony as the probability that a particular sequence element could be used. Even if the pair of landmark sequences contained the same landmark at identical positions, the sequence may not have terminated there, because the asynchrony probability prevented the first pair of matching landmarks in sequence from being compared, as if the robots had failed to rendezvous successfully despite attempting to do so at the same location at the same time.

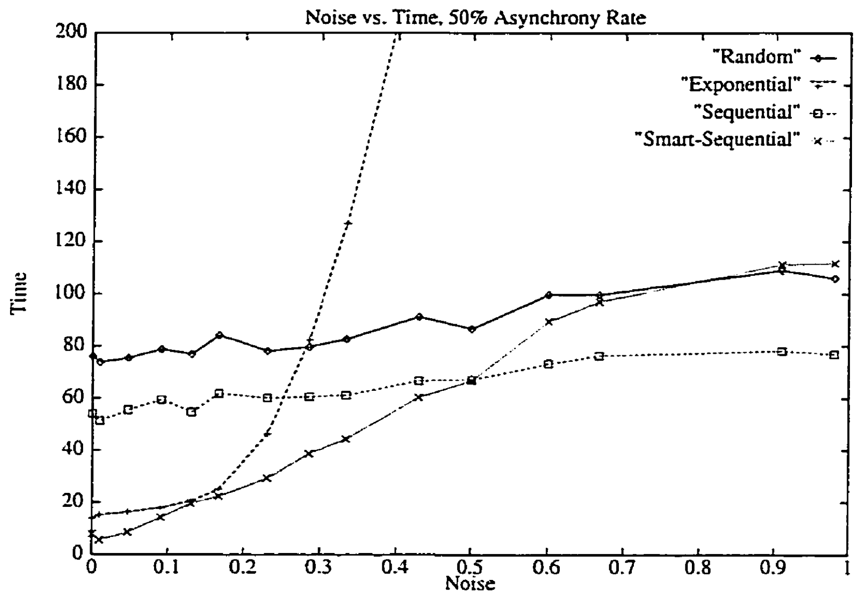


FIGURE 4.7. Performance with 50% Asynchrony rate

Figure 4.7 shows the performance of the algorithms given a 50% asynchrony rate, or a 50% probability of successfully making a rendezvous. In this case, the smart-sequential and exponential algorithms out-perform the sequential strategy, because the sequential form suffers from having to visit every other landmark before being able to return to the landmark that failed on a particular iteration, whereas the other two algorithms can return to landmarks relatively quickly. However, once noise dominates the values, ($\delta > 0.5$) the sequential algorithm outperforms the other algorithms because it does not rely heavily on particular landmark values — it is not returning to the same landmark over and over again.

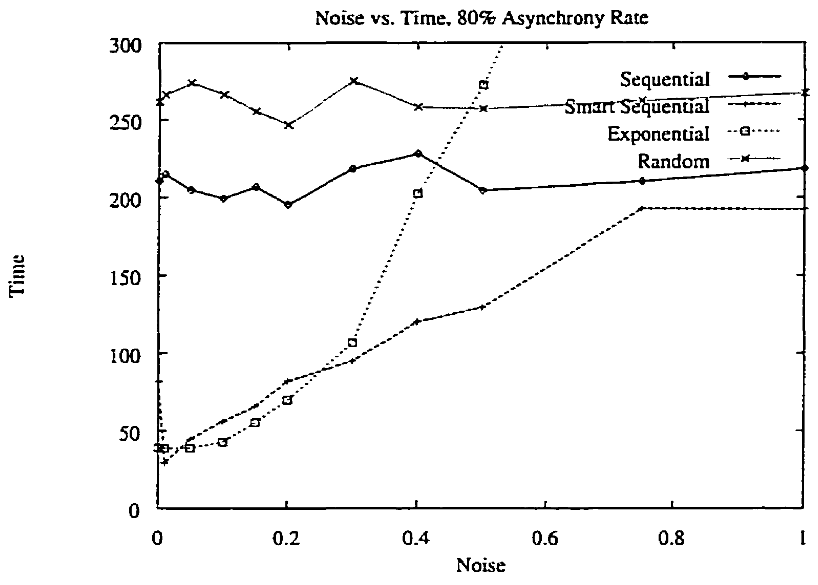


FIGURE 4.8. Performance with 80% Asynchrony rate

4.2.4. 80 % Asynchrony. Even more interesting in the case of very high (80%) asynchrony, Figure 4.8 shows that the exponential probabilistic function outperforms the deterministic algorithms in the face of low noise ($0.5 < \delta < 0.25$), but again fails rapidly in the case of high noise ($\delta > 0.25$). The exponential algorithm essentially forces the robot to return to the same landmark over and over again, which is the correct strategy when asynchrony is high. However, when noise is high, the odds that the recurrent landmark is the wrong one increase, and the deterministic algorithms, which do not return to the same landmark as often, perform better.

4.2.5. 75 % Landmark Commonality. Finally, Figures 4.9 and 4.10 show performance for maps with only 75% of the landmarks in common. The performance with non-identical landmark sets (akin to non-isomorphic maps) is very similar to performance under low- to medium-asynchrony. The smart-sequential algorithm performs better with low noise because it can return to landmarks faster than sequential, but in the case of high noise ($\delta > 0.35$), returning to landmarks too frequently can be costly, and the sequential algorithm again dominates.

4.3. Conclusion

In this chapter, we have described a numerical analysis of the time complexity of the algorithms presented in Chapter 3, and discussed some of the results. The parameter space of the rendezvous problem is sufficiently large that an exhaustive analysis of the behaviour of the algorithms under all conditions is beyond the scope of this paper. What has been described is a numerical analysis of

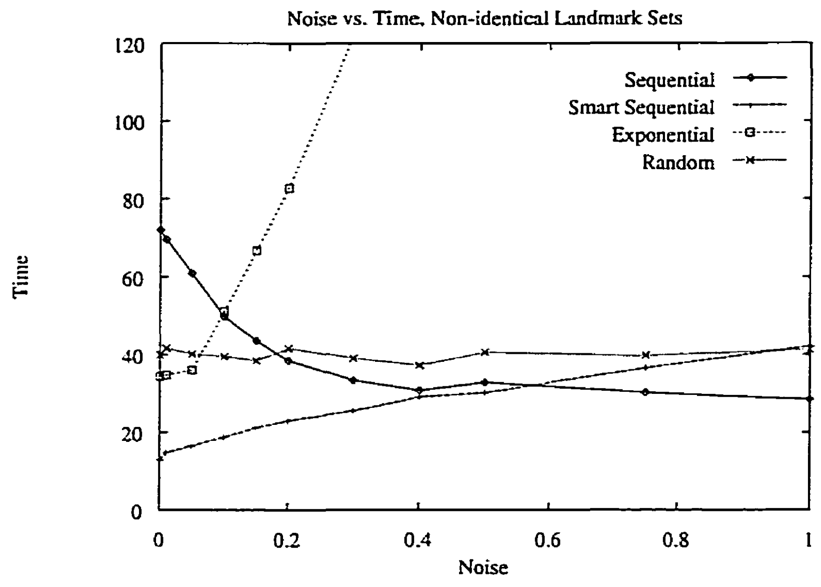


FIGURE 4.9. Performance with 25% non-commonality in landmark sets

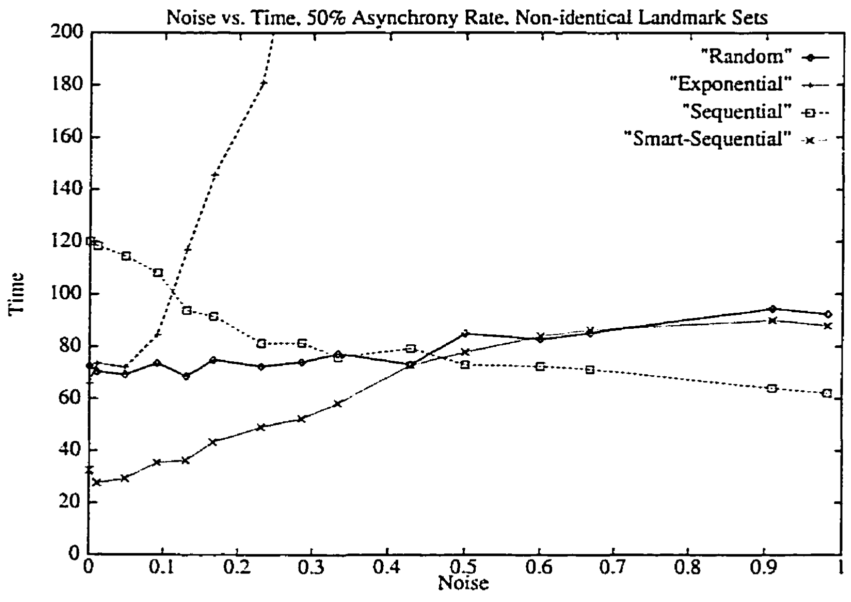


FIGURE 4.10. Performance with non-identical landmark sets, and 50% Asynchrony rate

the algorithms under some of the more realistic conditions that may be encountered by two mobile robots exploring an unknown environment.

In the presence of large amounts of sensor noise (which we are modelling as agent dissimilarity), the landmark selection will be essentially random, in which case the best strategy is simply to have one robot visit every landmark, and have the other robot sit and wait for it. However, this is also an unrealistically pessimistic scenario. If the robots have been constructed to facilitate rendezvous, they are likely to have a somewhat common perception of the environment and to have some commonality in their explored areas. In reality, the robots will probably experience some limited sensor noise, minimal dissimilarities, some asynchrony, and partial but not complete landmark commonality. So, the best strategy takes these factors into account, and chooses a series of landmarks to visit in some intelligent way.

Each of these methods has particular advantages and disadvantages. The sequential method is simple, but makes no effort to account for relative likelihoods, or asynchrony. In view of the potential shortcomings of the sequential method, we have proposed an alternative method, the probabilistic method, that has an increased chance of compensating for a missed rendezvous and also attempts to compensate for small variations in the respective rankings of the landmarks selected by the two robots. For instance, the distinctiveness of each landmark could be the same, which would lead to a uniform random visitation strategy. The probability distribution $f(D(x,y))$ could be a linear function of value, or, if we assume that the amount of sensor noise is low, an exponential strategy. However, if sensor noise is high and the two agents do not share the same ordering of landmarks, then the agents may be forced into revisiting the incorrect landmarks much too often. A good compromise between these two methods is the smart-sequential method. The advantage of this method is that, if $\vec{delta}$ is low, landmark combinations with high values are explored before landmark combinations where one landmark has a very high value, and the other has a relatively low value. This leads to an increased probability of meeting even with substantial asynchrony. The smart-sequential method is tantamount to guessing where the other robot might be, given relatively similar but not identical landmark rankings.

In the following chapters, we present an implementation of the algorithms in the context of both simulated and real mobile robot exploration, and show how the lessons learned in this chapter persist into the implementation on real robots.

CHAPTER 5

Exploration

In Chapter 4, we described some of the more interesting aspects of the performance of the algorithms proposed for the rendezvous problem. The numerical analysis technique showed that the probabilistic algorithms are in many respects much worse than the deterministic algorithms, although each type of algorithm has areas of best performance.

Recall that the context we are using for the rendezvous problem is exploration of an unknown environment. This chapter is a presentation of the particular exploration method used as the underlying application for the rendezvous problem. We show a number of different techniques used for the exploration, and some results of the exploration, before conducting the simulation experiment in Chapter 6.

5.1. Exploration Methods

There are many different algorithms that have developed for exploration of an unknown environment, however, most exploration algorithms are used to describe the spatial characteristics of the environment from one of two standpoints. The first describes the environment in terms of the objects in the environment. The other point of view describes space in terms of the connectedness of the free space. The first method of spatial description most often results in either low-level descriptions of discretised space with free and occupied pixels, or vertex-edge pairs of filled polygons. Alternatively, the spatial description might result in high-level descriptions assigning semantic labels to points in space, that represent objects recognised in the environment, such as a human might construct for describing space. The opposing view of space most often results in a similar low-level description of discretised space or vertex-edge pairs of free space polygons. Alternatively, the spatial description might result in a high-level route map, describing how to get from point to point in space, again such as a human might make for navigating in an unknown environment.

The exploration methodology used by an autonomous agent is driven then by the consideration of the particular spatial representation in use by that agent, but there is also the consideration of the sensors available to that agent. If the spatial representation of the robot demands polygon vertex descriptions, then the exploration algorithm must be geared to the precise and complete examination of as many polygon vertices as possible. Whether the robot has sonar sensors or laser range sensors makes considerable difference as to the speed and precision of acquisition of range data, which again should profoundly affect the choice of exploration strategy.

For the purposes of this research, there is in fact no commitment to one spatial representation over another for the exploration algorithm. The only constraint is that we would prefer an exploration strategy that allows us to explore the space as quickly and as broadly as possible. This is by no means an essential requirement - the sole purpose of the exploration algorithm, besides the meta-purpose of providing a test application of the rendezvous problem, is to provide a trajectory over which the landmark set is acquired. Since landmark acquisition is trajectory independent, the actual trajectory generated by the exploration is irrelevant - we simply want the landmarks along the path. If the path covers a substantial area of of space, so much the better - the experiments will have more chance to elicit interesting results. A second consideration is that the robots that will eventually be used in this research have sonar transducers as their primary spatial sensor. Therefore, the exploration algorithm must respect the limitations of the sensors in terms of range, precision and likely sources of error.

We therefore would like to devise an exploration algorithm that will cover large amounts of unexplored space reasonably rapidly using sonar data. Ideally, the limiting factor of the execution of the exploration algorithm should be the mechanical running time of the algorithm; any exploration method that is dominated by the time to compute the next position is undesirable for our methods. We shall also see in Chapter 7 that an algorithm that will allow also that data to be corrected over time will be very useful.

5.1.1. Topological Mapping. There are a number of different exploration algorithms developed for different purposes that could be used as the basis for our exploration algorithm. Exploration algorithms exist that use both metric and topological representations of space, or combinations thereof. Different algorithms have different purposes, such as speed, or accurate mapping.

One example of an exploration algorithm is the topological robot exploration strategy developed by Kuipers and Byun [**34**]. The notion of distinctive places that drives this exploration strategy is the basis for the landmarks we use for the rendezvous process. This exploration strategy describes the notion of "distinctive" places in the environment, much the same as our distinctive landmarks.

These distinctive places form nodes in the graph, and the nodes are connected by distinctiveness contours - paths through space that have equal distinctiveness.

However, none of the existing algorithms was appropriate to the purposes of this research. As a result, the method we have used is derived from traditional path-planning algorithms. The aim of our exploration method is to move the agent through known space as rapidly as possible to unexplored space. We therefore treat this as the problem of path-planning a route away from known obstacles to unknown space. The approach we use is a path-planning approach known as "potential fields". We also use on-line search to perform global path-planning for generating paths between explored regions of space. By using path-planning algorithms as the bases of the exploration algorithm, we ensure that the robots are moving towards unexplored space as much as possible.

5.2. Potential Fields

The potential field method is inspired from classical physics, and treats space as a potential field such as a gravitational field, or electrostatic field. Goals exert an attractive force, such as a positive charge or a gravity well. Obstacles exert a repulsive force, such as negative charge, or an infinite gravitational potential. Each point in space then has an associated potential, computed from the known obstacles and goals. Using techniques borrowed from optimisation, the agent then moves down through the potential field to the nearest minimum. If the placement of goals is correct, the local minimum should be the nearest goal.

An alternative view of the potential field method treats space as a rubber sheet, where obstacle poke the sheet upwards, and the goals pull the sheet downwards. The robot is ball-bearing on the sheet, allowed to roll freely through the force of gravity. Once the ball is released, it will roll between the obstacles without actually colliding with any of them, and roll into one of the pits where the goal has pulled the rubber sheet downwards. By using this technique, the robot will naturally move away from obstacles towards unexplored space. As it moves between obstacles, it will choose a path equidistant from the surrounding obstacles.

5.2.1. The Exploration Algorithm. The following describes each step of the exploration algorithm:

(i) Acquire sensor data

Given that the sensors available to us with our current robots are sonar transducers, the assumption is that the data returned by the sensors will be of the form (*range*, *angle*), where *range* is from the centre of the robot. Given the current pose of the robot, the (x, y) position of each data point can be easily computed

(ii) Update potential field to reflect new data

We are moving "downhill" through the potential field. The downhill motion should represent motion away from obstacles (e.g. walls, furniture, etc.); these points are therefore represented as points of high potential.

The data points also affect the surrounding area, with some function, just as mass has an effect on the surrounding gravitational field. The potential function used in this work is[1]:

$$\Phi(d) = -\frac{\xi}{\kappa}d + \xi \tag{5.1}$$

We set ξ to 1×10^3, the height of the potential field at the obstacle; κ is a constant that represents the maximum area over which the field is applied (75cm in this case). d is the distance from the data point. It should be noted that the robot may still roll close to a wall, unless additional precautions are taken. These are explained in step 4.

(iii) Update potential field to account for position

In order to prevent the agent from repeated exploration of regions, we allow the agent's position to affect the potential field as well. The agent itself raises the potential to some level, so that even in flat regions of the potential surface, the agent continues to roll forward (propelled into the future by its own history, as it were), until some environmental feature alters its path.

We used the same potential function as for the data points to represent the potential effect of the agent's history, with different constants:

$$\Phi = -\frac{\xi_r}{\kappa_r}d + \xi_r \tag{5.2}$$

where ξ_r is 700. ξ_r is lower then ξ of Equation 5.1 to reduce the effect of the robot potential function compared to that of the sonar data.

(iv) Search for location of minimum potential over local neighbourhood

If either the minimum potential is found to be the current location, or the minimum potential found is above some threshold, then this must be treated as a special case. This special case will be addressed in 5.2.3.

The local neighbourhood used was a circle of radius 30cm (the same radius of the robot). The potential at points around the robot is found by representing the environment as a

[1]This potential function was arrived at empirically in order to maximise the exploration algorithm efficiency; it should be noted that it is not related to the traditional potential functions of gravitation or electromagnetic fields, and does not have the same form.

discretised grid of potentials, much like the occupancy grid stores probabilities. While it would be possible to dispense with the potential grid and analytically determine the local (within the neighbourhood) potential minimum point from the occupancy grid, this would become extremely computationally costly as information accumulated in the occupancy grid. We have instead opted for computing the change in potentials at discrete points in the environment as data is accumulated, and searching the discretised grid.

It should be noted that when a potential is applied to a cell, it only changes the cell potential if the new potential raises the potential at that cell - it is not added, and two different potentials are never combined in any way. Therefore, the wall potential effects usually dominate over the robot potential effects.

(v) Move to the location of the local minimum

By keeping the local neighbourhood relatively small, and ignoring the possibility of dynamic obstacles, we can assume that the straight-line path from the current location to any point in the neighbourhood is also the best path (in terms of distance and time), and path-planning of any non-trivial sort is unnecessary.

(vi) Go back to Step 1.

The advantages to the potential field method are simplicity and effectiveness. The potential field descent automatically encapsulates both known-obstacle avoidance and exploration. Furthermore, the potential field method is a well-established and well-understood method; considerable literature exists both on its implementation and use.

However, the potential field method has some drawbacks, as well. The potential field relies upon a discretisation of space, and as such has the substantial storage requirements of any discretised map of space. Furthermore, it is computationally intensive. Finally, it has two substantial issues which are addressed in the following two subsections, namely the issues of sampling rate, and the presence of local minima.

Figure 5.1 shows an example trajectory through an environment, and the resulting potential field.

5.2.2. Sampling. In order to overcome this issue of computational complexity, the potential is stored for a sampling of points over space, and the local neighbourhood is searched over what points have been sampled in that neighbourhood. As new information about obstacles is acquired by the sensors, only those points in the sample set that are affected by the obstacles have their potential recomputed. It should be pointed out that it is this storage of sampled points that raises the storage

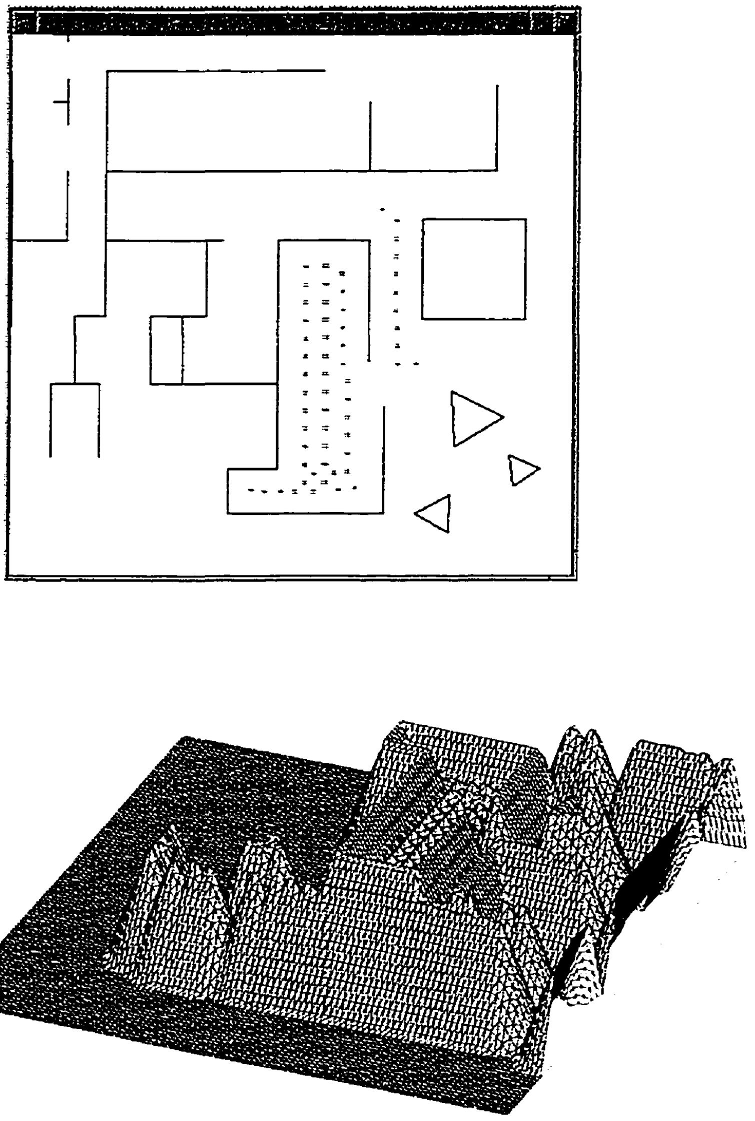


FIGURE 5.1. An example trajectory through an environment, and the resulting potential field.

requirements for the potential field planning method. If the potential could be recomputed quickly for each point in the neighbourhood there would not be the same storage constraints.

Besides the tradeoff between computation and storage, there is a performance balance within the sampling itself - how coarsely to sample space. If space is over-sampled, the storage requirements balloon with limited performance benefit. However, if space is undersampled, then existing paths may not be found within the undersampled representation, local maxima and minima may not be found, and other undesirable effects may occur.

Another issue is how large of a local neighbourhood to search, in order to find the next position. Too large of a local neighbourhood, and the search time dominates the running time of the algorithm. Too small a local neighbourhood, and the mechanical running time becomes unnecessarily high. The size of the local neighbourhood over which the potential is searched for a minimum is akin to the step size that the robot takes every iteration during exploration. The step size should be small enough that the the agent will not miss any environmental features between steps, but should be large enough that the robot can move from point to point in the environment in a reasonable amount of time. In the experiments performed in this thesis, the step size was 30 cm, which is a little bit larger than the robot radius. This is an *ad hoc* parameter, but one that appears to have worked reasonably well.

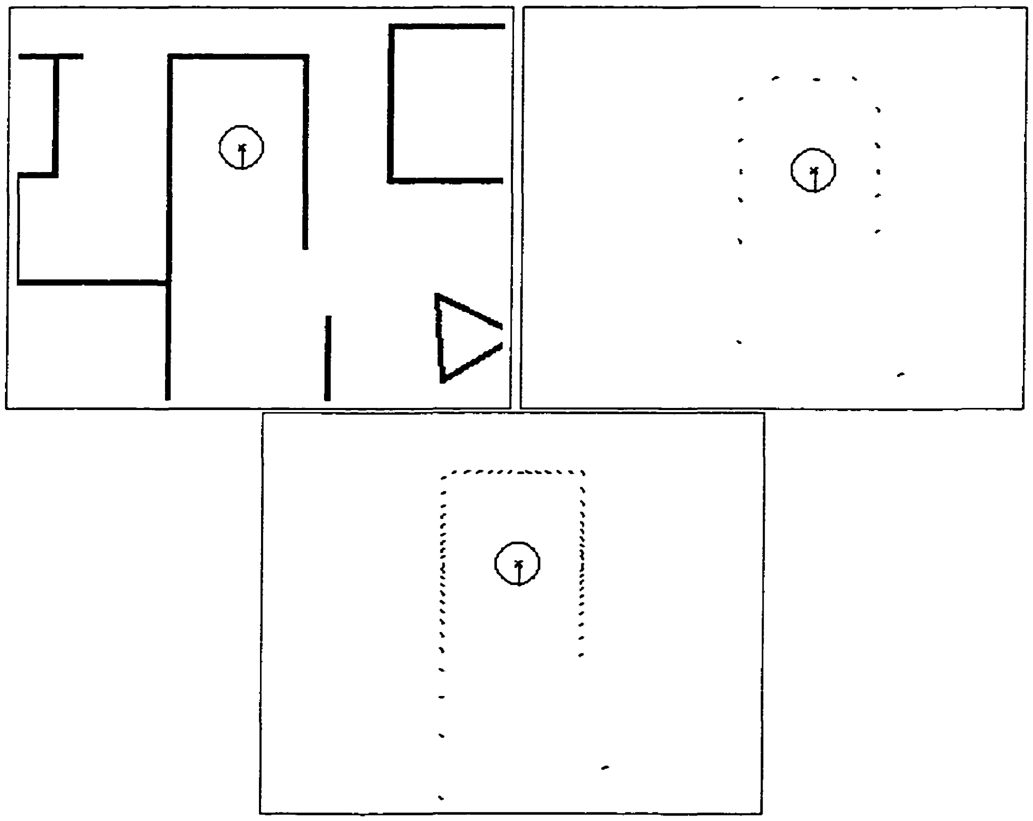


FIGURE 5.2. An illustration of the sensor undersampling problem. The circle in the center of each image is the robot. The upper left image is the actual map, the upper right image is the result of 16 range measurements, where the black dots correspond to where the range measurement was observed. The lower image is the result of 64 range measurements.

The final sampling issue is that of sensor sampling of the environment. At every step, the robot takes a certain number of sensor measurements in the environment. These measurements are range data from the centre of the robot, at equally spaced angular intervals. The robots used for these experiments have 12 (in the case of the RWI B12 robot) and 16 (in the case of the Nomad 200) sonar transducers mounted around the sonar base, at regular angular intervals. These transducers can all be used more or less simultaneously.[2]

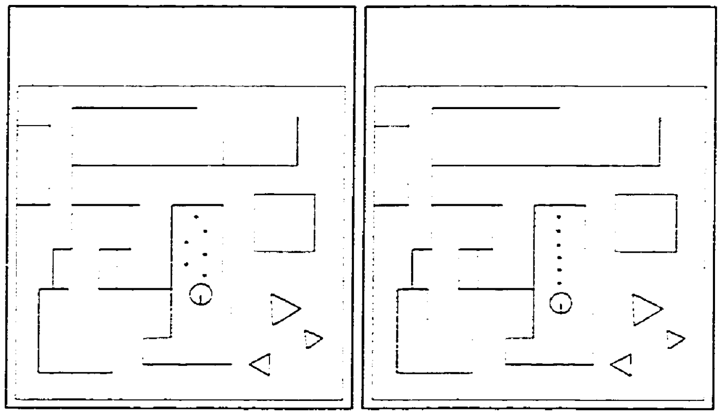


FIGURE 5.3. On left is the trajectory that results from using 16 sonar measurements per iteration. Each cross represents the position of the robot at an iteration. The start position is the same as in Figure 5.2. On the right is the exploration trajectory from the same location, using 64 sonars.

Sixty-four sonar samples were used for our potential field navigation, because fewer sonar samples could capture enough information about the environment to direct the robot in a reasonable fashion. Figure 5.2 demonstrates the sensor undersampling problem. In this example, the 16 range measurements (left box) capture most of the large structure of the corridor, however, they do not emphasise the major escape route down the middle of the corridor, as is emphasised using 64 measurements (right box). As a result, the trajectory of the robot moves between the interleaved "bumps" in the potential field surface. With 16 data points, these bumps are not close enough together to form a coherent barrier, so the surface is rippled. Figure 5.3 shows the robot trajectory moving from left to right across the corridor using 16 sensors, but moving correctly using 64. By using 64 sensors, the surface is still rippled, but the dominant direction is now towards the centre of the corridor, and down the length of the corridor. The ripples do not determine the direction.

The solution to this problem is to ensure that the space is being sampled sufficiently by the sonars. 64 sonar measurements was empirically determined to elicit the expected behaviour from the exploration algorithm. It should be noted that in reality, the sonar beam is not as precise

[2]In principle, adjacent sonar transducers cannot be fired simultaneously due to acoustic crosstalk, but the amount of time to acquire data from all transducers is on the order of a second or less.

as indicated by the diagrams in Figure 5.2. The sonar cone has a spread of about 12°, and the angular displacement between two consecutive sonar measurements of 64 is only 5.6°. Consequently, there will be overlap between measurements at anything more than a meter. How to handle these sonar artefacts is a topic of ongoing research. One possibility is to represent sonar measurements as something other than single range measurements. We have elected to use the perhaps over-simplistic but sufficient range point representation, and accept some inevitable artefacts in the data.

5.2.3. Local Minima. One final concern with potential-field based exploration is the problem of local minima. Recall that at each iteration of the algorithm, we are searching for the minimum potential over a local neighbourhood, and moving to that location. If, however, that position lies at the current location, then the algorithm will become stuck in an infinite loop, never moving the robot. We therefore must trap this special case, and use an alternate method to potential field descent for escaping local minima.

Recall from Step 4 of the algorithm, we also must use an alternate method if the local potential minimum is above some, arbitrary, threshold, τ. This threshold τ ensures that the potential field does not push the robot into a wall. τ was chosen such that the robot can never be close to an obstacle than 1.5 times its radius.[3] Note that this threshold τ also prevents the robot from moving too close to a previous location, but closer to a previous location than a wall.

5.2.4. Breadth-First Search. The failure of the local potential-field descent during exploration suggests a global approach to path-planning problem. We use the need for a more global strategy as an opportunity to perform a global search for distant potential minima - we expand the search neighbourhood to the entire map, searching every point for which we have gathered information. This strategy takes us from the recently explored regions to the fringes of the known environment.

The search region is now substantially larger than the local neighbourhood being searched during the potential field descent, and almost invariably irregular in shape. Consequently, we use an exhaustive breadth-first over the potential field originating from the position of the robot, examining every point in the potential field for which we have sensor information.

The technique for performing breadth-first search on a field such as this is well-established with time complexity $\mathcal{O}(n)$ in the number of cells to be searched; a good a reference is Cormen, Leiserson and Rivest [**16**]. However, there must be some method of determining which of the potential field cells is to be searched. We only wish to search those cells for which we have any sensor information at

[3]From Equation 5.1, given the values of $\xi = 1 \times 10^3$, $\kappa = 75$ and the robot radius of $r = 23$, the threshold must be $\tau = 530$.

all - searching the potential of cells for which we have no information is dangerous, since the potential is almost certainly going to be incorrect. We therefore turn to the metric map, the occupancy grid.

5.3. Occupancy Grids

The initial state of the occupancy grid has every point set to some value representing *unknown*. As the data points are added, the cells at those locations are set to the value representing the probability that the cell is *filled*, or occupied (hence the name). However, each data point represents information not only about one point in the environment that contains an obstacle, but also about every point in the environment between the robot and the obstacle. Inside the occupancy grid representation, these points are set to the value representing the probability that the cell is *empty*. These points are identified by performing scan-conversion across the occupancy grid between the robot position and the data point; the scan-conversion is accomplished using a technique borrowed from computer graphics, the Bresenham or mid-point algorithm. Figure 5.4 demonstrates the scan-conversion algorithm. The grey cells are assigned a low probability of being occupied, whereas the cross-hatched cell is assigned a high probability of containing an obstacle. No other square is affected by the sonar beam - if all other squares are currently unknown, then they remain *unknown*, and only the coloured squares in the image now contain some sensor information.

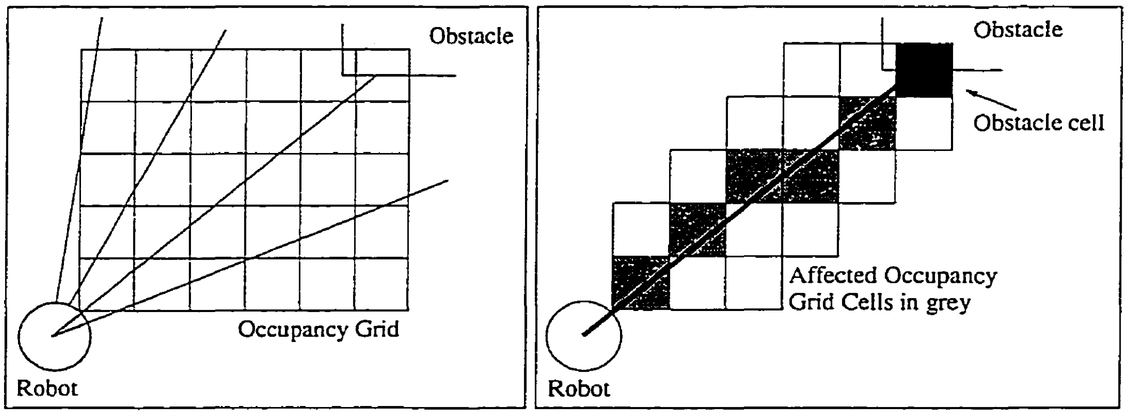


FIGURE 5.4. An illustration of the Bresenham algorithm. The dark grey cells are low-probability occupied, and the black cell is high-probability occupied.

Those cells that are unaffected by the sonar measurement, that remain in the *unknown* state, represent the boundary of the breadth-first search. Recall that the breadth-first search should not explore those cells for which no sensor information has been acquired, as the potential is likely to be wrong. Therefore, by performing flood-fill, breadth-first search, from the current position to the edges of the region that consists of low-probability ($P_i < 0.5$) cells, we can constrain the search. Note that we do not expand cells containing sensor information and have a high probability of being occupied ($P_i \geq 0.5$) because we are looking for an available place for the robot, with low potential. The robot clearly should not be sent to a location containing an obstacle.

5.3.1. Antialiasing. Because the sonar beam model has some finite thickness, the beam is anti-aliased, that is, the points one either side of the beam are given some probability indicating that they are empty as well. This is another technique borrowed from computer graphics, to prevent images displayed on low-resolution raster devices (such as low-resolution printers and monitors) from having jagged edges. We use it in this instance, to account for the width of the sonar cone.

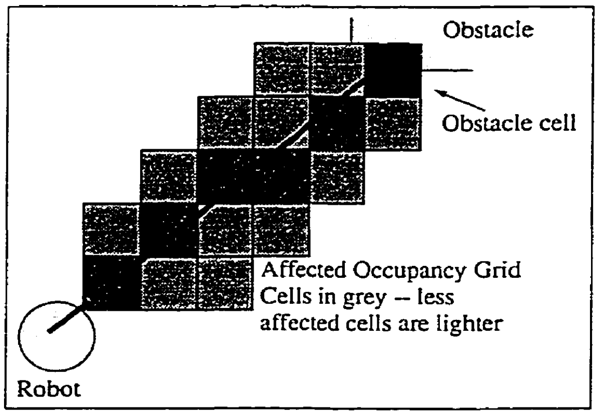

Obstacle
Obstacle cell
Affected Occupancy Grid Cells in grey -- less affected cells are lighter
Robot

FIGURE 5.5. An illustration of aliasing - the lighter squares receive less of a change in probability.

5.3.2. Bayesian Recombination of Data. Frequently, two successive measurements will provide information about the same location in the environment. If a probability is being assigned to an occupancy grid square that already contains a probability, the two probabilities must be recombined in some consistent manner. For this, we use Bayes' rule as formulated for combining independent data [47]:

$$P_{i+1}(x,y) = P_i(x,y)w(x,y) + (1 - w(x,y))v(x,y) \tag{5.3}$$

where $P_{i+1}(x,y)$ is the probability of occupancy at location (x,y) after the recombination, $P_i(x,y)$ is the probability before, $w(x,y)$ is the recombination weighting at location (x,y), and $v(x,y)$ is the new measurement. V will always be either 0, meaning empty, or 1, meaning occupied. The weighting recombination we used in this experiment was uniform for all positions, $w(x,y) = w = 0.5$. That is, the new measurement is equal in weight to the old value.

In principle, the value at a particular point should never change. However, the reality of using sonars as sensors means that some measurements will sometimes be wrong. This probabilistic recombination of data allows us to account for these errors, and by averaging the measurements through time, we should get a more realistic picture of the environment.

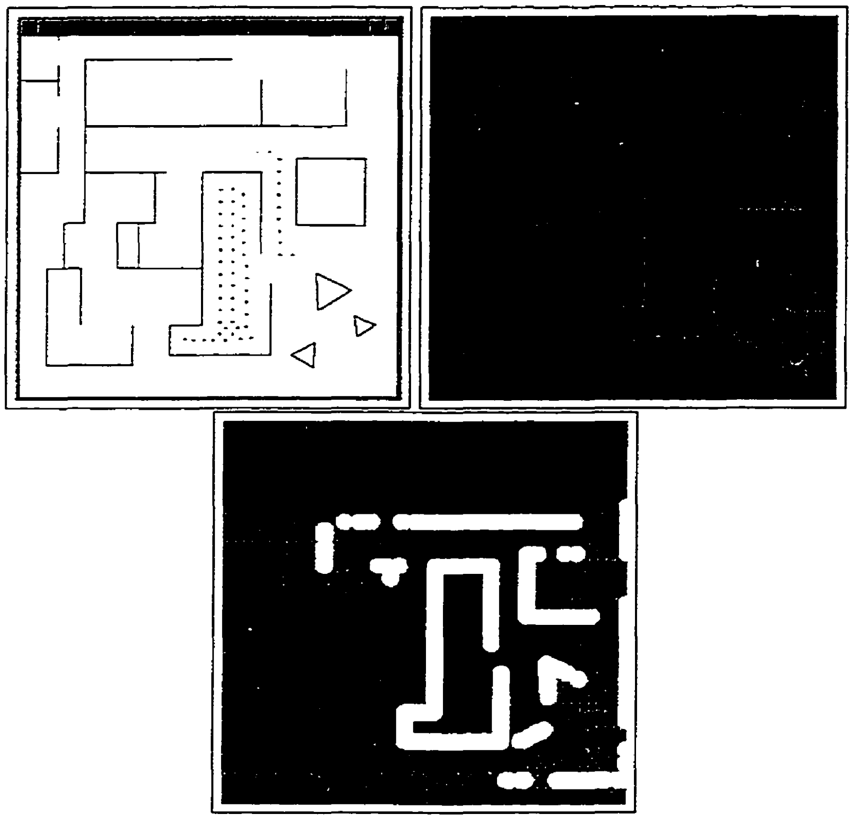


FIGURE 5.6. The diagram on the left is the original map, with an exploration trajectory marked in. The diagram on the right is the occupancy grid representation that results from the exploration. The third diagram is the c-space representation. The grey areas are pixels with no information. Black areas are free pixels (probability of occupancy < 0.5) and the white areas are occupied pixels (probability of occupancy > 0.5).

Finally, we convert the occupancy grid into a configuration space. This representation models the robot as a point in space, by expanding obstacles by the robot's radius. Since the robots have only holonomic motion constraints (within our domain of interest) and are regular cylinders, combining configuration space with the occupancy grid is a relatively simple proposition - each data point is expanded to a disc with radius equal to the robot's radius. The rotational symmetry of the robot allows us to use only two dimensions for the configuration space representation.

By expanding the occupancy grid to configuration space (or c-space), we can ensure (to the limit of sensor accuracy) that the robot will never come in contact with an obstacle. Every time the exploration algorithm moves the robot to a new location, we can verify that the location is free of obstacles using the occupancy grid. While the potential field alone should be sufficient to prevent the robot from driving through obstacles, the occupancy grid is an added security measure.

Figure 5.6 shows an exploration trace, and the resulting occupancy grid. The black regions are those that are free of obstacles. The white pixels are obstacle pixels. The grey areas are pixels that contain no sensors information.

Figure 5.6 also shows the result of converting the occupancy grid into c-space. The c-space map can now be used to perform such tasks as path-planning more easily, by treating the robot as a point. Notice that the white areas are much more pronounced in size, as a result of expanding the obstacles.

5.4. Exploration and Rendezvous

Finally, we can use the exploration method to explore space, and the occupancy grid will give us a metric map of the environment. (Although the occupancy grid is in fact in c-space, it is a relatively simple matter to extract the original map.)

Figure 5.7 shows a demonstration exploration trajectory that ran for 200 seconds. The first image taken is the initial position of the robot. Each subsequent image is immediately after the robot found itself in a local minimum, or had exceeded the potential threshold, either of which triggered a breadth-first search for a new region of low potential. Notice that each breadth-first search results in exploration of a new, often far-removed and disconnected region of space, which is in fact very much the desired result.

5.5. Conclusion

In this chapter, we have described the exploration method we will be using as the underlying task for the rendezvous experiments in the following chapters. We presented the techniques the exploration method used, namely potential field descent and occupancy grids, and how the data is used to generate both the potential field and recombined over time into the configuration space occupancy grid.

In the following chapter, we describe experiments using this exploration method with the rendezvous algorithms described in Chapters 3 and 4, and examine the results of those experiments.

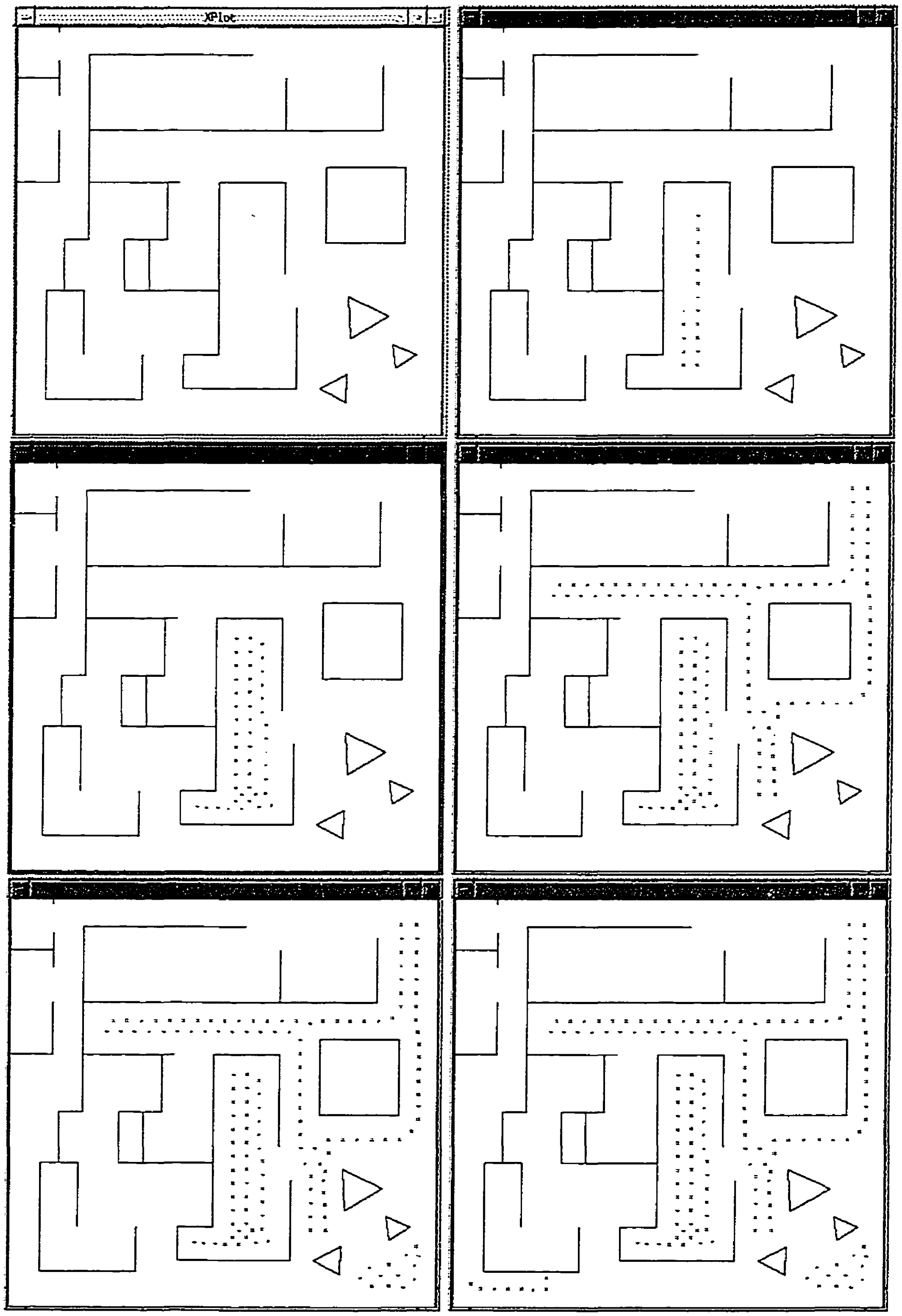


FIGURE 5.7. 6 snapshots of a 200 sec. exploration trajectory. Each small cross is a point on the trajectory. Connected lines have not been drawn in, for clarity.

CHAPTER 6

Simulation

In the preceding chapters, our discussion of the rendezvous problem has been on a theoretical basis. Although Chapter 4 encapsulated a number of practical issues with parameters such as sensor noise, our analysis did not address the problems of space. In this chapter, we will be simulating mobile robots in a two-dimensional environment, and performing an analysis of the algorithms similar to the numerical analysis of Chapter 4. However, this numerical analysis was abstracted from the domain of mobile robotics to a considerable degree.

In the previous chapter we proposed an exploration algorithm that will motivate our rendezvous algorithm. The context of this research is exploration of an unknown environment; the objective is, as always, to increase the speed of the exploration by using multiple robots, and we wish to overcome the inevitable communication limits by using the technique involving periodic rendezvous to bring the robots together to share information at regular intervals.

By performing the exploration with rendezvous first in simulation, we are able to address many issues that result from coupling rendezvous to a primary task such as exploration, and issues that result from the spatial domain of the problem. We will not be addressing the problems of physically-realisable robots, such as overcoming sensors artefacts. Once the simulation is working to our satisfaction, we then move to the experiments performed on real robots; these experiments are described in Chapter 7.

The experiments that we describe in this chapter have two goals; the first is to determine the behaviour of the various rendezvous algorithms (proposed in Chapter 3) under different experimental conditions. The second goal is to determine the speed-up of the exploration of two robots performing rendezvous, when compared with a single robot. First we outline the experimental method we used for the experiments, in particular the method of landmark acquisition and the rendezvous process. Secondly, we demonstrate the results obtained, and the relationship to previous results.

6.1. Experimental Method

The agents were modelled as idealised Nomad 200 robots with perfect (noise-free) sensing abilities and odometry.[1] The agents explored the unknown environment for a pre-determined length of time; at the end of this length of time the agents attempted rendezvous. The agents then took the n best landmarks seen so far, and used those for the rendezvous algorithm. Each agent was running the same rendezvous algorithm; where the algorithms demanded asymmetrical agents, the agents were assigned roles randomly *ab initio*. Instead of generating sequences of algorithms, as in Chapter 4, we simulate the robot motion between rendezvous points, and simulate communication attempts.

6.1.1. Acquiring Landmarks. The trajectory of the robot through the environment is given by the exploration algorithm as described in Chapter 5. In principle, all that remains to be done for landmark acquisition is to measure the distinctiveness function along the trajectory and retain the local maxima. This can be captured by naive algorithm depicted by the flowchart in Figure 6.1.

However, this simplistic description hides several complex issues, the first of which is choosing an appropriate distinctiveness function.

6.1.1.1. *The Distinctiveness Function.* Recall from Chapter 3 that we would like a distinctiveness function that is smooth and has few local extrema over the exploration space. However, in most non-trivial environments a number of factors such as sensor noise and occlusion make the distinctiveness surface highly non-convex and complicated. Our choice of a distinctiveness function was based human experience; we would like the function to peak in wide-open areas that correspond to large rooms, foyers, etc. These are more traditional meeting points from a human perspective than, for instance, corridors, alleys or alcoves. Figure 6.2 shows an example of a robot in a room, taking 16 sonar measurements.

We can measure the "openness", $\mathcal{R}$, of any point in the environment simply by summing the range returned by each sensor:

$$\mathcal{R}(x,y) = \sum_{i=1}^{n} R(x,y,\theta_i) \tag{6.1}$$

Figure 6.3 shows the result of the robot using 4 of its sonar transducers to compute the openness of its position, from Equation 6.1.

[1]While we did have the ability to simulate the sonars using a more realistic sonar simulator, this simulator was simply too computationally slow to be of much use - 64 data points took over 1 minute to compute, as opposed to 1 second, using the noise-free simulator, a speed difference of at least one order of magnitude.

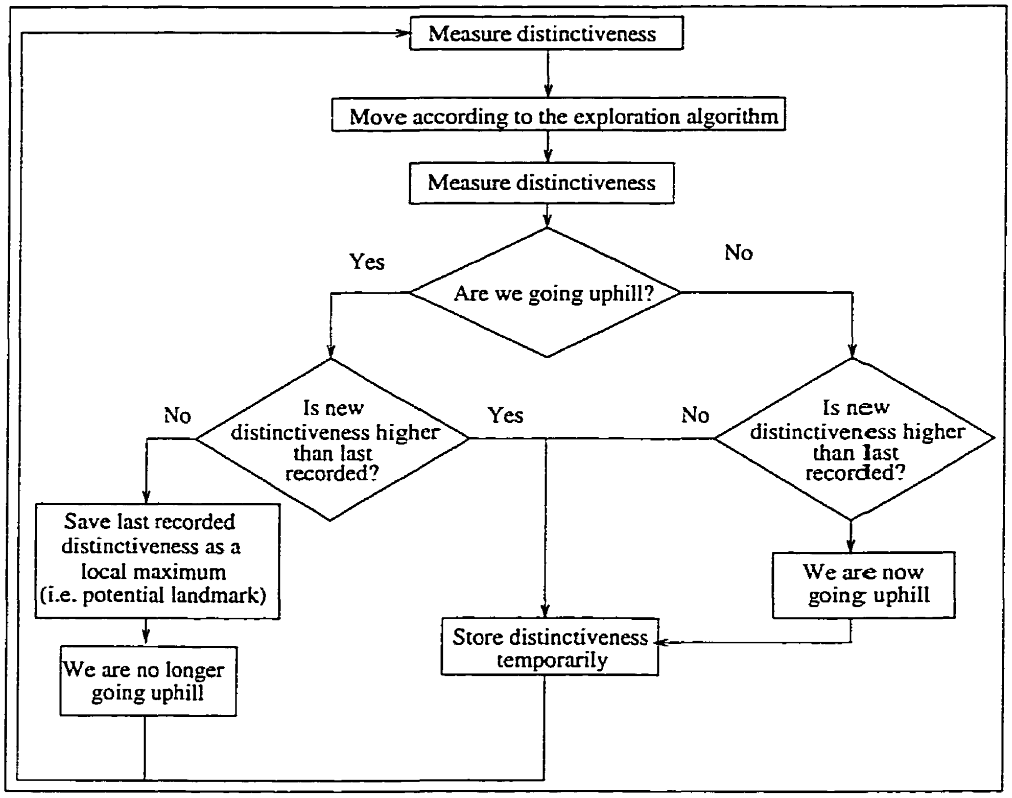

Measure distinctiveness
Move according to the exploration algorithm
Measure distinctiveness
Yes
No
Are we going uphill?
No
Is new distinctiveness higher than last recorded?
Yes
No
Is new distinctiveness higher than last recorded?
Save last recorded distinctiveness as a local maximum (i.e. potential landmark)
We are no longer going uphill
Store distinctiveness temporarily
We are now going uphill

FIGURE 6.1. The landmark acquisition algorithm.

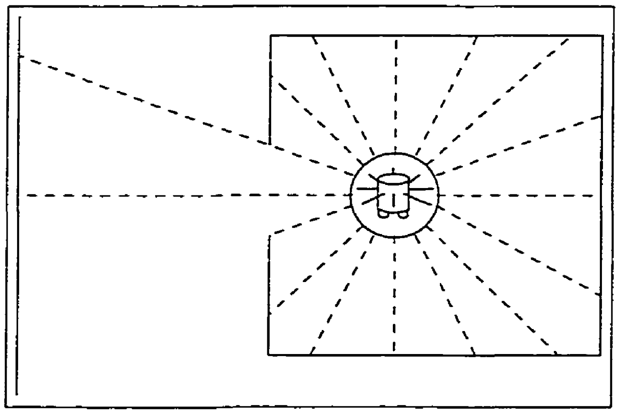


FIGURE 6.2. A typical measurement taken by a robot using 16 idealised sonar transducers.

Furthermore, we would like to pick the middle of these open spaces, that is to say, the most symmetrical points in the environment. Given that we have round robots with evenly spaced sensors, where each sensor has a diametrically-opposed mate, then measuring symmetry is a relatively easy task. We measure the asymmetry, $\mathcal{A}$, of each point by summing the absolute of the differences from

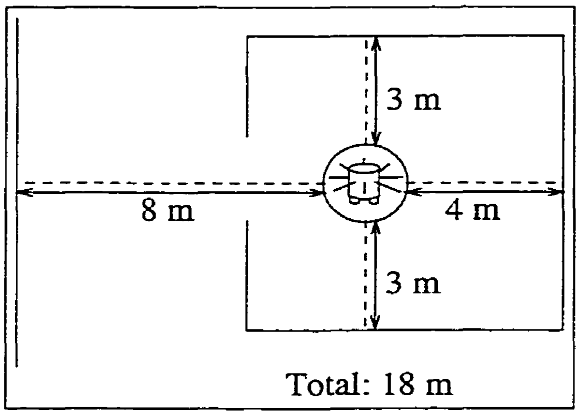


FIGURE 6.3. The result of using 4 transducers and Equation 6.1 to compute the openness of the surrounding space.

these diametrically-opposed pairs. If each pair of sensors measures the same, then the asymmetry falls to 0.

$$\mathcal{A}(x,y) = \sum_{i=1}^{n/2} \mid R(x,y,\theta_i) - R(x,y,\theta_{i+n/2}) \mid \tag{6.2}$$

Figure 6.4 shows the same robot as figures 6.2 and 6.3 using 4 transducers to compute the asymmetry of its position. Notice how the asymmetry falls to 0, when the robot is positioned exactly in the middle of two walls. The zero asymmetry case could only occur when the the robot is in the middle of a regular polygon with same number of sides as the robot has transducers.

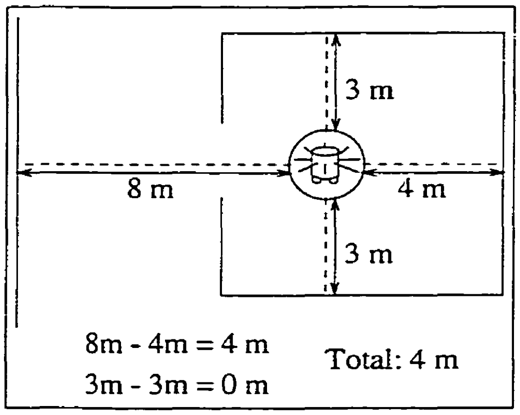


FIGURE 6.4. The result of using 4 transducers and Equation 6.2 to compute the asymmetry of the surrounding space at the robot's position.

We then combine the openness and the asymmetry by dividing $\mathcal{R}$ (6.1) by $\mathcal{A}$ (6.2) to get

$$D = \frac{\sum_{n=1}^{64} R_i}{\sum_{n=1}^{32} \mid R_i - R_{i+32} \mid} \tag{6.3}$$

Recall from Chapter 3 that we showed an example distinctiveness surface in Figure 3.1. We used this distinctiveness function (Equation 6.3) to compute that surface.

6.1.1.2. *The Distinctiveness Distribution.* The numerical analysis conducted in Chapter 4 had an underlying assumption that the distribution of the landmark values was approximately linear. Certainly, in the case of no noise, this was completely true, as the landmark values were computed using a line function. However, this assumption is not necessarily valid for landmark values acquired in real space, or even in simulation.

The main reason for suspending this assumption is that the environment and the distinctiveness function dictate the landmark values. Even with a knowledge of environmental features, it is extremely difficult to pick a distinctiveness function that has few, dominant local extrema, let alone determining a function that will give these extrema a linear distribution. With our assumption of a totally unknown environment, picking such a distinctiveness function is clearly impossible. The question, then, is whether or not the analyses under the assumption of a linear distinctiveness distribution are relevant to the real robot.

Figure 6.5 shows a sample landmark distribution. These landmark values were generated from a simulated robot trajectory through a sample environment that is morphologically similar to an indoor environment (rectilinear walls, etc.). This is the same map used to generate the sample distinctiveness surface (Figure 3.1).

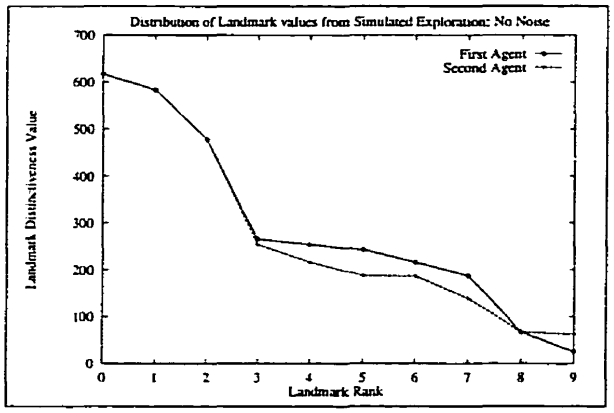


FIGURE 6.5. An example distribution of landmark values from a simulated exploration run, with no noise applied to the distinctiveness values.

Figure 6.5 shows the landmark distribution is roughly linear. It is a noisy line, but not unlike the landmark distribution produced by the linear function with 10% noise. Figure 6.6 shows a sample landmark distribution acquired in the same manner as Figure 6.5, but with roughly 10% noise (±25 on a maximum value of 250). Here the linear distribution is preserved in all but the first distinctiveness value.

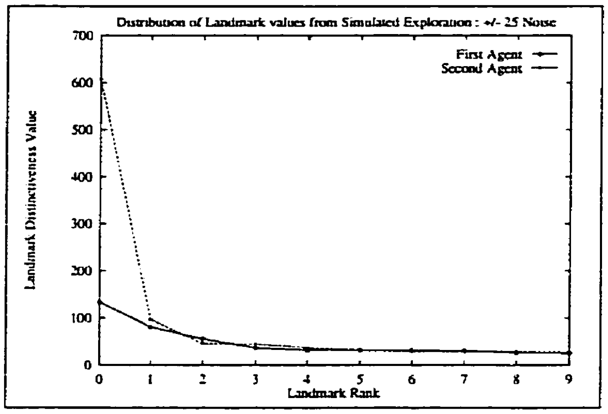


FIGURE 6.6. An example distribution of landmark values, from a simulated exploration run, with ±25 applied to the Distinctiveness.

However, the distribution for the very high noise simulation (±400 on a maximum value of 250) shows again the linear distribution of landmark values, plotted in Figure 6.7.

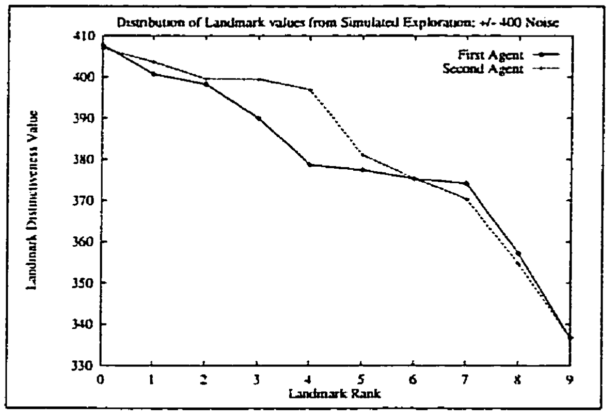


FIGURE 6.7. An example distribution of landmark values, from a simulated exploration run, with ±400 applied to the Distinctiveness.

Figure 6.7 shows a linear distribution, again with substantial noise. However, given the fundamentally linear natures of the three sample distributions shown here, it can be safely concluded that the assumption of linear distributions for the landmark values collected by the agents during exploration is tenable, and the numerical analysis performed in Chapter 4 is valid.

6.1.2. Trajectory Dependencies. One of the problems with using the exploration method described in Chapter 5 is that there is no guarantee that any particular point in the environment will be visited during the exploration process (in fact, there are no guarantees about anything, since no semantics are ever ascribed to any part of the spatial representation). Consequently, two agents that have explored the same areas of space may have trajectories that do not overlap in any manner. Even more troubling are the issues of landmark distributions, and accurately recognising peaks in the distinctiveness surface.

6.1.2.1. *Landmark Distributions in Space.* One of the difficulties of demanding trajectory-independent landmark acquisition, is that any notion of the landmarks' relative positions in Euclidean free space is lost. While it is possible to determine how close two landmarks are in absolute distance, obstacles may intervene, making two landmarks appear relatively close together, whereas they may be separated by some considerable distance through free space.

Ideally, however, only landmarks which are not mutually visible should be kept in the landmark set, otherwise a number of undesirable conditions may result. One such condition is that two landmarks (which are in reality distinctiveness maxima along the trajectory) may in fact be too close together, because two points along the trajectory that are separated in time (or distance along the trajectory) are in fact very close together. Figure 6.8 demonstrates this effect. The robot travels along the distinctiveness surface, storing only the local maxima that it sees *along the trajectory.* Recall that the robot is only considering the distinctiveness of points that it actually visits. The robot does not perceive that there is a peak in the distinctiveness surface between the two peaks in the trajectory.

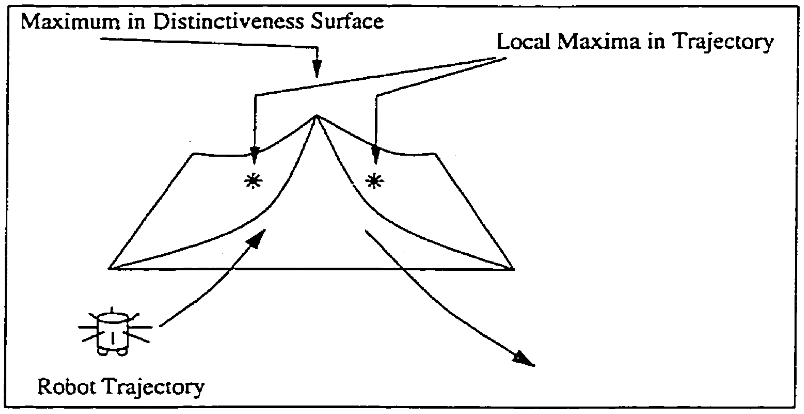

Maximum in Distinctiveness Surface
Local Maxima in Trajectory
Robot Trajectory

FIGURE 6.8. An example trajectory that covers both sides of the peak in the distinctiveness surface, but never sees, or measures the actual peak in the surface.

While this is not a problem for the algorithms, if the environment is large, or the area of the environment common to the agents is relatively small, then the time to rendezvous becomes a serious issue. Since the goal is to have the agents rendezvous in minimum time, it is undesirable for the agents to spend time visiting points in the environment that are close together. The algorithms, and the testing performed on them in Chapter 4 assume that two agents at two different landmarks cannot communicate, or "see" one another. By invalidating this assumption, the strengths of the algorithms we have developed are, in some sense, being defeated. Points in the environment are essentially being revisited, regardless of whether it is appropriate to do so.

There are a number of ways of dealing with this problem. Each method has its strengths and drawbacks. These methods are only necessary for weeding out those landmarks that have been already acquired and lie within some visibility range, e.g. radio range, etc.

- If the robot has a range sensor, it can be used to determine whether any of the previously acquired landmarks is visible from each potential landmark, as it is acquired. This has the advantage of being fast, and requiring minimal additional storage and computation. However, most range sensors are both inaccurate and have insufficient range to be practical.
- The robot can suspend the primary task (e.g. exploration, etc.) and attempt to travel in a straight line to each of the other landmarks that lie within the visibility range. If no obstacle is hit then the two landmarks can be considered mutually visible. The advantage to this method is the same as using the range sensor, but does not have the same accuracy or range problems of the range sensor. However, it does add substantial mechanical complexity to the rendezvous problem. As the number of landmarks grows, the complexity becomes a sufficiently large drawback as to make this method of determining mutual visibility worse than simply allowing the mutually visible landmarks to remain in the landmark set.
- If the primary task (e.g. exploration, etc.) provides a metric map, such as an occupancy grid, this can be used by the rendezvous algorithm to determine whether or not two landmarks are mutually visible. This method does demand substantial storage for such a map, but if one is already available from the primary task, then making use of the map is a computationally cheap option. However, the map must contain appropriate information, and must be reliable.

Since the task that we are performing is exploration and an occupancy grid is available, the third method was used in the experiments for eliminating mutually visible landmarks from the landmark set. As the results will indicate, it worked with success.

6.1.2.2. *Accurate Peak Measurement.* The method described in section 6.1.2.1 suffices for eliminating multiple landmarks that represent the structure in the distinctiveness surface. However, there still remain the issues of accurately recognising the distinctiveness peaks, and even more importantly, measuring the peak height accurately.

The problem of accurately measuring the peak height is related to ensuring that the agents have the same perception of the environment. In addition to noise and sensor differences which were discussed in Chapter 3, care must be taken to ensure that the agents view the environment from the same (or equivalent) standpoints.

If the agents share the same trajectories through the environment, then this issue is solved. Such a situation would occur, for instance, if the agents were employing Voronoi diagrams or freeway methods for navigation. While there will in practice be some positional error across agents, this will largely be due to sensor error and can be encapsulated in the sensor model. However, if the primary task does not involve navigation along mandated trajectories, then it is likely that the agents will, while capturing the same peaks in the distinctiveness, have very different perceptions of the height of the peaks, as Figure 6.9 demonstrates. [2]

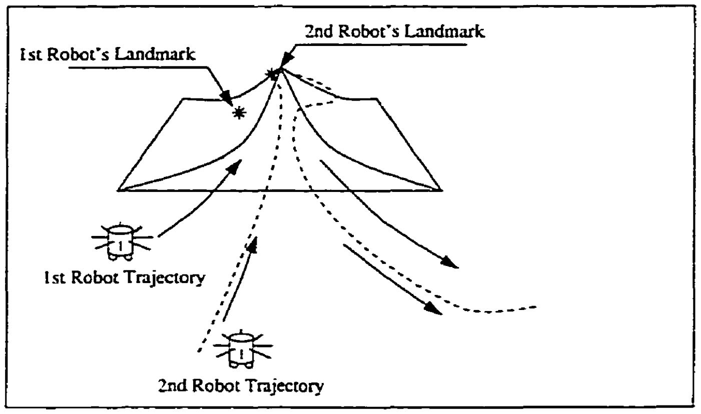


FIGURE 6.9. Two agents exploring the distinctiveness surface. Because of the nature of the exploration algorithm, one agent passes directly over top of the peak, and thus measures its height correctly. The other agent passes first to one side, and then the other, retaining only the higher of the two maximal measurements, never measuring the peak correctly at its maximum height.

In practice, the measurement of landmarks can be refined, thus solving the sampling problem, by performing gradient ascent over the distinctiveness surface every time a potential landmark is identified. There are three possibilities for when this landmark refinement process occurs:

(i) During the landmark acquisition process

Refining the landmarks during the acquisition process does not effect the mechanical complexity of the rendezvous process, although it increases the complexity of the acquisition

[2]This is, of course, a sampling problem. However, given the prevalence of high-frequency information in the distinctiveness surface, undersampling is inevitable without serious increases in mechanical complexity. In the worst case scenario, if the agents drastically undersample the distinctiveness surface, they will not only mis-measure the distinctiveness peaks, but miss some peaks altogether. If the distinctiveness function is also used for the primary task (as it is in this research, as the sonar is used both for the distinctiveness measurements as well as generating the map), the primary task must be aware that rendezvous is being performed, and must be willing to relinquish control of its sensors to the landmark acquisition process. This requires some coupling between the landmark acquisition process and the primary task, but the coupling can be eliminated if necessary by giving the rendezvous process a separate sensor.

process. While the rendezvous process has an accurate perception of the environment, and therefore no recalculation is necessary during the rendezvous algorithm, the primary task (e.g. exploration) is no longer completely decoupled from rendezvous; again, it must be aware that periodically it will be suspended while the landmark acquisition process refines the landmark measurements.

(ii) Between the primary task and the rendezvous process
Before the rendezvous process begins, the agent visits each landmark it has acquired, and performs gradient ascent over the distinctiveness surface starting from each landmark. Armed with the refined landmark measurements, it then computes the visit sequence according to the specified algorithm and attempts rendezvous. The advantage is that the rendezvous process is again decoupled from the primary task, but the rendezvous sequence is computed correctly, and only once. The disadvantage is the non-trivial mechanical complexity that results from revisiting every landmark, before beginning rendezvous. For a process that has many, widely separated landmarks, this will be unacceptable.

(iii) During the rendezvous process
The landmark measurements can be refined during the rendezvous process. This has the advantage that the rendezvous process and primary task are decoupled as in the previous method, but the additional mechanical complexity is low, as in the first method. The disadvantage is that the visit sequence must be recomputed in the majority of cases if a deterministic algorithm is being used. Furthermore, if the measurements are completely wrong, the measurement may not be corrected until after a substantial number of iterations.

The method chosen for the simulation and real robot experiments in this research was the first one, since maintaining the decoupling between the rendezvous and exploration tasks was not a primary concern.

Figure 6.10 shows the result of the hill-climbing operation.

Table 6.1 shows the change in the distinctiveness values, as a result of the refinement process. Note the reordering of the landmarks, especially landmark 5, which was initially at position 2. Also note the substantial increase in overall distinctiveness values.

As the diagram shows, some of the landmarks are positioned some distance away from the initial estimate. Since both agents perform the same landmark refinement, the trajectory dependency is overcome, and the only differences in landmark perceptions should be due to noise and sensor differences.

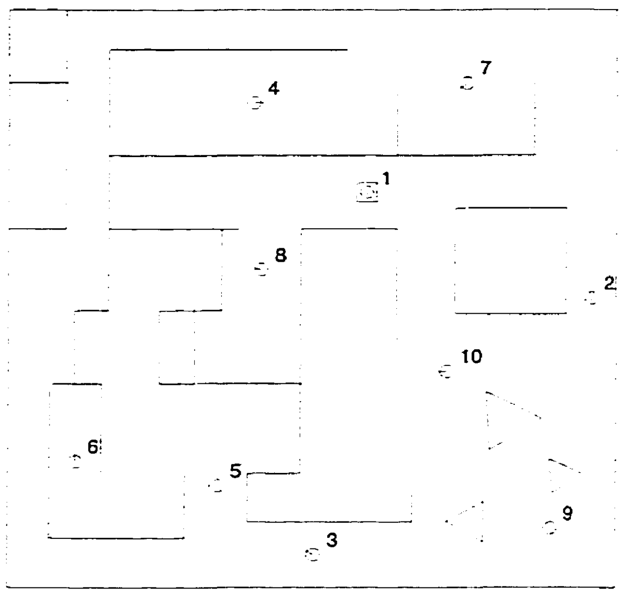


FIGURE 6.10. The result of the landmark refinement process. The dashed circles are the initial peak estimates, acquired during the exploration process. The solid circles are the final positions of the landmarks. The box is the best landmark.

Final Landmark Rank	Initial Distinctiveness	Final Distinctiveness
1	40.74 (3)	616.86
2	60.12 (1)	583.98
3	35.96 (4)	477.76
4	19.34 (6)	265.54
5	48.38 (2)	254.00
6	10.95 (8)	242.94
7	7.50 (9)	215.90
8	22.06 (5)	187.96
9	7.57 (10)	67.10
10	15.66 (7)	25.40

TABLE 6.1. The change in landmark values as a result of the landmark refinement process. The initial rank of each landmark is in parentheses.

6.1.3. Modelling Noise. Two of the formal parameters developed in Chapter 3 were δ, random noise, and $\eta(x,y)$, systematic noise at position (x,y). Recall that in Equation 3.10, we combined these two parameters into the one parameter, $\eta(x,y)$. For the purposes of the numerical analysis, $\eta(x,y)$ was generated independently for each (x,y) pair from the function $Random(0 : r \times max(F(i)))$.

During the robot simulation, the noise modelled by Equation 3.10 was applied only to the distinctiveness value in order to isolate the distinctiveness sensor from the exploration sensor. This explicit decoupling may appear counter-intuitive. However, since the primary task is not necessarily exploration, or even driven by sonar data, separating the noise model from the exploration sensors is more consistent with the separation between rendezvous and the primary task that has been maintained throughout this work. By treating the noise model as applicable only to the distinctiveness values, the distinctiveness sensor becomes a qualitatively different sensor to the exploration sensor. This approach has the added advantage of simplifying the exploration algorithm - no effort must be made to clean up noisy data for map generation. It should be noted that this clean decoupling was not used during the real robot experiments. There is no question of modelling noise on a real robot - the noise is the real thing.

Initially, the same approach as the numerical analysis was used for modelling the noise during the robot simulation. An independent random value was generated for each possible position on the distinctiveness surface, to create an equivalent error surface. When a simulation measurement was made at that position, the random value from the error surface was added to the same position on the distinctiveness surface. However, this method proved to be problematic, due to the non-smooth nature of the resulting error surface, as Figure 6.11 depicts.

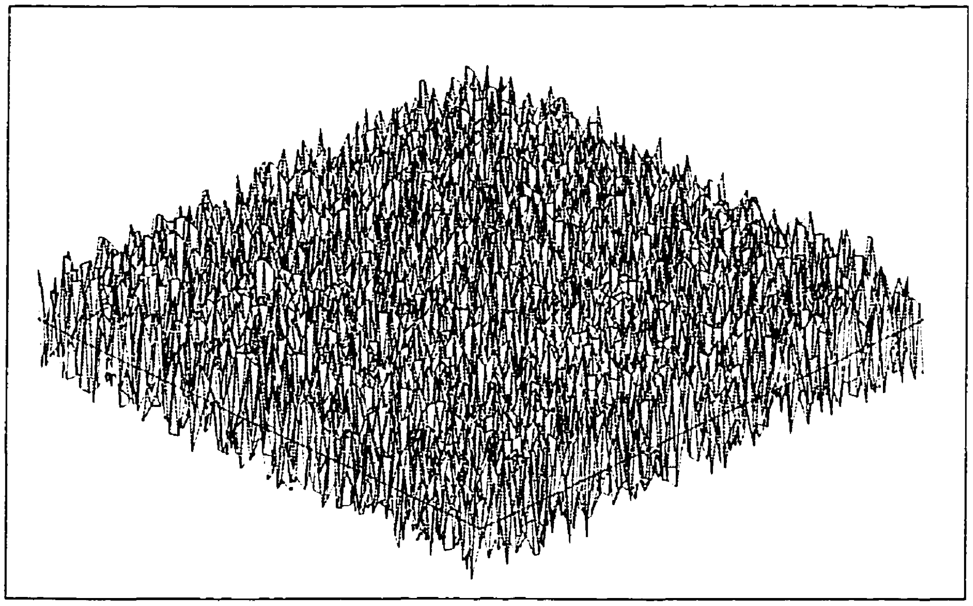


FIGURE 6.11. An example error surface, generated by a random error function, $|\eta(x,y)| = [0,50)$, 20% of $max(F(i))$.

This is clearly not a smooth surface, which does not affect the algorithms for low ranges of noise. However, as the noise level rose past 10%, maxima in the error surface started to appear as maxima in the noisy distinctiveness surface. While this is an intended side-effect, the discontinuous nature of the surface resulted in many more local maxima in the surface than had original been there, changing the distribution of the landmarks considerably between agents. At the macroscopic level, the same major features were present in the surface, yet small maxima form equally valid landmarks compared to large maxima – not as *distinct* landmarks, but landmarks nonetheless.

Consequently, another method was used to generate a smoother error surface. This method computes each error value as a change from the previous value. The change has two components: a random component, and a component that is dependent on the previous value. These two components are summed to get the change from position to position.

$$\eta(x,y) = Random(-.5:.5) + \delta_\eta(x,y)(|\frac{\eta(x,y-1)}{max(\eta)}|) \tag{6.4}$$

δ_η is a different δ from Equation 3.10. δ_η specifies a preferred direction of motion - away from the extrema of the error surface, and towards the zeropoint.

$$\delta_\eta(x,y) = -1 : \eta(x,y-1) > 0 \tag{6.5}$$

$$1 : \eta(x,y-1) <= 0 \tag{6.6}$$

Figure 6.12 shows an example error surface that results from this method of noise modelling. While still not a completely smooth surface, it is an improvement over the random noise surface at high noise values.

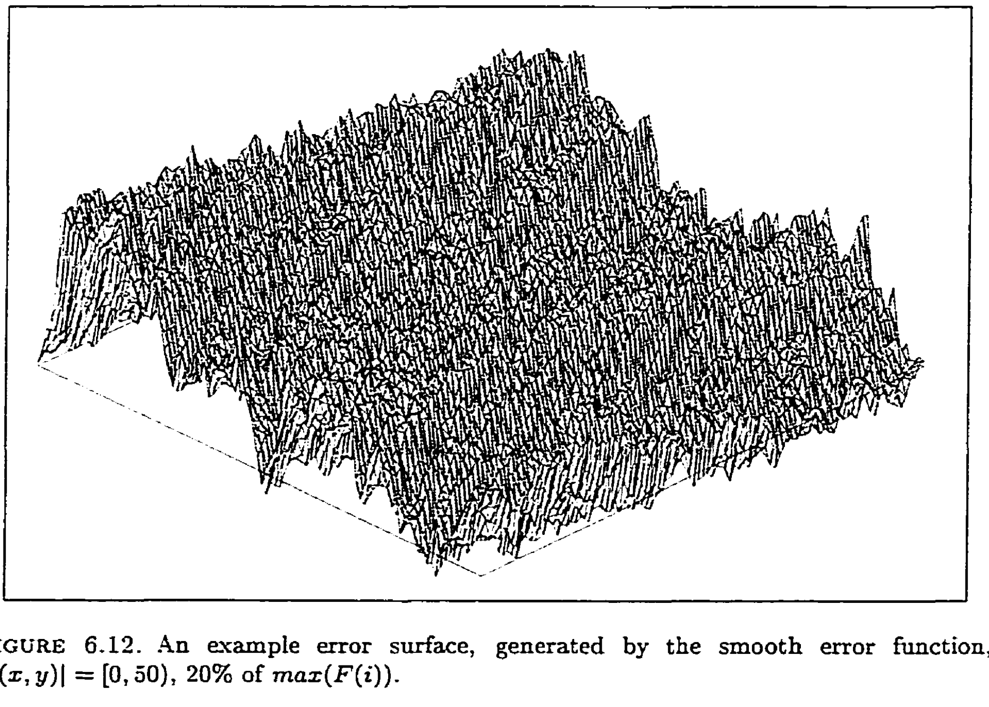


FIGURE 6.12. An example error surface, generated by the smooth error function, $|\eta(x,y)| = [0,50)$, 20% of $max(F(i))$.

This error function has two main advantages:

(i) Each value is dependent on the preceding value. There is a notion of time as the error surface grows from one corner across to the other corner, however, this is inevitable - some values have to be computed first. These values that are computed first are edge elements, which are a special case.

(ii) The surface tends toward the zeropoint (and therefore does not get clamped at one extremum), and moves faster towards the zeropoint, the further away it gets, thus giving some stability to the surface.

6.1.4. Modelling Landmark Commonality and Asynchrony. Modelling the landmark commonality, d, and asynchrony, j, explicitly was impractical, as these are not parameters that can be explicitly set without communication between agents. The landmark commonality parameter is a reflection of the degree to which the trajectories of the agents overlap; this parameter is a function of the trajectories, not the inverse. Similarly, the asynchrony is a function of environmental and robot characteristics; it is extremely difficult to extract the appropriate characteristics from the single parameter.

However, the simulation did model these characteristics indirectly. The landmark commonality parameter was set by altering the size of the bounded world, and altering the time allowed between rendezvous attempts. For example, if an agent explored the world completely in 600 seconds, then, in principle, the commonality, d, could be set to $d = n/2$ by setting time to rendezvous to 300 seconds, or doubling the world size. In both cases, the initial poses of the robots would have to selected carefully, to allow for some, but not complete overlap of trajectories.

Asynchrony was modelled using the radio communication simulator. The simulator had a locking mechanism that prevented the robots from moving to the next landmark, until both had made a communication request. By allowing the locking mechanism to operate probabilistically, the parameter j could be included in the simulation. The locking mechanism had a probability p of failing, so that the robot made a communication request which was instantaneously filled, regardless of whether the other robot was capable of accepting the request. The probability p of failing was set explicitly to equal the asynchrony, j. The communication request could only be filled if the other robot was available for communication (i.e. the locking mechanism had succeeded), and the two robots were mutually visible and within range. This implementation of asynchrony is somewhat limited, in that the rendezvous failure is attributed almost wholly to radio failure, as opposed to a myriad of factors including communication, but within the scope of this simulation, it served its purpose.

6.1.5. Simulating Rendezvous. The simulation of detecting other robots and achieving rendezvous was implemented using a special process that simulated radio communication. The processes controlling the simulated robots communicated to the radio simulator during the rendezvous phase of the simulation. Requests were made by each controlling process to the radio simulator, and the simulator then determined, based on its knowledge of the complete map and the current positions of the two simulated robots within the map, whether or not the robots were mutually visible (line-of-sight), and whether they were in radio range of one another (13.5 m [3]).

6.2. Experimental Results

There were three main experiments performed on the simulated exploration and rendezvous, and each was a variant of a test of the rendezvous algorithm performance vs. noise. Each data point is the average of 25 trials. The trial was terminated at 100 rendezvous attempts if the agents had not achieved rendezvous by then.

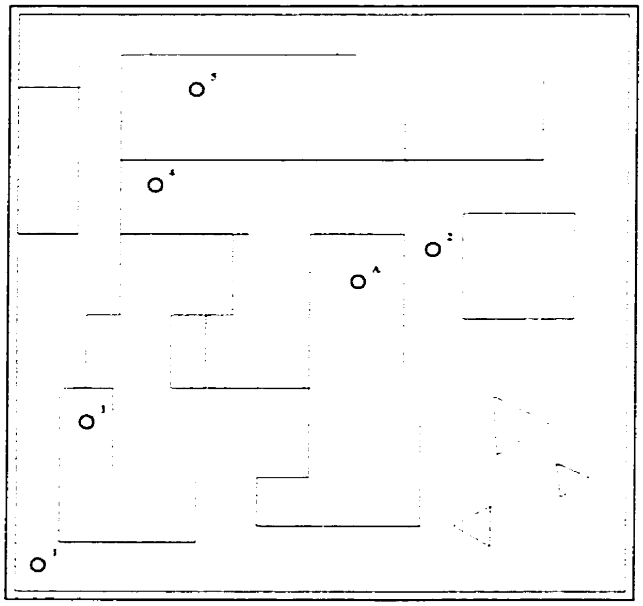


FIGURE 6.13. The map for the simulated experiments, with the starting positions marked as circles. One agent always started from the position labelled 'A', whereas the other agent started 5 times at each of the positions marked with numbers.

Figure 6.13 shows the map used for the first test suite, the baseline algorithm performance. In each set of 25 trials, one agent was started at the same point every trial, the **A** in Figure 6.13, and the other agent was put at one of 5 locations, the circles labelled **1-5** in Figure 6.13, for 5 trials per location. The trials were conducted for 15 values of δ.

[3]This number for the radio range was based on the radius of the smallest robot we used, the RWI B12. 13.5m is one hundred B12 diameters.

6.2.1. Baseline Performance. Figure 6.14 demonstrates the performance of the 4 main algorithms, in the face of increasing noise. The size of the landmark set was 10 landmarks and asynchrony j was 0. In order to have the agents have as close to 100% landmark commonality as possible, the simulation explored for 600 seconds - this proved to be sufficient for the agents to have explored almost all of the space. The four algorithms were sequential, smart-sequential and the probabilistic functions exponential and random.

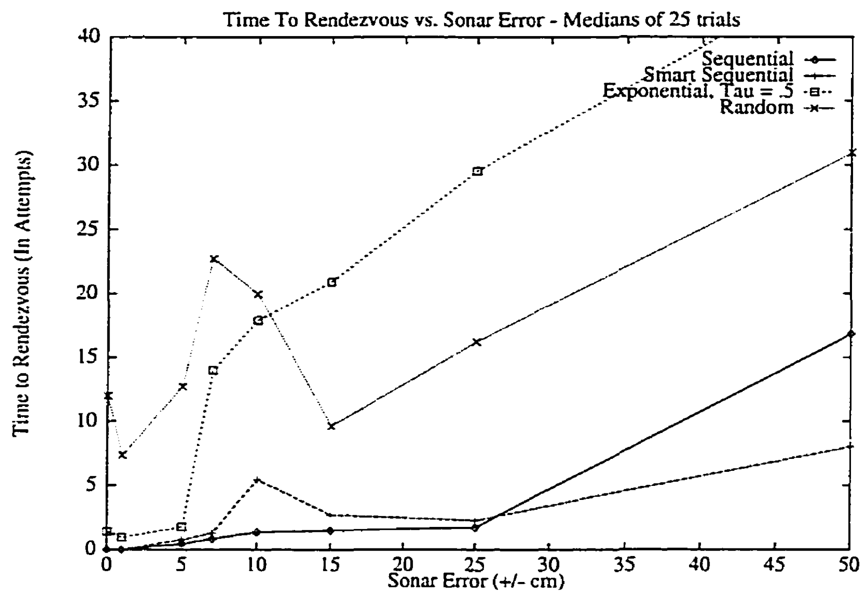


FIGURE 6.14. Baseline Performance - Time to Rendezvous as a function of Noise-level $\delta = [0, 50]$.

6.2.1.1. *Low-Noise Performance.* Figure 6.14 is the equivalent graph to Figure 4.4, although the noise range is much shortened. The graphs are superficially different, however, most of the main features are preserved. The deterministic algorithms are dominant throughout the noise range, and in particular, smart-sequential is the fastest algorithm in the lowest noise case, but as noise begins to dominate the measurements, the smart-sequential algorithm's ability to guess where the other agent might be is hampered by poor estimates of the environment. Smart-sequential was designed explicitly to handle small perturbations in the landmark orderings between agents; at the level of the simulated robots, it is succeeding. Note also that exponential very quickly fails, although the failure is not as dramatic as in the numerical analysis.

Figure 6.15 shows the performance of more variants of the exponential algorithm. With a higher τ constant in the exponent, the algorithm again fails very quickly. Clearly, exponential is not the algorithm of choice for high-noise situations. Recall that these data points are averages of 25 trials and so many of the individual trials failed to meet in the allotted time.

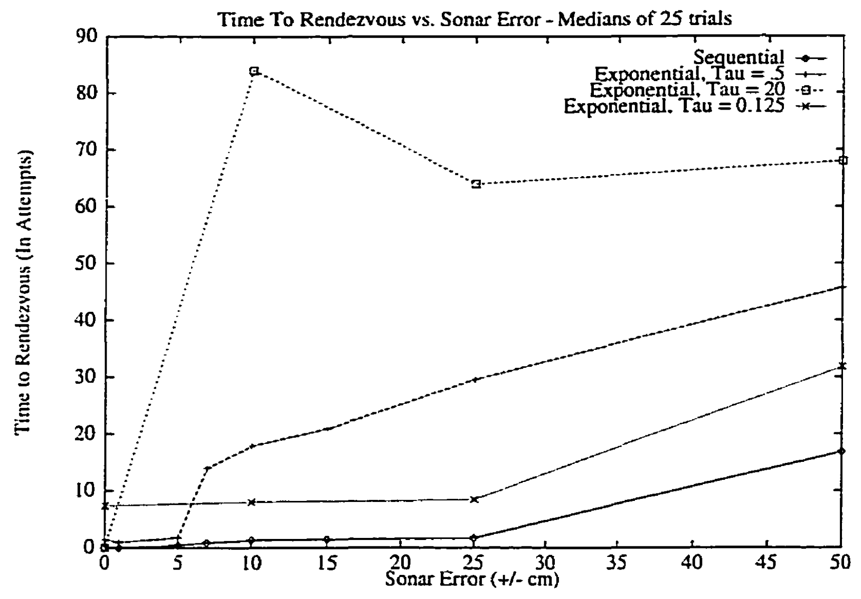


FIGURE 6.15. Baseline Performance - Time to Rendezvous as a function of Noise-level $\delta = [0, 50]$, with probabilistic (exponential) algorithms, sequential is also shown for reference.

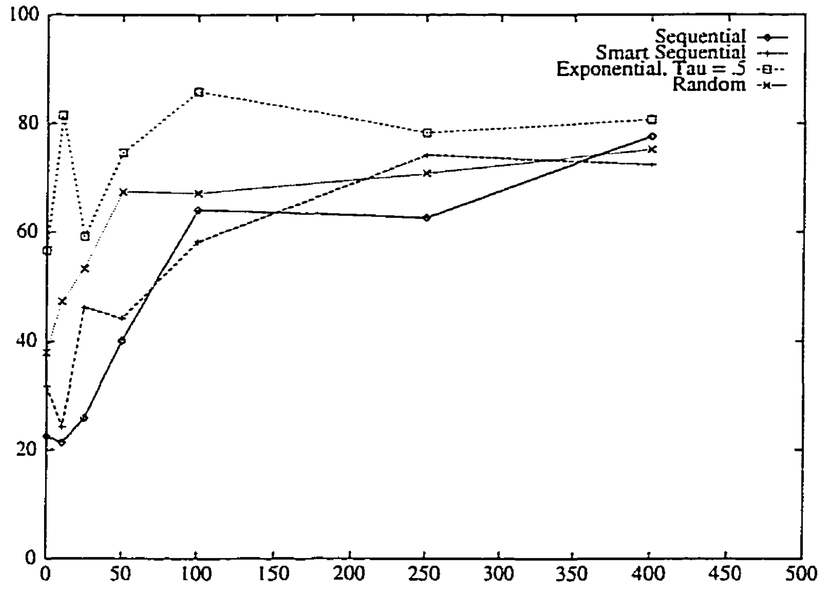


FIGURE 6.16. Baseline Performance - Time to Rendezvous as a function of Noise-level $\delta = [0, 500]$.

6.2.1.2. *High-noise.* Figures 6.16 and 6.17 demonstrate the effect of further noise on the performance of the algorithms. At the highest noise level in Figure 6.16, the noise is 80% of the highest noise-free peak in the environment; however, certain algorithmic characteristics manifest themselves. For example, sequential continues to out-perform all other algorithms.

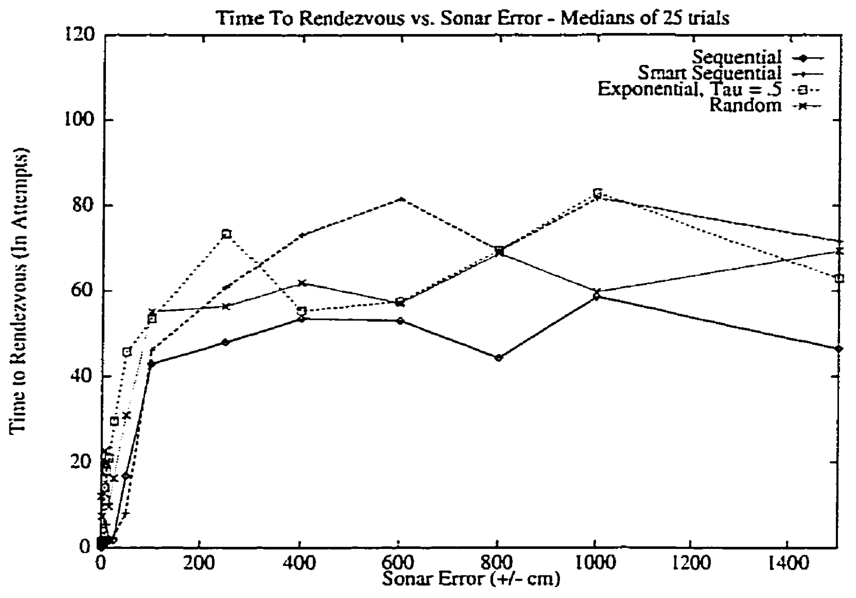


FIGURE 6.17. Baseline Performance - Time to Rendezvous as a function of Noise-level $\delta = [0, 1500]$.

However, in Figure 6.17, noise has become by far the dominant characteristic of the environment; the agents are essentially operating blind. There appear to be some interesting phenomena in the highest noise region, but further investigation revealed these to be artefacts of the initial conditions, the size of the environment and the fairly low number of trials. The only real conclusion that can be drawn from these figures is that sequential is a robust algorithm if the agents have no way of reliably quantifying the environment.

6.2.2. Disjoint Exploration Areas. This was the second of the three experiments performed using the simulated exploration and rendezvous was the explicit case of disjoint landmark sets, representing areas of the environment explored by only one agent. Again, each data point is the average of 25 trials. The trial was terminated at 100 iterations if the agents had not achieved rendezvous by then.

Figure 6.18 shows the larger map used for this test suite, the algorithm performance with disjoint exploration areas. Just as in the baseline tests, in each set of 25 trials, one agent was started at the same point every trial, the letter **A** in Figure 6.18, and the other agent was put at one of 5 locations, the digits **1-5** in Figure 6.18, for 5 trials per location. The trials were conducted for 15 values of δ.

Figure 6.19 shows an example of the exploration carried out by two agents in this environment. Clearly, the two agents have explored the majority of the environment, and yet the overlapping

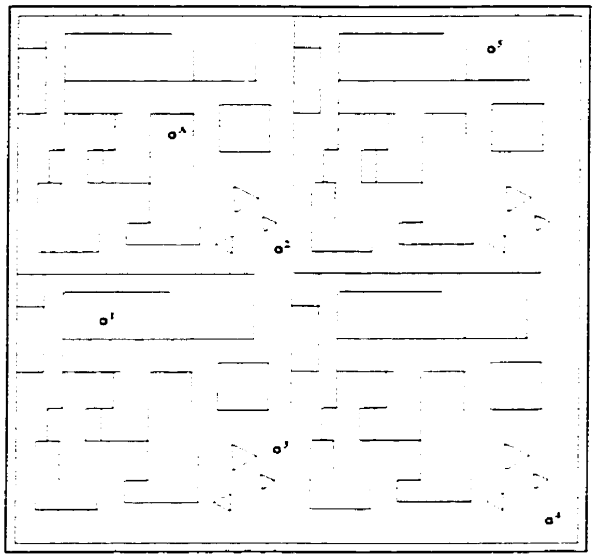


FIGURE 6.18. The map for the simulated experiments for the disjoint landmark set experiments, with the starting positions marked as circles. One agent always started from the position marked with 'A', whereas the other agent started 5 times at each of the positions marked with a digit.

areas of their trajectories is fairly minimal. This is the first experiment where the speed-up of the algorithms can be tested; the results of the speed-up of the algorithms will be in section 6.3.

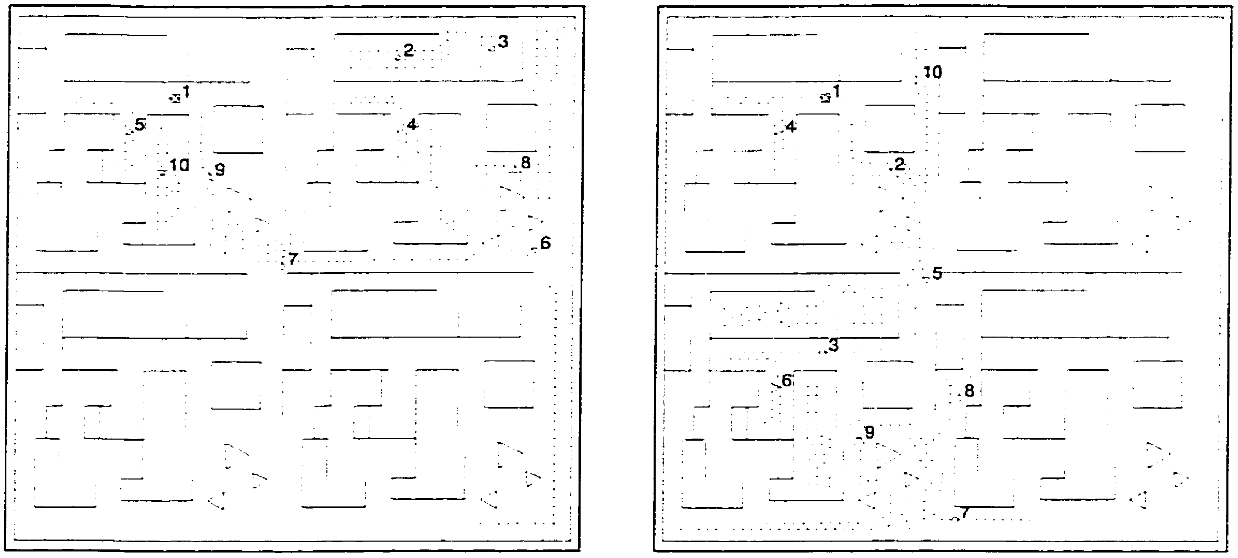


FIGURE 6.19. Two example trajectories through a larger space. The exploration was allowed to continue for 600 seconds, before rendezvous occurred. The circles indicate landmarks. Notice that the rendezvous occurred successfully, even though a large part of the trajectories were unique to the agent.

6.2.3. Algorithm Performance. Figure 6.20 demonstrates the performance of the 4 main algorithms, in the face of increasing noise. The size of the landmark set was 10 landmarks and asynchrony j was 0. The four algorithms were sequential, smart-sequential and the probabilistic functions exponential and random.

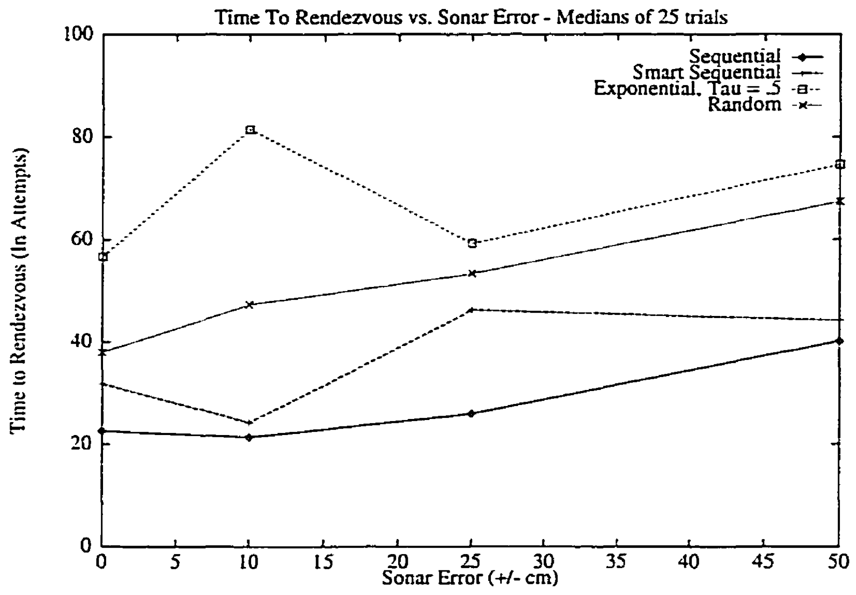


FIGURE 6.20. Non-Identical Landmark sets - Time to Rendezvous as a function of Noise-level, Low Noise $\delta = [0, 50]$.

Notice that smart-sequential is no longer the best algorithm, even in this low noise region of the parameter space. The ability of the smart-sequential algorithm to guess the location of the other agent is damaged by the incomplete knowledge that results from disjoint landmark sets.

The exponential algorithms fare somewhat better than they did in the previous experiments, as Figure 6.21 indicates. This is more a reflection of the fact that the deterministic algorithms cannot perform any better; it does not indicate an improvement on the part of the exponential algorithms.

Finally, in the high noise case shown in Figure 6.22, the algorithms are relatively close together. This is because the failure rate of all the algorithms was relatively high - much higher for the deterministic algorithms than in the numerical analysis. This is because with the high noise level, there is no way to guarantee that the the landmark sets will be identical. As the noise level increases, the likelihood of the two agents choosing two points in the environment to be landmarks falls considerable. As the noise level increases, the agents are essentially blind. The size of the bounded environment and the restrictions on the relative spacing of the landmarks prevent the landmark sets from being completely disjoint, so all the algorithms converge on a average time to rendezvous for

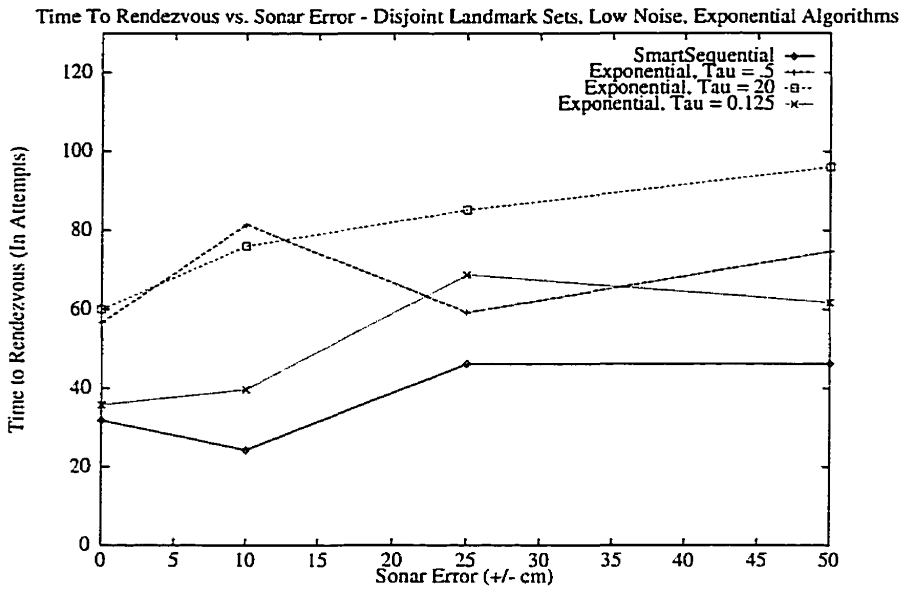


FIGURE 6.21. Non-Identical Landmark sets, Exponential Performance - Time to Rendezvous as a function of Noise-level, Low Noise $\delta = [0, 50]$.

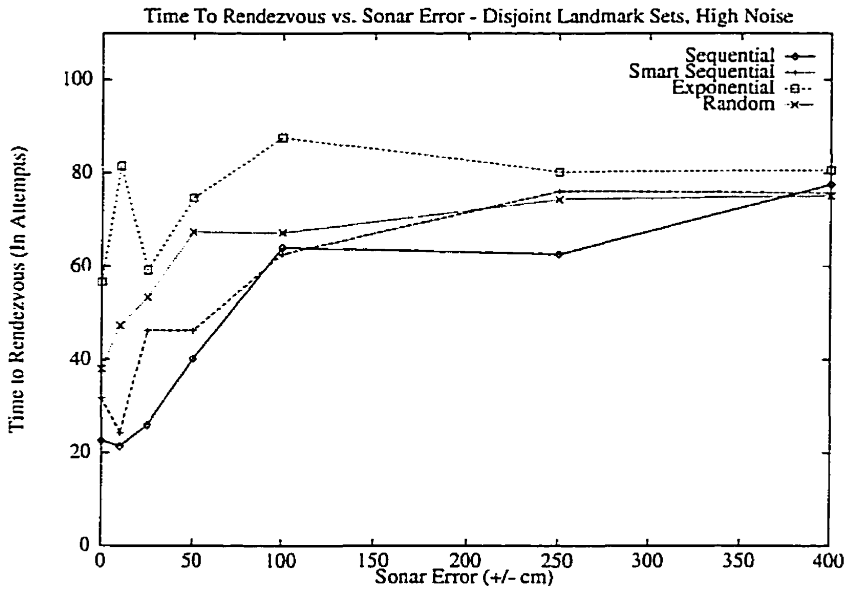


FIGURE 6.22. Non-Identical Landmark sets - Time to Rendezvous as a function of Noise-level, High Noise $\delta = [0, 400]$.

this particular environment. In this high noise case, none of the algorithms is any more useful than the other.

6.2.4. Asynchrony. In this final set of experiments, we test the ability of the agents to rendezvous under conditions where the robots would sometimes fail to meet successfully, even if they were at the same location. Each data point is the average of 25 trials, and trials were terminated at 100 rendezvous attempts if the agents had not met by that point. The map that was used was the same as for the baseline simulations; the robots were allowed to explore for 600 seconds, sufficient for the agents to have covered almost all of the space. The size of the landmark set was, again, 10 landmarks. The four algorithms evaluated were sequential, smart-sequential, random and exponential, and exponential was tested with three different exponential constants.

We are particularly interested in the low-noise region of the parameter space, as the numerical analysis indicated that the exponential algorithms performed best under these conditions. As Figure 6.23 indicates, the superior performance of the stochastic algorithms is present in the spatial simulation.

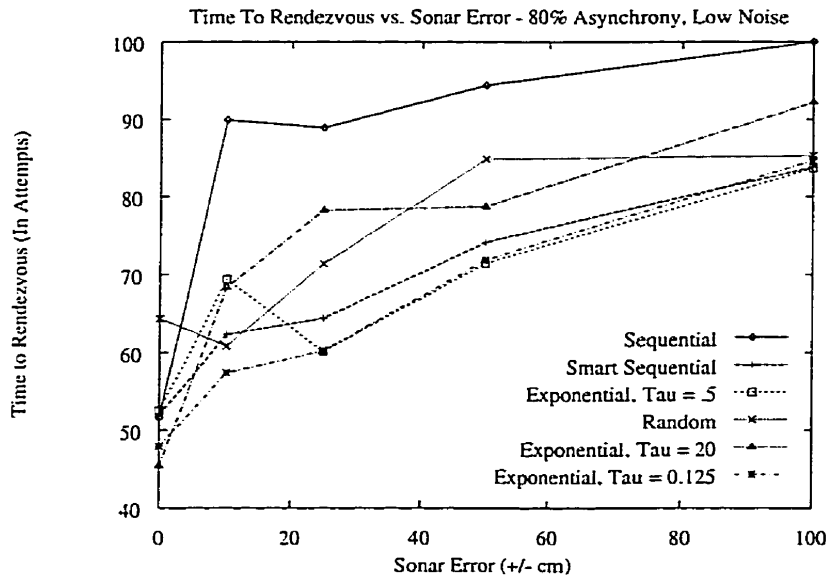


FIGURE 6.23. 80% Asynchrony - Time to Rendezvous as a function of Noise-level, Noise $\delta = [0, 100]$.

Focussing further on the region where δ is small, we see in Figure 6.24 that in the case of no noise, the algorithm that has the fastest performance is the exponential algorithm with a very large exponential constant. That this algorithm should be the fastest is expected, since the exploration suffers only from missed meetings - both agents should have chosen the same landmarks. Since this algorithm will revisit the best landmark more often than any other algorithm, it has the best chance of overcoming the asynchrony problem.

However, once any noise is present in the system, this highly tuned algorithm fails rapidly since the algorithm is unstable with even the slightest noise. The algorithms with much lower exponential constants, and thus are less tuned to the best landmark, perform much better. The smart-sequential

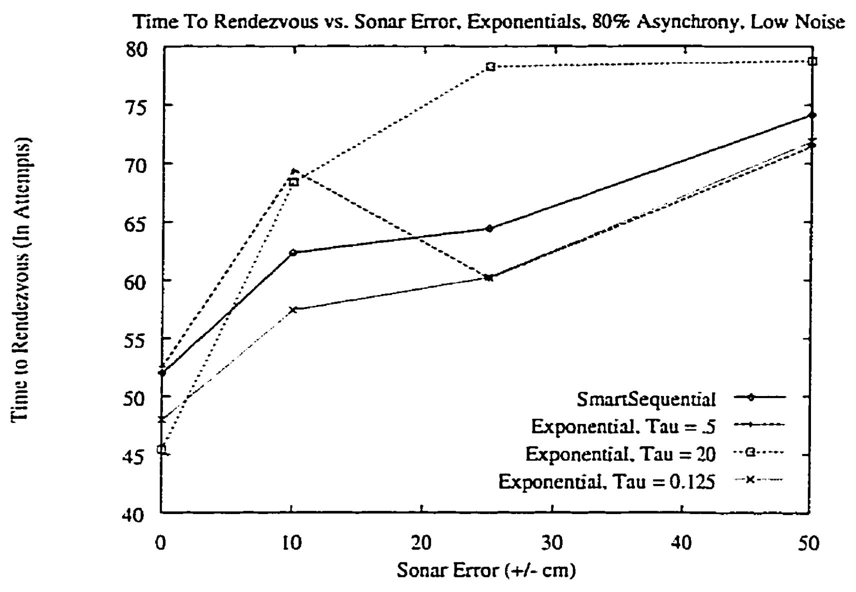


FIGURE 6.24. 80% Asynchrony - Time to Rendezvous as a function of Noise-level, Exponential performance, Noise $\delta = [0, 50]$.

algorithm is close to the behaviours of these exponential algorithms, and much better than the large constant exponential algorithm. Once the noise begins to dominate the signal, the deterministic algorithms prove superior. However, smart-sequential outperforms sequential for much of this part of the parameter space; the inability of the sequential algorithm to return to landmarks more often inhibits its performance much as it did in the previous experiment.

6.3. Multi-Agent Exploration

Of particular interest in this experiment is the ability for the rendezvous algorithm to overcome the communication restriction and yet maintain the increase in speed that multiple-agent robotics promises. What we would like to do is demonstrate a significant increase in exploration speed, even accounting for the time to rendezvous.

As our metric for measuring speed increase in exploration, we used the change in mapping speed, S,

$$S = \frac{A}{T} \tag{6.7}$$

where A is the percentage of the environment that has been mapped, and T is the time to complete the mapping.

Since the experiment was constructed so that the occupancy grid matched the size of the bounded environment, we used the number of cells in the occupancy grid that contained information of any kind (occupied or not) as our measure of the size of the mapped environment.

The increase in speed of the mapping process is then given by Equation 6.10,

$$\begin{aligned}\Delta S &= \frac{S_{combined} - S_{single}}{S_{single}} && (6.8)\\ &= \frac{\frac{A_c}{T_c} - \frac{A_s}{T_s}}{\frac{A_s}{T_c}} && (6.9)\\ &= \frac{A_c}{A_s}\frac{T_s}{T_c} - 1 && (6.10)\end{aligned}$$

We took the area of a single agent, A_s to be the area explored by the active agent, and the time of the single agent T_s to be the time allowed for the exploration process alone. The combined area, A_c was the explored area of the merged maps, and the combined time, T_c was the time to explore, T_s added to the time to rendezvous, T_r.

$$T_c = T_s + T_r \tag{6.11}$$

Recall that each data point in the preceding graphs represents the mean of 25 trials. These 25 trials were composed of 5 possible initial configurations, with 5 trials per configuration. Since the environment was bounded, it was possible to determine what percentage of the environment each agent had explored. Once the maps from the two agents had been merged, it was then possible to determine how much of the environment had been explored by the two agents together, and in fact the increase in explored speed, compared to the efforts of a single agent. The increase in explored area for each of these 5 configurations is given in Table 6.2.

Index	Active	Passive	Combined	% Increase in Area
0	48.0	42.2	69.1	44.2
1	48.0	58.4	72.0	50.2
2	48.0	66.5	74.6	55.6
3	48.0	59.5	68.5	42.8
4	48.0	67.7	73.9	54.1
Average				49.4

TABLE 6.2. The increase in explored area, as a percentage of the environment, for each of the 5 initial configurations.

As Table 6.2 shows, the increase in explored areas was a minimum of 42.8%, and on average 49.4%. If the agents were capable of merging their maps immediately after the exploration phase, then $T_c = T_s$, and the increase in area is exactly equal to the increase in speed. However, this ideal situation is equivalent to total communication, and is not realistic.

There are two possible ways to interpret the exploration speed results: the first treats each exploration iteration and rendezvous iteration as a single time increment, as if travelling through a graph where each arc is of time-length 1, and T_r is simply the number of rendezvous iterations.

Table 6.3 shows the speed increase in the algorithms in the zero-noise case, using this graph-like model of the exploration process. Each datum is the average of 25 trials; if the agents failed to meet (e.g. due to the exponential algorithm), then the change in mapping speed, ΔS was set to 1.0.

Algorithm	% Speed Increase
Sequential	49.1 %
Smart-Sequential	38.1 %
Exponential	21.1 %
Random	46.7 %

TABLE 6.3. The speed increase using the graph-like model of the world, in the zero-noise case. Each number is the average of 25 trials.

Only the exploration speed of the exponential algorithm was seriously degraded by the rendezvous process. Figure 6.25 shows the change in exploration speed as the noise is increased.

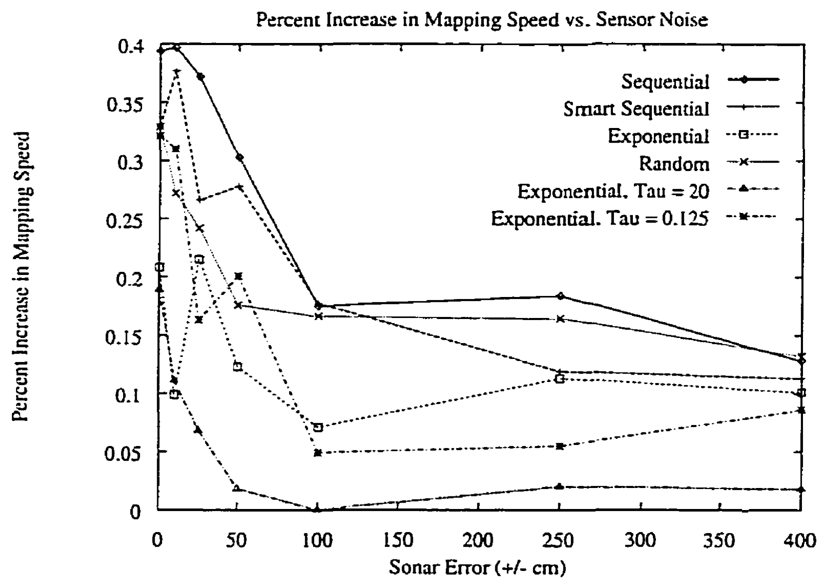


FIGURE 6.25. Increase in exploration speed as a function of noise. Environment modelled as a graph.

Characteristically, sequential performed extremely well over the majority of the noise range; smart-sequential did well in the low-noise range, however once the noise began to dominate the measurements, smart-sequential's performance was considerable degraded. These results reassuringly corroborated on a general level the numerical and simulation results.

However, a number of over-simplifying assumptions were made, most questionable of which was the assumption that each rendezvous iteration takes a single time unit. In order to further bolster these results, we computed the increase in speed, taking into account the mechanical complexity of the rendezvous process. Using the actual translation and rotation speeds of the robots that we used for the experiments using real robots (these experiments are given in Chapter 7), we determined that each exploration step took approximately 30 secs. For the rendezvous time, we used the robot simulator to provide the actual distance that the robots travelled through the simulated environment as they moved from landmark to landmark. The speed of the robots is 40 cm/sec; since the distance the two robots travelled each iteration was usually considerable different, the time to travel between landmarks on each iteration was assumed to be the longest of the two distances, divided by the speed of the robot.

Figure 6.26 shows the performance of the rendezvous algorithms, using this realistic model. As the graph indicates, the realistic model does not affect the speed adversely; despite the fact that the rendezvous iterations were considerably longer now, the exploration step was still sufficiently expensive that the speed gains from using two agents were preserved.

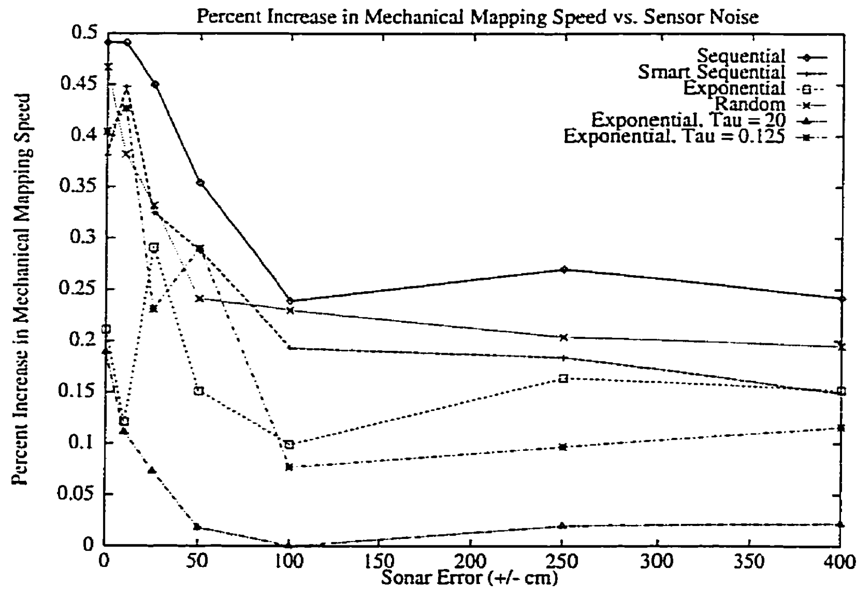


FIGURE 6.26. Increase in exploration speed as a function of noise. Times given using mechanical model of robots.

Figure 6.26 is compelling support for multiple-agent robotics in general; despite the necessity of considerable mechanical complexity to overcome the communication problems, an increase of speed of up to 50% in the exploration task is still available. It is a problem for future work to show that this increase in speed is possible in general.

6.4. Conclusion

In this chapter, we described the experiments we performed in simulation to test the numerical analysis of Chapter 4. In the course of outlining the experimental method, we also described a number of issues which affect the rendezvous algorithm as applied to spatial domains. These issues included trajectory dependence, our model of distinctiveness distributions, spatial sampling and our noise model.

We then described three main experiments we performed to test the algorithms in the face of noise, asynchrony, and disjoint exploration areas. The results were consistent with the previous numerical analysis, although there were some small differences. One important result is that the stochastic algorithms have their domain of superiority as expected, but inappropriate tuning of the algorithm parameters can make the algorithms fail very easily. The superiority of these algorithms is not stable.

We also analysed the rendezvous process in terms of increasing the speed of the exploration process. The analysis confirmed that even with the communication constrained to occur only during successful rendezvous, speed increases of up to 50% are attainable, when compared to a single-agent system performing exploration.

In the following chapter, we will describe an experiment in implementing the exploration and rendezvous algorithms on actual robots in a laboratory setting, in order to demonstrate the algorithms can be used in reality.

CHAPTER 7

Rendezvous using Physical Robots

In Chapter 3, we described the rendezvous problem, and motivated the problem with the context of exploration. In particular, we proposed two main classes of algorithms, and gave particular instances that exemplify these two classes. In Chapter 4, we gave a numerical analysis of these algorithms.

In the previous chapter, we analysed the algorithms running in a simulated spatial environment, using the exploration method described in Chapter 5. We performed these experiments in simulation in order to address a number of issues that relate to the spatial nature of the problem; we did not address any issues that resulted from using practical robots, as opposed to idealisations. We were, from these experiments, able to show some interesting features of the rendezvous problem with respect to our algorithms. In particular, we were able to show that communication problems, if addressed using rendezvous, do allow for the speed increases that are often assumed to accompany multiple robot systems.

However, some issues of using mobile robots must still be addressed. All of the prior simulation experiments assumed the simulation sensors were ideal; noise was explicitly applied in order to approximate real sensors. Odometric error was assumed to be negligible. Issues of path-planning were simplified to allow the robots to pass through each other in space, rather than investing time in allowing the simulated agents to detect each other during the exploration stage. These are all assumptions that are not valid once a real robot is being used.

In this chapter, we will show that the these and other assumptions can be dealt with without affecting the utility of the rendezvous procedure. We will not be interested so much in the performance of the algorithms under different environmental conditions; this chapter presents a proof of concept, that, in fact, the rendezvous method is possible and useful on real robots.

7.1. Experimental Method

The experiment was conducted using two mobile robots, a Nomad 200 and an RWI B-12. Both robots are essentially cylindrical, and quasi-holonomic, in that they are capable of turning with 0° radius. The Nomad 200 is 50cm in diameter, and has 16 sonar transducers equally separated by 22.5°. The RWI B-12 is 27cm in diameter, and has 12 sonar transducers equally separated by 30°. Although the Nomad 200 has an onboard 486 processor running Linux, all computation was performed off-board, on two SGI Indigo platforms running IRIX 5.2, and an Pentium platform running Linux 2.0.29. The communication between the robots and their controlling platforms was wireless.

Figure 7.1 show the robots moving through the maze in the laboratory. The right panel of the figure shows the robots standing next to each other, having made a successfully rendezvous.

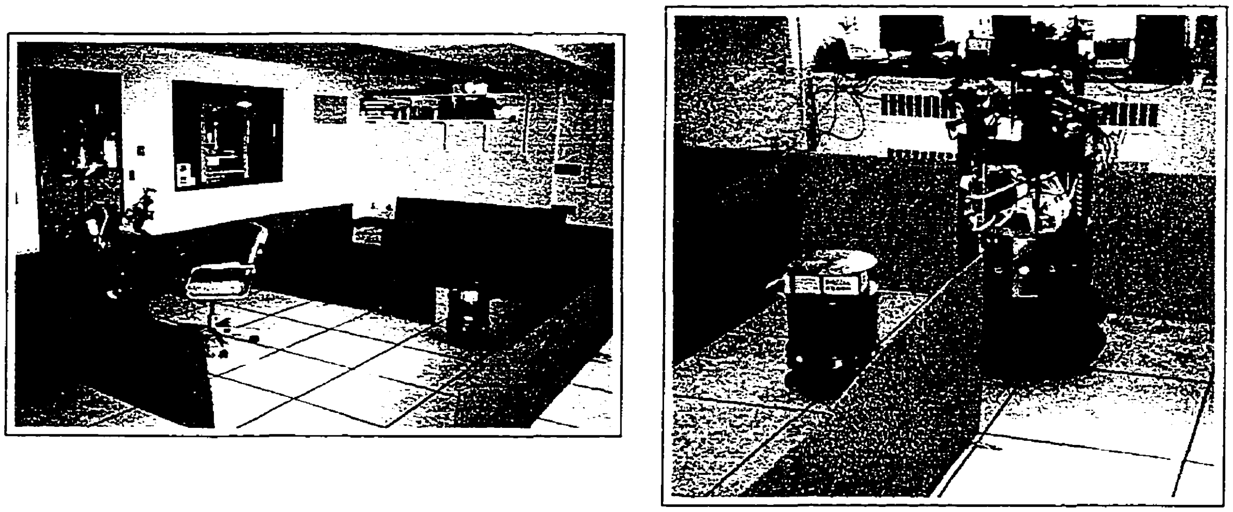


FIGURE 7.1. The robots exploring the maze, and then making a rendezvous.

7.1.1. The environment. The experiment was held in a laboratory space measuring 550cm by 840cm. The walls were free-standing corrugated plastic, 60cm high. The walls were taped together for structural integrity, and stood off the floor with angle-brackets, measuring 10cm long. The total wall length, including bounding walls, was 50.4m.

The small square box in the lower left corner was a desk chair; this was included to show the robustness of the exploration method, and is not especially relevant to the rendezvous problem.

7.1.2. Sonar sensors. The sensor that was used throughout these experiments was the sonar sensor, which is a range sensor only. Consequently, all our distinctiveness function candidates relied upon range information only. The sonar sensor operates by emitting a high-frequency sound

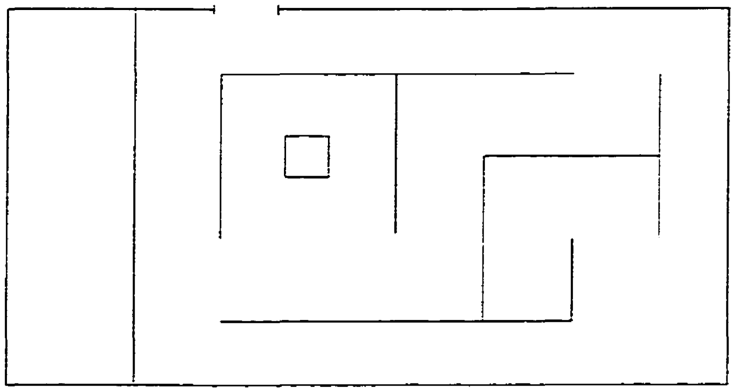


FIGURE 7.2. The map for the real experiments. The space at the left of the maze was used for observation, and also held the controlling equipment.

pulse (40 kHz, for our particular robots). The pulse propagates through the atmosphere until it strikes an object in the environment. The same transducer emitting the pulse then listens for a pulse at the same frequency echoing off an object in the environment. If the pulse echo is heard within a short duration, then the time of flight is used to calculate the distance to the obstacle, otherwise the pulse is assumed to have been lost. Figure 7.3 demonstrates a robot using sonar to measure the distance to the wall in front of it.

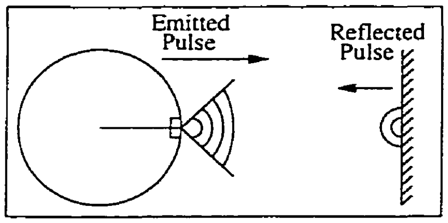


FIGURE 7.3. A robot using sonar to measure the distance to the wall in front of it.

The maximum range of the robots[1] is 8m for the Nomad, and 13m for the RWI. The range precision is ±2.54cm for the Nomad, and ±1.07cm for the RWI. By using the sonar to measure the distance to obstacles around it, the robot can acquire a metric map of its environment. There do exist more sophisticated sonar models such as developed by Kleeman and Kuc [31], Wilkes [53], and Lacroix and Dudek [36] that can recognise and deal appropriately with sonar artefacts in our model. However, our simple model of the sonar pulses, combined with some simple outlier handing, is sufficient for the limited purposes of our experiments; a more sophisticated sonar model would be more appropriate for long-term exploration and environment modelling.

[1]Assuming speed of sound at 330 m/s.

7.1.3. Sonar Filtering. Given the nature of real sonar sensors, some processing must be done on the sonar data. The sensors have a maximum range of 8m, but are extremely unreliable at that range. There are a number of different artefacts from using sonar sensors that must be dealt with. Fortunately, these fall into two main categories: outliers, and multiple-bounce reflections.

7.1.3.1. *Sonar outliers.* If a sonar beam strikes an object along the object's surface normal, then the sound beam is reflected back along its incidence path, the ideal trajectory for a sonar pulse. Figure 7.4 depicts this ideal case.

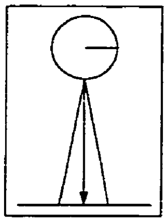


FIGURE 7.4. The ideal trajectory for a sonar pulse from a robot to an obstacle.

If the sonar beam intersects the surface along an oblique path and the surface is not specular with respect to sonar pulses, then the pulse will be scattered, and some of the pulse will be reflected in the direction of the emitting transducer. The sonar transducer that emitted the pulse is the only transducer that is used to detect the pulse, and so the reflected pulse must be heard by the same transducer that emitted it. The left panel of Figure 7.5 depicts the behaviour of a sonar pulse, bouncing off a scattering surface.

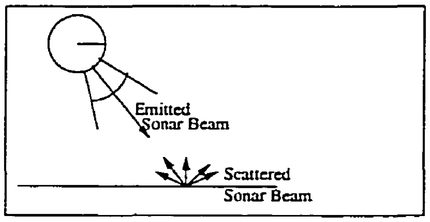


FIGURE 7.5. The behaviour of a sonar pulse bouncing off a scattering surface. Although the sonar pulse scatters in all directions, some is reflected in the direction of the transducer.

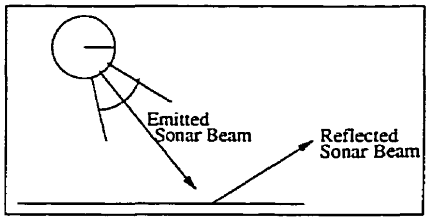


FIGURE 7.6. The behaviour of a sonar pulse bouncing off a specular surface. The majority of the sonar pulse bounces away from the robot and its transducer.

However, Figure 7.6 shows that if the surface is specular, then the pulse is reflected *away* from the emitting transducer and the pulse is lost. Figure 7.7 shows a typical sonar range scan, using the Nomad robot and 64 sonar measurements[2]. As the figure depicts, the majority of the sonar beams are either too long (along the corridor walls), or fall short (at the ends of the corridors). The walls have been drawn in black on top of the range data, to emphasise that as the sonar beam does not accurately measure the range to the walls except at angles that are close to the wall.

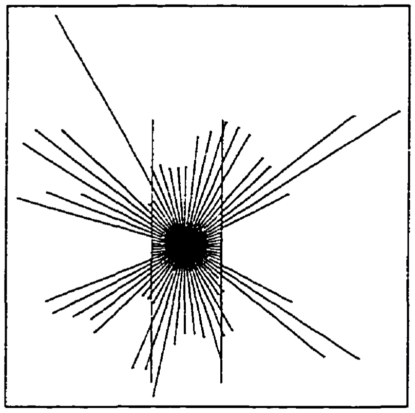


FIGURE 7.7. A typical sonar scan from the Nomad 200, in a hallway situation. The walls have been drawn in black. Note that most of the range data is completely inaccurate.

7.1.3.2. *Corner artefacts.* The other main class of sonar error that we will see in our environment as a result of our sonar model is corner artefacts. This results when a sonar pulse hits both walls of a corner, before returning to the emitting transducer [**36**]. Since this multiple bounce path takes longer than a simple one-bounce path (and range is measured as time-of-flight of the sound pulse) then the wall is perceived as being further than it is [**53**]. Again, note that this is a feature of highly specular environments. Figure 7.8 shows a schematic of the corner artefact problem, and some actual data acquired by the RWI B-12 robot, illustrating the error.

7.1.3.3. *Lowpass Filtering.* There exist a number of methods of dealing with poor sonar data such as this. It is beyond the scope of this work to deal with general solutions; since this is a proof of concept experiment, we will use the domain knowledge to construct a simple filter. The majority of sonar errors in our environment resulted from the sonar pulse either never returning, or returning along a complex, multiple-bounce path. As a result, the errors will almost always overestimate the true range. Consequently, by rejecting range data if the range is above an upper threshold, we eliminated the majority of errors. As a precaution, we also rejected all data that was below a lower threshold of the robot's radius. The upper limit was the same for both robots: 300 cm.

[2]Although the Nomad has only 16 transducers, it can take many more measurements than this by rotating on one point by small increments between measurements of 16. Sixty-four data points can be acquired by taking 16 data points 4 times, and rotating by 5.625° between each set of 16.

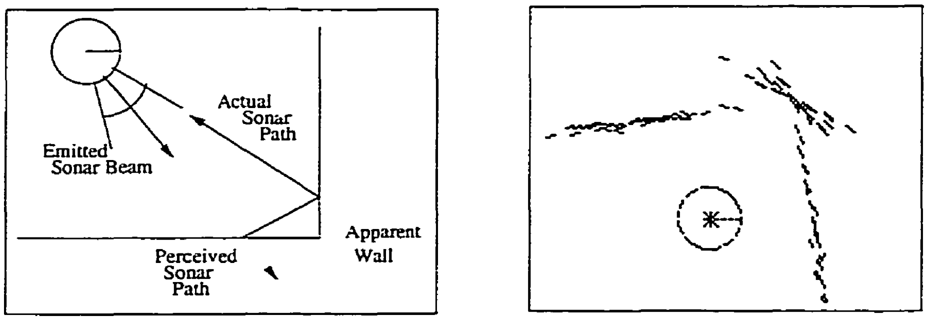


FIGURE 7.8. The left is a depiction of the corner artefact problem. Although the sonar pulse does return to the robot, it takes longer than if it had bounced only once off the wall. On the right is data gathered by the RWI B-12, illustrating the same problem. The RWI is the circle in the centre. The walls pass to its left and in front. However, there is no corner point, but a third, diagonal line where a corner should be.

This type of filtering was sufficient for most of the errors, but could not handle corner artefacts, since these errors are consistent across several points (unlike the outliers), and appear to have some structure that might exist in the environment. There is no easy way to eliminate these artefacts, without recognising them as such. So in addition to the low-pass filtering, Bayesian recombination of data was used. Recall from Chapter 5 that this method used Bayes' rule to allow us to combine occupancy estimates acquired over time for the same point in space. While this method did not immediately eliminate corner artefacts, they became much less damaging as information about the corners accumulated.

7.1.4. Acquiring Landmarks. The same landmark acquisition method was used as in the simulation in Chapter 6. We acquired local maxima along the trajectory during exploration, and then returned between the exploration and rendezvous phases to refine the landmark positions. There were three alternative possibilities for refining the landmark positions, but in the interests of simplicity, the same method as the simulation was used.

It should be noted that during both the exploration and rendezvous phases, there are times when the robot must move between points that are frequently separated by some distance, and often have intervening obstacles. For instance, when the exploration algorithm moves the robot into a potential minimum, it uses breadth-first search to move to a new region to explore. There is no explicit handling of the path to the new location. Similarly, there is no explicit handling of the robot's motion from rendezvous location to rendezvous location. There is an assumption that the robot is capable of finding a path through known space to any point in the environment, if one exists. We were able to maintain this assumption using potential field gradient descent through the known map [**21**]. The rendezvous process fed the range points, after processing, to the path-planner, so that the path-planner had an accurate picture of the environment. We were then able to assume that a path between any two points in the environment that had been visited were accessible, and relied upon the control software to execute the motion correctly.

7.2. Experimental Results

7.2.1. Trajectory. Figure 7.9 shows the trajectories of the robots moving through the maze. The RWI B-12's trajectory is shown in the left panel, and the Nomad 200's trajectory is shown in the right panel. It should be emphasised that the maps were overlaid by hand for clarity, and the robots had no embedded knowledge of the layout of environment.

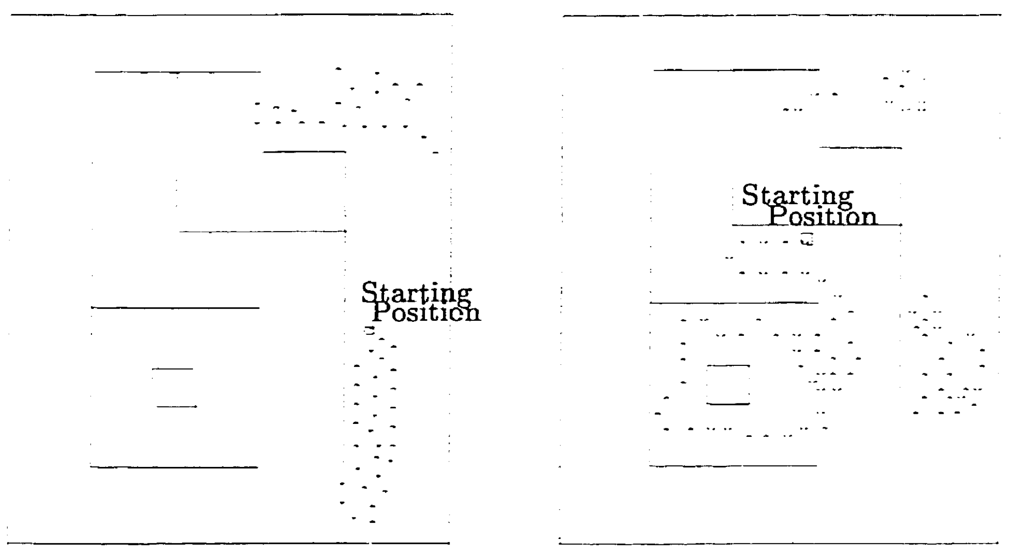


FIGURE 7.9. The trajectories of the two robots throughout the environment. The RWI B-12 trajectory is on the right, and the Nomad 200 is on the left.

The trajectories consist of collections of points, separated by large areas of space. These "islands" of points were areas of the environment explored using the local potential field descent. Once a local potential minimum had been reached, the robot used breadth-first search to find a new area that was known to be clear, yet low in potential (i.e. seen but unexplored).

Notice that the trajectories only overlap on the right hand side of the map; much of the B-12's trajectory is in the inner part of the maze, whereas the Nomad spent time moving up and down the corridor on the right. The B-12's trajectory is also much longer than the Nomad's, because the Nomad spent time performing breadth-first search in the lower part of the corridor. The limitations of the sonar, combined with the filtering, prevented the Nomad from using much of its data outside that small area of the map.

It is in these real experiments that we can observe some limitations of the exploration method. The RWI B-12 occupancy grid indicated that the right-hand corridor to be clear of obstacles, that is to say, the wall appeared to be relatively thin. We can see that the occupancy grid indicated this by the fact that the RWI moved to the far side the wall after reaching a potential minimum; the breadth-first search is limited to points in the environment that are known to have a low probability of being occupied. However, the only way the occupancy grid could contain any information about cells past the wall would be through sonar error. The control software would have prevented the robot from running into any unobserved obstacles, so the situation is not as dangerous as it would seem. However, the problem does highlight the difficulties of using the simple sonar model for any serious exploration.

7.2.2. Occupancy Grid. It is difficult to demonstrate the occupancy grid in any reasonable manner, because the useful data is visually swamped by the sonar artefacts. However, the occupancy grids at the end of the exploration runs are shown here for completeness sake, in Figure 7.10. The white pixels correspond to points in the environment for which there is no information, and the black pixels are points that have a low probability ($P(x, y) < 0.5$) of being occupied. Notice the substantial sonar errors present along the corridor explored by the Nomad 200 (right panel), despite the low-pass filtering in use.

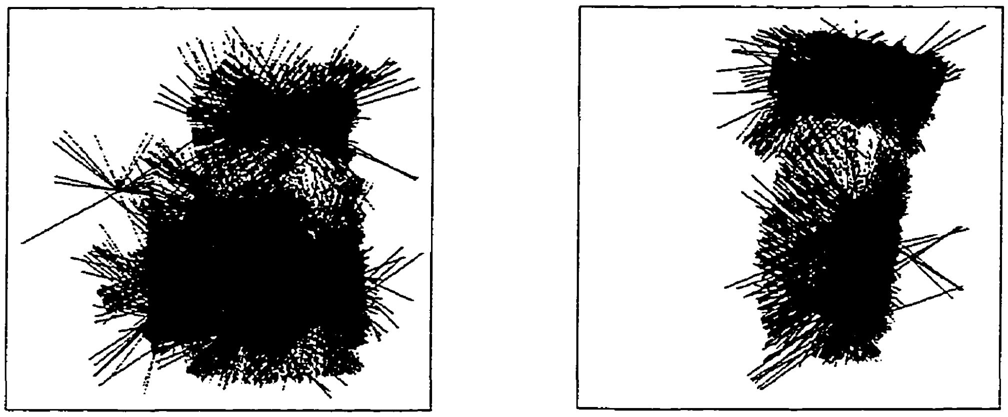


FIGURE 7.10. The occupancy grids generated by the exploration process. The RWI B-12 occupancy grid is on the right, and the Nomad 200 is on the left. Black pixels are empty, white pixels are unknown.

We were able to improve the occupancy grids however, by running the robots with the same trajectory, and lowering the rejection threshold from 300 cm to 150 cm, making the filter even tighter. Figure 7.11 shows the occupancy grids that resulted from this tighter threshold. The map contains fewer outliers, however, it is also much sparser.

Because of the trade-off between clarity and sparseness, we decided to maintain the map at two resolutions; one map was used for generating the potential field, and therefore absorbed much more information (erroneous or not) to prevent the robot from finding potential minima too often.

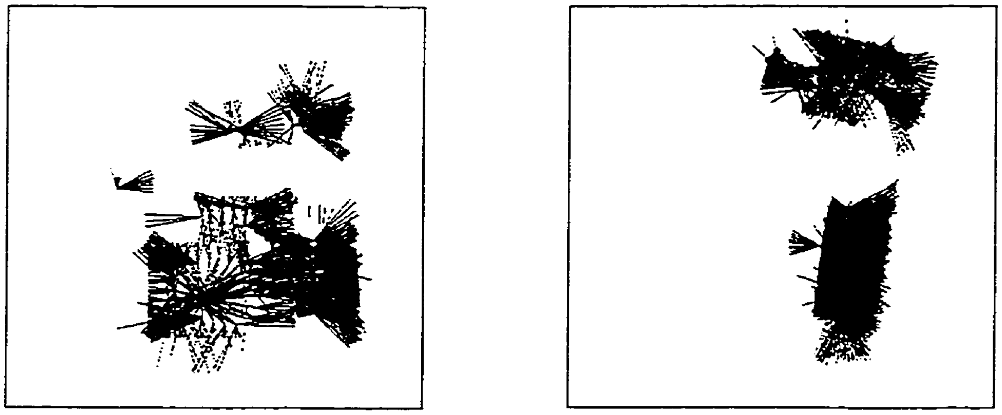


FIGURE 7.11. The occupancy grid generated by the exploration process by the robots, using a narrower threshold. Black pixels are empty, white pixels are unknown.

However, for the purposes of mapping, the data from the narrower filter was preserved, in the interests of keeping a cleaner map.

The maps generated by the exploration are depicted in Figure 7.12[3]

[3]These diagrams were generated from the robot control software, which was fed data points by the exploration algorithm from the narrow filter.

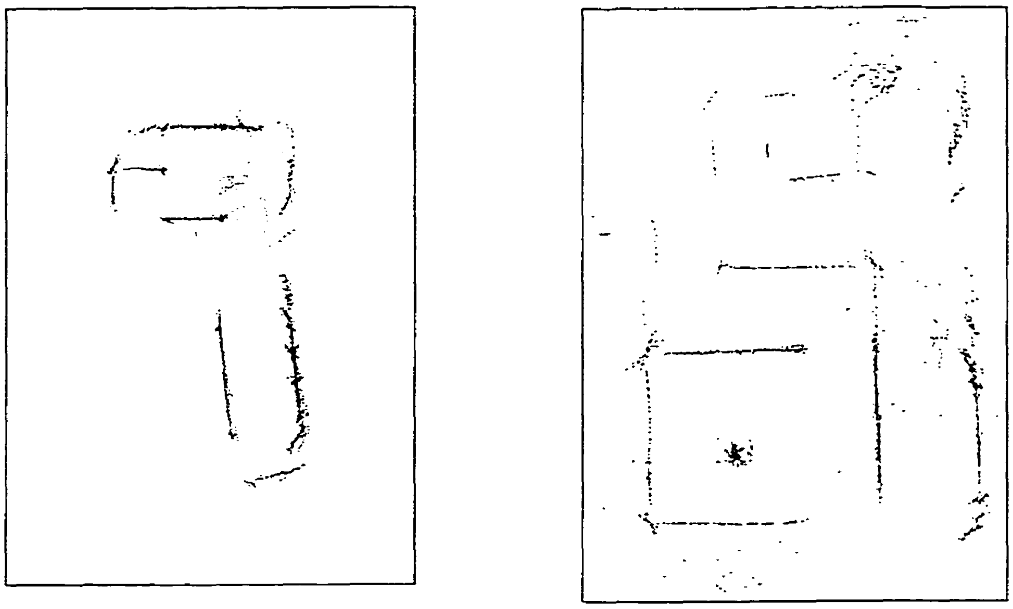


FIGURE 7.12. The maps generated by the exploration process. The RWI B-12 trajectory is on the right, and the Nomad 200 is on the left.

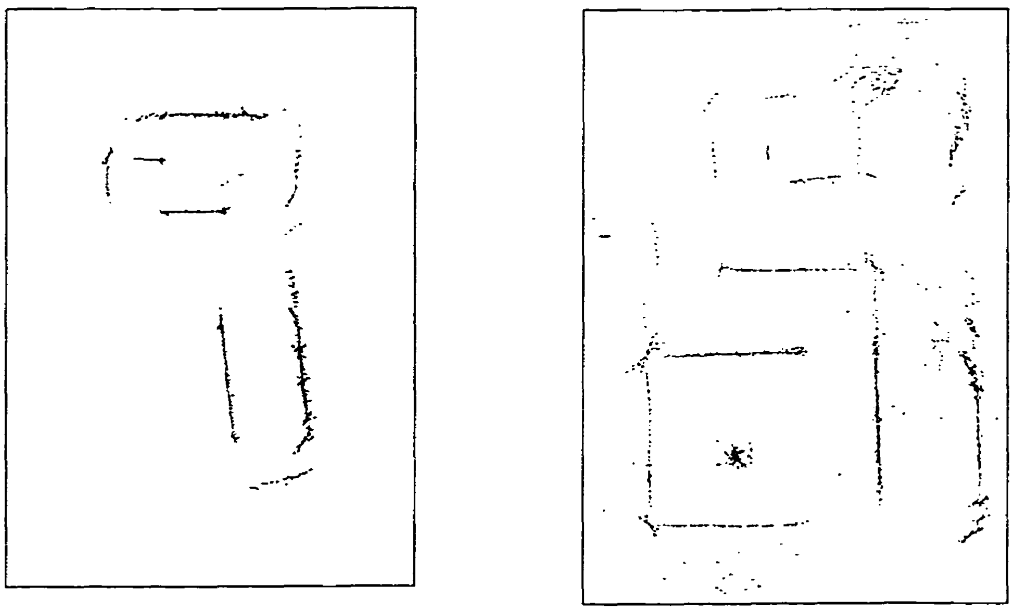


FIGURE 7.13. The same maps, after the neighbourhood filter. The RWI B-12 trajectory is on the right, and the Nomad 200 is on the left.

Notice however, that there are still a number of artefacts present, despite the Bayesian recombination of data, and the low-pass filtering. The presence of these artefacts, however, tend to be in areas where the robot was present for a very short time, (the artefacts in the right corridor of the RWI map), or where the robot never entered (the artefacts at the bottom of the RWI map). Interestingly enough, the Nomad 200 had very few such artefacts, but as a result, has a sparser map. This could be due to sensor noise, or actual differences in the sensors between the two robots. The differences between the sensors manifested themselves in the relative perceptions of the environments distinctiveness surface, as shall be shown during the discussion of rendezvous.

Another features is the angular offset of the lower corridor with respect to the upper corridor. This is due wholly to positional error, both translational and rotational. This error, over long distances, will cause serious problems to the robot's ability to accurately represent its environment. Several solutions are possible, however, this problem is outside the scope of this work.

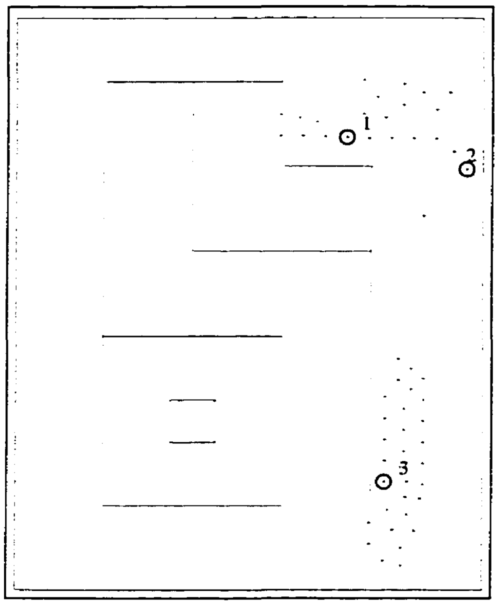
We were able to refine the map even further by applying a neighbourhood filter, which periodically eliminated any data points that had no neighbours within a certain neighbourhood. The neighbourhood chosen was 10cm. This filtering was not a part of the occupancy grid or the potential field in any way, as removing points from either of these two spatial representations is a major task, and beyond the scope of this work. Figure 7.13 shows the resulting maps after the thinning process was applied. The majority of points that were removed were those outliers where the robot had not been present much, if at all.

One of the shortcomings of this approach proved to be the memory requirement. At all times, three complete spatial representations were maintained (potential field, major occupancy grid and thinned map), in addition to any temporary representations for the breadth-first search. As an example, the potential field was typically 10m by 10m, with a discretisation of one cell per square cm. Storing this map as an array of 4 byte floats consumed 4 megabytes of memory. Combined with the occupancy grids (also stored as floats, since the grid represents probabilities), 12 megabytes were consumed. A further 4 megabytes were consumed by the breadth-first search.

7.2.3. Landmarks. Finally, Figure 7.14 shows the landmark positions that were chosen by the robots for rendezvous.

Although gradient ascent was used in the simulations, it was not used in these experiments due to the small size of the environment. Notice that the Nomad chose a point in the upper corridor as its best rendezvous location, whereas the RWI chose a point in the inner maze. This is no doubt due to sensor differences between the two robots.

7.2.4. Rendezvous. This single experiment provides the clearest support for this thesis, in demonstrating a need for establishing some appropriate behaviour if the initial rendezvous attempt is unsuccessful. As Figure 7.14 indicates, the two robots did not choose the same point in the



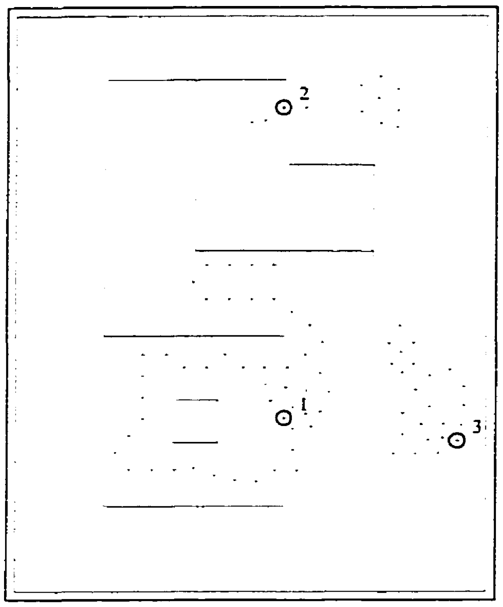


FIGURE 7.14. The landmark selections of the two robots overlaid on their trajectories. The RWI B-12 trajectory is on the right, and the Nomad 200 is on the left. The ranking of the landmarks is shown as well.

environment for the best rendezvous location. Table 7.1 shows the numerical values for the position [4] and distinctiveness of the landmarks.

Nomad 200	Landmark 1	(596.00, 189.00)	Distinctiveness: 11.15
	Landmark 2	(758.00, 234.00)	Distinctiveness: 9.73
	Landmark 3	(648.00, 676.00)	Distinctiveness: 9.29

RWI B-12	Landmark 1	(584.00, 715.00)	Distinctiveness: 4.86
	Landmark 2	(582.00, 279.00)	Distinctiveness: 3.00
	Landmark 3	(817.00, 747.00)	Distinctiveness: 2.99

TABLE 7.1. The positions and distinctiveness of the landmarks acquired by the Nomad and the RWI during the exploration.

The robots made a successful rendezvous on the 4th attempt among the three landmarks, since they were using the sequential method of exploration. As the environment in which the experiment was conducted was too small to allow the robots to use any real form of electromagnetic radio communication to detect each other, the rendezvous detection was performed using a special process.

[4]It should be noted that the relative co-ordinates of the landmarks are essentially meaningless, as there was some offset between the maps of the two robots, which has not been corrected.

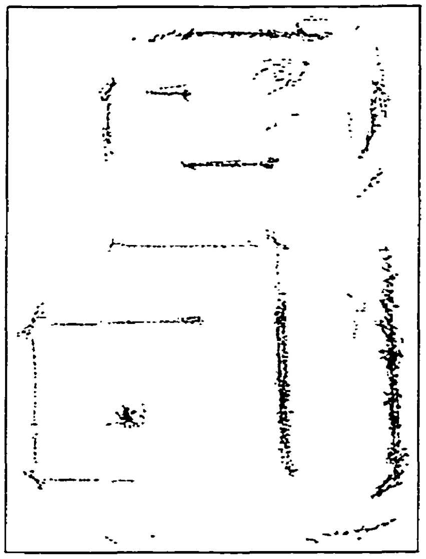


FIGURE 7.15. The final map created from the merged data acquired by the two robots.

Finally, Figure 7.15 shows that the maps were merged. The map merging was performed manually. Although algorithms exist to merge maps gathered by heterogeneous agents [30], that problem is not the focus of the present work.

7.3. Conclusion

In this chapter, we demonstrated multiple-robot exploration of an unknown environment, followed by a successful rendezvous, implemented on physical robots in the laboratory setting. The exploration method used was local potential field gradient descent combined with potential search, as described in Chapter 5. The rendezvous method was formalised in Chapter 3, and the sequential algorithm was used in the actual experiments.

While the experiments discussed in Chapters 4 and 6 were analyses of various algorithms under different conditions, the experiment discussed in this chapter was a proof of concept experiment, that rendezvous could be implemented on real robots, and the maps merged successfully. The two most important conclusions that were drawn from the experiment using the real robot is that the methodology we have chosen for achieving rendezvous is practical, and that we have addressed one of the most important issues in our methodology, the sensor differences. The fact that the robots failed to meet on the first iteration of the rendezvous cycle is a very convincing piece of evidence that the rendezvous problem is substantially more complex than simply choosing a place to meet in the environment.

This experiment underlined the issues of sensor differences in a number of ways. The maps created by the two robots, although structurally the same, differed in the details. Although the two robots were using the same type of sensor, the Nomad in general had fewer spurious measurements. Figures 7.12 and 7.13 show that the Nomad had more accurate data, and as a result, the filtering process gave the RWI the sparser map. Whether this difference was as a result of the sensors, a result of the physical configuration of the environment, or even a result of the physical configuration of the robots themselves, is unknown. However, it is clear that any algorithm that relies on the sensors of two different robots must take into account the substantial sensor differences, and cannot assume that the sensors, even if identical in nature, will perceive the environment the same way.

CHAPTER 8

CONCLUSION

8.1. Overview

In this work, we have described the new problem of performing rendezvous between multiple mobile agents. The objective was to overcome practical communication limits by periodically having the agents converge and share information. In this manner, we increase the speed of operation of the multiple robot system compared to the single robot system, while eliminating the traditional assumption of infinite range, full bandwidth communication between agents. We are specifically interested in multiple-robot exploration of an unknown environment where communication is limited to short-range line of sight. Although we dealt primarily with two-agent systems, the work is easily extensible to larger collections of agents, or swarms. Furthermore, we developed a methodology that does not depend on any particular task such as exploration, is trajectory independent and does not require any memory-intensive spatial representations. Although our implementation does take advantage of metric information that is provided by the exploration algorithm, our rendezvous methodology can be decoupled completely from the underlying primary task.

We divided the rendezvous problem into two separate subproblems. The first is determining what points in the environment constitute good rendezvous locations, or *landmarks*. We addressed this problem by modelling the environment as a function of the sensors; this function gave rise to a distinctiveness surface, defined over the domain of the environment. We then chose landmarks at the local extrema of the surface, limiting our knowledge of the surface only to those points that the agents have visited. Which points the robot visited was dictated by the trajectory prescribed by the underlying task, and so we demonstrated how to overcome these trajectory dependencies.

While the problem of rendezvous reduces, in the idealised case, only to the task of choosing the best point in the environment to which the robots should converge, this is in fact an inappropriate idealisation. In the formalisation of this problem, we identified 3 key parameters that characterise the problem. We showed that a number of different points in the environment must be chosen for meetings, and these points must be visited in some intelligent manner for rendezvous to be achieved reliably. These parameters we have called *sensor noise*, *map commonality*, and *asynchrony*.

This problem of which appropriate behaviour to use in choosing the landmarks to visit is the second of the two subproblems of rendezvous. We proposed two main classes of algorithms, *deterministic* and *probabilistic*, and gave examples of each class of algorithm. In order to determine the characteristics of the algorithms, we gave a closed-form analysis of the worst- and expected-case complexity of the algorithms at points in the parameter space. This closed-form analysis was complemented by a numerical description of the performance of the algorithms at a range of points in the parameter space.

We then described a particular exploration method that used a combination of potential field gradient descent, coupled with breadth-first search of the potential field in order to escape potential minima. This exploration method was devised to cover space quickly, and was the primary task of the rendezvous process.

Finally, we demonstrated the rendezvous algorithm in use both in simulation and on physical robots. The simulation tests were used as a confirmation of the numerical results. Within the class of deterministic algorithms, there were different regions that favoured different algorithms. An interesting conclusion from these results is that, depending on a combination of these confounding factors, no strategy is canonically a good or bad choice - under the correct circumstances, a heretofore poor choice of algorithm can outperform the erstwhile winner. These results were confirmed by both analytic closed-form solutions of Chapter 3, the idealised numerical simulations of Chapter 4 and the physically-based simulations of Chapter 6. The physical experiments served as a proof of concept for the exploration and rendezvous algorithms, and we concluded with a map of an environment that resulted from the collaborative exploration and subsequent successful rendezvous within our laboratory of two robots.

8.2. Acquiring Landmarks

The model we developed for multiple-agent systems has two distinct phases: the primary task phase, and the rendezvous phase. During the main task phase, in which our system explores the environment, the environment must also be examined for landmarks. We used the notion of "distinctiveness", that is, a locally convex function defined at every point in the environment. We defined this particular distinctiveness function as a function of the sensors directly, as opposed to computing the function from perceived features in the environment. Our distinctiveness function we used encapsulated the semantic notions of both open area, and symmetry, and was designed to be maximal at the centre of large areas. It should be noted again that there are a number of constraints on a useful distinctiveness function; it should be trajectory and rotationally invariant, locally convex and contain few discontinuities.

The landmarks were acquired concurrently with the main task, and in order to preserve the decoupling from the primary task, the acquisition is passive - the trajectory of the robot was determined wholly by the exploration algorithm. This posed two problems: the first, of determining whether landmarks that were separated in time along the trajectory, were in fact separated in Cartesian space, and the second, eliminating trajectory dependencies in the landmark measurements. We discussed a number of different methods of dealing with the first problem, and in our implementation made use of the metric information provided by the exploration method to eliminate the peaks in order to avoid the mechanical cost that would have resulted otherwise. The second problem could be solved only by gradient ascent over the distinctiveness surface at each landmark. However, in order to preserve the independence between the primary task and the rendezvous process we showed that there are several choices of when this landmark refinement process could occur, each with different costs and benefits. In our particular implementation, we performed the gradient ascent interleaved between the exploration and rendezvous phases. In this manner, we preserved the trajectory independence of the landmark acquisition, yet did not suffer the mechanical complexity or algorithmic performance degradation that the other methods entailed.

8.3. Exploration

We developed an exploration method that would motivate the rendezvous process; The objective of the exploration algorithm was to generate a map for each robot that could then be fused into a single map, combining the information from both agents.

The exploration algorithm was based on local potential field gradient descent. In this method, the robots model the obstacles in the environment as repulsive forces, and maintain an internal representation of the environment as an idealised potential field that results from the combination of the forces. As the robots use their sensors to acquire more information about the obstacles in the environment, they change their representation of the potential field to reflect the new obstacles. The robots then search their representation of the field for the potential minimum in the local area, bounded by some arbitrary distance, and move to that point.

One of the difficulties of potential field descent methods, however, is the issue of potential minima. A number of different environmental configurations can result in the robot becoming frozen in a local minimum, leaving a large part of the environment unexplored. Consequently, we developed a special case algorithm for handling these situations. By using breadth-first search, we were able to search the potential field for a global minimum in the field. The bounds of the search were the explored areas of the environment. Two spatial representations of the environment were maintained, a potential field for navigation, and an occupancy grid which was used as the map of the environment. The occupancy grid maintained a record of obstacles in the environment, points in the environment that the robot knew were likely to be unoccupied, and those points for which no information had been acquired.

The sensors that were used for this research were sonar sensors; consequently, a considerable degree of processing of the data had to be performed, especially for the experiments performed using real robots. Similar sonar errors were not present in the simulated experiments, so errors were added using a random noise model. It is difficult, and computationally extremely costly, to model most sonar artefacts in any reasonable, physically-based manner. One of the results of the physical robot experiments was that it became necessary to maintain the map at two levels of processing: one map was used for supplementing the potential field during navigation, and the other map (which underwent an additional level of processing) was used for merging with the other agents' during rendezvous.

It should be noted that although we took considerable effort to show that the rendezvous process could be separated from the primary task at all times, there are substantial costs in maintaining this separation. In fact, the implementation given in this research did not maintain the independence of the two processes, but instead the rendezvous process exploited the information acquired by the exploration process. Furthermore, there are efficiency gains that can be made by designing the primary task so that it is aware of the rendezvous process, and can make choices relevant both to the primary task and the rendezvous process.

8.4. The Rendezvous Cycle

Once the robots have explored the environment and acquired landmarks, in principle the rendezvous process should involve both robots visiting the most distinct landmark they have visited, and waiting for the other to arrive. However, as the experiment with the physical robots demonstrated, the robots may not share the same perception of the environment, and thus may not agree on the best landmark in the environment. The rendezvous problem is complicated by the three major subtleties of *sensor noise*, *map commonality*, and *asynchrony*. We formalised sensor noise as a combination of sensor differences and noise in the sensing process. Map commonality was defined as the extent to which the robots have explored the same areas of the environment. Asynchrony referred to the combination of factors that prevent an agent from reaching a landmark at the correct time.

In order to overcome these difficulties, we assumed that multiple attempts may have to be made at different locations in the environment, if any rendezvous failed. We proposed two main classes of algorithms, ***deterministic*** and ***probabilistic***, and gave two examples of each. The numerical analysis of Chapter 4 showed that each algorithm has considerable variance under different system and environment conditions. The deterministic class most often outperformed the probabilistic class, but under conditions of asynchrony, the probabilistic class showed that it had a region of superiority. The smart-sequential strategy exploited the distinctiveness measure or preference ordering on landmarks to attempt to compensate for missed rendezvous and was superior under substantial levels of asynchrony and limited noise. The pure sequential strategy was preferable when asynchrony is low, since without asynchrony a meeting is assured after n visits by avoiding visiting combinations that might otherwise compensate for asynchrony. With substantial levels of both asynchrony and moderate noise the stochastic search strategy was preferable to deterministic ones. That these small regions of parameter space exist indicates that the problem of rendezvous deserves further development.

The physically-based simulation demonstrated that, although it is much harder to isolate the parameters in a physical sense, many of the main conclusions were upheld, despite several complicating factors that were not part of the numerical simulation. Furthermore, the physically-based simulation demonstrated that an increase of speed is still attainable with a multiple robot system using the rendezvous approach to communication.

Finally, the experiments using physical robots gave a compelling demonstration that the rendezvous algorithms are an essential part of the rendezvous process; the assumption that the robots will meet on the first iteration is simply untenable. Despite very similar sensors and configurations, and a high degree of overlap between the agents, the robots required 4 rendezvous iterations before they could successfully meet and share information.

8.5. Future Work

8.5.1. Sensors. Possibly the biggest difficulty with this work is the fact that considerable effort must be made to overcome sensor error. Much of this is due to the fact that the distinctiveness function, and as a result the robot's ability to localise and rank the landmarks correctly, is based on the sensors' direct output. The sensors are demonstrably unstable, in that small changes in the pose of the robot can result in dramatic changes in the sensor output. Vastly preferable would be to construct a distinctiveness function that was a function of some stable, intermediate representation of the environment, computed from the unstable sensor output. The ability to attach semantic information to the sensor output and use that semantic information for determining landmarks would assist the rendezvous process immeasurably. Unfortunately, it is an open topic of research to find any general, stable representation of any sensor currently available. Indeed, such a system would go a long way to solving the computer vision problem.

8.5.2. Exploration. A second difficulty with this work lies in the nature of the exploration method. The exploration method was constructed to be implemented quickly and easily, be general and robust. While it succeeded in these measures, it has a number of shortcomings. Because it uses a discretised, metric map, it is extremely storage-intensive. Considerable effort was spent reducing the storage requirements of the rendezvous process, however all of this saved storage was consumed by the exploration process. Clearly, this can be addressed by reducing the granularity of the discretisation, however, there will always be an upper limit. A better solution might be to maintain only a local metric map, and combine the local potential field with a topological representation.

Another issue is the trajectory's sensitivity to initial conditions. The exploration algorithm used here could take substantial different paths through the same part of the environment, depending on the initial conditions. From the point of view of the rendezvous process, it would be preferable to have the robots take trajectories that are as similar as possible through the same region of the environment. Such a strategy, for example, would eliminate the need for landmark refinement. The undersampling problem would no longer be an issue, since both robots would be sampling the environment at the same places.

8.5.3. Landmark Acquisition. Only one distinctiveness function was investigated in this work. No consideration was given to its stability, or its ability to distinguish qualitatively different locations in the environment. Further investigation should consider different distinctiveness functions, especially with consideration given to different sensor systems.

8.5.4. Rendezvous. Only a small number of rendezvous algorithms were considered for this work. There is a body of literature on online search methods, of which rendezvous is a subclass. Algorithms that were not considered here may have surprising regions of utility.

The majority of this work was conducted with the assumption of a two-agent system. While many of the principles are easily extensible to multiple agents, the analysis may need to be refined for an arbitrary number of agents.

Duplicating the analysis of the algorithms in the physically-based simulation proved to be substantially difficult. For instance, without communication, there was no way to ensure that the landmark sets were 100% identical. Certainly, by the time sensor noise dominated the original signal, the landmark sets had lost a considerable degree of commonality. Although the formal parameters sufficed for an introduction to the problem, further work should be directed in refining the parameters. The sensor noise parameter could certainly be split into its original components of sensor differences and sensor noise. It may not even be possible to model the sensor differences as probabilities; the sensor noise model may need to be increased to a number of factors. Similarly, the asynchrony parameter should be divided into the degree of time synchronisation and also a probability of failed meeting. Currently, the asynchrony factor more closely models probability of failed radio communication, and in fact was modelled as such for the physically-based simulation.

Of the analysis presented in this work, only limited but critical parts of the parameter space were examined. Further examination is necessary for examining the behaviour of the algorithms under conditions of worsening noise, worsening asynchrony, and perhaps most importantly, conditions of landmark commonality. It is physically likely that as time passes, the areas explored by the agents will overlap more and more; analysis of the performance of the algorithms under these conditions would be useful.

One open problem is the ability of the agents to choose the appropriate rendezvous algorithm. A major part of this problem is allowing the agents to estimate the environmental parameters, and identify the correct portion of the parameter space that identifies the environment. At no time did the agent's attempt to estimate the experimental parameters; the agents did not use any environmental information in the algorithms. Allowing the agent to vary the parameters, such as constants in the stochastic algorithms, as rendezvous succeeds or fails, may have considerable power.

Finally, the only consideration used by the algorithms for choosing which landmark to visit was the distinctiveness of the landmark. Given the sometimes substantial mechanical complexity of travelling between two landmarks, a better algorithm would consider the mechanical complexity of visiting landmarks in addition to its distinctiveness, so that of two landmarks with similar distinctiveness, the closer landmark would be visited first.

REFERENCES

[1] Rachid Alami, Frederic Robert, Felix Ingrand, and Sho'ji Suzuki, *Multi-robot cooperation through incremental plan-merging*, Proc. IEEE International Conference on Robotics and Automation, 1995, pp. 2573–2579.

[2] Steve Alpern, *The rendezvous search problem*, SIAM Journal of Control and Optimization **33** (1995), no. 3, 673–683.

[3] Steve Alpern and Gal Shmuel, *Rendezvous search on the line with distinguishable players*, SIAM Journal on Control and Optimization **33** (1995), 1270–1276.

[4] Edward J. Anderson and Skander Essegaier, *Rendezvous search on the line with indistinguishable players*, SIAM Journal on Control and Optimization **33** (1995), 1637–1642.

[5] Kianoush Azarm and Gunther Schmidt, *Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation*, Proc. IEEE International Conference on Robotics and Automation (Albuquerque, NM), 1997, pp. 3526–3533.

[6] Tucker Balch and Ronald C. Arkin, *Communication in reactive multiagent robotic systems*, Autonomous Robots **1** (1994), no. 1, 27–52.

[7] ———, *Motor schema-based formation control for multiagent robot teams*, Proc. 1995 International Conference on Multiagent Systems (San Francisco), 1995.

[8] Ronen Basri and Ehud Rivlin, *Homing using combinations of model views*, Proceedings of the International Joint Conference of Artificial Intelligence (Chambery, France), Morgan Kaufman Publishers, August 1993, pp. 1656–1591.

[9] Gerardo Beni and Ping Liang, *Pattern reconfiguration in swarms - convergence of a distributed asynchronous and bounded iterative algorithm*, IEEE Transactions on Robotics and Automation **12** (1996), no. 3, 485–490.

[10] Rodney A. Brooks, *Visual map making for a mobile robot*, Proc. IEEE International Conference on Robotics and Automation, 1985, pp. 824–829.

[11] ———, *Intelligence without representation*, Artificial Intelligence **47** (1991), 139–159.

[12] B. Brumitt and A. Stentz, *Dynamic mission planning for multiple mobile robots.*, Proc. IEEE International Conference on Robotics and Automation (Minneapolis, MN), April 1996, pp. 2396–2401.

[13] Stephen J. Buckley, *Fast motion planning for multiple moving robots*, Proc. IEEE International Conference on Robotics and Automation (1989), 322–326.

[14] R. Chatila and J. Laumond, *Position referencing and consistent world modelling for mobile robots*, Proc. IEEE International Conference on Robotics and Automation, 1985, pp. 138–170.

[15] William W. Cohen, *Adaptive mapping and navigation by teams of simple robots*, Robotics and Autonomous Systems (1996), no. 18, 411–434.

[16] Thomas Cormen, Charles Leiserson, and Ronald Rivest, *Introduction to algorithms*, The MIT Press, 1990.

[17] R. Deveza, D. Thiel, A. Russell, and A. Mackay-Sim, *Odour sensing for robot guidance*, The International Journal of Robotics Research **13** (1994), no. 3, 232–239.

[18] Bruce Donald, *On information invariant in robotics*, Artificial Intelligence (1995), no. 72, 217–304.

[19] Bruce R. Donald, James Jennings, and Daniela Rus, *Analyzing teams of cooperating mobile robots*, Proc. IEEE International Conference on Robotics and Automation, 1994, pp. 1896–1903.

[20] Gregory Dudek, *Environment mapping using multiple abstraction levels*, Proceedings of the IEEE **84** (1996), no. 11, 1684–1704.

[21] Gregory Dudek and Michael Jenkin, *A multi-layer distributed development environment for mobile robotics*, Proc. Conf. on Intelligent Autonomous Systems (IAS-3) (Pittsburgh, PA), IOS Press, February 1993, pp. 542–550.

[22] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes, *Robotic exploration as graph construction*, IEEE Transactions on Robotics and Automation **7** (1991), no. 6, 859–865.

[23] ———, *A taxonomy for multi-agent robotics*, Autonomous Robots **3** (1996), 375–397.

[24] Alberto Elfes, Autonomous Robot Vehicles, ch. Sonar-based real-world mapping and navigation, Springer, Berlin, 1990.

[25] M. Erdmann and T. Lozano-Peŕez, *On multiple moving objects*, Algorithmica **2** (1987), no. 4, 477–521.

[26] F. Hara et al., *Effects of population size in multi-robots cooperative behaviors*, Proc. International Symposium on Distributed Autonomous Robotic Systems, 1992, pp. 3–9.

[27] M. B. Kadonoff et al., *Arbitration of multiple control strategies for mobile robots.*, Proc. SPIE Mobile Robots. (Cambridge, MA), 1986.

[28] S. Kato et al., *Coordinating mobile robots by applying traffic rules*, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 1992, pp. 1535–1541.

[29] K. R. Harinarayn and V. J. Lumelsky, *Sensor-based motion planning for multiple mobile robots in an uncertain environment*, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (Munich, GR), 1994, pp. 1485–1492.

[30] K. Ishioka, K. Hiraki, and Y. Anzai, *Coorperative [sic] map generation by heterogeneous autonomous mobile robots*, Proc. of the Workshop on Dynamically Interacting Robot (Chambery, France), 1993, pp. 58–67.

[31] Lindsay Kleeman and Roman Kuc, *An optimal sonar array for traget localization and classification*, Proc. IEEE International Conference Robotics and Automation (San Diego), May 1994, pp. 3130–3135.

[32] Eric Krotkov, *Mobile robot localization using A single image*, Proc IEEE of the International Conference on Robotics and Automation, 1989, pp. 978–983.

[33] Benjamin Kuipers, *Modelling spatial knowledge*, Cognitive Science **2** (1979), 129–153.

[34] Benjamin Kuipers and Yung-Tai Byun, *A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations*, Robotics and Autonomous Systems (1991), no. 8, 47–63.

[35] Benjamin J. Kuipers and Y. T. Byun, *A qualitative approach to robot exploration and map-learning*, Proc. Spatial Reasoning and Multi-Sensor Fusion Workshop (Chicago, IL), 1987, pp. 390–404.

[36] Simon Lacroix and Gregory Dudek, *On the identification of sonar features*, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (Grenoble, France), September 1997.

[37] Jean-Claude Latombe, *Robot motion planning*, Kluwer Academic Publishers, 1991.

[38] John J. Leonard and Hugh F. Durrant-Whyte, *Mobile robot localization by tracking geometric beacons*, IEEE Transaction on Robotics and Automation **7** (1991), no. 3, 376–382.

[39] T. Levitt and D. Lawton, *Qualitative navigation for mobile robots*, Artificial Intelligence **44** (1990), 305–360.

[40] Paul MacKenzie and Gregory Dudek, *Precise positioning using model-based maps*, Proc. IEEE of the International Conference on Robotics and Automation (San Diego, CA), IEEE Press, 1994.

[41] M. Mataric, *Minimizing complexity in controlling a mobile robot population*, Proc. IEEE International Conference on Robotics and Automation, 1992, pp. 830–835.

[42] David Miller, *A spatial representation system for mobile robots*, Proc. IEEE International Conference on Robotics and Automation, 1985, pp. 122–127.

[43] Hans P. Moravec and Alberto Elfes, *High resolution maps from wide angle sonar*, Proc. IEEE International Conference on Robotics and Automation, 1985, pp. 116–121.

[44] P.A. O'Donnell and T. Lozano-Peŕez, *Deadlock-free and collision-free coordination of two robot manipulators*, Proc. IEEE International Conference on Robotcs and Automation, 1989, pp. 484–489.

[45] Lynne E. Parker, *Heterogeneous multi-robot cooperation*, Ph.D. thesis, Massachusetts Institute of Technology, 1994.

[46] ______, *The effect of action recognition and robot awareness in cooperative robotic teams*, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (Pittsburgh, PA), vol. 1, 1995, pp. 212–219.

[47] J. Pearl, Artificial Intelligence Encyclopaedia, ch. Bayesian Decision Methods, pp. 49–56, Wiley-Interscience, 1987, pp. 49–56.

[48] Brian Pinette, *Qualitative homing*, Proc. IEEE Int. Symposium on Intelligent Control, 1991, pp. 318–323.

[49] Ioannis M. Rekleitis, Gregory Dudek, and Evangelos E. Milios, *Multi-robot exploration of an unknown environment, efficiently reducing the odometry error*, Proc. of the 15th International Joint Conference on Artificial Intelligence, 1997.

[50] R. Andrew Russell, *Heat trails as short-lived navigational markers for mobile robots*, Proc. IEEE International Conference on Robotics and Automation, 1997, pp. 3534–3539.

[51] Jacob T. Schwartz and Micha Sharir, *On the piano movers' problem: III Coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers*, The Intenational Journal of Robotics Research **2** (1983), no. 3, 46–75.

[52] Kazuo Sugihara and Ichiro Suzuki, *Distributed algorithms for formation of geometric patterns with many mobile robots*, Journal of Robotic Systems **13** (1996), no. 3, 127–139.

[53] David Wilkes, Gregory Dudek, Michael Jenkin, and Evangelos Milios, *Simulation of sonar mapping in complex environments using multiple reflecting surfaces*, Proceedings of Vision Interface 1991 (Calgary, AB), June 1991.

[54] Eiichi Yoshida, Masakazu Yamamoto, Tamio Arai, and Jun Ota, *A design method of local communication area in multiple mobile robot system*, Proc. IEEE International Conference on Robotics and Automation, 1995, pp. 2567–2572.

[55] S. Yuta and S. Premvuti, *Co-ordinating autonomous and centralized decison making to achieve cooperative behaviours between multiple mobile robots*, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (Raleigh, NC), 1992, pp. 1566–1574.

APPENDIX A

Derivation of Analytical Results

A.1. Expected-Case Results

A.1.1. Random.

$$P_{unsuccessful} = \frac{n-1}{n} \tag{A.1}$$

The expected-case time is the median number of iterations k, such that

$$\sum_{i=0}^{k} P(i) = 0.5 \tag{A.2}$$

where $P(i)$ is the probability of meeting in exactly i iterations. Therefore, probability of meeting on the kth iteration is 50%. For the probabilistic algorithms such as random, the probability of meeting on the ith iteration is $P_{unsuccessful}^{(i-1)} + P_{successful}$.

$$\begin{aligned}
P_{unsuccessful}^{(k-1)} + P_{successful} &= .5 && \text{(A.3)}\\
P_{unsuccessful}^{(k-1)} + P_{unsuccessful} &= .5 && \text{(A.4)}\\
P_{unsuccessful}^{(k)} &= .5 && \text{(A.5)}\\
\left(\frac{n-1}{n}\right)^{k} &= .5 && \text{(A.6)}\\
k \log^2 \left(\frac{n-1}{n}\right) &= \log^2 .5 && \text{(A.7)}\\
k &= \frac{1}{\log_2 \left(\frac{n}{n-1}\right)} && \text{(A.8)}
\end{aligned}$$

For asynchrony,

$$P_{unsuccessful} = \left(\frac{n-1}{n}\right) j \tag{A.9}$$

and from equation A.6,

$$\left(\frac{n-1}{n}j\right)^k = .5 \tag{A.10}$$

$$k \log_2 \left(\frac{n-1}{n}j\right) = \log_2 .5 \tag{A.11}$$

$$k = \frac{1}{\log_2\left(\frac{n}{n-1}\right) + log_2 j} \tag{A.12}$$

For disjoint landmark sets,

$$P_{unsuccessful} = \left(\frac{n-1}{n}\right)\left(\frac{d}{n}\right) \tag{A.13}$$

and from equation A.6,

$$\left(\left(\frac{n-1}{n}\right)\left(\frac{d}{n}\right)\right)^k = .5 \tag{A.14}$$

$$k \log_2 \left(\frac{dn-d}{n}\right) = \log^2 .5 \tag{A.15}$$

$$k = \frac{1}{\log_2\left(\frac{n}{dn-d}\right)} \tag{A.16}$$

A.1.2. Sequential. The sequential algorithm consists of the active agent cycling throughout its n landmarks, while the passive agents waits n rendezvous attempts before moving to the next landmark. From equation A.2,

$$\sum_{i=0}^{k}\left(\frac{n-i-1}{n-i}\right) = 0.5 \tag{A.17}$$

$$\sum_{i=0}^{k}\frac{n-i}{n-i} - \sum_{i=0}^{k}\frac{1}{n-i} = 0.5 \tag{A.18}$$

$$k - \frac{n}{2} = 0.5 \tag{A.19}$$

$$k \approx \frac{n}{2} \tag{A.20}$$

thus solving for k gives expected time $= \frac{n}{2}$.

For asynchrony, the probability of any given cycle failing to rendezvous is j. Each additional cycle involves a cost of n. So, the probability of k cycles meeting is the $P^(k-1)_unsuccessful + P(k)_successful$. Thus, the expected time is the number of sweeps plus the expected time of the final sweep.

$$j^k = 0.5 \tag{A.21}$$

$$k \log_2 j = -1 \tag{A.22}$$

$$k = \frac{-1}{\log_2 j} \tag{A.23}$$

Unsuccessful time $= j^k n$, successful time $= \frac{n}{2}$.

$$i = j^k n + \frac{n}{2} \tag{A.24}$$

$$i = j^{\frac{-1}{\log_2 j n + \frac{n}{2}}} \tag{A.25}$$

Similarly, for disjoint landmark sets, the probability of any given cycle failing to rendezvous is $\frac{d}{n}$. Each additional cycle involves a cost of n. So, the probability of k cycles meeting is the $P^{(k-1)}_{unsuccessful} + P^k_{successful}$. Thus, the expected time is the number of sweeps plus the expected time of the final sweep.

$$\left(\frac{d}{n}\right)^k = 0.5 \tag{A.26}$$

$$k \log_2 \left(\frac{d}{n}\right) = -1 \tag{A.27}$$

$$k = \frac{-1}{\log_2\left(\frac{d}{n}\right)} \tag{A.28}$$

Unsuccessful time $= \left(\frac{d}{n}\right)^k n$, successful time $= \frac{n}{2}$.

$$i = \left(\frac{d}{n}\right)^k n + \frac{n}{2} \tag{A.29}$$

$$i = \left(\frac{d}{n}\right)^{\frac{-1}{\log_2\left(\frac{d}{n}\right)}} n + \frac{n}{2} \tag{A.30}$$

$$\tag{A.31}$$

A.1.3. Smart-Sequential. The smart-sequential algorithm consists of the two agents cycling throughout its n^2 landmark pairs. The probability of an unsuccessful rendezvous on the ith iteration is:

$$P(i) = \frac{n^2 - i - 1}{n^2 - i - 1} \tag{A.32}$$

From equation A.2,

$$\sum_{i=0}^{k} \left(\frac{n^2 - i - 1}{n^2 - i} \right) = 0.5 \tag{A.33}$$

$$\sum_{i=0}^{k} \frac{n - i}{n - i} - \sum_{i=0}^{k} \frac{1}{n^2 - i} = 0.5 \tag{A.34}$$

$$k - n = 0.5 \tag{A.35}$$

$$k \approx n \tag{A.36}$$

thus solving for k gives expected time $= n$.

A.2. Worst-Case Results

A.2.1. Random. The worst-case result for the random algorithm is always ∞, because no guarantee exists that the robots will meet on any particular iteration. However, it should be noted that the worst-case has 0 probability of occurring.

A.2.2. Sequential. For the simplest case, the worst-case occurs if the passive robot goes first to the active robot's last landmark. Therefore, the active robot will visit go through its entire cycle of n landmarks, and find the passive robot on the last iteration.

For the asynchrony case, the worst-case occurs if the robots fail to rendezvous due to asynchrony on every iteration, thus the time to rendezvous is again ∞. Again, this scenario has a probability subset of measure 0.

For the disjoint landmark case, the worst-case occurs if the passive robot *first* visits each of its unique landmarks (i.e., landmarks unknown to the active robot). Thus, d cycles of length n must occur before the passive robot is at a landmark the active robot will visit. During the final cycle, the worst-case occurs if the passive robot is now waiting at the active robot's last landmark; the active robot goes through its entire cycle of size n first before finding the passive robot on the last attempt. Thus, the worst-case is $nd + n$ visits.

A.2.3. Smart-Sequential. The smart-sequential algorithm creates a list of pairs of landmarks of length n^2. Of these n^2 pairs, n of them contain a potential rendezvous. If all n pairs are at the end of the list, then the preceding $n^2 - n + 1$ pairs do not contain a rendezvous. Therefore, the worst case for smart-sequential is $n^2 - n + 1$.

The asynchrony derivation is the same as for sequential.

If the landmark sets of the two agents are not identical, then the number of potential rendezvous pairs in the list of n^2 pairs created by the smart-sequential algorithm is $n - d$. That is to say, of the n^2 ways of selecting two landmarks, one from each agent, only $n - d$ pairs will match. If all $n - d$ pairs are at the end of the visit sequence, then $n^2 - n + d + 1$ attempts must be made before every remaining attempt will result in a rendezvous. Therefore, the worst-case for smart-sequential if the landmark sets are disjoint is $n^2 - n + d + 1$.

Document Log:

Manuscript Version 1 — July 1st, 1997
Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX — 10 October 1997

Nicholas Roy

Center for Intelligent Machines, McGill University, 3480 University St., Montréal (Québec) H3A 2A7, Canada, *Tel.* : (514) 933-5795

E-mail address: nickr@@cim.mcgill.ca

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX

IMAGE EVALUATION
TEST TARGET (QA-3)