A RANDOMIZED APPROACH TO MINIMUM DISTANCE LOCALIZATION

Malvika Rao

School of Computer Science
McGill University, Montréal

February 2004

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

Abstract

This thesis addresses the problem of minimum distance localization in environments that may have structural self-similarities. In other words, how can a robot most efficiently collect sensor data to estimate its position and rule out ambiguity (as opposed to merely increasing accuracy). The formalism is that a mobile robot is placed at an unknown location inside a $2D$ self-similar environment modeled by a simple polygon P. The robot has a map of P, can sense its environment and hence compute visibility data. However, the self-similarities in the environment mean that the same visibility data may correspond to several different locations. The goal, therefore, is to determine the true initial location of the robot by distinguishing amongst several possibilities consistent with the sensed visibility data, while minimizing the distance traveled by the robot. We present two randomized approximation algorithms that efficiently solve the problem of minimum distance localization. The performance of our localization algorithms is validated and explored via extensive experiments on a range of simulated environments.

Résumé

Cette thèse s'attaque au problème de localisation à distance minimum dans des environnements qui peuvent être auto-semblables. En d'autres termes, comment un robot pourrait-il amasser des informations le plus efficacement possible dans le but d'estimer sa position et annuler toute ambiguité (et non pas dans le but d'améliorer son exactitude). Le problème qui se pose est le suivant: un robot mobile est placé dans un lieu inconnu à l'intérieur d'un environnement à deux-dimensions auto-similaire et devra alors calculer des données de visibilité. Cependant, l'auto-similarité dans un environnement signifie que les donnes de visibilité pourraient correspondre à plusieurs positions différentes. Le but est alors de déterminer la bonne position initiale du robot en distinguant parmi les possibilités consistantes avec les données de visibilité tout en minimisant la distance parcourue par le robot. On présente deux algorithmes d'approximation de nature aléatoires qui résolvent efficacement le problème de localisation à distance minimum. La performance de nos algorithmes de localisation est validée et explorée à travers un grand nombre d' expérimentations sur une plage d'environnements simulés.

Acknowledgements

First and foremost, I would like to thank my two supervisors Prof. Sue Whitesides and Prof. Gregory Dudek. I want to extend my gratitude to Prof. Whitesides for her scientific guidance and financial support throughout the course of my research. I would especially like to thank Prof. Whitesides for providing me with a rich and stimulating reseach experience by giving me the opportunity to travel and work on other research projects.

I would like to thank Prof. Dudek for introducing me to the topic of minimum distance localization and for always being a great source of intellectual inspiration. I am most grateful to Prof. Dudek for his boundless energy and patient supervision which got me through the often frustrating periods of thesis work.

I would like to thank Prof. Luc Devroye for useful discussions. I owe special thanks to Rob Sim for providing me with code used in generating environments and for proof-reading my thesis on short notice. I am indebted to the Systems Staff at SOCS and at CIM for their help. In particular, I would like to thank Ron Simpson and Andrew Bogecho who made it possible for me to run tons of experiments.

A heartfelt thanks to my colleagues and friends in Computer Science for their dependable help and for making school a fun place to be. In particular, I would like to thank Eric Bourque, Saul Simhon, Tallman Nkgau, Matt Garden, Alessio Salerno, and Ioannis Rekleitis.

I would like to extend my appreciation to Danielle Azar for helping me translate the abstract in French as well as for being a wonderful friend. I want to thank Jean-Francois Minardi for always coming through for me, regardless of the circumstances. This thesis would not be what it is without his intelligent advice and steadfast moral support. I am grateful to my brother, Abhijeet Rao, for making me laugh during stressful times.

Above all, I want to thank Mum and Dad for their unconditional love and support which made everything possible, and for many more things that have nothing to do with mobile robots.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

With the Mars rovers Spirit and Opportunity transmitting satellite images of the surface of Mars in unprecedented granular detail, there is no question that the field of mobile robotics has indeed come of age. In fact, for the first time in history, two mobile robots are exploring Mars at the same time. Breakthroughs in technology mean that increasingly, we have the option of sending mobile robots on missions that are too dangerous or simply infeasible for human beings. There is no dearth of uses for mobile robots. For instance, underwater robots have been deployed near active faults in the Pacific ocean to observe plate tectonic events. Another potential mission for a mobile robot could be to clear landmines in war-torn regions.

And yet, despite the obvious advances, simple questions remain, to which no satisfactory answers exist to date. One such question fundamental to the operation of a mobile robot is *"Where am I?"*. Known as the *localization* problem, it comes in two main flavours: *local localization* and *global localization*. Local localization, also referred to as pose maintenance, describes the case where the robot knows its current position approximately, and simply seeks to refine its current pose estimate. More challenging is the global localization problem, sometimes referred to as the *lost robot* or *kidnapped robot* [1] problem, where the robot must determine its initial or current pose without any prior estimates.

[1]This term was first coined by Engelson [**Eng94**]

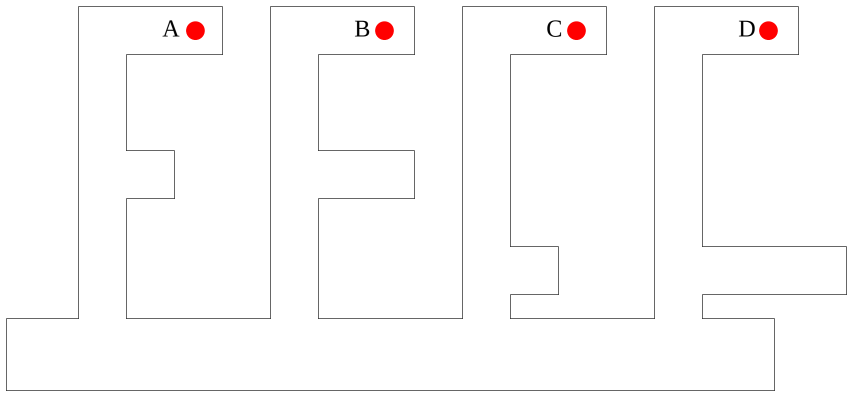


FIGURE 1.1. A self-similar environment. Visibility data at locations A, B, C, D is the same.

As anybody who has experienced a maze can attest, it is not easy to determine one's location when the surrounding environment contains few distinguishing landmarks. Likewise, global localization of a mobile robot is made even more difficult when the robot's environment is *structurally self-similar*[2]. Clearly, the thing to do in such situations is to travel, so that we may find some distinguishing feature that sheds light on our location. This thesis focuses on the problem of global localization of a mobile robot in environments that may contain self-similarities, with the condition that the distance traveled by the robot in order to achieve localization is minimized.

We shall begin with a description of the problem that we wish to solve in Section 1. Following that, we discuss the reasons that motivated us to tackle this particular problem in Section 2. In Section 3, we present our approach and identify the contributions of this thesis. Finally, we outline the organization of the rest of this thesis in Section 4.

1. Problem Statement

We consider the problem of *minimum distance localization* which is defined as follows: A mobile robot is placed at an unknown location inside a $2D$ environment modeled by a simple polygon P which may be structurally self-similar. The robot

[2]See Section 2 for a more detailed definition. A geometric definition of self-similar polygons is provided in Chapter 4

has a map of P, can sense its environment and hence compute expected visibility data. However, the self-similarities in the environment mean that the same observed visibility data may correspond to several different locations (see Figure 1.1). The goal, therefore, is to determine the true initial location of the robot by distinguishing amongst several possibilities consistent with the sensed visibility data, while minimizing the distance traveled by the robot.

In other words, we are posing the question: what path should the robot select to travel along which will allow it to most efficiently collect sensor data in order to correctly estimate its location and rule out ambiguity. The objective of our research is to find an efficient solution to the problem of minimum distance localization - one that is both efficient to implement, yet also efficient in terms of performance measures such as the length of the localization trajectory and the running time.

2. Motivation

Before proceeding further, we will define more precisely what we mean by *self-similarity* or *ambiguity* in an environment, two terms that will be used interchangeably in the rest of this thesis. In particular, we would like to distinguish between ambiguities in the structure of the environment and ambiguities perceived by the robot which can be attributed to sensor noise. In our work, we employ a perfect sensor model. In other words, the sensor contains no noise. Hence, we are solely concerned with structural ambiguity. As shown in Figure 1.1, an environment may contain several places that look alike, thus making it self-similar. Realistic examples of such environments include office buildings and hotels where several sets of rooms and even entire floors can look identical. Consequently, if a robot were to "wake up" in one such room, its estimate of its initial location would be ambiguous since it can not be certain which of the many similar-looking rooms it really is in. Environment ambiguity necessarily entails travel with the intent to search for distinguishing landmarks. For example, in Figure 1.1, the locations A, B, C, and D can be disambiguated as we travel downward to find, that for each location, the next bend in the passage occurs at different distances and consists of different sizes. The geometric definition of a self-similar polygon is deferred to Chapter 4.

Global localization refers to estimating one's position, where the space of possibilities is the entire environment [**DJ00**]. In general, global localization must contend with the fact that the observations used to estimate one's position can be ambiguous. Even with a perfect map and a perfect sensor model where the sensor is able to detect objects at infinitely long distances (provided that these objects are not occluded from the robot's line of vision) this ambiguity can be serious [**GMR97**]. It follows then, that as visibility gets poorer (i.e. the distance at which the sensor can detect objects gets shorter), which is the case with real robots, the level of ambiguity in the environment can only increase. Note that we still have not taken into account sensor noise. If we were to incorporate noise in addition to limited distance visibility in a sensor, then the level of ambiguity in the environment would obviously increase even further. We conclude, therefore, that ambiguity in an environment is not only a difficult problem but also a common one, and hence deserving of our attention [3].

Global localization intrinsically entails combining sensor measurements from multiple vantage points along some path. The problem of selecting an optimal length trajectory for minimum distance localization has been demonstrated to be NP-hard [**DRW98**].

Simple heuristic strategies for localization have been used in practice but these can sometimes exhibit strikingly poor performance [**DRW98**]. Consider Figure 1.2 adapted from [**DRW98**]. The robot, represented by the small shaded square, is situated in one of many similar corridors. However, each corridor contains a distinctive (signature) room that uniquely identifies it from the other corridors. In Figure 1.2, the distinctive room is situated a distance $d+\epsilon$ to the left of the robot' initial position. To the right of the robot's initial position are a long sequence of rooms that provide some information regarding the robot's location, although none can uniquely identify the location. If the robot were to make a purely greedy choice and decide to go to

[3]In Chapter 3 we describe experiments that demonstrate the increasing level of ambiguity in an environment with deteriorating visibility. Actual experimental results are presented in Chapter 5.

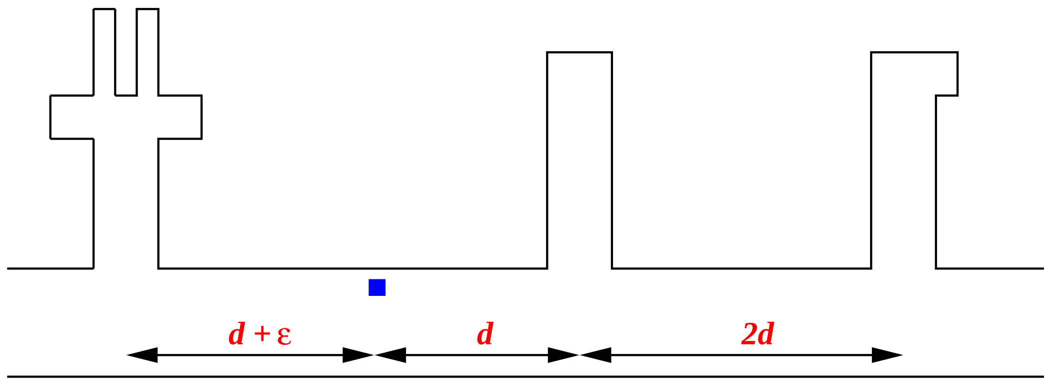


FIGURE 1.2. Simple heuristic leads to exponential path length.

the nearest place from its current location that can provide any information, then it would choose to travel a distance d to the first room to the right instead of traveling a distance $d + \epsilon$ to the signature room. Next, it would once again choose to travel a distance $2d$ instead of traveling a distance $2d+\epsilon$. If the rooms to the right were placed at distances $d, 2d, 4d, 16d, ...$ successively from one another, the robot could end up traveling an exponentially increasing distance instead of just the $d+\epsilon$ required to move from its initial position to the signature room. This example clearly demonstrates that a simple heuristic can fail quite badly to localize the robot within even a reasonable, if not near-optimal, distance margin. Hence we are motivated to search for a more intelligent localization strategy which is nonetheless efficient to compute.

3. Approach

Previous work in minimum distance localization [**DRW98**] requires the robot to make observations that are arbitrarily difficult to achieve in practice. That is, the robot is directed to visit a series of visibility cells - places where specific combinations of landmarks in the environment can be seen - even though these cells may be arbitrarily small [4]. For any real robot with any odometry error whatsoever, maneuvering into such cells may not be feasible. As a result, errors in the robot's position estimate

[4]Visibility cells are defined in Chapter 2.

might be introduced which would only propagate as the robot continues to travel, thereby compromising its chances of effective localization.

We present two randomized approximation algorithms that efficiently solve the problem of minimum distance localization based on visiting a series of randomly selected sample points in the environment from which distinguishing landmarks may be observed. In the context of our environment model, we consider landmarks to be combinations of vertices and edges that define the perimeter of the environment. Out of a set of randomly selected locations in the environment, the robot chooses the location that promises the most information gain regarding its initial location. The robot then proceeds to visit this chosen location where it collects sensor data in order to disambiguate its initial position in the environment. This process is repeated in an effort to further rule out remaining ambiguities until the robot is able to uniquely determine its initial location. We argue that by virtue of random sampling, our strategy would direct the robot to visit a particular visibility cell with a probability directly proportional to the area of that cell. Hence vanishingly small visibility cells have vanishingly small probabilities of being visited [5].

The localization algorithms we propose are distinct from the previous work in global localization in that they are the first algorithms to not only address the problem of path optimization, but to also be implemented practically with significant results. Randomized approaches to a multitude of hard problems in Computer Science have been known to yield simple and efficient algorithms where deterministic techniques have failed [6]. Our case is no exception. The algorithms we have developed are efficient to compute and yet also efficient in their performance.

4. Thesis Outline

The remainder of this thesis is organized as follows:

[5]The case where the only disambiguating position happens to be in fact a tiny cell requires special sampling techniques and is addressed in the Discussion section of Chapter 5. Note however, that if the sole disambiguating cell is so small that the robot can not navigate to a position within it, then localization is not possible with any algorithm.

[6]For a more complete listing see [**MR95**].

In Chapter 2, we introduce formally the geometric and algorithmic concepts required to understand the work done in this thesis and examine related literature that makes use of similar concepts to solve different problems.

In Chapter 3, we explore in depth previous work in global localization, including competing approaches to global localization in ambiguous environments, with the intent of placing the research done in this thesis in context.

In Chapter 4, we describe the algorithms we have developed and provide an analysis of our work. We begin by formally defining the localization problem that we are solving and stating any assumptions we have made. Next, we present two localization algorithms that solve minimum distance localization. Following that, we provide a complexity analysis and a discussion of our algorithms. Finally, we describe the techniques we used to generate random self-similar environments for the purposes of testing and experimentation.

In Chapter 5, we present experimental results obtained for a set of self-similar environments.

Finally, we conclude with a discussion of the work, as well as an outline of the open questions and possible directions for future research in this area.

CHAPTER 2

Background

The work presented in this thesis brings together three distinct research areas:

(i) Visibility
(ii) Randomized Algorithms
(iii) Global Robot Localization

Our algorithms solve the problem of global localization of a mobile robot using geometric techniques rooted in the fields of visibility and randomized algorithms. Research conducted to date on global localization, including competing approaches, will be discussed in depth in the next chapter. In this chapter however, we introduce formally the geometric and algorithmic concepts required to understand the work done in this thesis. As well, we examine related literature that makes use of similar concepts to solve different problems.

1. Visibility

The question of whether two objects can *see* each other is a fundamental one in Computational Geometry. The study of visibility spans a wide range of issues from $2D$ ray shooting, art gallery theorems, and visibility queries in polygons to their more complicated $3D$ counterparts. Visibility problems arise in a diverse array of situations. For example in Computer Graphics, when rendering a scene containing

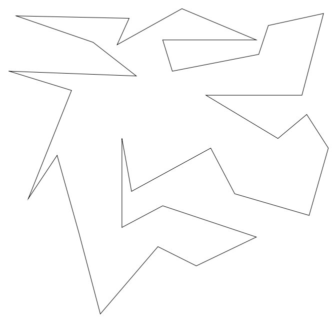


FIGURE 2.1. A simple polygon.

a collection of objects and a light source, we would like to know which objects are occluded and which objects receive and reflect light rays from the light source. In mobile robotics, determining the visibility of the robot's sensors is imperative to the robot fulfilling its mission successfully, and without hazard to itself or other objects in the surrounding environment.

Our approach to global localization involves visibility in the $2D$ plane and takes inspiration from applications of the art gallery problem. In this section we will offer precise definitions for necessary terminology and describe relevant results.

DEFINITION 1.1 (Polygon [**O'R98**]). *A* polygon *is the region of a 2D plane bounded by a finite collection of line segments forming a simple closed curve. Let $v_0, v_1, v_2, ..., v_{n-1}$ be n points in the plane where $n \geq 3$. Let $e_0 = v_0v_1, e_1 = v_1v_2, ..., e_i = v_iv_{i+1}, ..., e_{n-1} = v_{n-1}v_0$ be n segments connecting the points. Then these segments bound a* simple polygon *if and only if:*

- *The intersection of each pair of segments adjacent in the cyclic ordering is the single point shared between them: $e_i \bigcap e_{i+1} = v_{i+1}$, for all $i = 0, ..., n-1$.*
- *Nonadjacent segments do not intersect: $e_i \bigcap e_j = \emptyset$, for all $j \neq i+1$.*

The points v_i are referred to as the *vertices* of the polygon, and the segments e_i are referred to as the *edges*. A polygon of n vertices has n edges. Figure 2.1 depicts a simple polygon.

DEFINITION 1.2 (Visible). *Two points x and y contained in a polygon P are* visible *to each other or can see each other if and only if the closed straight line segment connecting them does not intersect the exterior of P: $xy \subseteq P$* [**O'R98**] *(see Figure 2.2).*

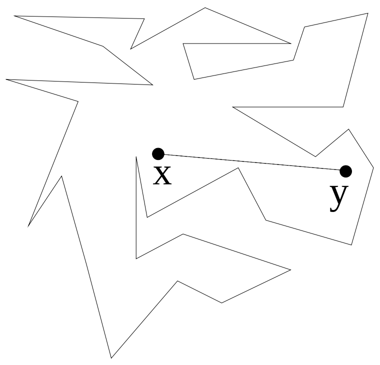


FIGURE 2.2. Points x and y are mutually visible.

In the context of a robot equipped with a range sensor or probe (such as a camera), the definition of what is visible to the robot must be modified in accordance with the capabilities of the sensor. Since all sensors are limited in range, let us assume that only those objects lying within a distance r_{max} can be detected. This gives rise to the following definition for the visibility of point x from point y, both contained in a simple polygon:

DEFINITION 1.3 (Limited Range Visible). *Let P be a simple polygon. Point $x \in P$ is* limited range visible *from point y if the closed straight line segment connecting them does not intersect the exterior of P: $xy \subseteq P$, and $d(x, y) \leq r_{max}$, where $d(x, y)$ is the Euclidean distance between x and y, and $r_{max} > 0$.*

DEFINITION 1.4 (Visibility Polygon). *The* visibility polygon *$V(q)$ from a point $q \in$ a polygon P is defined as the set of all points in P that are visible from q (see Figure 2.3).*

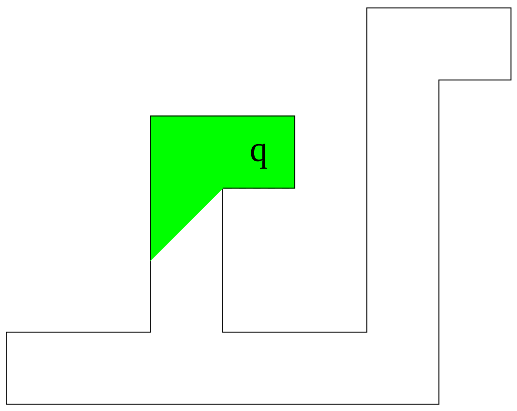


FIGURE 2.3. Shaded region represents the visibility polygon of point q in polygon P.

Visibility polygons can be computed in linear time [**GA81, Lee83**]. Next, we define a visibility polygon in the context of a range sensor which is only able to detect objects lying within a distance r_{max} as follows:

DEFINITION 1.5 (Limited Range Visibility Polygon). *The* limited range visibility polygon $V'(q)$ *from a point* $q \in P$ *is defined as the set of all points* $p_i, i = 0, 1, , , , , n-1$ *in* P *that are visible from* q, *and* $d(q, p_i) \leq r_{max}$, *where* $d(q, p_i)$ *is the Euclidean distance between* q *and* $p_i, i = 0, 1, , , , , n-1$, *and* $r_{max} > 0$.

Thus far, we have defined visibility from a point. Now, we will introduce the notion of weak visibility from an edge [1].

DEFINITION 1.6 (Weakly Visible [**Tou86**]). *A set of points* Q *is said to be* weakly visible *from an edge* e *if for each point* $q \in Q$ *there exists a point* $z \in e$ *(depending on* q*) such that* q *and* z *are visible.*

DEFINITION 1.7 (Weak Visibility Polygon). *Given a polygon* P *the* weak visibility polygon $W(e)$ *of an edge* $e \in P$ *is defined as the set of all points* $y \in P$ *that are visible from some point on* e *(see Figure 2.4).*

[1] A set of points Q is said to be *completely visible* from an edge e if for every point $q \in Q$ and every point $z \in e$, q and z are visible. A set of points Q is said to be *strongly visible* from an edge e if for every point $q \in Q$ there exists a point $z \in e$ such that q and z are visible [**AT81**].

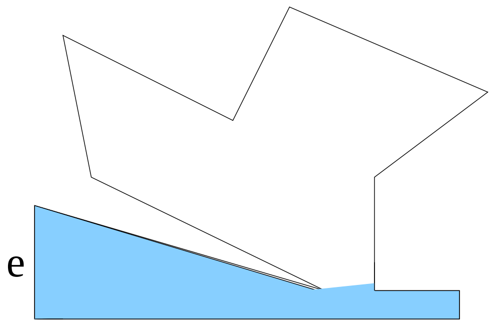


FIGURE 2.4. Shaded region represents the weak visibility polygon of edge e.

Several authors have proposed algorithms for computing the weak visibility polygon of an edge $e \in$ a polygon P that run in time $O(n \log n)$. Shortest path algorithms compute the Euclidean shortest path from a source vertex in a polygon to a destination vertex in P. Toussaint [**Tou86**] and Guibas *et al.* [**GHL$^+$87**] present linear-time algorithms for computing the weak visibility polygon of an edge $e \in$ a polygon P by making use of the relationship between visibility and shortest path problems. These solutions are linear-time because they use the linear-time triangulation algorithm of Tarjan and Van Wyk [**TW86**].

DEFINITION 1.8 (Visibility Cell [**GMR97**]). *Given a polygon P a* visibility cell *C of P is a maximally connected subset of P with the property that any two points in C see the same subset of vertices of P.*

DEFINITION 1.9 (Visibility Cell Decomposition [**GMR97**]). *A* visibility cell decomposition *of a polygon P is a subdivision of P into visibility cells (see Figure 2.5).*

The subdivision is created by introducing line segments inside P. Each line starts a reflex vertex u and is collinear with a vertex v which is either visible from u or adjacent to it in P. Hence, each such line partitions P into two regions, one where v is not visible due to the obstruction created by u and another region where v is unobstructed by u. The lines are also referred to as *visibility cell edges* (see dashed lines in Figure 2.5). Each time we move from one visibility cell to another, crossing a visibility cell edge, our view of P changes.

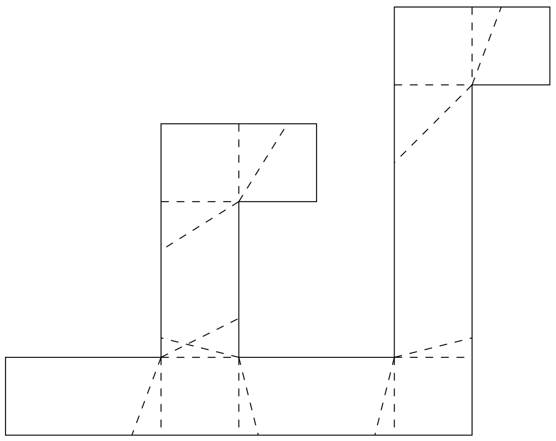


FIGURE 2.5. The visibility cell decomposition of polygon P.

1.1. The Art Gallery Problem. Consider the following scenario: The owner of an art gallery wants to place guards such that the entire gallery is thief-proof.

- *Where should the guards be placed so that the entire gallery is covered?*
- *What is the minimum number of guards required to protect the art gallery?*
- *What is the optimal placement of the guards which would ensure that only the minimum number necessary are used?*

These questions illustrate some of the issues that arise in what has come to be known as the art gallery problem. It originated during a discussion between Victor Klee and Vasek Chvatal in 1973. The original art gallery problem was as follows: determine the minimum set of points G in a polygon P such that every point of P is visible from some point of G [**She92**]. Lee and Lin showed this problem to be NP-hard [**LL86**]. Chvatal proved that the number of points of G will never exceed $\lfloor n/3 \rfloor$ for a simple polygon of n sides [**Chv75**] (see Figure 2.6).

Several variations on the art gallery problem have since been studied [**CN88, ST88, BGL$^+$97, She89**], giving rise to a large body of literature collectively referred to as art gallery problems and theorems. In addition, the practical nature of the art gallery problem has inspired other problems of a similar geometric flavour. It is easy to see the applications of art gallery theorems and results in the field of mobile robotics - we need only replace those guards with robots! In particular, art

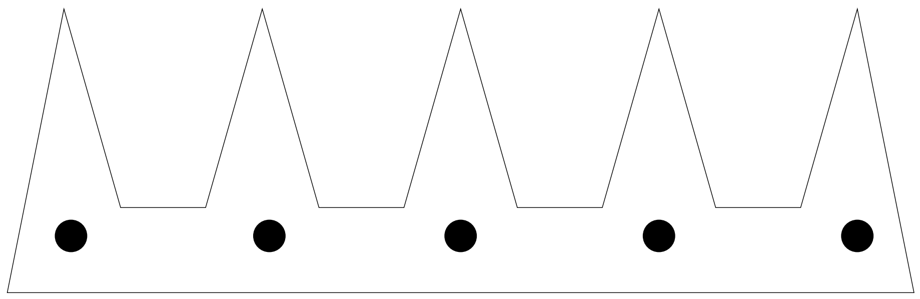


FIGURE 2.6. Chvatal's comb polygon requiring n/3 guards.

gallery algorithms have been used to compute efficient sensing locations for mobile robots. Sensing is a fundamental task imperative for robot exploration, mapping, and localization. Therefore, it is important to determine at what locations the robot should sense in order to obtain the maximum information regarding its surroundings. This formulation is referred to as the *next best-view (NBV)* problem. We will examine some art gallery algorithms in robotics in the next section.

2. Randomized Algorithms

Algorithms that use random numbers to make choices during computation are called randomized algorithms [**MR95**]. The running time and the quality of the output of a randomized algorithm, therefore, differs from one execution to another despite a fixed input. Randomized algorithms originated with Monte Carlo methods used in numerical analysis and simulation. The notion of a probabilistic Turing machine in complexity theory was first put forth by de Leeuw *et al.* [**LMSS55**]. The earliest randomized algorithms were developed by [**Ber70, Rab76, SS77**]. Since then, randomized algorithms have grown in popularity and now enjoy extensive use in different fields including mobile robotics. A comprehensive survey can be found in the book by Motwani and Raghavan [**MR95**]. The two most important advantages of randomized algorithms are simplicity and speed. They are simple to implement and their expected time performance is often substantially better than that of deterministic algorithms. Karp [**Kar91**] lists some of the principles guiding the design of randomized algorithms. We describe a subset of these principles relevant to our work as follows:

- *Random Sampling*: Small random samples are viewed to be representative of a much larger population. Hence, these samples can be used in algorithms to compute some feature of the population as a whole.
- *Foiling an Adversary*: An adversary may pick a bad input which induces a deterministic algorithm to perform poorly. Randomizing the input makes all inputs behave more or less alike on average. As a result, the adversary's strategy is less critical.
- *Abundance of Witnesses*: In some cases, a solution can be obtained by finding a witness which could verify a hypothesis. However, the search space containing the witness may be too large to be searched exhaustively. If a reasonably large number of witnesses were present, then a randomly chosen element might well turn out to be the witness.

Hence, due to the principles underlying the design of randomized algorithms, such algorithms are ideally suited to tackle NP-hard problems. For example, in linear programming, the best known combinatorial bounds were exponential in either the number of constraints or the number of variables. The first subexponential algorithms were randomized algorithms, obtained independently by Kalai [**Kal92**] and by Matousek *et al.* [**MSW92**]. Next, we will describe hard problems in robotics that employ randomized techniques to arrive at a solution.

2.1. The Next Best-View Problem. The *next best-view (NBV)* problem can be formulated as follows: determine the next pose which will allow us to extract the greatest amount of scene information. This issue arises in several fields including Computer Vision and Graphics where careful viewpoint selection becomes crucial for puposes such as object recognition, $3D$ scene reconstruction, and $3D$ modeling. In mobile robotics, next best-view algorithms have been used for a variety of tasks. Amongst other applications, NBV techniques are employed to compute near-optimal sensing locations for exploration and map-building. The NBV problem has been shown to be isomorphic to the set covering problem which is known to be NP-complete [**SRR01**]. Local planning methods are often used to solve instances of the NBV problem. As a result, computing a succession of (local) NBV positions may accumulate a rather large number of sensing operations. In this section we will describe some randomized algorithms that address the NBV and art gallery problems in the context of mobile robot navigation and data acquisition.

Gonzalez-Banos and Latombe [**GBL98, GBL01a**] present two randomized algorithms that use a $2D$ map to estimate the locations where sensing will yield the most information through random sampling of the workspace. These randomized algorithms solve an extended version of the art gallery problem. The first algorithm picks random positions in the interior of a polygonal workspace and computes the portions of the workspace boundary that would be covered by "guards" located at these positions. In order to choose the near-optimal number of positions such that the boundary is completely covered, the algorithm solves the set-coverage problem using a greedy strategy. Observing that guards located too deeply in the interior of the workspace can not see the boundary leads to the second algorithm. The second algorithm begins by sampling a point located on the boundary of the workspace. Next, the region from which this point can be observed is computed. This region is then sampled and the location with the highest coverage is retained. As a result, the unobserved perimeter of the workspace is reduced and the process is repeated until the entire workspace boundary has been covered. The authors obtain an upper bound on the running time of $O(nm^2)$ for the first algorithm, where n is the number of edges in the workspace and m is the number of random samples, and a logarithmic upper bound for the second algorithm.

In [**GBL02, GBL01b, GBML$^+$00**] Gonzalez-Banos and Latombe propose navigation strategies for exploring and building polygonal layouts of indoor environments. The robot has no a priori information about the environment and must construct a model of the environment simultaneously as it explores its surroundings. The authors address the next best-view problem and put forth an algorithm that directs the robot to construct and navigate through *safe regions* - the largest regions guaranteed to be free of obstacles, given the sensor measurements obtained so far. These safe regions are used to predict the overlap between future views and the current environment model, and to compute locations that are likely to see large unexplored areas (in other words, these are the next-best view locations). Safe regions are expanded iteratively. A first layout model is built from the data sensed at the robot's initial position. Subsequently, at each iteration, the algorithm updates the layout model by calculating the union of the existing safe region with the local safe region computed at the new sensing position of the robot. Potential sensing or next best-view positions are generated by randomly sampling the regions within the visibility range of unobserved edges of the current safe region. The candidate positions are evaluated according to the expected gain of information that will be sensed at this position and the cost of moving there. The authors also describe methods for polyline generation from sensor data and model alignment.

2.2. Probabilistic RoadMaps. Robot motion planning has been shown to be a hard problem, requiring time exponential in the number of degrees of freedom of the robot to solve it [**KL98**]. While there exists a wide range of practical path planners, the one we are interested in is the *Probabilistic RoadMap planner (PRM).* A probabilistic roadmap is defined as a network of simple paths connecting collision-free configurations generated by randomly sampling a robot's configuration space [**HKL$^+$98**]. Computing a probabilistic roadmap is divided into two phases: the preprocessing phase and the query phase. During the preprocessing phase, random free configurations of the robot are generated and connected where possible using a local motion planner. Thus a probabilistic roadmap is constructed and stored as a graph where the nodes represent the free configurations of the robot and the edges represent feasible paths computed by the local motion planner. Subsequently, queries, asking for a path between two free configurations of the robot, can be answered by searching the graph for a sequence of edges connecting the start and goal configurations.

Narrow passages in a robot's configuration space pose a threat to the connectivity of a probabilistic roadmap. By virtue of random sampling, very few configurations lying in narrow spaces are likely to be picked. Hsu *et al.* [**HKL$^+$98**] describe a random sampling strategy for finding narrow passages with probabilistic roadmap planners. Let F denote the free space. A new roadmap R' is constructed in a dilated free space F', generated by permitting the robot some penetration distance into the obstacles present in the workspace. R' now provides information regarding which areas of the original free space F require more dense sampling. As a result, local resampling operations are carried out in the neighbourhood of certain milestones and links of R' that are not in F. This leads to the discovery of new milestones located in F, thereby giving rise to a roadmap R which is wholly contained in F.

CHAPTER 3

Multiple Hypothesis Global Localization

In order to successfully carry out its tasks, a mobile robot must be able to estimate its location with respect to some representation of its environment. There exist two main types of localization problems: *local localization* and *global localization.* Local localization, also referred to as pose maintenance, describes the case where the robot's initial pose is known and the localization process seeks to refine the robot's estimate of its pose and correct any incremental errors in the robot's pose that may arise. More challenging is the global localization problem, where the robot must infer its initial or current pose without any a priori estimate of its pose given that the space of possibilities is the entire environment [**DJ00**]. The global localization problem is made even more difficult when the environment the robot resides in contains ambiguities. Previous work in global localization in the context of ambiguous environments models ambiguity in terms of multiple hypotheses, where the set of similar-looking locations in the environment constitutes the set of hypotheses. This chapter surveys competing works on multiple hypothesis global localization and seeks to place the research done in this thesis in context.

As we established in Chapter 1, ambiguity in an environment is not only a difficult problem but also a common one, which can only be exacerbated by limited sensor visibility and noise. We carried out experiments on a set of randomly generated environments measuring the level of self-similarity in each environment. Our experimental results demonstrate that while the number of self-similar locations in an environment is large enough with unlimited visibility [1], this number increases significantly as visibility gets poorer which is the case with real robots [2].

Most research to date has focused on the relatively simpler problem of pose maintenance [**Sug88, SD98, NMN94, LDW91, MD94**]. Common techniques involve Kalman filtering, triangulation, learning from landmarks, principal components analysis (PCA) and vision-based models, to name a few. A detailed survey of the various approaches to pose maintenance can be found in the book by Dudek and Jenkin [**DJ00**].

In contrast, multiple hypothesis global localization has received comparatively little attention. Existing approaches can be divided into two main categories: the probabilistic approach and the geometric approach. The most well-known family of probabilistic methods is Markov and Monte Carlo localization and their variations. These methods consider practical issues such as sensor noise, drift, and incorrect maps. Moreover, they have been implemented on real robots operating in real environments. However, some techniques utilize simple heuristics for robot motion, like wall following, which can be shown to produce unnecessarily long trajectories and can sometimes even fail to localize the robot. Other methods employ *passive localization* which has no strategy for controlling the robot's motion or its sensors. Thus the robot drifts and senses randomly in its environment, hoping to eventually localize itself.

The purely geometric research on global localization in self-similar environments assumes, for the most part, ideal conditions such as perfect sensor visibility, a correct map, and zero drift. Issues considered involve near-optimal strategies for localization that can detect all hypothetical locations and minimize the distance traveled by the robot as well as schemes to improve the computational complexity of existing algorithms.

[1]This is only to be expected given that we are dealing with environments generated to be self-similar.
[2]See Chapter 5 for actual experimental results.

1. Minimum Distance Localization

In this section we will focus on research that addresses the problem of minimizing the length of the trajectory followed by a mobile robot to achieve global localization.

Dudek, Romanik, and Whitesides [**DRW98**] consider the problem of localizing a robot in a known environment with minimum travel. They show that the problem of constructing an optimal localizing decision tree is NP-hard by giving a reduction from the Abstract Decision Tree (ADT) problem, which was proven to be NP-complete by Hyafil and Rivest [**HR76**]. In addition, Dudek *et al.* present an approximation algorithm in which the localization problem is treated in two phases: hypothesis generation and hypothesis elimination. The hypothesis generation phase computes the set of hypothetical locations that match the observations sensed by the robot at its initial location. The hypothesis elimination phase rules out incorrect hypotheses thereby determining the true initial location of the robot. For the hypothesis generation phase, the solution proposed by Guibas, Motwani, and Raghavan [**GMR97**] is used.

Guibas *et al.* address the problem of global localization by embedding. They show how to preprocess a map polygon P, by computing its visibility cell decomposition, so that given the robot's visibility polygon V, the set of points in P whose visibility polygon is congruent to V and oriented similarly can be returned. Their technique preprocesses P in $O(n^5 \log n)$ time and $O(n^5)$ space, where n is the number of vertices of P. Subsequently, it answers queries in $O(m + \log n + k)$ time, where n is the number of vertices of P, m is the number of vertices of V, and k is the number of places in P at which the visibility polygon is V. Without preprocessing, a single localization query can be answered in $O(mn)$ time.

The preprocessing phase mentioned above decomposes the polygonal map into visibility cells. This visibility cell decomposition is used in the hypothesis elimination phase in the algorithm of [**DRW98**]. First, the different hypothetical locations are overlayed together with their visibility cell decompositions [3]. The resulting overlay arrangement can yield up to $O(n^6)$ cells. Each cell in the overlay arrangement corresponds to a potential probe position which can distinguish between different hypothetical locations. Consequently the robot greedily travels to the nearest distinguishing visibility cell from its starting location in order to rule out hypotheses. The robot travels a path of at most $(k-1)d$, where d is the length of an optimal verification tour. Dudek *et al.* prove that this competitive ratio is the best possible. However, with space and time complexity of $O(n^6)$, their algorithm is impractical to use.

Kleinberg [**Kle94**] approaches robot localization by modeling the environment as a bounded geometric tree. The sensing model is based on determining the relative location and degree of nodes in the tree, where each node corresponds to a specific location in space. Schuierer [**Sch96**] proposes a technique that uses geometric overlay trees to speed up the implementation of the greedy localization strategy put forth by [**DRW98**]. While his approach reduces the time complexity to $O(kn^2)$ and space complexity to $O(kn)$, no implementation results are shown and it is unclear how to extend the technique to address more practical issues.

Buck, Schafer, and Noltemeier [**BSN99**] address the hypothesis elimination phase of the global localization problem by using voronoi diagrams of the environment. The mobile robot is assumed to be circular, with a diameter r, and uses a safety margin ϵ to all obstacles and walls. A shortest-path tree with respect to the voronoi diagram as well as an overlay tree are built. Next, the robot is moved along the voronoi vertices and senses its environment at target nodes. In this fashion, incorrect hypotheses are discarded. A running time of $O(kn\log(n))$ is achieved for an environment containing obstacles. The authors also give a competitive ratio of $2(k-1)$ times the optimum verification path restricted to voronoi paths. Clearly, a path that adheres to the voronoi diagram is longer than one that does not have to conform to this restriction. In

[3]see [**DRW98**] for more details.

fact, a voronoi path, while having the property of maximum clearance to all obstacles in the environment, may even preclude feasible, shorter paths.

The authors also describe two alternative decision strategies to that proposed by Dudek *et al.* [**DRW98**] (in which the robot greedily travels to the nearest distinguishing visibility cell from its starting location in order to rule out hypotheses). The first decision strategy, called *residual path length (RPL)* weights the choice of the next sensing point with the expected number of hypotheses to be eliminated at that destination. This strategy assumes, rather unrealistically, that all hypotheses are equally distributed. Decision strategy *inverse expected entropy (IEE)* does not assume an equal distribution of the hypotheses. Instead, it assumes that a set of ranked hypotheses have been generated with different probabilities according to their rank. A weighting function determines the final decision.

In realistic scenarios, range sensors do not provide exact visibility polygons which we may compare with the visibility cell decomposition of the polygonal environment. Moreover, range sensors have a limited sensing range in practice and minor obstacles, too insignificant to be included in the environment map, or more unpredictably still, dynamic obstacles, may affect the robot's view. Karch *et al.* [**KW99, KNW97, KNW98**] examine polygon distances for modeling the similarity between a range scan and the preprocessed visibility information used for robot localization in a polygonal map. Given a range scan and a pre-processed visibility cell decomposition, their algorithm looks for the visibility skeleton that is most similar to the scan. It does so by performing a nearest neighbour query in the set of pre-processed visibility skeletons with respect to a chosen distance function. The authors describe a distance function for star-shaped polygons called the *polar coordinate metric (PCM)*. The *polar coordinate function (PCF)* of a star-shaped polygon P corresponds to a description of P in polar coordinates with its kernel point as the origin and with period 2π. Thus, the PCM between two star-shaped polygons P and Q is the minimum integral norm between their respective PCFs in the interval $[0, 2\pi]$ over all horizontal translations. Experimental results show a 90% success rate with small scenes.

2. Markov Localization

Markov localization is a probabilistic approach to pose estimation developed by various authors in slightly differing forms [**BFHS96, CKK96, NPB95, SK95**]. This technique is based on the *Markov assumption* which states that sensor measurements at the current time are independent of past or future sensor measurements and depend solely on the robot's current pose. Markov localization proceeds by maintaining a probability distribution (also known as *belief*) over the entire configuration space of the robot. Initially the state of uncertainty of the robot's location is represented by a uniform probability distribution over all positions. As the robot moves around and senses its environment, the probability density is weighted and modified to reflect a higher probability at those hypothetical locations that appear to be more consistent with the sensor observations. Eventually most of the probability is centered around a single location, which the robot may assume to be its true position with a high degree of certainty.

Fox, Burgard, and Thrun [**FBT98a, FBT98b**] use Markov localization to determine a mobile robot's pose from sensor data. Their approach is one of *active localization*, where the localization routine assumes control over the robot's motion and sensors. The environment state space is discretized into a position probability grid and optimal actions are chosen on the basis of their utility versus their cost. Experimental results demonstrate that global localization is achieved with this technique, but the length of the localizing trajectory relative to the optimum is not considered. The authors also conducted experiments replacing their active navigation strategy with random motion, resulting in several instances where the robot failed to localize itself.

An obvious disadvantage of grid-based Markov localization is that the state space for even medium-sized environments can get exceedingly large. Moreover, increasing the grid resolution decreases the level of accuracy of the solution. If the state space is continuous, which is indeed the case with real mobile robots operating in real environments, updating the belief as the robot moves and senses new information is expensive.

2.1. Monte Carlo Localization. In order to overcome the high computational complexity of Markov localization, *Monte Carlo localization (MCL)* was introduced [**FBDT99, DFBT99**]. In Monte Carlo localization, the belief is represented by a set of N weighted, random samples or *particles*. When the robot moves, MCL generates N fresh samples that approximate the robot's new position. Each sample set is generated by randomly picking a sample from the previously computed set of samples. The likelihood of a sample being drawn is determined by its current weighting factor. Sensor readings are taken into account by re-weighting the samples.

Experimental results indicate that Monte Carlo algorithms are able to localize a robot more efficiently than Markov methods in certain cases, even succeeding where grid-based Markov localization fails [**TFBD01, FTBD01**]. A drawback of Monte Carlo methods lies in the number of samples generated in those regions of the environment with high probability distribution. The technique performs poorly if insufficient samples are generated in regions where the belief is large [**TFB00**].

Several papers [**TFB00, MSW02**] present global localization results based on the theme of Monte Carlo localization. The sampling problem noted above is addressed in [**TFB00**]. Milstein, Sanchez, and Williamson [**MSW02**] extend MCL to introduce the idea of clusters of particles. They point out that a significant limitation of MCL is that it converges too quickly on a single pose of high probability. This is undesirable in highly symmetrical environments with very few distinguishing features that allow for global localization. In such instances, it is preferable to track multiple distinct hypotheses for long periods of time. In the *Cluster-MCL* algorithm described in [**MSW02**], the different clusters represent the different hypothetical locations of the robot. The probability of each cluster is tracked by multiplying the prior probability of the cluster by the average of the likelihoods of the points that compose that cluster. At any instant in time, the cluster with the highest probability is used to determine the robot's location. Experimental results indicate that Cluster-MCL outperforms MCL in highly symmetric environments but some drawbacks exist. Clusters may all be generated at the same location or no clusters may be generated in the correct location. Also, the number of clusters may grow out of bounds and must, therefore, be limited by a pre-defined value in order to stay efficient.

3. Kalman Filter-based Localization

Position tracking using an extended Kalman filter is a local localization technique. The standard argument against using Kalman filters for global localization is that the restrictive nature of Gaussians allows only for unimodal distributions, thereby representing one pose hypothesis only [**FBDT99**]. Consequently, the case where the robot's pose is ambiguous can not be handled. In contrast, Markov and Monte Carlo localization methods represent the robot's belief in a discrete but multi-modal way. However, adaptations of the extended Kalman filter approach have been used for global localization.

Jensfelt and Kristensen [**JK01**] point out that multiple unimodal distributions can be used to represent ambiguous pose. They use a hybrid approach, that consists of Kalman filtering of Gaussian pose hypotheses and Bayesian probability, to globally localize a mobile robot. An incomplete topological world model is utilized and the robot's motion is dictated by a simple, greedy strategy. Results from experiments performed with a real robot indicate that although the robot is able to localize itself, in certain cases, it travels farther and takes longer than necessary (a shorter localizing path can be shown to exist). The authors note that some exploration strategy is necessary since random motion is not likely to lead to successful localization.

Arras, Castellanos, and Siegwart [**ACS02**] take a probabilistic feature-driven approach to multi-hypothesis global localization. An interpretation tree is maintained which pairs locally observed measurements to a global map of model features. Hypotheses are generated by searching the interpretation tree for likely pairings that match the observations sensed by the robot to the features in the global map. Geometric constraints pertaining to the properties of a feature reduce the complexity of the search. Examples of such constraints are intrinsic properties, such as feature colour, texture or dimension, relative measures such as distance and angle, and visibility constraints indicating whether a model feature is visible from a robot location. The robot's location is estimated using the *extended information filter (EIF)*, a reformulation of the extended Kalman filter to handle the case where no a priori pose estimate is available. Experiments conducted with a simulated environment demonstrate successful localization.

Gutmann and Fox [**GF02**] compare the performances of the Markov, Monte Carlo, Kalman filters as well as various combinations and extensions of these techniques in a series of localization experiments.

4. Other Approaches

Duckett and Nehmzow [**DN97**] describe a perception-based localization method where the robot is expected to rely solely on its own perceptions. Hence, no map is provided and the robot has to first explore its surroundings and build a map using a neural network. Subsequently the robot is moved to an arbitrary, unknown location within the same environment and must re-localize itself. It does so by comparing old and new information and considering the change in its perceptions over time as it moves around. A set of competing hypotheses are maintained as stored place memories, each associated with a confidence level reflecting the robot's certainty in that hypothetical location being its true location. Evidence accumulated through sensory perception narrows down the hypotheses until a single one remains. Experiments were carried out with "wall following" as well as "random wandering" as motion strategies. While better results were achieved with wall following than with random wandering, this strategy is not efficient in terms of distance traveled by the robot to achieve localization. As observed in [**FBT98a**], in certain types of environments, wall following may even fail to direct the robot to informative places.

Demaine, Lopez-Ortiz, and Munro [**DLOM02**] consider the problem of placing reflectors in a $2D$ polygonal environment in such a way that the robot is able to localize itself from any point in the environment. The robot is permitted to rotate at its current location and use its laser sensor to determine the angles at which certain reflectors are visible. The authors demonstrate that there is always a placement of reflectors, computed in polynomial time, that allows the robot to localize itself from any point in the environment. Betke and Gurvitz [**BG97**] solve robot localization in a known environment by using angles subtended by landmarks and assuming a correspondence between landmarks and points in the map. Avis and Imai [**AI90**] present localization algorithms where the environment is assumed to contain identical markers, of which the robot takes angle measurements.

Brown and Donald [**BD00**] describe algorithms for robot localization which allow for uncertainty in the data returned by the range sensor. They implement their technique on a real mobile robot and present experimental results. Tovey and Koenig [**TK03**] analyze Greedy Mapping, a mapping scheme that moves the robot from its current location on a shortest path toward the nearest unvisited or informative location until the terrain is mapped.

CHAPTER 4

Methodology and Algorithms

In this chapter we will describe the algorithms we have developed and provide an analysis of our work. We begin by formally defining the localization problem that we are solving and stating any assumptions we have made. Next, we present two localization algorithms that solve minimum distance localization. Following that, we provide a complexity analysis and a discussion of our algorithms. Finally, we describe the techniques we used to generate random self-similar environments for the purposes of testing and experimentation.

1. Assumptions

In this section we formally define the localization problem and state the assumptions we have made about the robot and its environment.

We are given a random environment modeled by an n-vertex simple polygon P without holes positioned somewhere in the $2D$ plane. A mobile robot is placed at an unknown initial location within P. The robot has a map of P (see Figure 4.1). First, the robot must determine if its initial location is unique by sensing its surroundings and matching the resulting visibility data W to the map of the environment. Given P and W, the robot must generate the set H of all hypothetical locations p_i in P such that the visibility at p_i is congruent under translation to W. Next, the robot

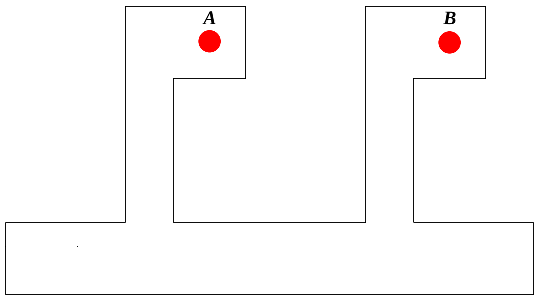


FIGURE 4.1. Polygonal environment P. Visibility data at locations A and B is the same.

must determine its true initial location by sensing and traveling in order to eliminate all hypothetical locations but one from H, while minimizing the distance traveled.

The robot is assumed to be a point robot moving in this static $2D$, obstacle-free environment. The robot is able to make error-free motions between arbitrary locations in the environment. The movement of the robot is restricted to the inside and along the boundary of the environment. As well, it is able to determine its orientation. Otherwise it would be impossible for the robot to uniquely determine its exact location in an environment with non-trivial symmetry such as a square [1].

The robot is also equipped with a range sensing device. The sensor is able to behave as a perfect sensor in that it can detect distances to those points on the boundary of the environment for which the robot has an unobstructed line of sight. However, the distance the sensor can "see" can be bounded by a constant Z. Consequently, the sensor can detect distances to those points on the boundary of the environment for which the robot has an unobstructed line of sight and which lie within a distance Z. If no finite value Z is specified, then the sensor is able to detect objects that are infinitely far provided that these objects are not occluded from the robot's line of vision.

The visibility data W sensed by the robot is composed of the counter-clockwise ordering of vertices and edges seen by the robot (see Figure 4.2). Geometric relationships amongst the data such as vertex angles, vertex and edge adjacencies, distances between vertices and edges, distances from the robot to edges and vertices, and the

[1]Note: the robot could still solve the problem up to rotational symmetry.

robot's relative position with respect to the data sensed are available. Alternately we can define W as the *visibility polygon* computed from the robot's location.

The robot must be able to identify those vertices of the visibility polygon that are also vertices of P and must know its location within the visibility polygon. Either type of data is consistent with the data that can be measured by real sensors such as laser range finders.

The polygonal environment P may be self-similar. In geometric terms, a polygon P is considered self-similar if there exists at least one set of points $q_i \in P, i > 1$ such that no two points are equivalent and the visibility polygon $V(q_i)$ of any point is congruent under translation to the visibility polygon of any other point in the set.

2. Description of Algorithms

We propose two randomized algorithms to solve minimum distance localization. Our localization algorithms (like that of [**DRW98**]) comprise two phases: hypothesis generation and hypothesis elimination. The hypothesis generation phase computes the set of hypothetical locations that match the observations sensed by the robot at its initial location. The hypothesis elimination phase rules out incorrect hypotheses thereby determining the true initial location of the robot. However, unlike the hypothesis generation phase of [**DRW98**], ours does not involve any preprocessing of the map polygon P, nor is the visibility cell decomposition of P computed. Instead we generate hypotheses using an online method. Our hypothesis elimination phase also differs from that of [**DRW98**]. In contrast to the deterministic evaluation of each visibility cell of P as a potential probe location, performed in [**DRW98**], we choose potential probe locations by randomly sampling points in certain regions of P and checking to see if the sampled location provides any new information. This avoids the computational complexity of calculating the visibility cell decomposition together with the overlay arrangement as performed in [**DRW98**]. Moreover, rather than pursue a greedy choice, which would move the robot to the nearest location from its initial location that can distinguish amongst various hypotheses, our strategy directs

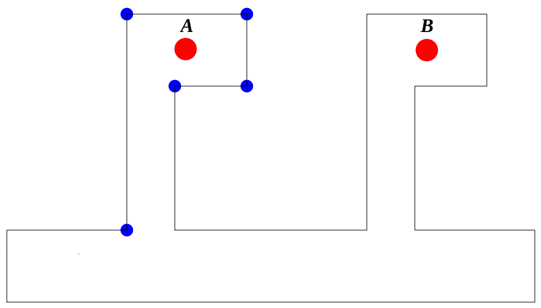


FIGURE 4.2. Visible vertices from location A.

the robot to make a weighted choice. The utility or information gain of each potential probe destination is weighted by its distance from the robot's initial location. We also investigate in our experiments, an alternate weighting strategy where the information gain at each probe location is weighted by its distance from the robot's *current* location. In general, while we believe that some form of weighted choice must be made, the actual weighting formula can vary. Our algorithms are flexible and hence, can seamlessly incorporate different decision strategies depending on factors such as the type of environment, restrictions placed on the robot, and any other objectives that the robot might simultaneously be trying to fulfill. Nonetheless, in the descriptions of our localization algorithms we will adhere to one weighting strategy - which is that the utility of each probe location is weighted with the distance from the robot's initial location. Before we proceed to describe the localization algorithms in their entirety, we will present the component procedures that make up the final algorithms and give some definitions.

2.1. Hypothesis Generation. Guibas *et al.* [**GMR97**] propose a technique for hypothesis generation which first preprocesses a polygon and then answers queries. They also outline a method for answering single shot queries without preprocessing. We chose not to use the preprocessing technique of Guibas *et al.* because of its enormous time complexity (see Chapter 3). In addition, the data structures employed in their technique appear to be complicated and therefore difficult to implement

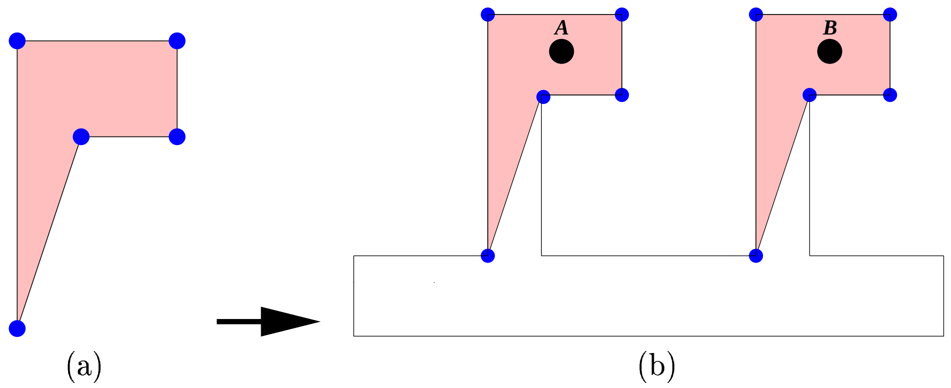


FIGURE 4.3. (a) Visibility polygon sensed by a robot; (b) Hypothetical locations.

practically. Hence we opted instead for a more intuitive and easily implementable solution. Next, we describe our algorithm for hypothesis generation.

Let W be the counter-clockwise ordering of vertices and edges sensed by the robot at its unknown initial location. Given an environment P and visibility data W, we want to determine the set H of all points or *hypothetical locations* $p_1, p_2, ...p_k \subset P$ such that the visibility data or visibility polygon computed at p_k matches W (see Figure 4.3). The hypothesis generation algorithm proceeds as follows:

- Find as many sets V of vertices in P that match W as possible. Start with the first vertex in the counter-clockwise ordered set of vertices of P and the first vertex w_0 in W. While there remain unmatched vertices in W:
- If w_0 finds a match v_0 then compute the hypothetical initial location p_k with respect to this vertex.
- Select the next vertex w_j in W that we want to match. Determine the vertex v_i in P most suitable to match w_j with. This is done as follows:
  - Let the current incomplete set of map vertices matching the vertices w_0-w_j in W be called V_k. If w_{j-1} and w_j are adjacent then select v_i in P such that it is adjacent to v_{j-1} in V_k. If w_{j-1} and w_j are not adjacent and w_j is convex, then get the next vertex in the counter-clockwise ordered set of vertices of P such that it falls at the same distance vector from v_{j-1} as w_j does from w_{j-1}.

  - If w_{j-1} and w_j are not adjacent and w_j is concave, then shoot a ray from p_k through v_{j-1}. Determine the ray's first exit point. Select the next vertex occurring counter-clockwise after the exit point such that it falls at the same distance vector from v_{j-1} as w_j does from w_{j-1}. We must ensure that all intermediate vertices occurring counter-clockwise from v_{j-1} to v_i are not visible from p_k.

- Finally check that the vertex properties of v_i (angle, incident edges, etc.) match those of w_j. Also check that v_i is at the same distance vector from p_k as w_j is from the robot's current unknown location and ensure that there is an unblocked line of sight between v_i and p_k. If all checks prove successful, proceed to matching w_{j+1}. Else start over at w_0 and the vertex in P following current (erroneous) match for w_0.
- If our trace brings us back to v_0 of V_0, exit. All possible hypotheses have been generated. Note that at least 1 valid hypothesis should be generated.

2.2. Overlay Arrangement. As we described in the previous section, the hypothesis generation phase generates a set $H = p_1, p_2, ...p_k \subset P$ of hypothetical locations in P at which the robot might be located initially. We select an arbitrary hypothetical location p_i from H to serve as a reference point or origin. Next, for each hypothetical location $p_j, 1 \leq j \leq k$, a translation vector $t_j = p_i - p_j$ is defined that translates location p_j to p_i. As a result, we compute a set of copies $P_1, P_2, ...P_k$ of the environment polygon P, corresponding to the set of hypothetical locations H such that P_j is equivalent to P translated by the vector t_j. The point in each translated polygon P_j corresponding to the hypothetical location p_j is now located at the origin p_i. We can now define the overlay arrangement as follows:

DEFINITION 2.1 (Overlay Arrangement [**DRW98**]). *The* overlay arrangement *for the environment polygon P corresponding to the set of hypothetical locations H is the structure obtained by taking the union of the edges of each translated polygon $P_j, 1 \leq j \leq k$.*

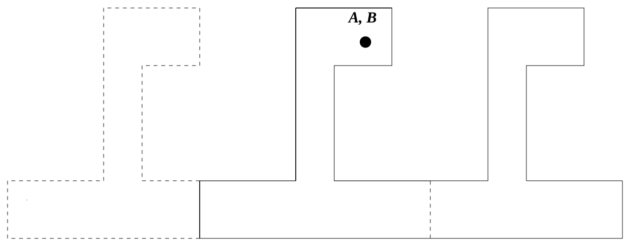


FIGURE 4.4. An Overlay Arrangement.

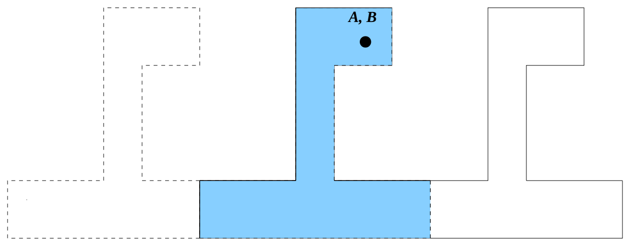


FIGURE 4.5. Shaded area represents the overlay intersection region.

Figure 4.4 illustrates an overlay arrangement. We consider only the *connected component of the intersection region of the overlay arrangement that contains the origin* since it is the area known to exist in all hypotheses (see Figure 4.5). We will refer to this connected overlay intersection component containing the origin as OI. In this respect, we differ from [**DRW98**] in recognizing that clearly, we do not need to examine visibility cells lying outside this overlay intersection region, common to all hypothetical locations, for new information. Since we only require the connected component of the overlay intersection region containing the origin, the more complicated simple polygon intersection algorithms are not required. Instead the overlay intersection area is computed using an algorithm similar to the linear time algorithm by O'Rourke *et al.* for convex polygon intersection [**OCON82**]. The algorithm performs a counter-clockwise traversal of the edges of the polygons containing the origin. It begins with an arbitrary edge e which is either partially or fully visible to the origin. Every time an intersection point of two or more edges is encountered, the edge that makes the sharpest counter-clockwise turn with respect to the origin is selected. If no intersection points are encountered then the algorithm follows the edge it is currently

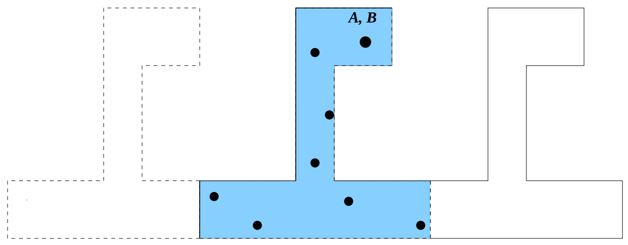


FIGURE 4.6. Random points chosen within overlay intersection.

traversing to its counter-clockwise adjacent edge. The algorithm terminates when it returns to its starting point on e.

2.3. Hypothesis Elimination. A set R of points is picked randomly from the connected intersecting region of the overlay arrangement surrounding the origin (see Figure 4.6). R is then evaluated to see if any of the points contained in R yield new information that could help to disambiguate the robot's initial location.

DEFINITION 2.2 (Random Point). *A* random point *refers to a location in the connected component OI of the overlay intersection region containing the origin for a given P, H, and origin, chosen by randomly sampling the interior of OI according to a uniform distribution.*

DEFINITION 2.3 (Useful Point). *A random point q is termed* useful *if sensing at this location is guaranteed to yield new information that distinguishes amongst the different hypothetical locations. In other words, the $Vis(q)$ with respect to the different hypotheses are not all congruent under translation.*

For each random point picked, $r \in R$, a value function $Value(r) = info/distance_{OI}$ is computed, where $info$ is the expected number of hypotheses that could be eliminated at r, assuming all the hypothesized initial locations are equally likely, and $distance_{OI}$ is the shortest path trajectory, constrained to lie within OI, from the robot's initial location at the origin of the overlay to r. We calculate $info$ for a point r as follows:

We assume that all hypotheses are equally likely. We say two hypotheses h_i and h_j are equivalent at r if $Vis(h_i, r)$ is congruent to $Vis(h_j, r)$ and has the same orientation. $Vis(h_i, r)$ refers to the visibility data computed at a point z such that the relative position of z with respect to the hypothetical location p_i is equivalent to the relative position of r with respect to the overlay origin. If there exist b equivalence classes of hypotheses at r of sizes $s_1, s_2, ..., s_b$ respectively, where the total number of hypotheses $k = s_1 + s_2 + ... + s_b$, then

$info(r) = (s_1/k)(k - s_1) + (s_2/k)(k - s_2) + ... + (s_b/k)(k - s_b)$.

The robot is moved to the random point r' in the overlay with the highest non-zero value of $Value(r')$. Those hypotheses h_i where $Vis(h_i, r')$ does not match the visibility data sensed by the robot at its new location are ruled out.

2.4. Common Overlay Localization Algorithm. We can now present the *common overlay localization (COL)* algorithm. Given an input polygonal environment P and a robot placed at an unknown initial location in P, the COL algorithm proceeds as follows:

(i) Sense visibility data W from the robot's current unknown initial location.
(ii) Generate the set of all hypothetical locations H in the environment P that match the visibility data sensed W.
(iii) Choose an arbitrary hypothetical location in H as the origin.
(iv) Construct an overlay arrangement centered on the origin.
(v) Compute the connected overlay intersection component containing the origin, OI.
(vi) Randomly choose a predetermined number of locations or points within OI.
(vii) For each random point picked, r, compute the value function $Value(r) = info/distance_{OI}$.
(viii) Observe that at each overlay intersection, there is latent information to be gained that is guaranteed to eliminate some hypotheses. Therefore, if none of the random points yield non-zero $info$, then the number of random points required is increased and chosen all over again within the current overlay

intersection area OI. Steps 6, 7 and 8 are repeated for a predetermined number of trials [2].

(ix) The robot moves to the random point r' in the overlay with the highest non-zero value of $Value(r')$.

(x) Now, eliminate hypotheses by comparing visibility data sensed by the robot at r' with the visibility data computed at all the equivalent random points corresponding to all the active hypotheses.

(xi) Let us call the set of eliminated hypotheses E. We repeat the overlay arrangement with the reduced set of hypotheses $H - E$. Steps 3-10 are repeated until only 1 hypothesis, corresponding to the true initial location of the robot, is left in $H - E$.

In step 8, we observe that each overlay intersection contains latent information that may be used to disambiguate hypotheses. Recall that in order to calculate the overlay intersection region OI, we first compute an overlay arrangement with the set of remaining, active hypothetical locations. OI comprises the connected area surrounding the origin common to all the remaining, active hypotheses. As hypotheses are eliminated, OI gets larger and larger until it consists of the entire polygonal environment P, corresponding to the case when only one hypothesis remains (the true initial location of the robot). Therefore, while $OI \subset P$, some random points lying *within* OI should be able to see distinguishing landmarks or features lying *outside* OI. The useful points that are closest to the robot's initial location clearly lie inside OI. The robot need not travel outside the current OI to gather disambiguating information. As a result, the COL strategy repeatedly chooses greater numbers of random points within the current OI until at least one useful point is uncovered.

2.5. Useful Region Localization Algorithm. The *useful region localization (URL)* algorithm differs from the COL algorithm with respect to the region where random points are chosen. Recall that the COL algorithm randomly chooses

[2]In our implementation, we terminate the algorithm if no useful points are obtained after this predetermined number of trials. Hence we proceed to the next step only if useful points exist.

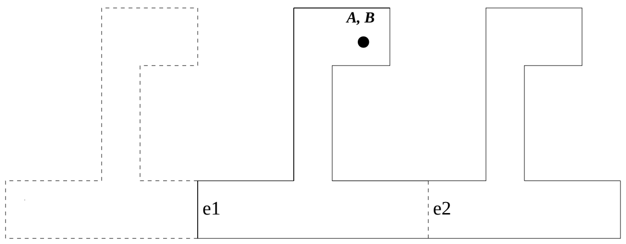


FIGURE 4.7. e1 and e2 are internal edges.

points within the overlay intersection area OI. Naturally, not all of these points are useful in eliminating hypotheses. The COL algorithm therefore, selects only those points that yield non-zero information and then, chooses amongst these select few, points with the highest weighted ratio. It turns out that we can do even better. In fact, we can determine precisely the portions of OI where any random point chosen is guaranteed to yield non-zero information.

Let us first observe that in large sections of OI, the visibility cells provide zero information gain for unambiguous localization. We must look to the boundaries of OI which is where the area common to all hypotheses ends and the"geography" changes. But the robot can spot this change from a distance as it approaches the afore mentioned boundaries. We refer to these boundaries as *internal edges*, which we define as follows:

DEFINITION 2.4 (Internal Edge). *An* internal edge *of an overlay intersection area OI is defined as an edge (one of many) that separates the inside of OI from other parts of the overlay arrangement, as opposed to those edges of OI that pertain to the outer silhouette of the overlay arrangement which separates the inside of OI from the rest of the 2D plane (see Figure 4.7).*

Once we have determined the set of internal edges of the overlay intersection area *OI*, the useful portions *U* of *OI* can be computed using the following procedure:

- Let us call the set of internal edges of the overlay intersection area *OI*, *B*. For each edge in *B*, compute its weak visibility polygon within *OI*. The union of all such weak visibility polygons should give us a region or set of

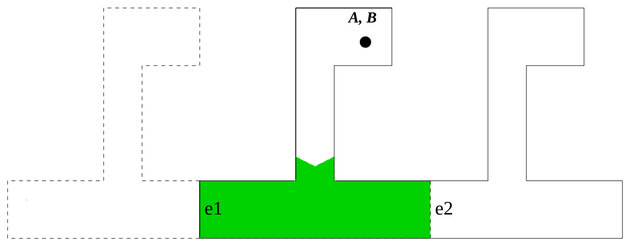


FIGURE 4.8. Shaded area represents the useful region.

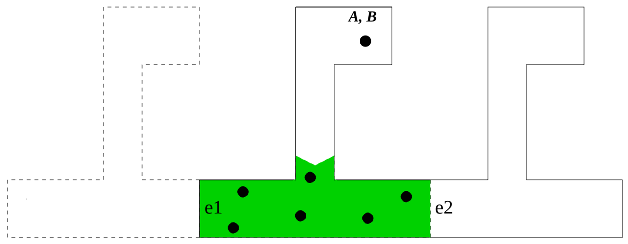


FIGURE 4.9. Random points chosen within the useful region.

disjoint regions U where U is a subset of OI. We claim that any random point chosen in U provides non-zero information whereas any random point chosen in OI-U provides zero information.

Figure 4.8 depicts the useful region of polygon P. Note that in the background chapter we have described algorithms that can be used to calculate the weak visibility of an edge. Steps 1-5 of the URL algorithm remain the same as in the COL algorithm.Consequently, the URL algorithm proceeds as follows:

(i) Sense visibility data W from the robot's current unknown initial location.
(ii) Generate the set of all hypothetical locations H in the environment P that match the visibility data sensed W.
(iii) Choose an arbitrary hypothetical location in H as the origin.
(iv) Construct an overlay arrangement centered on the origin.
(v) Compute the connected overlay intersection component containing the origin, OI.
(vi) Compute the useful region U of OI.

(vii) Randomly choose a predetermined number of locations or points within U (see Figure 4.9).

(viii) For each random point picked, r, compute the value function $Value(r) = info/distance_{OI}$.

(ix) The robot moves to the random point r' in the overlay with the highest non-zero value of $Value(r')$. Note that we are guaranteed that all the random points chosen provide non-zero information for hypothesis elimination. As a result, we do not need to choose more random points repeatedly as is done in the COL algorithm.

(x) Now, eliminate hypotheses by comparing visibility data sensed by the robot at r' with the visibility data computed at all the equivalent random points corresponding to all the active hypotheses.

(xi) Let us call the set of eliminated hypotheses E. We repeat the overlay arrangement with the reduced set of hypotheses $H - E$. Steps 3-10 are repeated until only 1 hypothesis, corresponding to the true initial location of the robot, is left in $H - E$.

It must be noted that once we have computed the useful region U, we need choose only one random point lying within U at each stage in order to guarantee that we eventually localize successfully (we could even just move the robot to the closest vertex or edge of U). A larger number of random points only serves to improve the performance of the algorithm by eliminating several hypotheses in one shot or by reducing the distance traveled by the robot. Unlike the COL algorithm, the URL algorithm has computable and finite time bound. Consider a situation where the useful region comprises a small fraction of the entire overlay intersection area OI (this need not mean that the useful region is in itself a small area where a robot can not navigate - simply that it is relatively small compared to the entire overlay intersection area). In such cases, choosing points randomly from the entire overlay intersection region may yield a useful point only with a large number of random points and after several trials. The advantage of the URL algorithm is that we can instantly access the useful portions of the overlay intersection and hence the useful points.

2.5.1. *Proof of Correctness.* We prove the validity of our claim that any random point chosen in the region *U* provides non-zero information for hypothesis elimination and points lying in the region *OI-U* provide zero information. Before we can proceed, however, we must first show that being able to see an internal edge partially or fully from a point in OI implies that that point provides disambiguating information. Likewise, any useful point in OI must be able to see an internal edge partially or fully.

PROOF. OI is defined as the connected component of the intersection region of the overlay arrangement that contains the origin and is common to all hypotheses. Pick an arbitrary point p located in OI and such that it can see some point(s) on an internal edge e of OI. Let us assume that p provides no new information. This implies that the visibility polygon at p, $Vis(p)$, is wholly contained in OI and therefore common to all hypotheses [3]. However, p was chosen such that it could see some point(s) on e and e is defined as an edge separating the connected area common to all hypotheses from the area that is not common to all hypotheses. Without loss of generality, we may assume that p can also see some point(s) on an edge $e', e' = e + \delta$, where δ represents a small displacement of e in a direction away from OI such that e and e' remain parallel and e' is not contained in OI. No point on e' is common to all hypotheses. Hence we have a contradiction.

Likewise, choose an arbitrary point p located in OI and such that it can not see any points on any internal edges of OI. Let us assume that p provides new information. This implies that the visibility polygon at p, $Vis(p)$, includes areas of the overlay arrangement not common to all hypotheses and therefore lying outside OI. Since an internal edge is defined as an edge separating the interior of OI from

[3]Disambiguating information necessarily consists of landmarks/features that differ amongst the set of hypotheses

other parts of the overlay arrangement, it follows that $Vis(p)$ must include some point(s) on at least one internal edge. Hence we have a contradiction.

□

We can now proceed to prove that any random point chosen in the region U provides non-zero information for hypothesis elimination and points lying in the region *OI-U* provide zero information.

Proof. Pick an arbitrary point q located within the useful region U. Let us assume that q *provides no new information* to rule out hypotheses. As we have already established, this implies that q can not see any point on any of the internal edges of the current overlay intersection area *OI*. In turn, this implies that no point lying on any internal edge e can see q. In other words, the weak visibility polygon of e does not include q. However, U is defined as the union of all the weak visibility polygons of all the internal edges. Hence we have a contradiction.

Similarly, pick an arbitrary point q' located within the region *OI-U*. Let us assume that q' *does provide new information* to rule out hypotheses. As we have already established, this implies that q' can indeed see at least one point on at least one of the internal edges of the current overlay intersection area *OI*. Therefore, there exists at least one point on an internal edge e' that can see q'. Hence the weak visibility polygon of e' must include q'. However, U is defined as the union of all the weak visibility polygons of all the internal edges and we picked q' from the region *OI-U*. Hence we have a contradiction.

□

2.6. Limited Visibility. When the sensor visibility is limited, the robot sees only a subset of the vertices and edges of its environment that it could otherwise see with unlimited visibility. Let S_i denote the set of vertices seen by the robot at visibility d_i. In general, for visibility values $d_1 < d_2 < d_3 < ..... < d^*$ where d^* represents unlimited or infinite visibility, the corresponding sets of vertices seen by the robot have the following relationship: $S_1 \subset S_2 \subset S_3 \subset ..... \subset S^*$. Consider Figure 4.10.

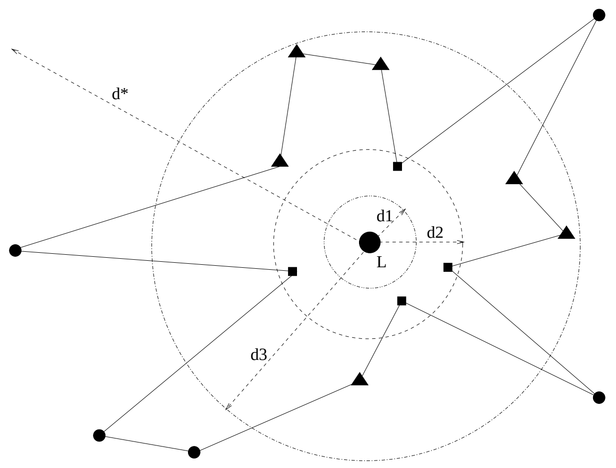


FIGURE 4.10. What the robot sees under a range of visibilities.

The robot, labeled L, is situated at the centre of the polygonal environment. With visibility d_1 the robot sees none of the vertices of the polygonal environment. With visibility d_2 the robot sees only those vertices represented by squares. With visibility d_3 the robot sees the vertices represented by squares as well as those represented by triangles. With unlimited visibility d^* the robot is able to see all the vertices of the polygon but not beyond since it cannot see through walls or solid edges.

All visibility computations and comparisons are restricted by the distance limit d beyond which the robot is assumed to not see. When computing the initial visibility data W sensed by the robot, those vertices and edges lying outside the circular area of radius d centered at the robot's initial location are not included. Likewise, in the hypothesis generation phase, we only look for patterns of vertices that conform to W as calculated incorporating the distance limit. During the hypothesis elimination phase, the robot's visibility at a particular random point is calculated in the same way as W. In addition, the distance limit is taken into account when we determine which random points provide non-zero information. As a result, in the COL algorithm, the number of random points yielding any information gain as a proportion of the total number of random points chosen in the entire overlay intersection region decreases as the visibility gets poorer.

In the URL algorithm, when computing the weak visibility polygon of the internal edges, we clip the polygon by a factor equivalent to d. This is accomplished by aligning an infinitely long rectangle along each internal edge, where the width of the rectangle (the axis aligned to the width is perpendicular to the internal edge) is equivalent to the distance limit d. The intersection of the rectangle with the original weak visibility polygon produces the clipped polygon.

As long as the set of points comprising the initial visibility data W sensed by the robot contains at least one vertex, both localization algorithms are able to successfully estimate the robot's initial location. However, if W is an empty set then localization is not feasible since the number of potential hypothetical locations is infinite. This situation can only occur if the robot's visibility is limited to a finite value d such that the circular area of radius d centered at the robot's initial location does not cover any vertices or edges (see Figure 4.10).

It follows that the more limited the visibility of the robot's sensor, the more the number of hypothetical locations that are likely to be generated. As well, the length of the localization trajectory is likely to be longer. The increased length of the localization trajectory can be attributed to the robot's "short-sightedness" which requires it to move much closer to distinguishing features in order to be able to see them than otherwise. Simply having a larger number of hypotheses to eliminate does not necessarily imply longer path lengths.

3. Complexity Analysis

In this section we will analyze the running times of the component algorithms that comprise our localization algorithms and estimate the overall complexity of the two localization algorithms. We begin by determining the running times of procedures common to both localization algorithms.

The number of hypothetical locations k is bounded above by the number of reflex vertices in the polygonal environment [4]. Let n denote the number of vertices in the

[4]For a proof see [**GMR97**].

map polygon P. Hypothesis generation takes time $O(mn)$ where m is the number of vertices in the visibility polygon W.

The intersection of two simple polygons takes $O(n \log n)$ time [**CE92**]. While we are using a simpler algorithm in our implementation to compute only the intersecting portion around the origin, for the purposes of this complexity analysis we will adhere to the time of $O(n \log n)$. Therefore, calculating the overlay intersection area around a chosen origin takes time $O(fn \log n)$ where f is the number of hypotheses remaining ($f \leq k$), and therefore, the number of translated polygons. Since $k = O(n)$ and $f = O(k)$, we get $O(n^2 \log n)$ as an upper bound on the time taken to compute the overlay intersection. Observe that as hypotheses are ruled out the value of f will decrease. If we only rule out only one hypothesis per intersection then the overlay intersection is computed a maximum of k times.

To compute visibility data at each random point takes $O(n)$ time. Likewise, to compare the visibility data at two points also takes $O(n)$ time. Since we will have to make f comparisons, where f is the number of hypotheses remaining ($f \leq k$), in order to calculate the information that each random point can give, we take $O(fn)$ time per random point.

3.1. Common Overlay Localization Algorithm. At each intersection let us say that we drop X random points on average. We say "on average" because this number could increase or decrease with respect to the size of the overlay intersection area and depending on our strategy. Experimental results described in the next chapter give empirical measures of the range of values we may expect for X to achieve localization with different environments.

We spend a total time of $O(Xfn)$ computing and comparing visibility data in order to determine which of the X random points provide new information. Of the X points chosen, let only Y be useful points where $Y \leq X$. The useful points are sorted according to their weighted ratio. In order to compute the distance of each point from the initial location, a shortest path algorithm is used. This amounts to $O(n \log n)$ cost per point and so $O(Yn \log n)$ for all the useful points.

Assuming that we only eliminate one hypothesis at each overlay intersection, we will have to calculate the overlay intersection area, compute and compare visibility data of all the random points and calculate the shortest path distance of the useful random points, a total of k times. Therefore we estimate the overall time complexity of the COL algorithm to be $O(mn) + k(O(fn \log n) + O(Xfn) + O(Yn \log n))$. A looser but simpler bound for the overall time complexity of the COL algorithm is $O(n^3 \log n + Xn^2)$.

3.2. Useful Region Localization Algorithm. The maximum number of internal edges possible at any overlay intersection is $O(fn)$ where f is the number of hypotheses remaining ($f \leq k$). We argue our claim as follows:

Proof. By definition, the overlay intersection area OI, being the region common to all the hypotheses, is a subset of the entire polygonal environment P. As hypotheses are eliminated, OI gets progressively larger, until there is only one hypothesis left in which case OI is equivalent to P. OI is calculated by traversing the boundary of the overlay arrangement lying around the chosen origin in a counter-clockwise direction. Everytime a point is encountered where two or more edges intersect, the edge that makes the sharpest counter-clockwise turn with respect to the origin is picked. As a result, each edge of the polygon translates comprising the overlay arrangement is traversed either once (fully or partially) or not at all. In the instance where an edge is traversed partially, it is never re-visited since the algorithm would have abandoned that edge in favour of one that makes a sharper counter-clockwise turn, and given that these polygon edges are straight-line segments. Hence, the number of internal edges present in any overlay intersection is bounded above by the total number of edges of all the polygon translates. If we have f hypotheses remaining ($f \leq k$), and each polygon consists of n edges, then the maximum number of internal edges possible at any overlay intersection is $O(fn)$.

□

Computing the weak visibility of an edge takes $O(n \log n)$ time. As a result the total cost of computing the useful region is $O(fn^2 \log n)$. Therefore we estimate the overall time complexity of the URL algorithm to be $O(mn) + k(O(fn \log n) + O(fn^2 \log n) + O(Yfn) + O(Yn \log n))$ which in turn reduces to $O(n^4 \log n + Yn^3)$.

If the number of random points gets very large then the running time likewise increases for both localization algorithms. However, in the URL algorithm we compute the region where any random point chosen is guaranteed to be useful. While this computation comes at a cost, it nonetheless reduces the number of random points required to obtain path lengths equivalent to those produced by the COL algorithm. Besides, with the URL algorithm, we can fix the number of random points to be a constant and the algorithm will still localize the robot successfully. This is not the case with the COL algorithm where increasing numbers of random points have to be chosen again if no useful points exist in the current lot. In environments where the useful region comprises a very small fraction of the entire overlay intersection area, the COL algorithm might require a large number of random points in order to achieve successful localization.

4. Environment Generation

In order to test the localization algorithms, random self-similar environments, any number of which could be generated automatically, were needed. Moreover, the environments were required to contain challenging, and potentially large number of self-similarities, while at the same time, displaying sufficient variation in shape and size to be interesting. We generated simulated office environments using three different techniques.

The first set of environments were modeled after mazes. The algorithm that generates these environments proceeds as follows: First a fixed initial space is divided into a rectangular grid of cells of a specified size (e.g. $100units \times 100units$). A predetermined quantity is chosen which limits the number of grid cells in the environment (and hence its size) we are about to generate. Next we randomly picked a grid cell

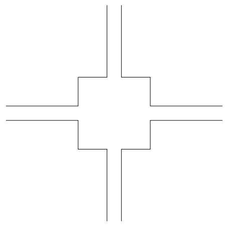


FIGURE 4.11. A room cell.

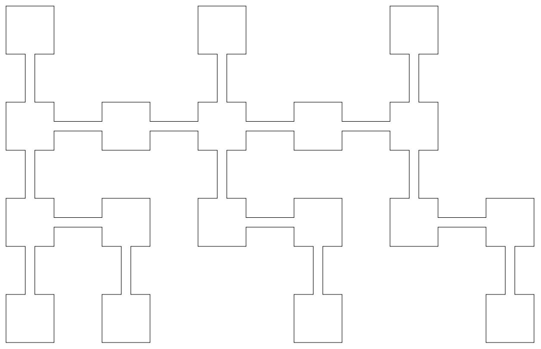


FIGURE 4.12. A simulated office environment.

as the start cell. This cell serves as a "room cell" (see Figure 4.11). We randomly determine the number of passages leading to other room cells from this start cell. There can be a maximum of 4 passages, corresponding to the available adjacent grid cells of the start cell. For each of the destination room cells, we repeat this procedure until the environment size limit is reached or until the canvas boundaries are encountered. Finally the room cells and passage cells are connected to form a closed simple polygon (see Figure 4.12).

Signature step environments, where each step contains a distinctive structure (signature) identifying it uniquely from the other steps, were created as well [5]. The signature is produced by randomly generating a sequence of characters and determining the ASCII number representation for each character. The number of characters in the sequence determines how many steps will be built whereas the binary encoding of each character determines the signature on the step corresponding to that character. This signature takes the form of a sequence of protrusions on the step, where

[5]The code for generating these environments was implemented by Rob Sim.

the length of each protrusion indicates whether that binary bit is on or off in the encoding (see Figure 4.13).

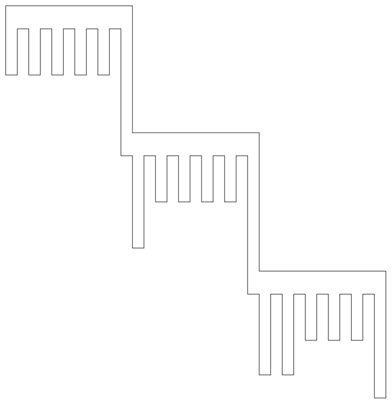


FIGURE 4.13. A signature step environment.

We also considered environments that consist of a "basic unit" polygon which is replicated and placed in several recurring positions in a fixed space. An input polygon or basic unit is initially placed at a randomly chosen position in a given space. Next, this polygon is replicated and the duplicate is placed at one of a set of fixed positions that are offsets from the first polygon's position. The two polygons are then connected via a passage. The resulting simple polygon then becomes the basic unit input for the next iteration. This process is repeated for a pre-determined number of iterations or until the size limit for the final environment is reached. Figure 4.14 displays an example polygon.

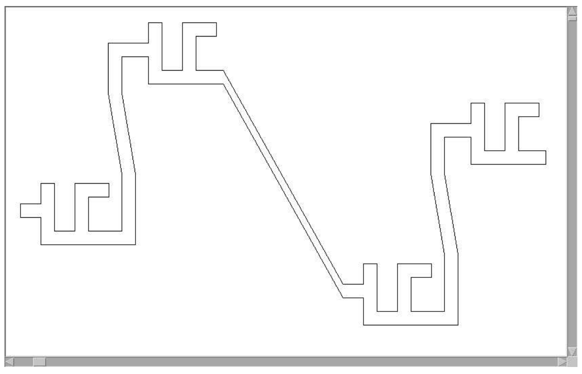


FIGURE 4.14. Another type of environment.

CHAPTER 5

Experimental Results

This chapter presents experimental results that demonstrate the performance and scope of our algorithms. Each set of experiments will highlight different aspects of our solution to minimum distance localization, such as the variation in path lengths for a multitude of environments with increasing numbers of random points, the effect on path length of a range of sensor visibility values, comparisons of the performance of different decision strategies, measures of self-similarity in a set of environments, etc. In addition to these statistical experimental results, we also include visualizations of our localization technique for a few sample environments. Finally we provide a discussion of our work.

Unless otherwise stated, the reader may assume that the algorithm used to achieve localization in a particular experiment was indeed the COL algorithm. Also, all experiments were performed on the simulated office environments that were modeled after mazes (see Figure 4.12). Visibility ranges were measured in pixel units [1].

1. Measure of Environment Ambiguity

The objective of these experiments was to measure the average number of hypothetical locations that may correspond to a randomly chosen initial location. In other

[1]The algorithms were implemented in Java Version 1.4.1 and the simulations were run on Linux machines.

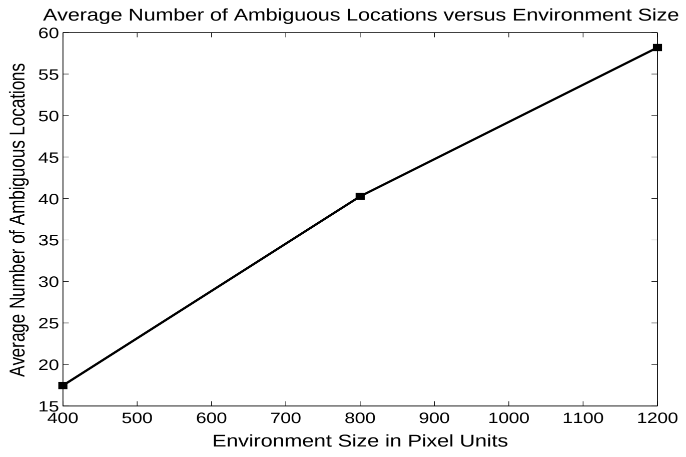


FIGURE 5.1. Level of ambiguity increases with the size of the environment.

words, we wanted to estimate the level of ambiguity in the environments. The procedure was as follows: For each environment in our set of environments, we randomly chose 10 initial locations. We then generated hypotheses for each of these 10 locations. We picked the maximum number of hypotheses generated for an initial location as representative of the level of ambiguity possible for that particular environment. Finally we took the average of all the maximum number of hypotheses obtained for all the environments in the set as the average value for that set of environments and for a particular range of visibility.

We generated 100 simulated office environments of 3 sizes. The first set were of size equivalent to 100 grid cells or approximately 400 vertices on average. The second set of environments were of size 200 grid cells or 800 vertices on average. The third set of environments were of size 300 grid cells or 1200 vertices on average.

Two sets of experiments were conducted. The first set of experiments measured the variation in the level of ambiguity as the environment size gets bigger. The visibility of the robot's sensors was assumed to be unlimited. Figure 5.1 shows the level of ambiguity plotted against environment size . These results indicate that the level of ambiguity is directly proportional to the size of the environment. In fact, we observe an almost linear increase in the number of hypotheses as environment

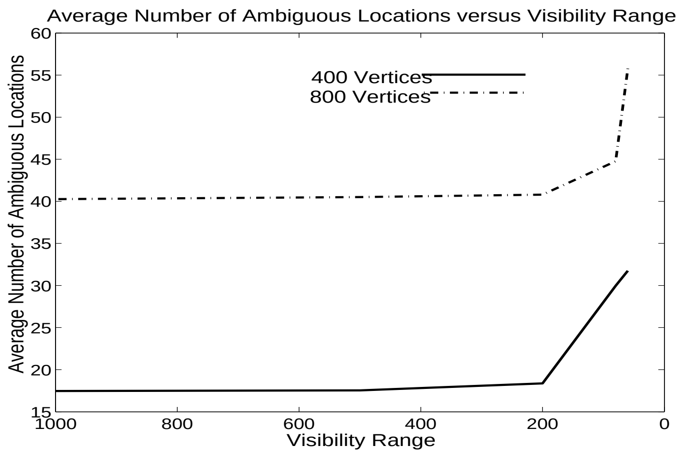


FIGURE 5.2. Level of ambiguity increases as visibility gets poorer.

size gets bigger. This makes sense given that each set comprised the same type of environments - only the size varied.

Next, we carried out experiments to evaluate the effect of varying visibility on the level of ambiguity. Two sets of environment were used, those of size 400 vertices and those of size 800 vertices. Figure 5.2 shows the level of ambiguity plotted against decreasing visibility range. The level of ambiguity increases very sharply as visibility dips below the 200 mark. As the size of each "cell" in the environment is 100 pixels, visibility range values below 100 give rise to the case where the robot is increasingly, not able to see the edges and vertices that comprise its immediate surroundings. In the limit, we approach the case where the robot sees only empty space which corresponds to an infinite number of hypotheses.

2. Path Lengths with Unlimited Visibility

We generated 73 simulated office environments, each with average number of vertices approximately 400. For each of the 73 environments an initial robot location was randomly selected. We then ran the COL algorithm with these environments and their respective initial locations for a series of different quantities of random points. The sensor visibility distance was unlimited or infinite (perfect sensor model). The objective was to measure the average distance traveled by the robot or average path

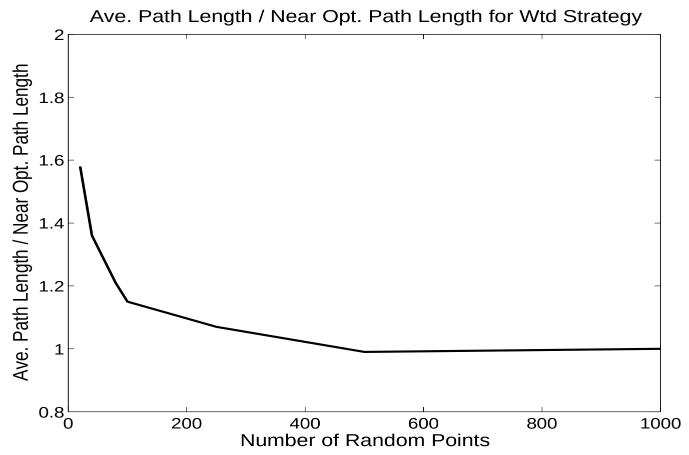


FIGURE 5.3. Performance of weighted strategy with simulated office environments based on 73 trials.

length as the number of random points are increased and to compare these average path lengths to the near-optimum result (recall that computing the optimum path is NP-hard). The number of random points was varied from 20 to 1000 and the average path length obtained for 1000 points was used as the near-optimum result. Figure 5.3 shows the ratio of the average path length to the near-optimum path length plotted against the number of random points. Our results indicate that the average path length gets significantly shorter initially as the number of random points is increased. However, after approximately the 500 random point mark, the incremental reduction in path length decreases, eventually settling at a more or less steady value.

Note that the number of random points specified in our plots showing the performance of the COL algorithm refers to the number of points chosen initially at every overlay intersection. Since this algorithm chooses increasing numbers of random points by doubling the quantity each time none of the existing points prove useful, we do not know exactly how many points were chosen in total at each stage of the localization process. In particular, as hypotheses are ruled out, the overlay intersection areas get larger, leading to a greater likelihood of increasing numbers of random points being picked repeatedly at each stage.

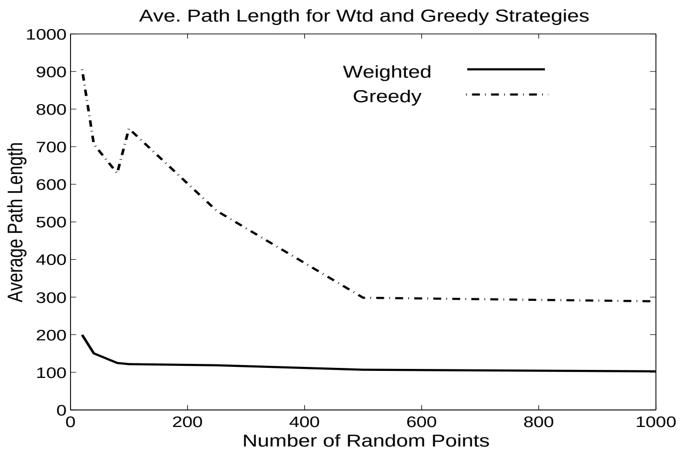


FIGURE 5.4. Performance of weighted versus greedy strategies based on 120 trials.

3. Weighted decision strategy versus greedy strategy

In the next set of experiments we compared the performance of our decision strategy (which we will refer to as the weighted strategy) versus its greedy version, where the robot is directed to move to the nearest location to its initial location which provides any non-zero information. This greedy alternative *approximates* the greedy strategy of [**DRW98**] but is not equivalent to it since we require an appropriate minimum number of points in order to adequately cover all the visibility cells in the overlay intersection region.

20 simulated office environments of average number of vertices approximately 400 were generated for this purpose. For each of the 20 environments, 3 initial robot locations were randomly selected. We then ran both weighted and greedy strategies with each of the 3 initial locations of each environment for a series of different quantities of random points, where each quantity was repeated twice to balance out any abnormal distributions. The total number of experimental trials was, therefore, $20 \times 3 \times 2 = 120$ trials [2].

Once again, we used the perfect sensor model (unlimited or infinite visibility). The initial set of random points picked for the very first overlay intersection was

[2]This set of trials will be used repeatedly for the experiments described in the remainder of this chapter.

kept the same for both strategies in order to ensure a fair comparison. Figure 5.4 shows the average path length for weighted and greedy strategies plotted against the number of random points. Even with very large numbers of random points (see 1000 points case), where we may reasonably assume an adequate sampling of most of the visibility cells in the overlay intersection region, on average the weighted strategy still outperforms its greedy counterpart.

Theoretically, the greedy technique of [**DRW98**] has been shown to have the best possible worst case bound on the distance travelled by the robot. In our weighted algorithm as well as its greedy variant, it is possible that a tiny visibility cell which might hold the key to an optimally short path was never sampled. This is a theoretical disadvantage but a practical advantage as the probability that the robot is directed to visit an arbitrarily tiny, and hence inaccessible, visibility cell is small.

4. Weighted decision strategy versus traditional heuristic

A set of experiments was performed where our weighted algorithm was compared with a traditional heuristic used in many localization papers, where the robot is directed to simply visit the closest point from its *current* location which provides any non-zero information. We used the same 20 environments with 3 initial locations per environment as described in Section 3, repeating each quantity of random points twice to balance out any abnormal distributions. The total number of trials was 120. Also we kept the same perfect sensor model as for the previous set of experiments. Figure 5.5 shows the average path length for our weighted algorithm and the traditional heuristic plotted against the number of random points. According to these results, the weighted algorithm produces shorter path lengths than the heuristic. Recall that the traditional heuristic can produce pathlengths of potentially exponential complexity as was demonstrated in Chapter 1 (see Figure 1.2).

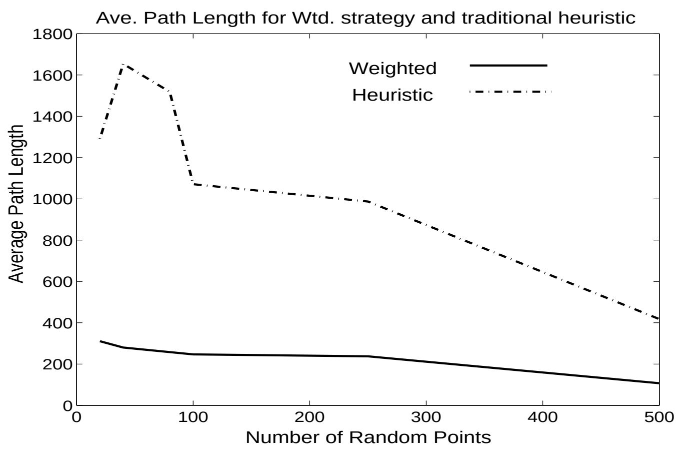


FIGURE 5.5. Performance of weighted strategy versus traditional heuristic based on 120 trials.

5. A suggestion for the quantity of random points

As hypotheses get eliminated, the successive overlay intersection areas get larger and larger. Selecting the same number of random points at each stage of the localization process, therefore, may not sample all the overlay intersection areas equally. Although the COL algorithm proceeds to choose increasing numbers of random points by doubling the quantity each time none of the existing ones prove useful, this still does not guarantee an adequate sampling of a given overlay intersection area. In this set of experiments we considered choosing the number of random points to be proportional to the size of the overlay intersection area which we measured in terms of the number of vertices. In choosing a size-proportionate number of random points we expect that useful points would be uncovered in the first few sets of points picked, without having to re-select points too many times. We performed the experiments over 120 trials as described in Section 3 with an unlimited visibility sensor model. Figure 5.6 shows the average path length plotted against the proportion of the size of the overlay intersection which specifies the number of random points selected. We seem to obtain fairly short path lengths when we choose the number of random points

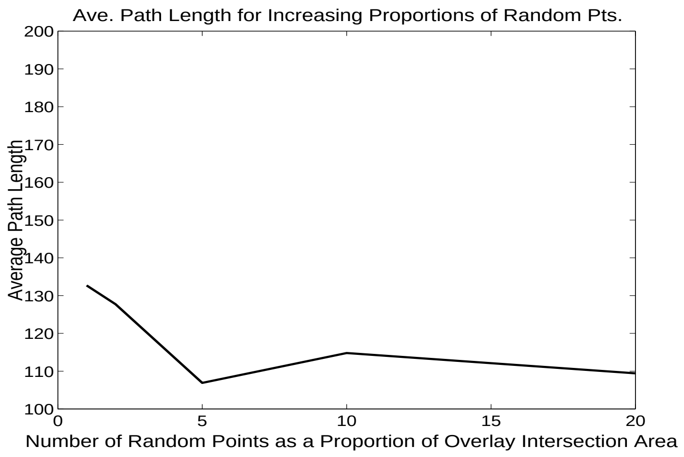


FIGURE 5.6. The number of random points are chosen to be proportional to the number of vertices of the overlay intersection region.

to be equal to the number of vertices of the overlay intersection region. The near-optimal value is attained when the number of random points is chosen to be 5 times the number of overlay intersection vertices.

6. Experiments with Limited Sensor Visibility

The following set of experiments measured the effect of limited sensor visibility on the localization path length for *two weighted* decision strategies. The first decision strategy is the strategy that we have described in Chapter 4 and have used in most of our experiments. Here, the information gain of each random point is weighted by its distance from the robot's initial location. As before, we will continue to refer to this strategy as the weighted strategy or the standard weighted strategy. The second decision strategy, which we will refer to as the *hybrid* strategy, operates by weighting the information gain at each random point by its distance from the robot's *current* location. We performed the experiments over 120 trials as described in Section 3. Note that the size of each grid cell in the simulated office environments we used is 100 pixel units.

Figure 5.7 depicts the average path length obtained for both the standard weighted strategy as well as the hybrid strategy with unlimited visibility. While both strategies

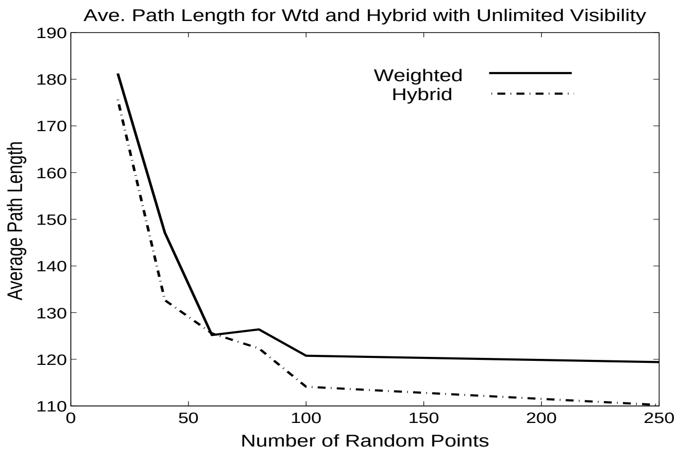


FIGURE 5.7. Weighted versus hybrid strategies with unlimited visibility. 120 trials were performed.

exhibit an improvement in performance as the number of random points is increased, the hybrid strategy produces significantly shorter path lengths.

Figures 5.8, 5.9, and 5.10 show the path lengths obtained for both strategies with visibility ranges 500, 200, and 90 respectively. For the limited sensor visibility value of 500, the path length for the standard weighted strategy gets shorter as the number of points is increased. Somewhat curious however, is the fact that for visibility values 200 and below the path length for the standard weighted strategy appears to be more or less the same even as the number of random points are increased. We will explain this behaviour in the next section.

Figures 5.11, 5.12, and 5.13 plot the average path length produced by the two strategies with decreasing visibility range and number of random points 40, 100, and 250 respectively. As expected, the results indicate that as the visibility gets poorer the path lengths for effective localization get longer. The poorer the robot's vision the closer it has to get to a distinguishing feature in order to eliminate hypotheses. As a result, it ends up travelling longer trajectories.

In terms of minimizing path length, the hybrid strategy soundly outperforms the standard weighted strategy for all the values of visibility and the different numbers of random points. In fact, the difference in path length between the two decision

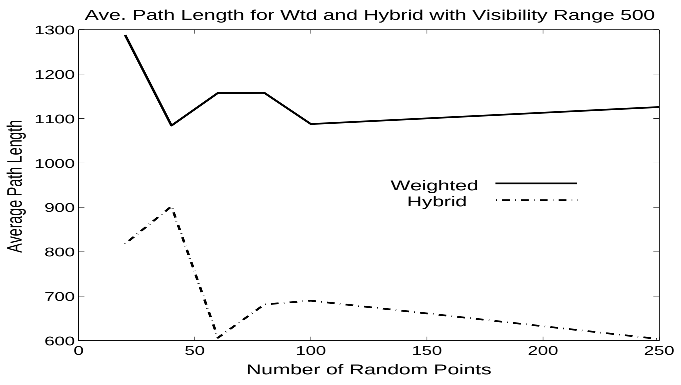


FIGURE 5.8. Weighted versus hybrid strategies with visibility range 500. 120 trials were performed.

strategies seems to get larger as the visibility gets poorer. Since the overlay intersection gets increasingly larger as hypotheses are ruled out, it follows that the robot moves progressively further away from its initial location each time it makes a decision to eliminate hypotheses. If the distance of a potential destination point is always measured from the robot's initial location, then it is possible that this could lead to extreme "zigzagging" of the robot's trajectory, depending on the shape of the environment and the placement of the initial location within the environment (see Figure 5.22). In addition, the distance travelled by the robot to observe a disambiguating feature gets larger as visibility decreases.

On the other hand, a weighted strategy that directs the robot to move to the most informative point nearest to its current location might avoid such a zigzagging effect on the robot's trajectory. Hence, as the visibility gets poorer the hybrid strategy produces path lengths that are shorter than those of the standard weighted strategy by a wider margin. Unlike the traditional heuristic which we demonstrated in Chapter 1 to yield potentially exponential path lengths, there is no reason we can come up with to show a similar theoretical worst-case bound for the hybrid strategy. If we were to use the very same example as in Figure 1.2 to evaluate the hybrid strategy, it would clearly direct the robot to travel to the signature room as its very first destination and thereby achieve localization.

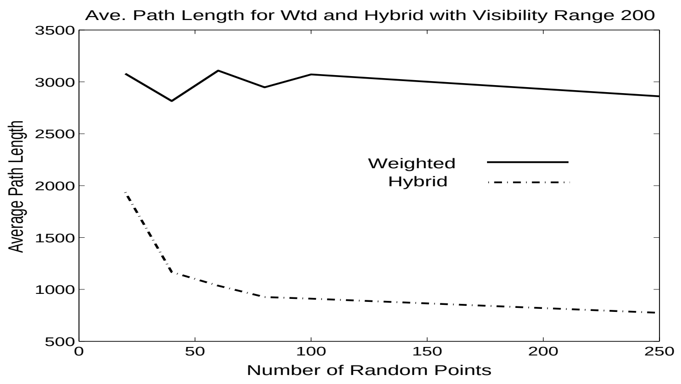


FIGURE 5.9. Weighted versus hybrid strategies with visibility range 200. 120 trials were performed.

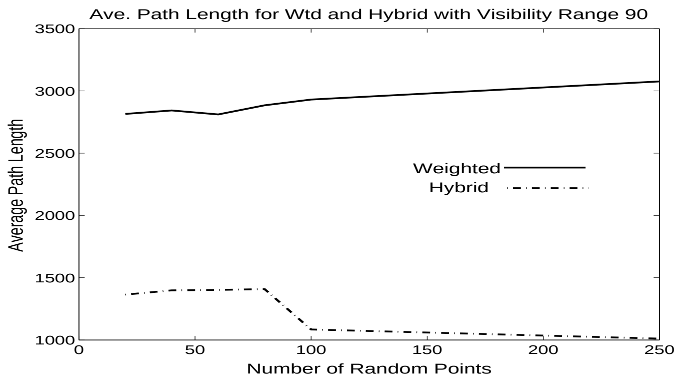


FIGURE 5.10. Weighted versus hybrid strategies with visibility range 90. 120 trials were performed.

7. Performance of Useful Region Localization Algorithm

Experiments were carried out to evaluate the performance of the URL algorithm with respect to different numbers of useful random points as well as various values of limited sensor visibility. We performed 120 trials with simulated office environments as described in Section 3. Figure 5.14 depicts the average path length obtained for number of useful random points ranging from 1 to 500, with unlimited visibility.

Since we are choosing points directly from the useful region, the algorithm is able to effectively localize the robot with just one random point, although the path length is understandably high in that case. Increasing the number of random points serves to reduce the path length. We observe that the values of pathlength for the different

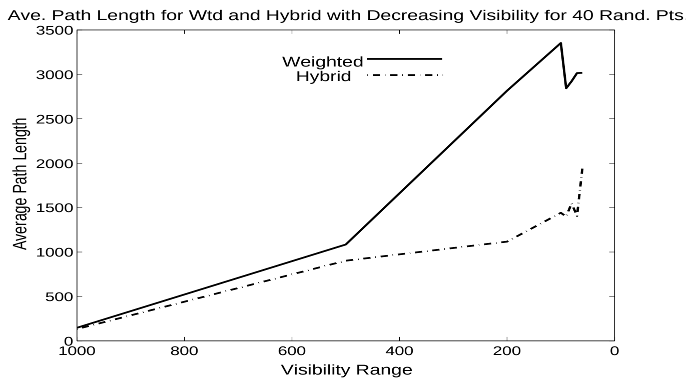


FIGURE 5.11. Performance of weighted versus hybrid strategies as visibility decreases, with 40 random points. The results are based on 120 trials.

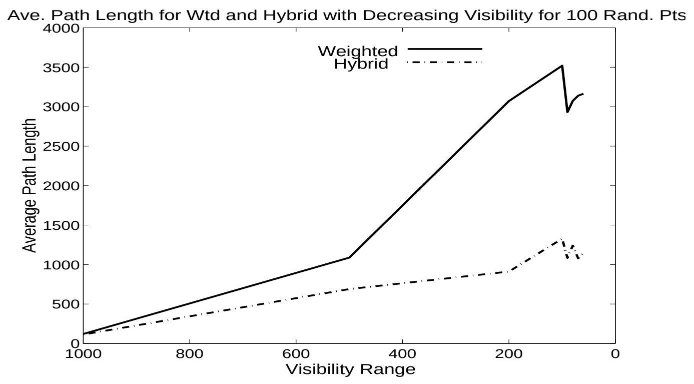


FIGURE 5.12. Performance of weighted versus hybrid strategies as visibility decreases, with 100 random points. The results are based on 120 trials.

numbers of random points in the case of unlimited visibility are somewhat higher than those obtained for the same number of points when running the COL algorithm. This is because the COL algorithm chooses increasing numbers of random points in iterations by doubling the quantity each time none of the existing points prove useful. As a result, we do not know exactly how many points were chosen in total at each stage of the localization process. In particular, as the overlay intersection areas get larger, it is more likely that a greater number of iterations would be required before any useful points appear. Since we double the number of points picked at each successive iteration, useful points, when they finally appear, they may do so in large quantities. On the other hand, the URL algorithm adheres to exactly the same number of points initially specified, regardless of the size of the subsequent overlay intersection regions

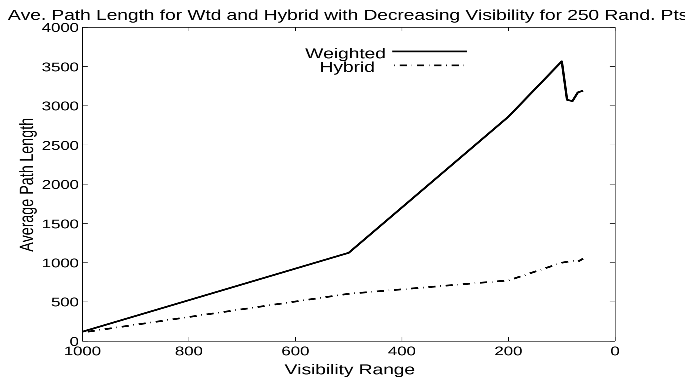


FIGURE 5.13. Performance of weighted versus hybrid strategies as visibility decreases, with 250 random points. The results are based on 120 trials.

(and hence the size of the subsequent useful regions). Given that these regions get larger as hypotheses are eliminated, this leads to potential undersampling.

Figure 5.15 plots the variation of average path length with increasing numbers of useful random points, for different visibility ranges. Poorer visibility produces longer path lengths. And unlike the COL algorithm, with any visibility range, increasing the number of random points leads to shorter path lengths. Figure 5.16 shows the effect of deteriorating visibility on the average path length for 500 useful points as well as for 1 useful point. Figure 5.17 compares the performance of the COL and URL algorithms with decreasing visibility range, for 250 random points.

When we examine the results for limited sensor visibility, we find that upwards of 80 random points, the URL algorithm produces path lengths that are at least equal to those produced by the COL algorithm, and at times much shorter. For example, in Figure 5.17 the average path length obtained for the URL algorithm starts out to be longer than that of the COL algorithm but very quickly becomes much shorter than that of the COL algorithm for lower visibility range values. As visibility gets poorer, the size of the useful region gets smaller since the robot is compelled to approach the disambiguating landmarks at very close proximity in order to see them. In such circumstances, choosing points from the entire overlay intersection region might result in very few of them actually coming from the useful region. The path lengths generated by the COL algorithm for the entire range of random point

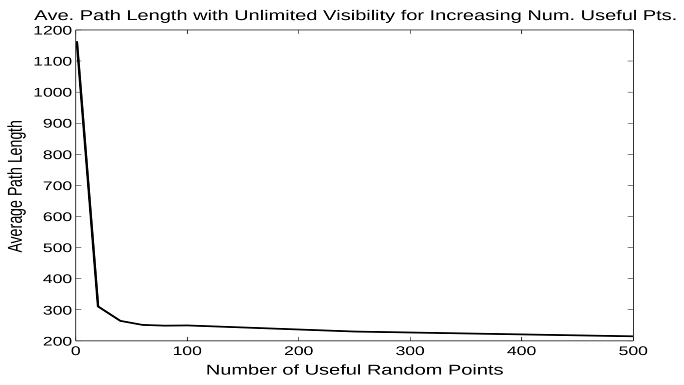


FIGURE 5.14. Performance of URL algorithm with unlimited visibility. The results are based on 120 trials.

quantities for limited visibility values 200 and below appear to be more or less the same. The COL algorithm functions by selecting increasing numbers of points in iterations until some happen to be useful. Since the useful region itself gets smaller as visibility is reduced, the number of useful points uncovered by the COL algorithm remains approximately the same regardless of whether it starts off with a relatively small number of points which are augmented in iterations in order to yield useful ones, or it starts off with a relatively large number of points which might not require much reinforcement in order to uncover some that are useful. Incrementing the total number of points chosen in the overlay intersection will tend to increase the number that prove useful but the relative increase is not sufficient to really make a difference to the path length.

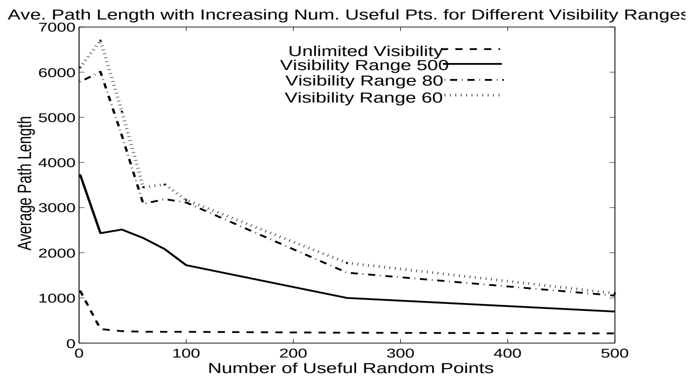


FIGURE 5.15. Performance of URL algorithm with different visibility ranges. The results are based on 120 trials.

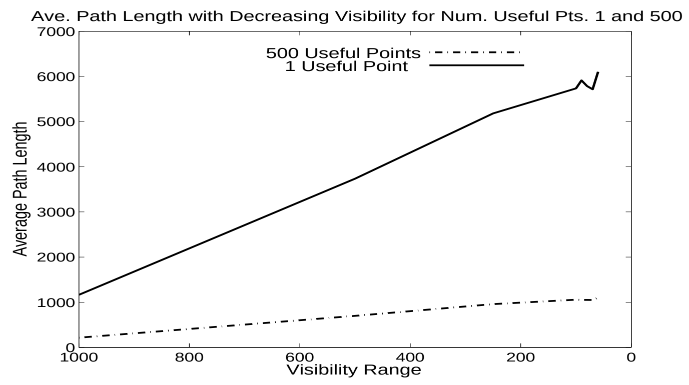


FIGURE 5.16. Performance of URL algorithm with only 1 useful random point versus 500 useful random points. The results are based on 120 trials.

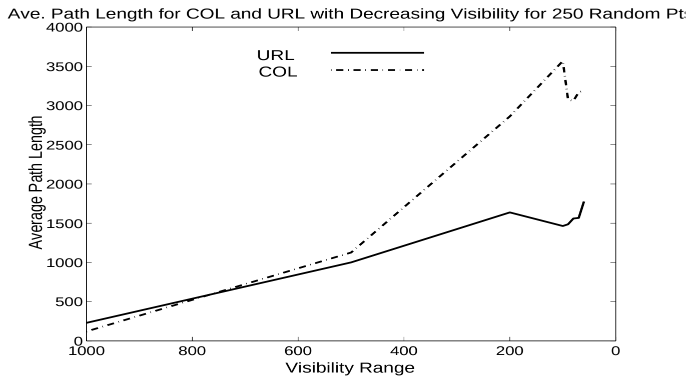


FIGURE 5.17. Performance of URL algorithm versus COL algorithm as visibility decreases, with 250 random points. The results are based on 120 trials.

8. Discussion

Since global localization in the context of ambiguous environments intrinsically entails travel, the decision strategy influencing the robot's motion is crucial for efficient localization. Several authors [**FBT98a, DN97**] have reported that naive strategies for robot motion such as wall-following and random drifting can fail to localize the robot. In addition, we established in Chapter 1 that a traditional heuristic can produce pathlengths of potentially exponential complexity (see Figure 1.2). Moreover, our experimental results have shown that the weighted strategy performs better than the traditional heuristic. Although the greedy strategy of [**DRW98**] has been proven to have the best possible worst case bound for the distance travelled by the robot, experimental results showed that our weighted strategy produces shorter pathlengths on average. Finally, the hybrid decision strategy that we evaluated in Section 6 outperformed the standard weighted strategy [3].

In general, while we believe that some form of weighted choice must be made, the actual weighting formula can vary. It is important to note that the choice of decision strategy is not critical to our technique. On the contrary, the flexibility of our algorithms means that different decision strategies, depending on factors such as the type of environment, restrictions placed on the robot, and any other objectives that the robot might simultaneously be trying to fulfill, can be accommodated. A probabilistic decision strategy proposed by [**BSN99**] assumes that the hypothetical locations possess different probabilities. In realistic scenarios, such a practical decision strategy might be effective as clearly, we would rather move to a probe position where a few highly probable hypotheses may be eliminated as opposed to a position where a large number of hypotheses of low probability may be eliminated.

The main drawback of our work is that it is set in a rather idealized context. We rely on a perfect sensor, a perfect map, and a point robot that makes error-free motions. Consequently, our technique risks being too sensitive to any sort of noise that may occur, especially in the sensed data and the map of the environment. However, we do not suffer much from the error-free motion model of the robot due to the randomized approach we have taken. Random sampling of the environment means that the probe destinations that our algorithm might choose will have a higher likelihood of coming from large, navigable visibility cells rather than small ones. Therefore, even if the robot does not position itself precisely at the point where it intended to, it is nonetheless likely to remain within the same visibility cell. Besides, a real robot with non-zero mass would be able to manoeuvre itself better in the large spaces where random points are more likely to be selected from. The randomized

[3]Recall that the hybrid strategy weights the information gain per random point with the distance of that point from the robot's current location whereas the standard weighted strategy weights the information gain with that point's distance from the robot's initial location.

approach also enables our technique to avoid horrendous worst-case scenarios in terms of path length (as depicted in Figure 1.2).

Relying on a perfect map of the environment may lead the robot to spot too few ambiguities. A small glitch in the map may make two hypothetical locations appear different when, in reality, they are identical. In order to effectively recognize and eliminate hypothetical locations, undeterred by any noise that may be present in the map, we might consider assigning probablities to different hypotheses as discussed above. Fairly accurate polygonal layouts might be available for certain indoor environments but this restricts us to only such indoor environments. Karch *et al.* [**KW99, KNW97, KNW98**] examine polygon distances for modeling the similarity between a range scan and the preprocessed visibility information needed for localization in an attempt to adapt the theoretical scheme to more realistic scenarios. Gonzalez-Banos and Latombe [**GBL02, GBL01b, GBML$^+$00**] describe methods for polyline generation from sensor data and model alignment. They argue that due to range-finding technology being now more precise and reliable, polygonal representations can be considered, especially given the added advantage of polygons possessing geometric properties that can be efficiently computed. The perception-based localization scheme of [**DN97**] requires the robot to first build its own map before attempting localization. This approach might have the advantage of imposing consistency between the map and the robot's perceptions of its environment during localization.

Special sampling strategies for narrow passages as described in [**HKL$^+$98**] might be required in order to adequately cover regions of the overlay intersection that are relatively small compared to the entire overlay intersection area. This issue is relevant only for the COL algorithm, since in the URL algorithm we sample only within the computed useful region. It is exactly this narrow passage effect that occurred when we evaluated the performance of the COL algorithm with low visibility ranges. When the visibility is poor, the size of the useful regions relative to the overlay intersection area becomes very small, and therefore hard to sample. Hence we observed that the length of the localization trajectory did not reduce much as the number of random points was increased.

We have not established any correlation between the number of hypotheses generated and the path length, keeping visibility range and environment size constant. A larger number of hypotheses may give rise to shorter path lengths than a smaller number of hypotheses. As visibility gets poorer, the number of hypotheses increases as does the path length. However, the increased length of the localization trajectory can be attributed to the robot's "short-sightedness" which requires it to move much closer to distinguishing features in order to be able to see them than otherwise.

9. Visualization of Common Overlay Localization Algorithm

In this section we present visual simulation results illustrating the localization trajectory taken by the robot in a multitude of environments. The shaded area in these figures represents the overlay intersection area. The square black points labelled $H0, H1, ..., Hk$ indicate the different hypothetical locations. The small round dots scattered across the shaded region represent the random points. The localization path taken by the robot goes through the most useful of these random points.

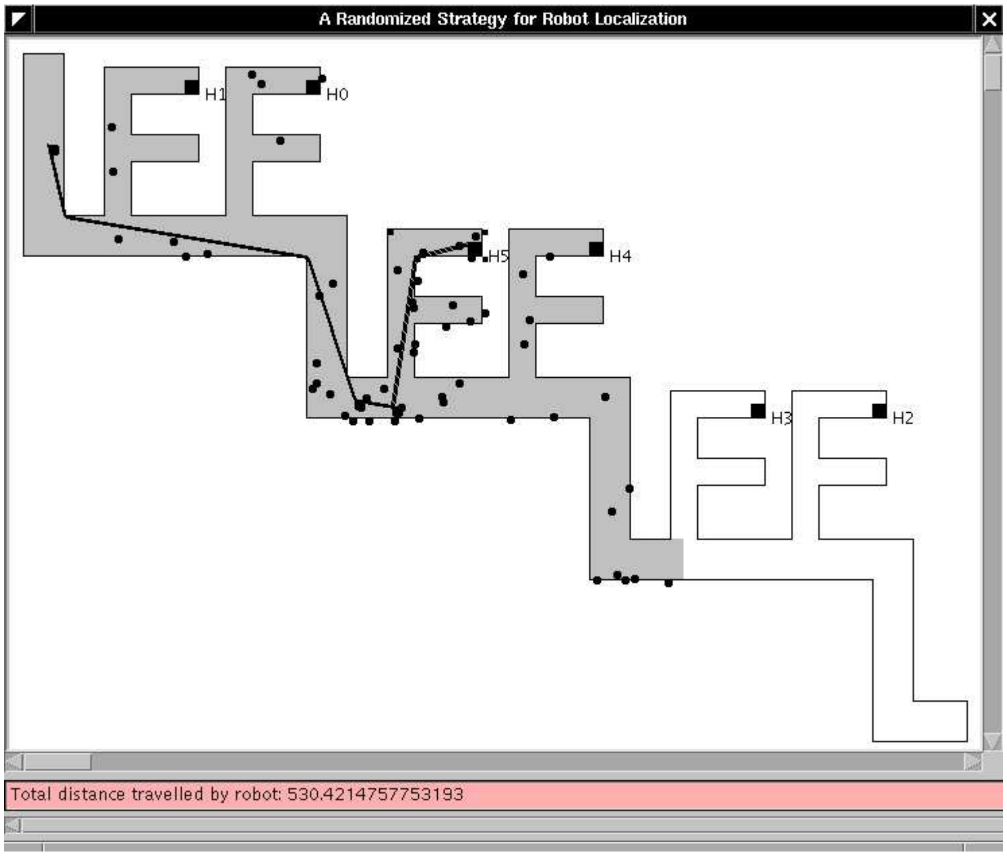
Figure 5.18 depicts the localization path taken by a robot with unlimited visibility for a staircase-like environment. Figures 5.19 and 5.20 show simulated office environments of the type modeled after mazes, where the size of each grid cell is 100 pixel units. Localization is achieved with unlimited visibility.

Figures 5.23, 5.21, 5.22, and 5.24 illustrate various simulated office environments of the type modeled after mazes, where the size of each grid cell has been reduced to 50 pixel units instead of 100 as before [4]. Each environment contains approximately 400 vertices on average. The visibility range was reduced to 30 pixel units. Figures 5.23, 5.22, and 5.24 show localization trajectories obtained with the standard weighted decision strategy. Figure 5.21, on the other hand, depicts the same environment and initial location as Figure 5.22 but with a trajectory obtained using the

[4]The grid cells used to generate these environments had to be made smaller in order to fit an entire environment on a page.

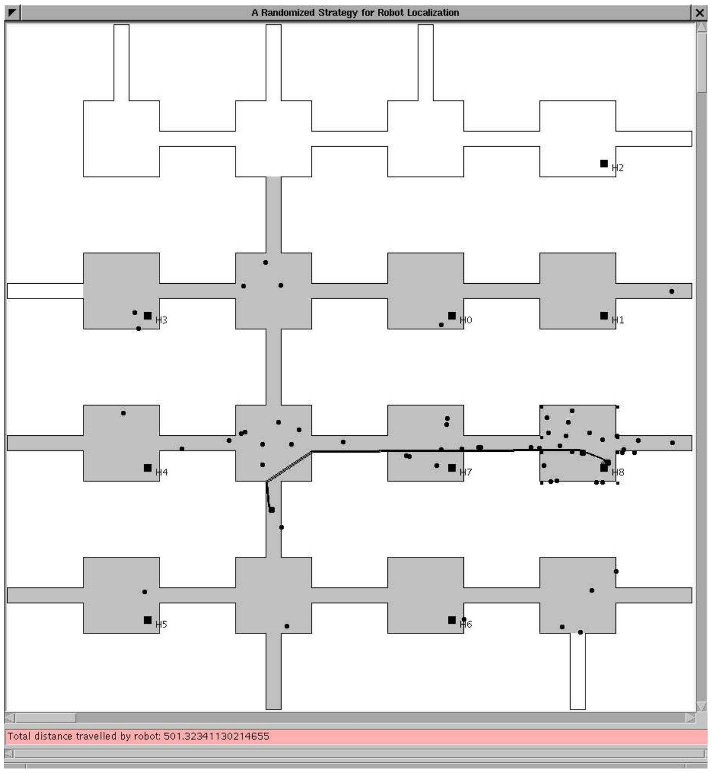
hybrid decision strategy. The initial location is the centre "room cell" in the top row of room cells. Observe that in Figure 5.21 the robot proceeds to successfully localize itself by moving in one direction from its initial location. The total distance traveled by the robot amounts to 614.74 units. In contrast, Figure 5.22 shows the robot moving in a "zigzagging" fashion from the initial location, resulting in a total pathlength of 3429.59 units. This example illustrates clearly the types of situations where measuring the weighted distance of a potential probe destination from the robot's current location produces significantly shorter localization trajectories, while simultaneously avoiding the pitfalls faced by the traditional heuristic.

Figures 5.25 and 5.26 depict the localization path taken by the robot in two signature step environments with visibility range 50. In both cases, the robot explores only the distinctive structure on the step it is situated on in order to estimate its true location.



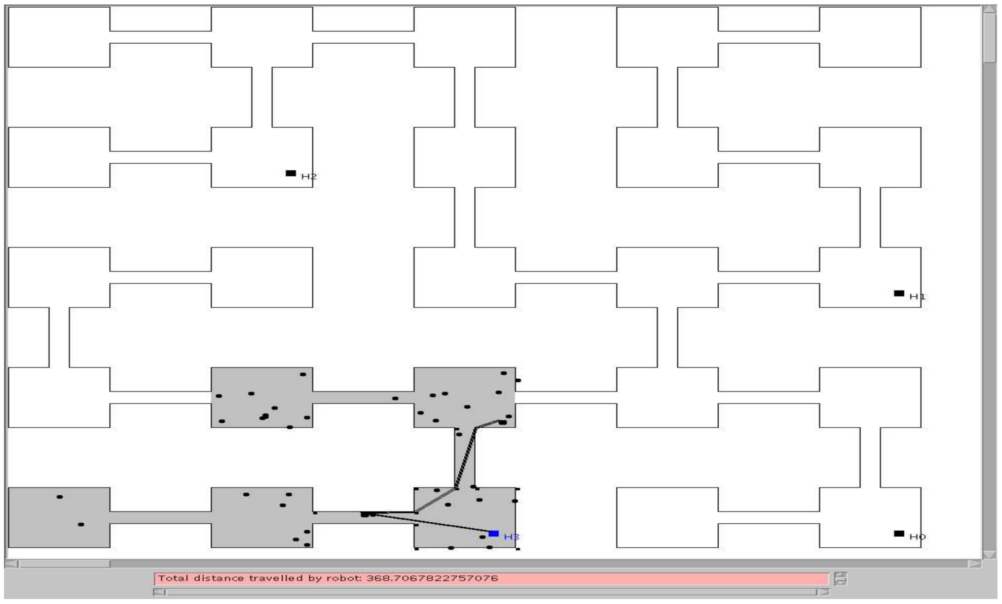
Total distance travelled by robot: 530.4214757753193

FIGURE 5.18. Localization trajectory in staircase environment with unlimited visibility.



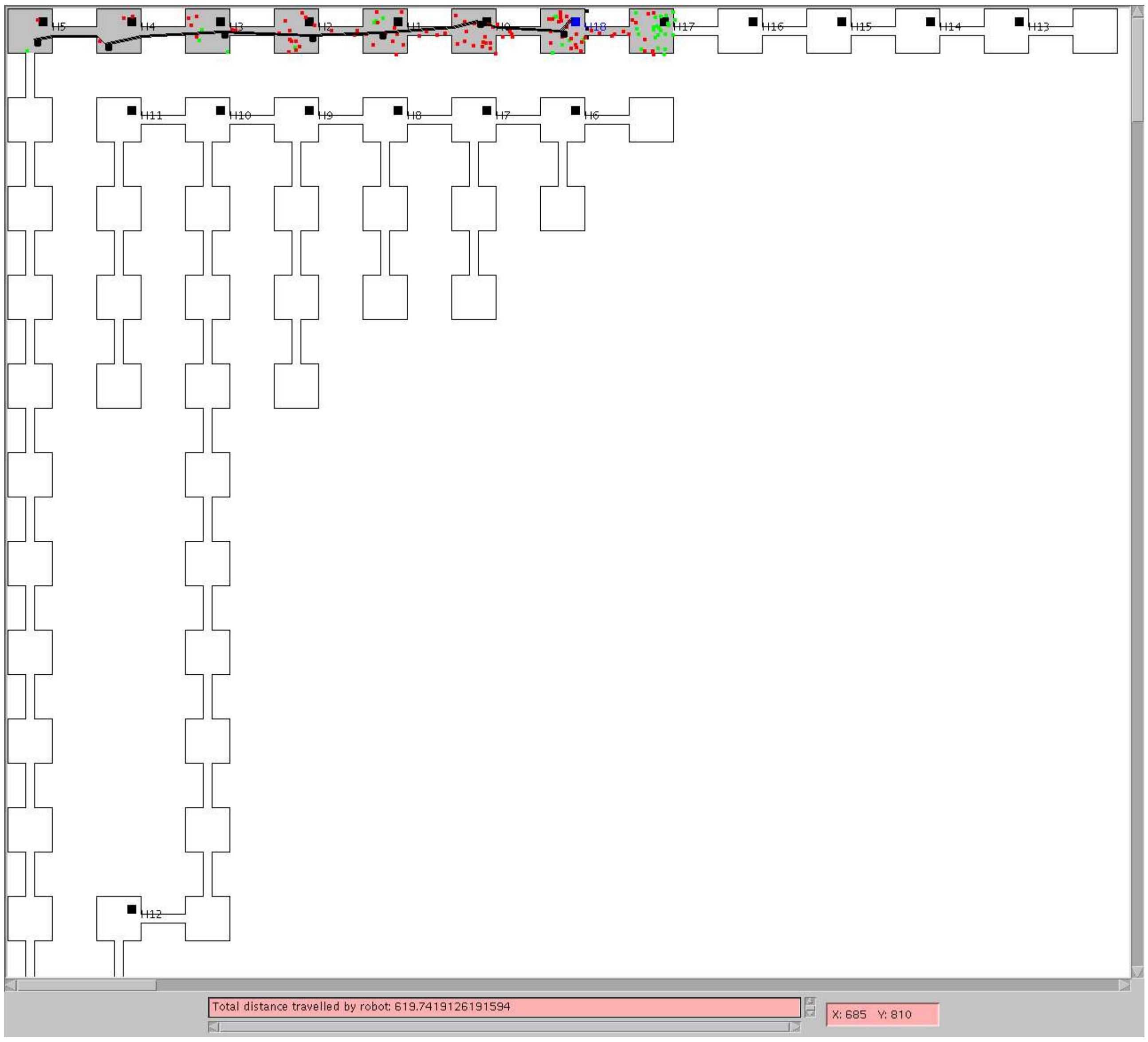
Total distance travelled by robot: 501.32341130214655

FIGURE 5.19. Localization trajectory in simulated office environment with unlimited visibility.



Total distance travelled by robot: 368.7067822757076

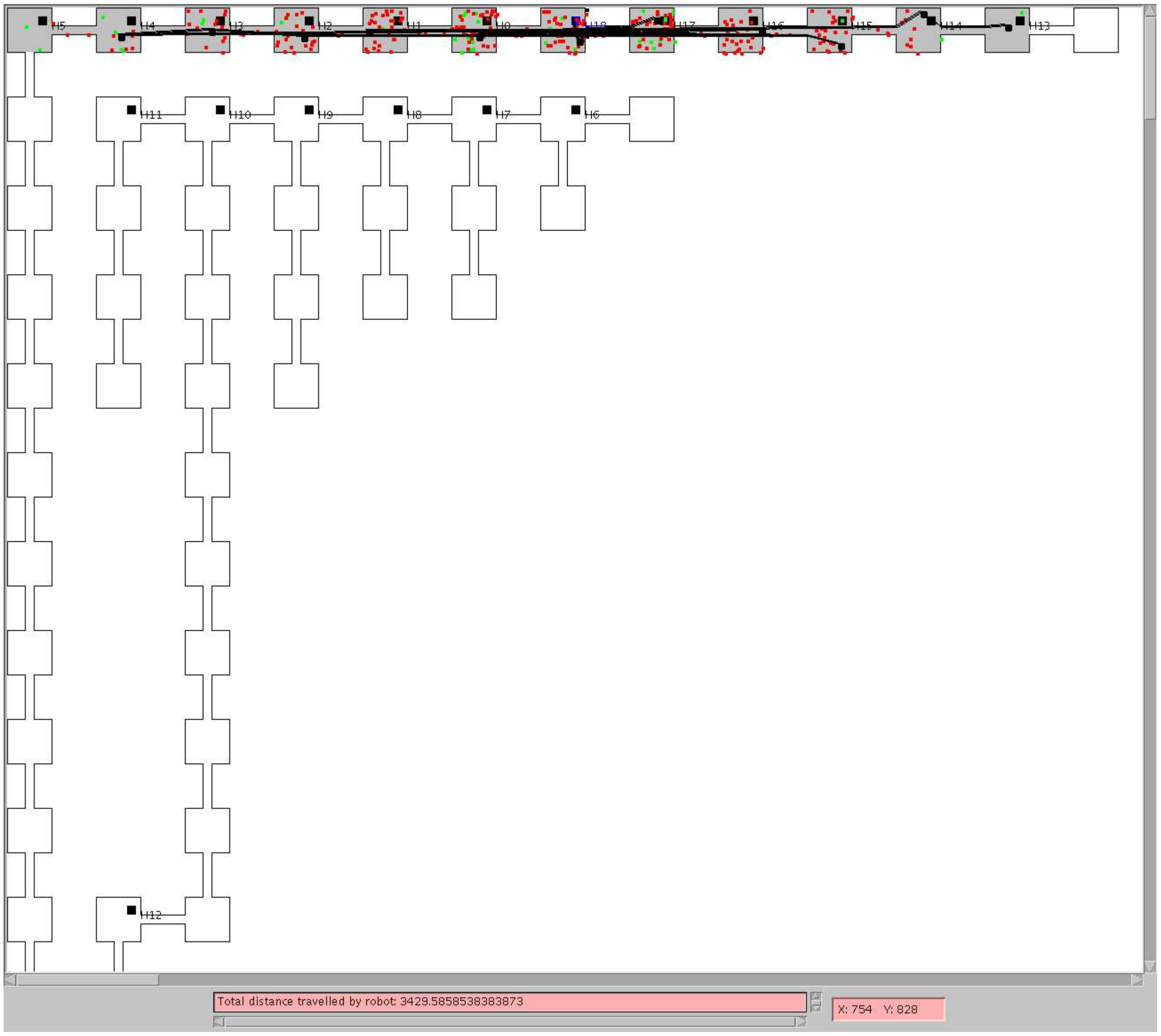
FIGURE 5.20. Localization trajectory in simulated office environment with unlimited visibility.



Total distance travelled by robot: 619.7419126191594

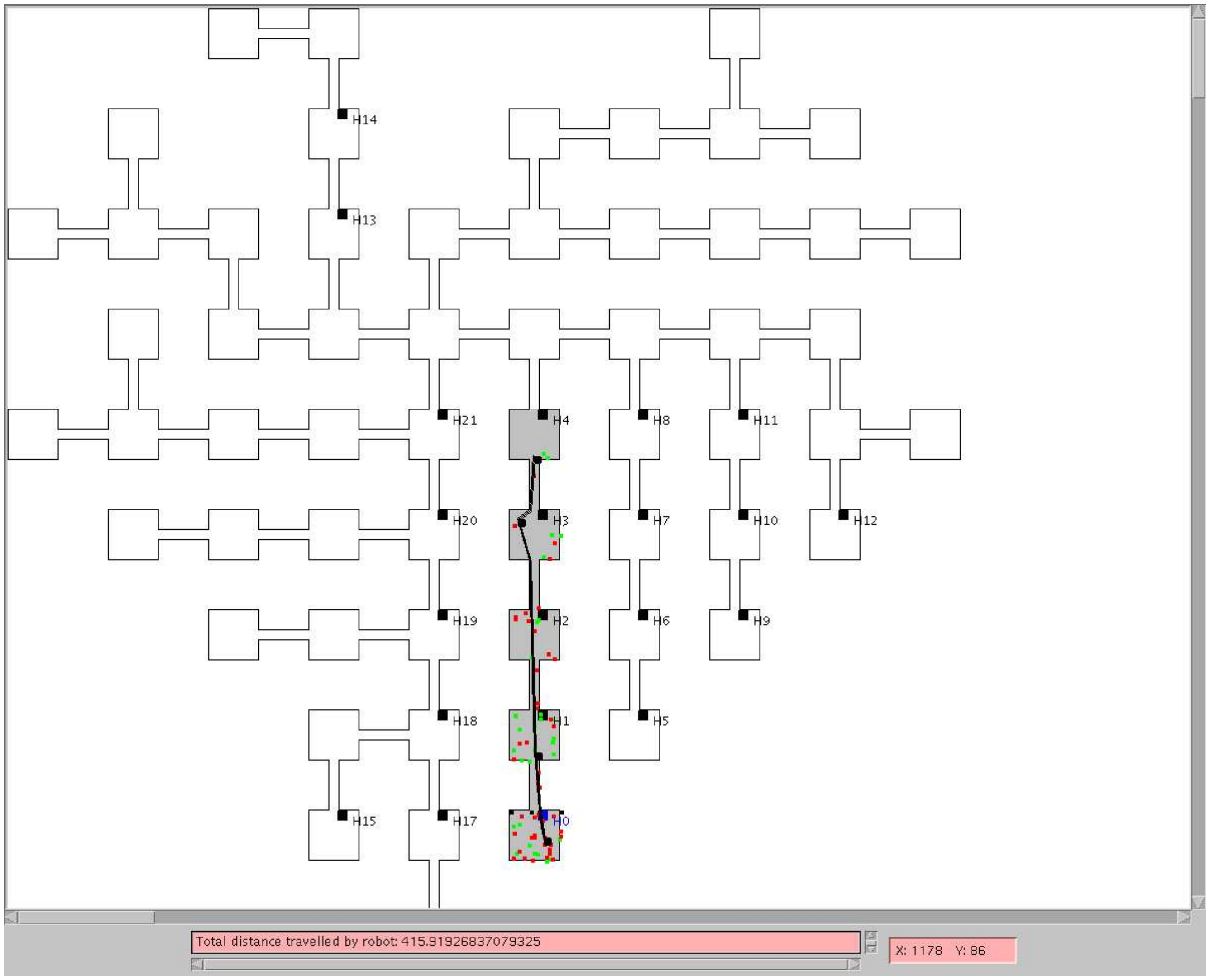
X: 685 Y: 810

FIGURE 5.21. Localization trajectory in simulated office environment with limited visibility. The hybrid decision strategy was used which directs the robot to weight the information gain at a potential probe destination with its distance from the robot's *current* location.



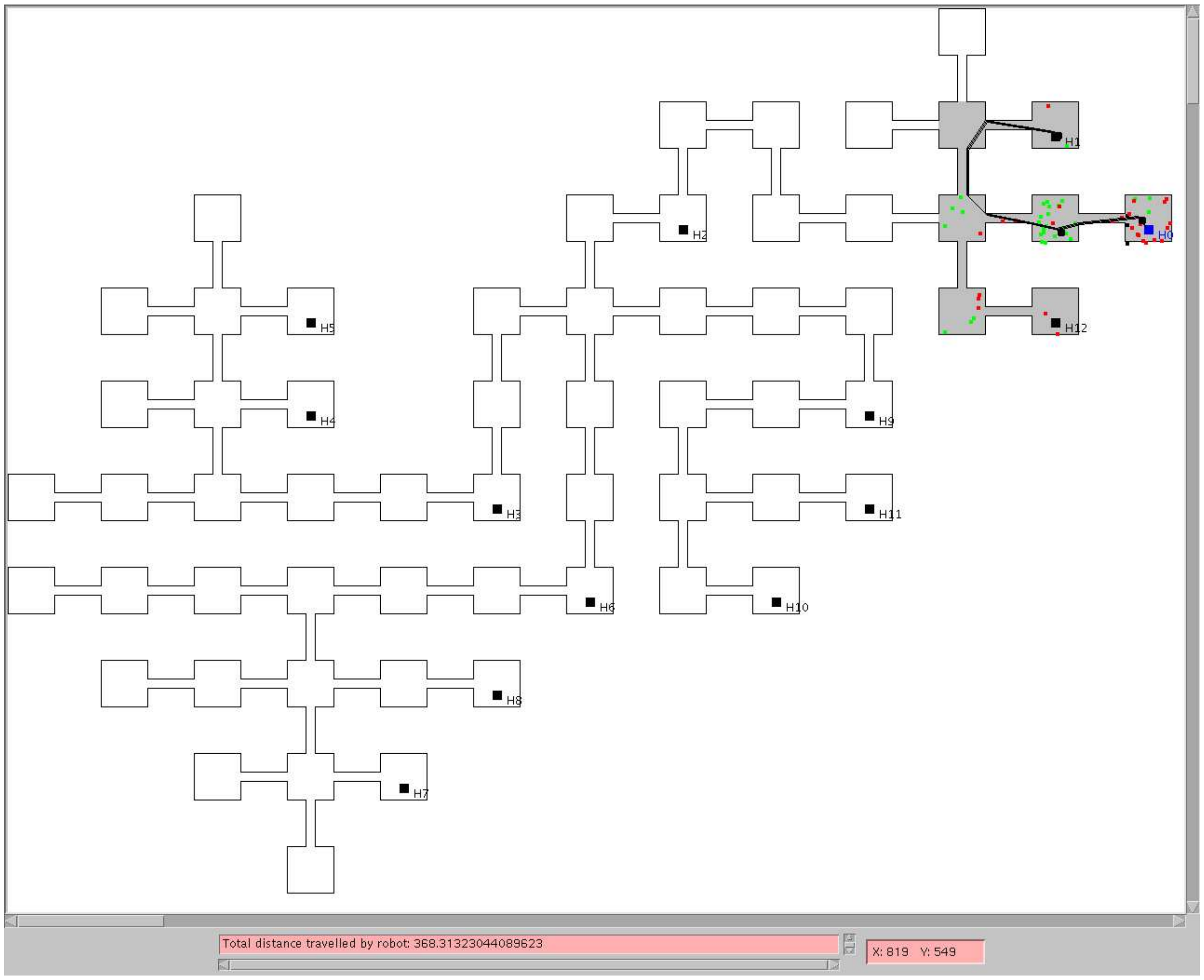
```
Total distance travelled by robot: 3429.5858538383873
X: 754   Y: 828
```

FIGURE 5.22. Localization trajectory in simulated office environment with limited visibility. The standard weighted decision strategy was used which directs the robot to weight the information gain at a potential probe destination with its distance from the robot's *initial* location. The robot travels in a zigzagging fashion, resulting in a long trajectory.



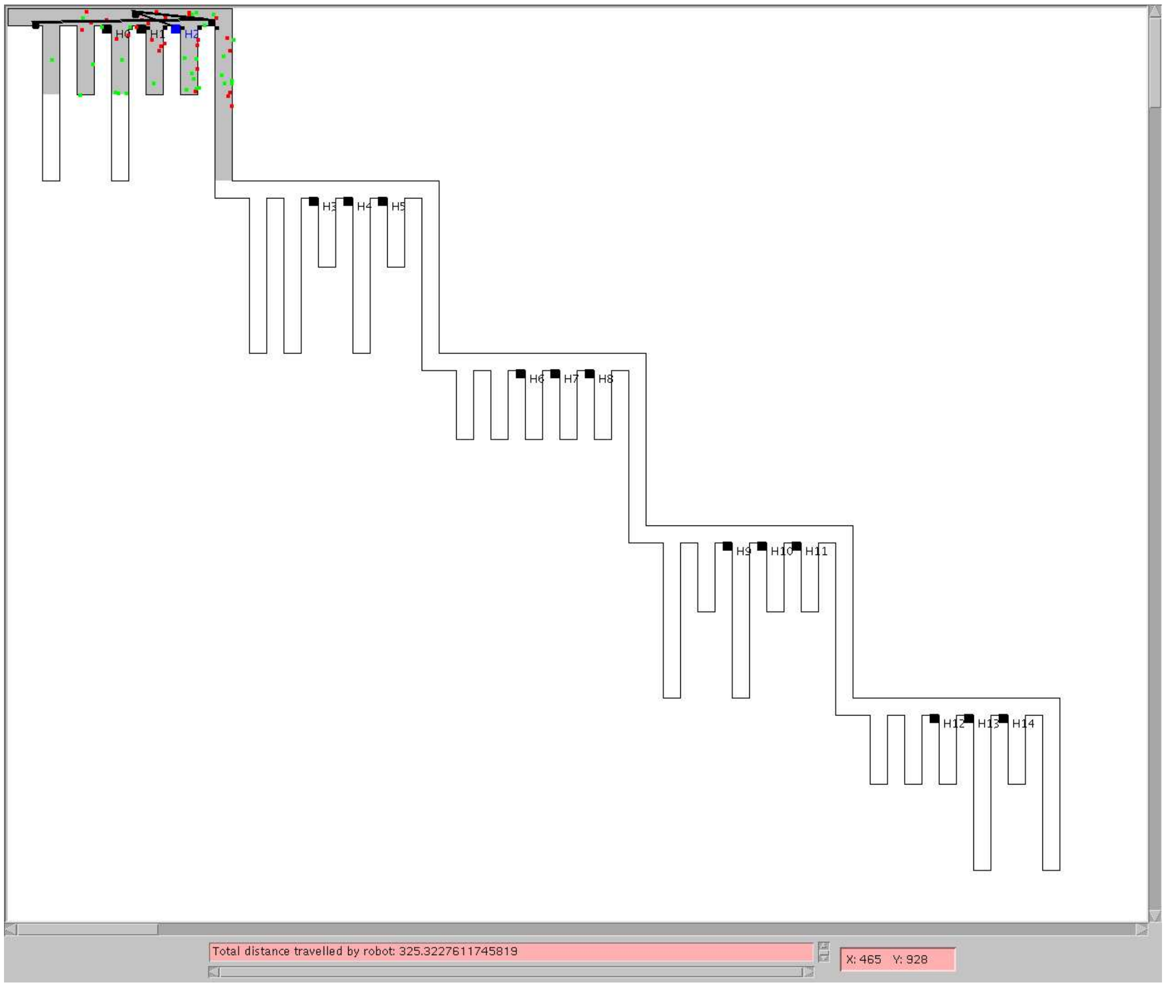
Total distance travelled by robot: 415.91926837079325

FIGURE 5.23. Localization trajectory in simulated office environment with limited visibility.



```
Total distance travelled by robot: 368.31323044089623
X: 819   Y: 549
```

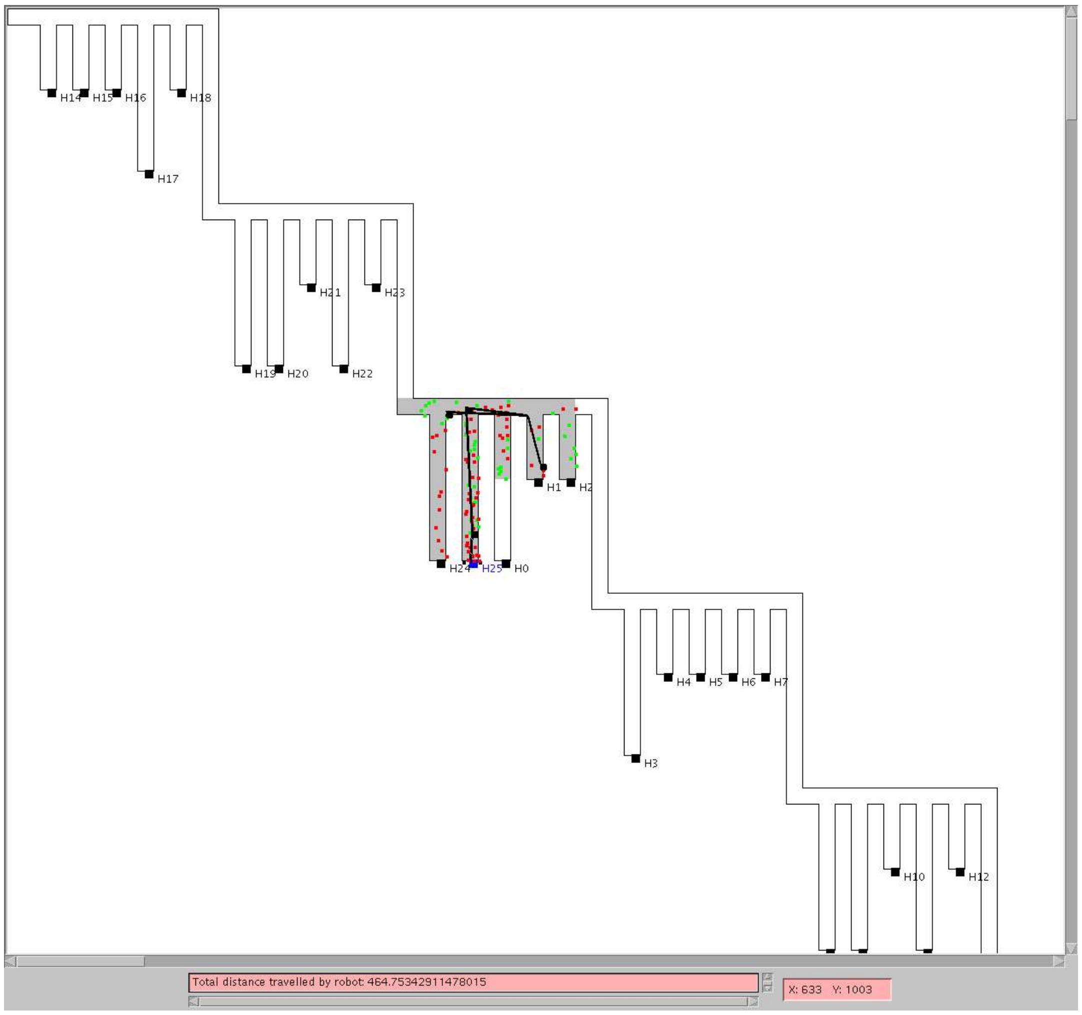
FIGURE 5.24. Localization trajectory in simulated office environment with limited visibility.



Total distance travelled by robot: 325.3227611745819

X: 465 Y: 928

FIGURE 5.25. Localization trajectory in signature step environment with limited visibility. The robot explores only the distinctive structure on the step it is situated on in order to estimate its true location.



Total distance travelled by robot: 464.75342911478015

X: 633 Y: 1003

FIGURE 5.26. Localization trajectory in signature step environment with limited visibility. The robot explores only the distinctive structure on the step it is situated on in order to estimate its true location.

CHAPTER 6

Conclusion and Future Work

1. Conclusion

In this thesis, we have presented two randomized approximation algorithms that efficiently solve the problem of minimum distance localization. Ours is the first work to address the problem of path optimization in global localization theoretically as well as practically. Our techniques are able to effectively localize a robot in large-scale environments with high degrees of ambiguity and limited sensor visibility. The algorithms we have developed are efficient to compute and yet also efficient in their performance.

Global localization in the context of ambiguous environments intrinsically entails combining sensor measurements from multiple vantage points along some trajectory. Therefore an intelligent motion strategy is imperative to achieve unambiguous localization. Theoretically, while the greedy decision strategy of [**DRW98**] has been proven to have the best possible worst case bound on the distance traveled by the robot, our weighted strategies produce shorter path lengths on average. Existing deterministic techniques may direct the robot to visit arbitrarily small visibility cells. In practice, reaching such cells may be infeasible and error-prone for a real robot. The algorithms we have proposed are based on visiting a series of random sample points from which distinguishing landmarks can be seen. Thus, it is certainly possible that a tiny visibility cell which might hold the key to an optimally short path is never sampled. This is a theoretical disadvantage but a practical advantage.

Simple heuristic strategies have been shown to exhibit strikingly poor performance with respect to the length of the localization trajectory. In addition, naive motion strategies such as wall-following and random wandering have been reported to fail to localize the robot. The weighted decision strategies that we employ are able to effectively localize the robot with competitive path lengths, while at the same time avoiding worst-case pitfalls. In general, although we believe that some form of weighted decision strategy must be used, the actual weighting formula can vary, depending on factors such as the type of environment, restrictions placed on the robot, and any other objectives that the robot might simultaneously be trying to fulfill. Our technique is flexible and can therefore accommodate different decision strategies.

We generated a variety of self-similar environments in order to validate and explore the performance of our localization algorithms. Experimental results show that although the performance of our algorithms improves with the number of random sample points used, the incremental improvement decreases as the number of samples increases, so that a fairly limited number of samples typically is sufficient. In the case of the URL algorithm, where we directly compute the portion of the overlay intersection region where any point chosen is guaranteed to yield new information, we require as little as 1 random point in order to unambiguously localize the robot. A greater number of random points only serves to minimize the path length.

As the sensor visibility range decreases, the robot is required to travel increasingly longer distances in order to achieve localization. Poor vision means that the robot has to move much closer to distinguishing features in order to be able to see them than otherwise. Experimental results indicate that for any fixed visibility range, the path length produced by the URL algorithm becomes shorter as the number of random points is increased. In contrast, at low visibility range values, the path length produced by the COL algorithm improves only marginally as the number of random points is increased. Since the regions of the overlay intersection that are likely to provide new information get smaller as visibility gets poorer, the number of useful points uncovered by the COL algorithm does not increase sufficiently to really make a difference to the path length. In such situations the URL algorithm seems to clearly be the better choice.

2. Future Work

The flexible nature of our technique suggests natural extensions to more realistic contexts. It would be interesting to adapt our algorithms to handle noisy sensor data and a noisy map. Hypotheses would then have to be generated in a probabilistic fashion and the decision strategy might have to be modified to reflect the likelihood estimates for the different hypotheses.

Polygonal environments containing holes might be accommodated with our existing technique. It would be worthwhile to consider what modifications are required in order to implement minimum distance localization in environments with obstacles.

Due to time limitations, we were not able to perform as many experiments as we would have liked. In particular, we would like to carry out experiments with different types of environments and in larger quantities. In addition the performance of the URL algorithm could be more thoroughly explored.

Some open questions raised are: Can we express the path length as a function of the number of concave vertices in the environment? What would such a function look like?

One aspect of this work that deserves more attention is the average case behaviour of our randomized algorithms and their expected time complexity.

Ultimately, we would like to implement our technique on a real mobile robot.

REFERENCES

[ACS02] K. O. Arras, J. A. Castellanos, and R. Siegwart, *Feature-based multi-hypothesis localization and tracking for mobile robots using geometric constraints*, In Proc. IEEE International Conference on Robotics and Automation (Washington DC, USA), 2002, pp. 1371–1377.

[AI90] D. Avis and H. Imai, *Locating a robot with angle measurements*, J. Symbolic Computation **10** (1990), 311–326.

[AT81] D. Avis and G. Toussaint, *An optimal algorithm for determining the visibility of a polygon from an edge*, IEEE Transactions on Computers **C-30, No. 12** (1981), 910–914.

[BD00] R.G. Brown and B.R. Donald, *Mobile robot self-localization without explicit landmarks*, Algorithmica **26** (2000), no. 3/4, 515–559.

[Ber70] E. R. Berlekamp, *Factoring polynomials over large finite fields*, Math. Comput. **24** (1970), 713–735.

[BFHS96] W. Burgard, D. Fox, D. Hennig, and T. Schmidt, *Estimating the absolute position of a mobile robot using position probability grids*, AAAI/IAAI, vol. 2, 1996, pp. 896–901.

[BG97] M. Betke and L. Gurvits, *Mobile robot localization using landmarks*, IEEE Trans. on Robotics and Automation **13** (1997), no. 2, 251–263.

[BGL$^+$97] J. Bose, L. Guibas, A. Lubiw, M. Overmars, D. Souvaine, and J. Urrutia, *The floodlight illumination problem*, Int. Journal in Computational Geometry **7** (1997), 153–163.

[BSN99] M. Buck, D. Schfer, and H. Noltemeier, *Practical strategies for hypotheses elimination on the self-localization problem*, 1999.

[CE92] B. Chazelle and H. Edelsbrunner, *An optimal algorithm for intersecting line segments in the plane*, J. Assoc. Comput. **39** (1992), 1–54.

[Chv75] V. Chvatal, *A combinatorial theorem in plane geometry*, J. Combinatorial Theory Series B **18** (1975), 39–41.

[CKK96] A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien, *Acting under uncertainty: Discrete bayesian models for mobile robot navigation*, In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 1996.

[CN88] W. Chin and S. Ntafos, *Optimum watchman routes*, Information Processing Letters **28** (1988), 39–44.

[DFBT99] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, *Monte carlo localization for mobile robots*, In Proc. IEEE Conference on Robotics and Automation, 1999.

[DJ00] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*, Cambridge University Press, 2000.

[DLOM02] E. D. Demaine, A. Lpez-Ortiz, and J. I. Munro, *Robot localization without depth perception*, In Proc. 8th Scandinavian Workshop on Algorithm Theory, Lecture Notes in Computer Science (Turku, Finland), vol. 2386, 2002, pp. 249–259.

[DN97] T. Duckett and U. Nehmzow, *Experiments in evidence-based localisation for a mobile robot*, In Proc. AISB-97 Workshop on Spatial Reasoning in Mobile Robots and Animals, 1997.

[DRW98] G. Dudek, K. Romanik, and S. Whitesides, *Localizing a robot with minimum travel*, SIAM J. Computing **27** (1998), no. 2, 583–604.

[Eng94] S. Engelson, *Passive map learning and visual place recognition*, Ph.D. thesis, Computer Science Department, Yale University, 1994.

[FBDT99] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, *Monte carlo localization: Efficient position estimation for mobile robots*, AAAI/IAAI, 1999, pp. 343–349.

[FBT98a] D. Fox, W. Burgard, and S. Thrun, *Active markov localization for mobile robots*, Robotics and Autonomous Systems **25** (1998), 195–207.

[FBT98b] ______, *Markov localization for reliable robot navigation and people detection*, Sensor Based Intelligent Robots, 1998, pp. 1–20.

[FTBD01] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, *Particle filters for mobile robot localization*, Sequential Monte Carlo Methods in Practice (New York) (A. Doucet, N. de Freitas, and N. Gordon, eds.), Springer, 2001.

[GA81] H. El Gindy and D. Avis, *A linear algorithm for computing the visibility polygon from a point*, Journal of Algorithms **2** (1981), 186–197.

[GBL98] H.H. Gonzalez-Banos and J.C. Latombe, *Planning robot motions for range-image acquisition and automatic 3d model construction*, In Proc. AAAI Fall Symposium Series, 1998.

[GBL01a] ______, *A randomized art-gallery algorithm for sensor placement*, In Proc. 17th ACM Symposium on Computational Geometry, 2001, pp. 232–240.

[GBL01b] ______, *Robot navigation for automatic model construction using safe regions*, Experimental Robotics VII. Lecture Notes in Control and Information Sciences (In Proc. of Int. Symposium on Experimental Robotics 2001) **271** (2001), 405–415.

[GBL02] ______, *Navigation strategies for exploring indoor environments*, International Journal of Robotics Research **21(10-11)** (2002), 829–848.

[GBML$^+$00] H.H. Gonzalez-Banos, E. Mao, J.C. Latombe, T.M. Murali, and A. Efrat, *Planning robot motion strategies for efficient model construction*, Robotics Research – The 9th International Symposium (J.M. Hollerbach and D.E. Koditschek, eds.), Springer, 2000, pp. 345–352.

[GF02] J. Gutmann and D. Fox, *An experimental comparison of localization methods continued*, In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (Lausanne, Switzerland), 2002.

[GHL$^+$87] L. J. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan, *Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons*, Algorithmica **2** (1987), 209–233.

[GMR97] L. Guibas, R. Motwani, and P. Raghavan, *The robot localization problem*, SIAM J. Computing **26** (1997), no. 4, 1120–1138.

[HKL$^+$98] D. Hsu, L. Kavraki, J. C. Latombe, R. Motwani, and S. Sorkin, *On finding narrow passages with probabilistic roadmap planners*, In Proc. Int. Workshop on Algorithmic Foundations of Robotics, 1998.

[HR76] L. Hyafil and R. L. Rivest, *Constructing optimal binary decision trees is np-complete*, Information Processing Letters **5** (1976), 15–17.

[JK01] P. Jensfelt and S. Kristensen, *Active global localisation for a mobile robot using multiple hypothesis tracking*, IEEE Transactions on Robotics and Automation **17(5)** (2001), 748–760.

[Kal92] G. Kalai, *A subexponential randomized simplex algorithm*, In 24th ACM STOC, 1992, pp. 475–482.

[Kar91] R. M. Karp, *An introduction to randomized algorithms*, Discrete Appl. Math. **34** (1991), 165–201.

[KL98] L. Kavraki and J. C. Latombe, *Probabilistic roadmaps for robot path planning*, Practical Motion Planning in Robotics: Current Approaches and Future Challenges (1998), 33–53.

[Kle94] J. Kleinberg, *The localization problem for mobile robots*, In Proc. 35th IEEE Conference on Foundations of Computer Science (Santa Fe, NM), IEEE Computer Society Press, 1994, pp. 521–533.

[KNW97] O. Karch, H. Noltemeier, and T. Wahl, *Robot localization: Theory and implementation*, In Abstracts 13th European Workshop Comput. Geom., 1997, pp. 17–19.

[KNW98] ______, *Using polygon distances for localization*, 1998.

[KW99] O. Karch and T. Wahl, *Relocalization – theory and practice*, Special Issue of Discrete Applied Mathematics on Computational Geometry **93** (1999).

[LDW91] J. J. Leonard and H. F. Durrant-Whyte, *Mobile robot localization by tracking geometric beacons*, IEEE Trans. on Robotics and Automation **7(3)** (1991), 376–382.

[Lee83] D. T. Lee, *Visibility of a simple polygon*, Computer Vision, Graphics, and Image Processing **22** (1983), 207–221.

[LL86] D. T. Lee and A. K. Lin, *Computational complexity of art gallery problems*, IEEE Transactions on Information Theory **32** (1986), 276–282.

[LMSS55] K. De Leeuw, E. F. Moore, C. E. Shannon, and N. Shapiro, *Computability by probabilistic machines*, Automata Studies (1955), 183–212.

[MD94] P. MacKenzie and G. Dudek, *Precise positioning using model-based maps*, In Proc. IEEE Int. Conf. on Robotics and Automation, 1994, pp. 1615–1621.

[MR95] R. Motwani and P. Raghavan, *Randomized algorithms*, Cambridge University Press, New York, 1995.

[MSW92] Jiri Matousek, Micha Sharir, and Emo Welzl, *A subexponential bound for linear programming*, In Symposium on Computational Geometry, 1992, pp. 1–8.

[MSW02] A. Milstein, J. N. Snchez, and E. Williamson, *Robust global localization using clustered particle filtering*, AAAI/IAAI, 2002.

[NMN94] S. Nayar, H. Murase, and S. Nene, *Learning, positioning, and tracking visual appearance*, In Proc. of IEEE Int'l. Conf. on Robotics and Automation, 1994.

[NPB95] I. Nourbakhsh, R. Powers, and S. Birchfield, *Dervish: an office navigating robot*, AI Magazine **16(2)** (1995), 53–60.

[OCON82] J. O'Rourke, C. Chien, T. Olson, and D. Naddor, *A new linear algorithm for intersecting convex polygons*, Comput. Graph. Image Process. **19** (1982), 384–391.

[O'R98] J. O'Rourke, *Computational geometry in c*, Cambridge University Press, 1998.

[Rab76] M. O. Rabin, *Probabilistic algorithms*, Algorithms and Complexity, Recent Results and New Directions (1976), 21–39.

[Sch96] Sven Schuierer, *Sensing, modelling and planning*, Intelligent Robots, ch. Efficient robot self-Localization in simple polygons, pp. 129–146, World Scientific Publ., 1996.

[SD98] R. Sim and G. Dudek, *Position estimation using principal components of range data*, In Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems, 1998.

[She89] T. Shermer, *Hiding people in polygons*, Computing **42** (1989), 109–131.

[She92] ______, *Recent results in art galleries*, In Proc. IEEE, Special Issue on Computational Geometry, vol. 80(9), 1992, pp. 1384–1399.

[SK95] R. Simmons and S. Koenig, *Probabilistic robot navigation in partially observable environments*, In Proc. International Joint Conference on Artificial Intelligence, 1995, pp. 1080–1087.

[SRR01] W. Scott, G. Roth, and J.-F. Rivest, *View planning as a set covering problem*, Tech. report, National Research Council of Canada (NRCC), Institute for Information Technology, 2001.

[SS77] R. Solovay and V. Strassen, *A fast monte carlo test for primality*, SIAM J. Computing **6** (1977), no. 1, 84–85.

[ST88] J. Sack and G. Toussaint, *Guard placement in rectilinear polygons*, Computational Morphology (1988), 153–175.

[Sug88] K. Sugihara, *Some location problems for robot navigation using a single camera*, Computer Vision, Graphics, and Image Processing **42** (1988), 112–129.

[TFB00] S. Thrun, D. Fox, and W. Burgard, *Monte carlo localization with mixture proposal distribution*, AAAI/IAAI, 2000, pp. 859–865.

[TFBD01] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, *Robust monte carlo localization for mobile robots*, Artificial Intelligence **128** (2001), no. 1-2, 99–141.

[TK03] C. Tovey and S. Koenig, *Improved analysis of greedy mapping*, In Proceedings of the International Conference on Intelligent Robots and Systems, 2003.

[Tou86] Godfried T. Toussaint, *A linear-time algorithm for solving the strong hidden-line problem in a simple polygon*, Pattern Recognition Letters **4** (1986), 449–451.

[TW86] R. E. Tarjan and C. J. Van Wyk, *A linear-time algorithm for triangulating simple polygons*, In Proc. 18th annual ACM Symposium on Theory of Computing (Berkeley, California, United States), 1986, pp. 380–388.

Document Log:

Manuscript Version 0

Typeset by $\mathcal{A}\mkern-5mu\mathcal{M}\mkern-4mu\mathcal{S}$-LaTeX — 10 May 2004

MALVIKA RAO

McGILL UNIVERSITY, 3480 UNIVERSITY ST., MONTRÉAL (QUÉBEC) H3A 2A7, CANADA, *Tel.* : (514) 398-7071

E-mail address: rao@cs.mcgill.ca

Typeset by $\mathcal{A}\mkern-5mu\mathcal{M}\mkern-4mu\mathcal{S}$-LaTeX