Vision-Based Motion Planning and Exploration Algorithms for Mobile Robots

Camillo J. Taylor, David J. Kriegman

Abstract—
This paper considers the problem of systematically exploring an unfamiliar environment in search of one or more recognizable targets. The proposed exploration algorithm is based on a novel representation of environments containing visual landmarks called the *boundary place graph*. This representation records the set of recognizable objects (landmarks) that are visible from the boundary of each configuration space obstacle. No metric information about the scene geometry is recorded nor are explicit prescriptions for moving between places stored. The exploration algorithm constructs the boundary place graph incrementally from sensor data. Once the robot has completely explored an environment, it can use the constructed representation to carry out further navigation tasks. In order to precisely characterize the set of environments in which this algorithm is expected to succeed, we provide a necessary and sufficient condition under which the algorithm is guaranteed to discover all landmarks. This algorithm has been implemented on our mobile robot platform RJ, and results from these experiments are presented. Importantly, this research demonstrates that it is possible to design and implement provably correct exploration and navigation algorithms that do not require global positioning systems or metric representations of the environment.

Keywords— **exploration, navigation, mobile robots, landmarks**

I. Introduction

Every year millions of tourists descend upon the Louvre museum in Paris hoping to catch a glimpse of the Mona Lisa, the Venus De Milo and I.M. Pei's controversial pyramid. On arriving at the museum they are faced with the problem of finding these famous artworks in a large, unfamiliar, maze-like building. The problem encountered by these hapless tourists is very similar to the kinds of exploration and navigation tasks we would like mobile robots to solve autonomously.

Consider the problem of programming a mobile robot to load all of the white boxes in a warehouse onto a truck. Or the problem of designing a mobile robot to perform autonomous inspections of nuclear waste facilities in search of leaking barrels of toxic waste [2] In each of these situations neither the designer nor the user knows precisely where the objects of interest are located. They must rely instead on the robot's ability to discover these targets automatically.

This paper presents an exploration algorithm that enables a mobile robot equipped with a visual recognition system to carry out a systematic exploration of an unfamiliar environment in search of one or more recognizable targets. This algorithm could be employed to accomplish the real-world tasks described in the previous paragraph.

The problem of searching for a particular target object is actually equivalent to the problem of finding all of the recognizable objects in the environment since the robot may have to discover every object before it can decide whether or not the target object is present. Therefore, the exploration algorithm presented in this paper is actually designed to search for all of the recognizable objects.

The proposed algorithm is based on a novel representation of environments containing recognizable objects (visual landmarks) called the *boundary place graph*. A *visual landmark* is simply a distinctive object or pattern that the robot can recognize with its vision system whenever it is in view. In a typical office environment tables, chairs or trash cans could serve as landmarks. In an art museum, paintings or sculptures may be used for this purpose. This paper does *not* address the problem of recognizing or selecting an appropriate set of landmarks in a given environment. In the sequel we will assume that the robot can recognize some set of objects and proceed to design algorithms based on this capability.

The boundary place graph records the set of landmarks that are visible from the boundaries of various configuration space obstacles in the environment. It does *not* record any representation of the geometry of the environment nor does it store explicit prescriptions (trajectories) for moving between places. The exploration algorithm constructs this representation incrementally from sensor data. Once the robot has completed its exploration, it can use the boundary place graph to plan and execute further navigation tasks within the environment.

It is important to note that the exploration algorithm presented in this paper does *not* assume any prior knowledge about the geometric structure of the environment. It is not provided with any prior information about the absolute or relative positions of the obstacles or of the landmarks and it does not attempt to measure these quantities. Nor does the algorithm assume *a priori* knowledge about the number of obstacles or landmarks it might encounter.

In order to precisely characterize the set of environments in which this algorithm can be expected to succeed, we provide a necessary and sufficient condition under which the algorithm is guaranteed to discover all of the landmarks in its environment. Importantly, this research demonstrates that it is possible to design and implement provably correct exploration and navigation algorithms that do not require global positioning systems or metric representations of the environment.

Camillo J. Taylor is with the GRASP Laboratory, CIS Dept. University of Pennsylvania, Philadelphia, PA USA. E-mail: cjtaylor@central.cis.upenn.edu

David J. Kriegman is with the Center for Computational Vision and Control, Dept. of Electrical Engineering, Yale University, New Haven, CT 06520, USA, E-mail kriegman@cs.yale.edu

The remainder of this section is devoted to reviewing related work. Section II describes all of the assumptions made about the structure of the environment and the capabilities of the mobile robot. It also contains the definition of the boundary place graph representation. Section III presents an analysis of a simple, sensor-based strategy that the mobile could use to navigate between nodes in the boundary place graph. Section IV presents an exploration algorithm that enables the robot to construct the boundary place graph from sensor data. Completeness and complexity results for the algorithm are also presented in this section. Section V describes the implementation of the exploration and navigation algorithms on our mobile robot platform, R.J., and discusses the experiments that were carried out. Finally, Section VI discusses some of the conclusions drawn from this research.

A. Related Work

In recent years, the problem of exploring an unknown environment has received considerable attention from researchers in the computer science theory community. Chin and Ntafos [4] considered the "night watchman's problem" of finding a closed route through the interior of a simple polygon such that every point on the interior of the polygon is visible from some point on the path. They showed that the general problem is NP-hard, however, for the special case of a rectilinear polygon they described an $O(n \log n)$ procedure for computing such a tour. Deng and Papadimitriou [6] investigated the problem of exploring an unknown polygonal room with a bounded number of polygonal obstacles. The length of the path taken by a robot that learns the environment for the first time is compared to the length of the shortest night watchman's tour. Kalyanasundaram and Pruhs [9] considered the problem of conducting a systematic exploration of an unknown environment containing a number of convex polygonal obstacles.

All of these algorithms assume that the environment is populated with polygonal obstacles and that the robot can accurately determine its position with respect to a global frame of reference. In practice, it is quite difficult to accurately estimate the position of a mobile robot with respect to an arbitrary frame of reference. In most mobile robot systems, some form of odometry or dead reckoning is used to determine the robot's global position. Every odometric system suffers from the problem of cumulative error; as the robot moves further and further from its starting point, errors in the estimates of the robot's position grow monotonically. There are several robot localization systems that require the user to go through the trouble and expense of installing a set of beacons at known locations in the robot's workspace. Global positioning techniques based on GPSS are inapplicable to indoor environments where structural elements of the building may occlude the signals from the satellites. One of the main advantages of the algorithms presented in this paper is that they do not require a global positioning system, which simplifies their implementation.

Most of the research that has been reported in the robotics literature casts the exploration problem in terms of constructing a global metric map of the robot's workspace [10],[1],[14] If the robot were able to construct such a map, then it could employ classical path planning algorithms [3],[16] to navigate through its environment.

Two systematic techniques (named the Sightseer and Seed Spreader strategies) for exploring the geometry of a two-dimensional configuration space were proposed by Lumelsky, Mukhopadhyay and Sun [19] They assume that that the robot is equipped with a tactile sensor and that it can accurately determine its position in a global frame of reference.

Iyengar and Rao [8],[22],[23] developed exploration algorithms inspired by the visibility graph approach to path planning [16],[17]. The problem is modeled in terms of a point robot moving through a 2-D configuration space populated with polygonal obstacles. In this case, perfect sensing is assumed, and the robot learns the visibility graph online.

Choset and Burdick recently developed the generalized Voronoi diagram which is based on a deformation retract and can represent configuration spaces of arbitrary dimension [5]. An on-line method is presented for constructing the representation from a sensor that can measure range in the configuration space. They also show how this representation can be used for motion planning.

The main criticism of the algorithms described in the previous paragraphs centers on their assumption that the robot can construct a reasonably accurate metric map of its workspace in a global coordinate system. In practice, this is extremely problematic. Once again the main difficulties stem from the fact that it is not easy to determine the position of the robot with respect to an absolute coordinate frame of reference. Whenever the robot encounters new features in the environment, it uses its estimate for its current position to determine where these features should appear in the map. This implies that any errors in the positioning system will be reflected in the map that the robot constructs.

Other researchers have developed navigation algorithms that rely on recognition. Kuipers and Byun [11],[12] proposed a scheme based on a place graph where a place is defined as a specific point in the world that the robot can recognize from sonar readings. The edges in this place graph represent navigation operations, like wall following, that take the robot from one place to another. Mataric later designed and implemented a similar algorithm that constructs a place map of portions of an office environment from sonar data [21]. The basic idea behind both of these approaches is to recast the navigation problem in terms of finding a route through the place graph from one region to another. Neither of these approaches addressed the problem of conducting a systematic exploration of an environment containing multiple obstacles.

The Achilles heel of these place graph algorithms is their reliance on heuristics to subdivide the world into recognizable places. It is not entirely clear that these heuristics will produce a useful or meaningful partitioning of an arbitrary environment. In this paper, we explain how the obstacle

boundaries and the visibility of specific landmarks induce a well-defined partitioning of the freespace which the robot can learn online.

Recognizable landmarks are the basis of many other navigation strategies. Levitt et. al. [15] proposed a method for partitioning an outdoor area into regions based on the visibility of pairs of recognizable landmarks. A robot can determine its position with respect to a set of *Landmark Pair Boundaries* which are virtual lines drawn between pairs of landmarks in the environment. A path from one place to another is planned by deducing the sequence of landmark pair boundaries that the robot would cross to get from one region to another. However, the method does not account for occlusion of landmarks or unexpected obstacles.

Lazanas and Latombe [13] have developed provably correct navigation algorithms based on landmark recognition and localization, assuming that the positions of the landmarks and the obstacles are known *a priori*. The task is to construct a navigation plan that will pilot the robot to the target location even in the presence of significant control uncertainty. The problem considered in this paper is quite different since the robot will *not* be provided with any prior information about the structure of its environment or the landmarks contained therein.

II. World Model

This research was motivated by a desire to produce algorithms that would enable mobile robot systems to navigate successfully in cluttered, unstructured office environments. In order to tackle this problem, we needed to develop a model of the environment which was simple enough to be tractable, but realistic enough to allow us to implement the resulting algorithms on our mobile robot platform. This section describes *all* of the assumptions that were made about the capabilities of the robot and the structure of the environment.

Figure 1 shows the major aspects of the world model. The robot is modeled as a holonomic vehicle with a circular cross section traveling through a planar workspace. These assumptions allow us to represent the robot as a point moving in a two-dimensional configuration space.

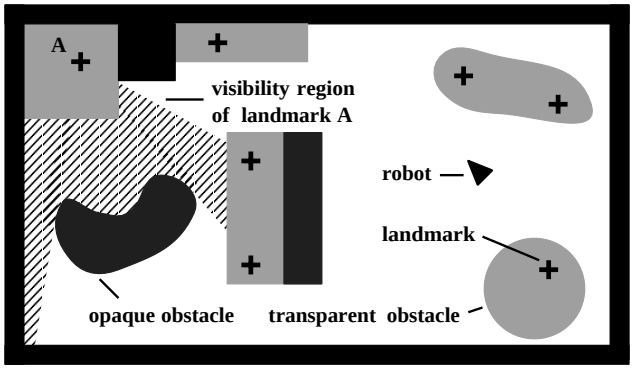


Fig. 1. Major features of the world model.

The robot is equipped with a vision-based recognition system which allows it to recognize and localize some (but not all) of the objects in the environment. More specifically, we assume that there is a set of distinct, recognizable, fixed objects in the world which will be termed *landmarks*. These landmarks are modeled as points in the workspace. We do *not* assume that the robot has any prior information about the absolute or relative positions of these landmarks.

We assume that the robot can measure the bearing to every landmark that is visible from its current position as shown in Figure 2 which implies that the robot is capable of looking in all directions. This capability can be realized by mounting the camera system on a pan-tilt head or using an omni-directional camera [25].

We also assume that the robot can measure some quantity that is monotonically related to the distance between the robot and the landmark. For example, the robot may be able to measure the height of the landmark in the image which decreases monotonically as the robot gets further from the landmark. It can use this measurement to determine whether it is closer or further away from the landmark than it was at a previous time. In the sequel we will refer to this measured quantity as the *relative range*. It is important to note that the robot *cannot* compute its coordinates with respect to the landmark using these measurements from its vision system. Nor do we assume that the robot has any means of estimating its position with respect to a global frame of reference.

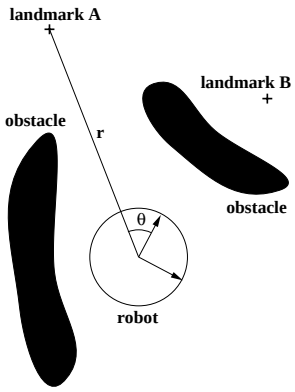


Fig. 2. The robot can measure the bearing, θ, to every visible landmark. It can also measure some quantity that is monotonically related to the range from the landmark, r. Note that landmark B is not visible from the robot's current position.

The robot must also contend with a set of obstacles, and these are modeled as simple closed curves in a planar configuration space; no assumptions are made about the shapes of the obstacles. There are a finite number of obstacles in the configuration space, and each of these has a finite length perimeter. Since the robot has nonzero diameter, a single configuration space obstacle may actually represent several disjoint obstacles in the workspace. For the sake of simplicity, we assume that the freespace is bounded by one of the configuration space obstacles.

The obstacles in our model come in two flavors, opaque and transparent. Opaque obstacles occlude landmarks from the view of the robot, while transparent ones do not. In the real world, opaque obstacles might include walls or

bookshelves; transparent obstacles will block the robot's progress, but the robot can either see through or over them (e.g. tables, waste paper baskets, windows). A configuration space obstacle may contain components that are both opaque and transparent. Every landmark in the environment must be contained within the boundary of an obstacle. A single obstacle may contain any number of landmarks or none at all.

We also assume that the robot is equipped with a proximity sensor that can detect imminent collisions and can be used to perform boundary following. We further assume that the robot will be able to determine when it has completely circumnavigated an obstacle.

A. Definitions and Observations

Given the world model described above, we can make the following definitions and observations:

Definition: Visibility Region The *visibility region* of a landmark is the set of points in the workspace from which the landmark is visible. As seen in Figure 1 , the visibility region of a landmark is always a simply connected, closed, star-shaped set. Due to sensor resolution and measurement noise, it is possible that a landmark will only be visible when the robot is within some finite radius.

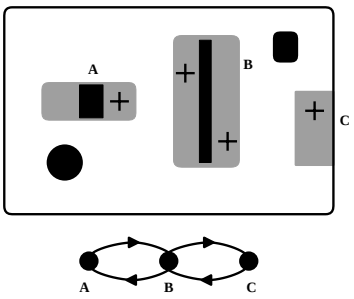


Fig. 3. Boundary place graph representation for a simple environment.

Definition: Boundary Place Graph
A *boundary place* is defined as the boundary of a configuration space obstacle that contains at least one landmark. Boundary place B is said to be *connected* to boundary place A iff a landmark contained inside boundary B is visible from some point on boundary A (i.e., the boundary of A intersects the visibility region of a landmark within B). These definitions allow us to define a *boundary place graph* as shown in Figure 3 where the nodes represent boundary places, and the directed arcs represent connections between these places. Note that there may be any number of configuration space obstacles in the environment that do not contain landmarks; however, these obstacles will *not* be included in the boundary place graph.

Observation 1: In order for a landmark to be visible in the robot's freespace, it must be visible along some portion of the enclosing obstacle boundary.

Proof: This proposition can be proved by constructing a straight line between the landmark and the point in freespace where the landmark is visible. Since light travels in straight lines, the landmark must also be visible from every point along that line and since the landmark lies in the interior of an obstacle, the line must cross the that obstacle's boundary at least once by the Jordan curve theorem. □

Observation 2: If the robot circumnavigates all the obstacles that contain landmarks, it will eventually discover all the visible landmarks in the workspace.

Proof: Since every landmark must be visible from the boundary of the obstacle that encloses it (from Observation 1), a robot that circumnavigates all the obstacle boundaries will eventually find all the landmarks. □

This observation is particularly relevant to our task since it means that the robot does not have to investigate every point in the 2-dimensional configuration space in order to find all the landmarks; it can accomplish its goal simply by following a finite length path around the boundaries of the obstacles. The exploration algorithm described in Section IV is based on this observation.

III. Navigation Algorithm

This section presents an analysis of a simple, sensor-based control strategy that allows the robot to navigate between two boundary places that are connected by an arc in the boundary place graph. Once this basic capability has been established, it can be employed by the exploration and navigation algorithms, described in the following sections.

Consider two boundary places, A and B, that are connected by an arc in the boundary place graph. Since the two places are connected, we know by definition that at least one landmark located inside boundary place B is visible from boundary place A. The robot can simply circumnavigate boundary place A until one of these landmarks became visible and then employ the approach algorithm outlined below to move to boundary place B. Since a transparent obstacle may lie between place A and the landmark in place B (See Fig. 4), the robot cannot simply travel in a straight line towards the landmark.

Approach Algorithm:
1) Head towards landmark until obstacle encountered.
2) Circumnavigate the obstacle boundary and let L denote the point on the section of the boundary from which the landmark is visible where the robot comes closest to the target.
3) **If** the target landmark appears to lie inside the obstacle boundary at the point L
   then
   terminate,
   else
   follow the boundary back to L.
4) Goto step 1.

Figure 4 shows an example of the execution of this approach algorithm. This example demonstrates that the robot may leave the visibility region of the landmark during

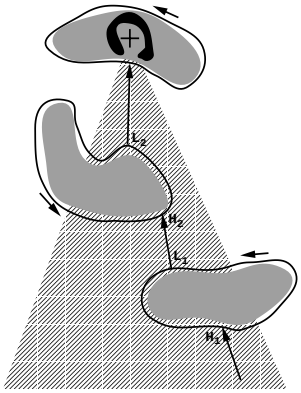


Fig. 4. Execution of the approach algorithm.

the execution of this algorithm. Note that this algorithm relies on the robot's ability to determine when it has completely circumnavigated an obstacle. Note also that the robot does not need to measure the actual distance to the target. It only needs to be able to detect when it is closest to the landmark.

This approach algorithm is similar to the Bug1 algorithm proposed by Lumelsky and Stepanov [18],[20]. However, the Bug1 algorithm is based upon the assumption that the robot is equipped with a global positioning system which allows it to measure its position with respect to the target at all times. In our model, the robot can only measure the relative range and bearing to the landmark when it is in view; it does not have any way of measuring its coordinates with respect to the target.

The difference between the two approaches can be illustrated by considering the problem faced by a tourist in Paris who wishes to visit the Eiffel tower. In the Bug1 algorithm, the tourist would be provided with a GPSS unit and would be given the exact latitude and longitude of the tower. In the approach algorithm described in this paper, the tourist would be shown a photograph of the historic landmark so that she could recognize her target. During the course of her journey, she would use the apparent size of the tower to gauge the relative range to her goal whenever it was in view. Note that she cannot always see the landmark nor can she measure her position with respect to the tower or her *actual* distance from the landmark, but she can use the apparent size of the tower to decide whether she is nearer or further away than she was at some previous time.

Both algorithms will get the tourist to her destination, but the underlying assumptions about her sensory capabilities are markedly different. We are obliged to demonstrate that the approach algorithm does in fact converge under these more restrictive assumptions.

The remainder of this section is devoted to proving that, when guided by this approach algorithm, the robot will converge in a finite amount of time to the boundary of the obstacle containing the landmark. The proof has been divided into two parts: the first part discusses the procedure that the robot uses to decide whether the target landmark lies inside or outside of a configuration space obstacle; the second part demonstrates that the robot will, in fact, converge to the correct obstacle boundary and gives an upper bound on the total distance that the robot would have to travel before termination.

Consider the portion of the obstacle boundary from which the landmark is visible, and let A denote the point in this section that is closest to the target. In the sequel, this point will be referred to as the *closest observable point*. Note that this point may be different from the point on the boundary that is actually closest to the target since the landmark may not be visible from every point on the obstacle boundary.

In general, the closest observable point on a boundary will be unique; however, it is possible to construct situations where this is not the case. Consider, for example, a landmark placed in the center of a circular obstacle. This special case does *not* affect the correctness of the arguments advanced in this section since the important properties of the closest observable point are shared by all the points that satisfy the definition. In the sequel we will discuss the case for a single closest observable point without loss of generality.

The robot can locate the closest observable points as it circumnavigates the obstacle boundary by keeping track of the relative range to the target whenever the landmark is visible. Whenever the robot needs to return to one of the closest observable points, it can simply track the obstacle boundary until the landmark is visible and the relative range to the target is equal to the smallest relative range recorded along the obstacle boundary. Note that the robot does not have to record the actual coordinates of the closest observable points.

If the landmark appears to be inside the boundary at the closest observable point, then it must lie inside the obstacle boundary, otherwise it must lie outside. This implies that the termination condition given in step 3 of the algorithm will only succeed when the obstacle has circumnavigated the obstacle that contains the target.

Lemma 1: The line segment between the closest observable point and the target landmark only intersects the obstacle boundary at the closest observable point.

Proof: If there were another point on the obstacle boundary that was also on the line segment between the closest observable point and the landmark, that point would also lie in the visibility region of the landmark and it would be closer to the target than the supposed closest observable point. □

Observation 3: If the line segment between the closest observable point and the landmark is directed into the obstacle at the closest observable point, then the landmark must lie within the obstacle boundary, otherwise it must lie outside.

Proof: Lemma 1 states that the line segment between the closest observable point and the landmark will never intersect the obstacle boundary more than once. So if the line segment is directed into the obstacle at the closest observable point, the target must lie inside the obstacle boundary

by the Jordan Curve Theorem. Similarly, if the line segment is directed away from the obstacle at that point, the landmark must lie outside. □

The following Lemmas prove that the robot will converge to the boundary of the obstacle containing the target landmark in a finite amount of time. This section of the proof follows the structure of the analysis provided in [20].

Lemma 2: When the robot leaves an obstacle boundary in order to head towards the target landmark, it never returns to any point on that obstacle.

Proof: Let O be an obstacle that the robot encounters along its path that does *not* contain the target landmark. Let H_i denote the point where the robot first encounters the obstacle and L_i denote the point where it leaves that obstacle boundary to head towards the target as shown in Figure 4. The integer i indicates the order in which the obstacles are encountered.

Since the robot encounters H_i while it is moving in a straight line towards a visible target, H_i must lie in the visibility region of the landmark. The algorithm will choose the leave point, L_i, to be the closest observable point on the obstacle boundary.[1] From the definition of the closest observable point, we can infer that $d(H_i) > d(L_i)$ where $d(P)$ denotes the distance between the point P and the target landmark. We can also deduce that $d(L_i) > d(H_{i+1})$ since the robot travels directly towards the target when it leaves the obstacle boundary.

Taken together, these observations imply that if we consider the sequence of hit and leave points that the robot encounters along its journey, the distance between the robot and the target landmark decreases monotonically until the robot reaches the boundary of the obstacle enclosing the landmark.

The only way that the robot could return to a previously visited obstacle is if it encounters that obstacle at a new hit point H'. This hit point H' would have to be closer to the target than the previous leave point L_i associated with that obstacle, which would imply that L_i was not the closest observable point on the boundary after all. □

Lemma 3: If d_i denotes the perimeter of an obstacle that the robot encounters during its journey, then the robot will travel a distance no more than $2d_i$ along the boundary of that obstacle.

Proof: The robot will circumnavigate every obstacle it encounters during its journey which means it will travel a distance of at least d_i along the obstacle boundary. In addition, it may have to travel a maximum distance of d_i along the boundary to get back to the leave point associated with that obstacle. □

If the robot had some means of measuring the distance it has traveled along the obstacle boundary, it could choose the shortest path along the boundary back to the leave point which would reduce this upper bound to $1.5d_i$.

[1] If there are many points on the obstacle boundary that could serve as the closest observable point, the robot can choose any one of these without affecting the correctness of the algorithm.

Lemma 4: The robot can only encounter obstacles that intersect the portion of the landmark's visibility region that lies within a disc of radius D centered around the target landmark where D denotes the distance between the robot's start point and the landmark.

Proof: In proving Lemma 2, we showed that the distance between the hit points and the target landmark decreases monotonically over time which implies that every hit point must be less than D units away from the target landmark. We also noted that every hit point must lie within the landmark's visibility region. Taken together, these observations imply that every obstacle that the robot encounters must have some section of its boundary intersect the portion of the landmark's visibility region that lies within a disc of radius D centered around the target landmark. □

Observation 4: The maximum distance that the robot will travel before it converges to the boundary of the obstacle that contains the target landmark is given by $D+2\sum d_i$ where $\sum d_i$ represents the sum of the perimeter of the obstacles that intersect the portion of the landmark's visibility region that lies within a disc of radius D centered around the target landmark.

Proof: If there were no extraneous obstacles, the robot would have to travel a distance of at most D before it encountered the obstacle boundary enclosing the target landmark. Lemma 4 states that the robot will only encounter those obstacles that intersect the portion of the landmark's visibility region lying within a disc of radius D centered around the target landmark. Lemmas 2 and 3 imply that the robot will travel no more than $2\sum d_i$ along the boundaries of those obstacles. □

IV. Exploration Algorithm

Observation 2 states that if the robot is able to circumnavigate all of the obstacles that contain landmarks, it would eventually discover all visible landmarks. The algorithm presented below is based on this observation. Effectively, the algorithm will cause the robot to perform a tour of the boundary place graph of the environment where visiting a node in the place graph corresponds to circumnavigating the boundary of that obstacle. We will term a particular landmark in the environment *explored* iff the robot has circumnavigated the configuration space obstacle that encloses that landmark. By exploring each of the landmarks that it sees, the robot can incrementally learn the entire boundary place graph of the environment. Once the entire graph has been explored, the robot can use the constructed representation for further navigation tasks. The entire exploration algorithm is outlined below in pseudo code. At the beginning of the exploration, we assume that the robot can see at least one landmark from its current position. If not, it would carry out some variant of a random walk until it found its first landmark.

Note that this exploration algorithm does *not* require any *a priori* information about the structure of the environment. It does not require any information about the

number of obstacles in the environment nor is it given any information about the number of landmarks in the environment or their positions in the scene.

Exploration Algorithm:
Find first landmark.
While ⟨ unexplored landmarks ⟩
    Select unexplored landmark, λ.
    Plan path through explored part of the
        boundary place graph to region where
        λ is visible.
    Approach λ.
    Circumnavigate boundary that contains λ, and
        record any observed landmarks.
    Update the place graph.

The robot maintains two data structures during this exploration procedure: $\mathcal{L}$ the list of landmarks it has seen and $\mathcal{B}$ the list of boundary places it has circumnavigated. For each boundary place, the robot records which landmarks are visible from that obstacle boundary. These landmarks are divided into two categories: interior landmarks which lie inside the obstacle boundary and exterior landmarks which lie outside. These lists, $\mathcal{L}$ and $\mathcal{B}$, represent the portions of the boundary place graph that the robot has explored so far.

Figure 5 shows the progress of the exploration algorithm on a typical environment at various stages. In this figure, the thicker lines along the robot's path denote sections where the robot would employ the approach algorithm while the thinner segments indicate places where the robot would use simple boundary following. The graphs below each figure denote the current state of the boundary place graph that the robot has constructed. In stage 1, the robot circumnavigates obstacle A and determines that there are two unexplored landmarks visible from the boundary of A which need to be explored. These unexplored landmarks correspond to unexplored edges in the robot's representation of the boundary place graph. In stage 2 the robot traverses one of the unexplored edges and ends up circumnavigating obstacle B; after it has finished its circuit around this obstacle, it concludes that there is still one unexplored edge in the graph, and so it plans a path back through the graph to boundary A and traverses the unexplored edge to boundary place C. In the final stage it visits obstacle D and concludes that there are no more unexplored landmarks in the environment. By this stage, the robot has built a complete representation for the boundary place graph of the environment which it can use in future navigation tasks.

Notice that by circumnavigating an obstacle the robot discovers all of the landmarks that can be seen from the boundary of that obstacle. These landmark sightings correspond to edges in the boundary place graph. Since the landmarks are all distinct, the robot can determine which of the landmarks lie within obstacles that it has already explored and which lie in unexplored nodes. To completely instantiate the boundary place graph, the robot does not have to traverse every arc, just visit every node.

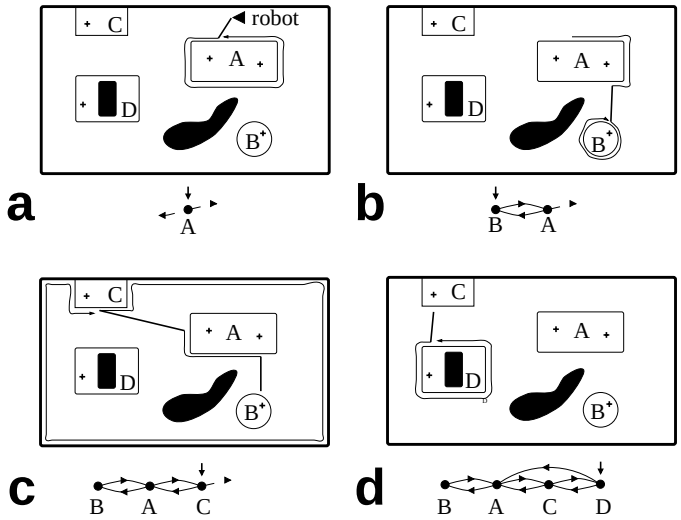


Fig. 5. Progress of the exploration algorithm: The thicker lines along the robot's path denote sections where the robot would employ the approach algorithm while the thinner segments indicate places where the robot would use simple boundary following. The dashed lines in the graph denote unexplored edges while the vertical arrow indicates the current position of the robot in the graph.

Since the exploration problem has been cast in terms of a graph traversal, it should not be surprising that the necessary and sufficient condition required to ensure success should be framed in the following manner.

Necessary and sufficient condition: The exploration algorithm presented above is guaranteed to discover all the landmarks that are visible from the robot's freespace regardless of which landmark it discovers first iff the boundary place graph of the environment is *strongly connected*. A directed graph is termed strongly connected iff it is possible to construct a path from any node in the graph to any other node.

Proof: Proving that the condition is necessary follows from the definition of a strongly connected graph. If the graph is not strongly connected, then we can always find two nodes A and B such that there is no path from A to B through the graph. This means that if the exploration procedure started at node A, it would never be able to carry out a complete tour of the graph since it could never reach node B.

In order to prove that this condition is in fact sufficient, we first divide the place graph into two parts, the explored section and the unexplored section. If there is no unexplored section, the exploration algorithm will terminate normally. Otherwise, we can select any node T in the unexplored section of the graph and consider a path from the robot's current location, S, to that node; since the place graph is strongly connected, we know that such a path must exist. Since the robot's current position, S must lie in the explored part of the graph, we can conclude that at some point along this path there must be an unexplored edge that connects a node in the explored part of the graph to a node in the unexplored part in the graph. This implies

that the robot can always plan a path through the explored part of the graph to some unexplored node. □

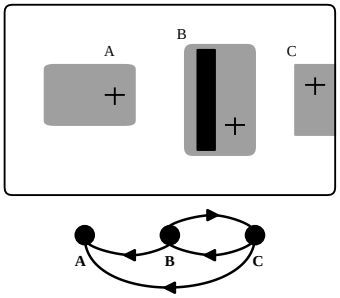


Fig. 6. The exploration algorithm will find all the visible landmarks iff the boundary place graph of the environment is strongly connected. This environment, where the boundary place graph is not strongly connected, could cause problems.

Figure 6 shows an example of an environment that could cause problems for the exploration algorithm. Notice that boundary place A is not connected to any of the other boundary places in the environment since the landmarks contained within B and C are occluded from all viewpoints on the boundary of A. If the robot were to start from boundary place A, it would terminate its exploration after circumnavigating that obstacle and would not discover the other landmarks. If the boundary place graph is strongly connected, the exploration will succeed regardless of which node in the graph is selected as the starting point.

A. Exploration Complexity

This section discusses some of the complexity issues related to the online graph exploration algorithm described earlier in this section.

Observation 5: In order to carry out a complete exploration of a boundary place graph with n nodes the robot will have to traverse at least $(n-1)$ edges.

Proof: Given a graph with n nodes the robot will have to traverse at least one edge for each new node it visits, this implies that the robot will have to traverse at least $(n-1)$ edges in order to visit all n nodes. □

Observation 6: In order to carry out a complete exploration of a boundary place graph with n nodes the robot will have to traverse at most $n(n+1)/2$ edges.

Proof: At any stage in the exploration process, the worst that could happen is that the robot might have to travel through all the previously visited nodes in order to get to an unexplored edge. That is, at stage i it might have to traverse i edges which implies that the total number of edges required to explore the entire graph would be $\sum_{i=1}^{n} i = n(n+1)/2$. □

It is quite simple to construct graphs where the number of edges that the robot has to traverse in order to visit all of the nodes is $O(n^2)$. Consider the graph shown in Figure 7a where the n nodes are divided evenly between two sets: the trunk nodes and the leaf nodes. To visit a leaf node in this graph the robot has to travel through all of the trunk nodes. This means that the total number of edges that the algorithm will traverse is given by $(n/2) \times (n/2) = (n^2/4)$.

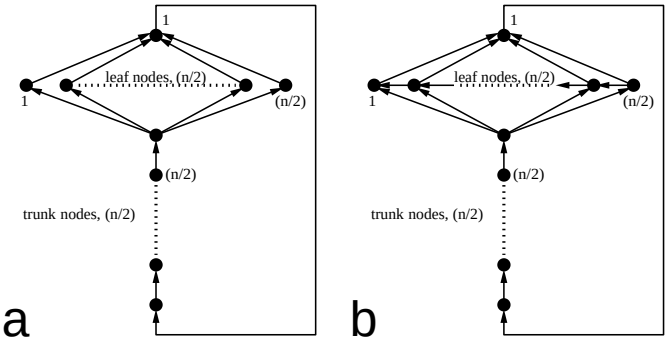


Fig. 7. (a) It will take $O(n^2)$ edge traversals to completely explore this graph. (b) The ratio between the number of edge traversals taken by the best online algorithm and the worst offline algorithm on this graph is $(n-1) : (n/2+1)(n/2)$.

By a similar construction, we can show that there is no competitive deterministic online algorithm for exploring an arbitrary strongly connected graph. An online algorithm is termed competitive iff the number of steps required by that algorithm is no more than a constant times the minimum number of steps required by an offline algorithm with complete information [6],[24].

Observation 7: There is no competitive deterministic online algorithm for exploring an arbitrary strongly connected graph.

Proof: Consider the graph shown in Figure 7b. This graph is similar to the one shown in Figure 7a, the only difference being that in this case the leaf nodes are connected together by a sequence of edges. By adding these edges we have made it possible to completely explore the graph with the minimum number of edge traversals $(n-1)$. The online algorithm, however, does not have the benefit of complete knowledge of the graph, so in the worst case it would have to carry out the same exploration sequence that it would have used if the extra edges were not there. That is the leaf nodes would get visited in the order indicated on Figure 7b, 1 through $n/2$. The ratio between the cost of the online algorithm and the offline algorithm is $O(n)$ so the online algorithm is *not* competitive. □

V. Experimental Results

The algorithms presented in the previous sections have been implemented on our experimental mobile robot platform, RJ which is shown in Fig. 8a. The simple targets shown in Figure 8b were employed as landmarks in our experiments. These targets were recognized in real time by making using of a projective invariant known as the cross-ratio [7]. The height of the bar-code target in the image was used as the relative range measurement.

Figure 9 shows the progress of the robot through an office complex that was used to test the exploration algorithm. This environment consisted of 4 distinct configuration space obstacles and 11 landmarks distributed over 3700 sq. ft. of floor space. The robot circumnavigated each

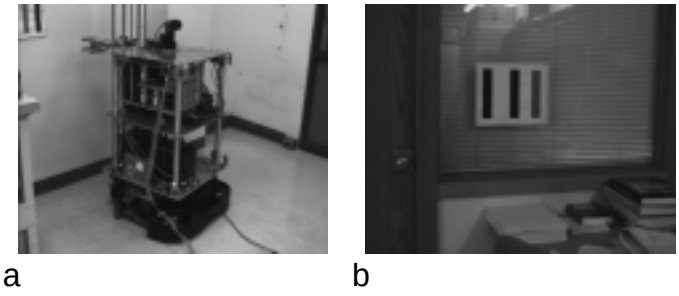


Fig. 8. (a) RJ, an experimental mobile robot platform. (b) Bar code landmarks used in our experiments. Note that the recognition algorithm must find these targets even in cluttered indoor environments with many vertical edges.

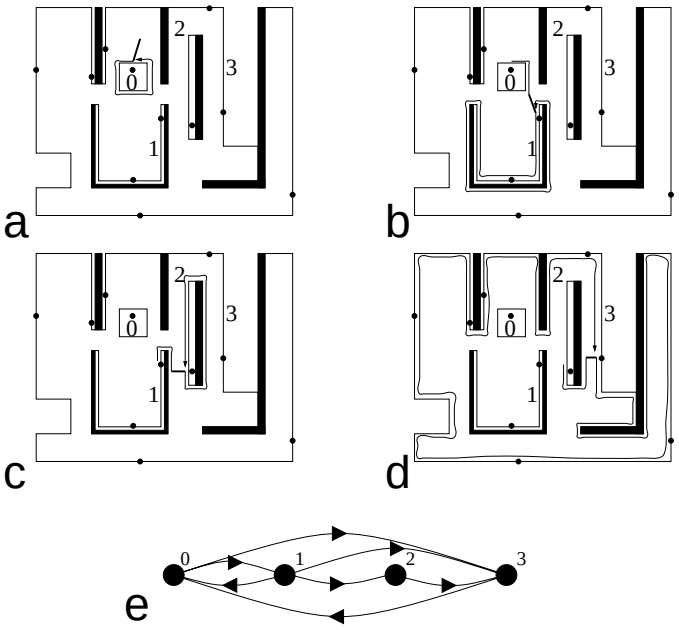


Fig. 9. This figure shows the progress of our mobile robot during its exploration of an office environment. During the course of the exploration, the robot circumnavigated each of the c-space obstacles in this office complex in the order indicated by their labels (0 thru 3). The boundary place graph that the robot constructs is shown in (e). The small black dots indicate the positions of the landmarks while the dark regions indicate opaque obstacles.

of the configuration space obstacles as indicated in the figure and eventually discovered all of the landmarks. The robot traveled several hundred meters during the course of this exploration procedure which took a little over half an hour. The robot was successful even in the presence of visual clutter and numerous small obstacles such as chairs, trash cans, etc. Since the robot was not equipped with a global positioning system, it had no means of plotting its course on a global map. Figure 9 is simply a sketch of the robot's path as seen by an outside observer. Once again, the darker lines denote segments of the path where the robot is executing the approach algorithm while the lighter lines denote the segments where the robot is carrying out simple boundary following.

Another experiment that was run on RJ is shown in Figure 10. In this experiment, the robot was first instructed to carry out a complete exploration of its environment; the resulting boundary place graph is shown in Figure 10e. The robot then used this place graph to navigate from one boundary place to another. Figures 10a-d show some of the paths that the robot generated and executed autonomously. Figures 10a,b,d show the robot executing paths that involve only one edge traversal of the boundary place graph while Figure 10c shows a plan with two edge traversals.

Recall that each edge traversal is composed of two parts (A) circumnavigating a boundary place until the target is in view and (B) approaching the target using the algorithm of Section III. As illustrated in Figure 4 the robot may encounter other transparent obstacles during the approach phase. This situation occurred when the robot executed the paths shown in Figure 10c (while approaching Boundary Place 1 from Boundary Place 0). Note that the approach algorithm implemented for the purposes of these experiments differs from the one described in Section III since the robot did not circumnavigate the intervening obstacles, it simply followed the boundary until it discovered a leave point that was closer to the target than the initial hit point. This version of the approach algorithm is analogous to the Bug2 algorithm proposed by Lumelsky and Stepanov [18],[20] and has similar convergence properties.

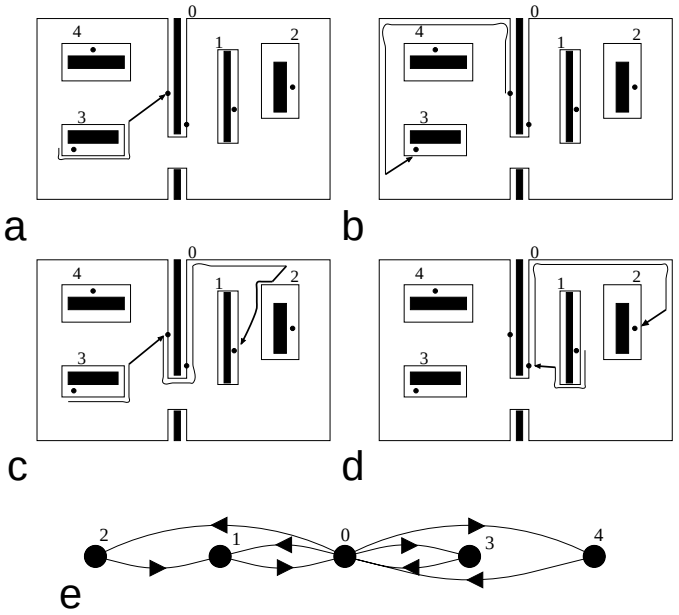


Fig. 10. Paths generated and executed by the robot are shown in a-d. The thicker lines along the robot's path denote sections where the robot employed the approach algorithm while the thinner segments indicate places where the robot used boundary following. Note that the robot may perform some boundary following while executing the approach algorithm.

VI. Conclusion

The main contribution of this research has been to provide novel exploration and navigation algorithms that would enable a mobile robot equipped with a vision-based recognition system to carry out a systematic exploration

of its environment in search of one or more recognizable objects. These algorithms are based upon a surprisingly simple representation for environments containing visual landmarks, the boundary place graph. For each of the configuration space obstacles that have been visited, the exploration algorithm simply records a list of the landmarks visible from the boundary of that obstacle. This representation differs substantially from others proposed in the literature since it does *not* attempt to record any metric information about the structure of the environment nor does it store any explicit prescriptions for getting between places. This approach offers a considerable advantage over previously proposed exploration algorithms since it does *not* require a global positioning sensor. Section IV presents a detailed analysis of this algorithm and describes a necessary and sufficient condition under which the robot will discover all of the landmarks.

Another important feature of these exploration and navigation algorithms is the fact that they only require the robot to perform two types of navigation operations: boundary following and navigation with respect to a visible landmark. Both of which can be carried out using simple, closed loop control strategies that only rely on local sensor data.

Acknowledgments

Thanks to Jeffery Westbrook for providing the examples shown in Figure 7. Support for this work has been provided in part by NSF Young Investigator Award, IRI-9257990 and DDM-9112458.

References

[1] N. Ayache and O. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Trans. on Robotics and Automation*, 5(6):804–819, December 1989.

[2] E. Byler, W. Chun, W. Hoff, and D. Layne. Autonomous hazardous waste drum inspection vehicle. *IEEE Robotics and Automation Magazine*, 2(1):6, March 1995.

[3] J. Canny. *The Complexity of Robot Motion Planning*. MIT Press, 1988.

[4] W.-P. Chin and S. Ntafos. Optimum watchman routes. *Information Processing Letters*, 28:39–44, May 1988.

[5] H. Choset and J. Burdick. Sensor based planning, part ii: Incremental construction of the generalized voronoi graph. In *IEEE Int. Conf. on Robotics and Automation*, Nagoya, Japan, 1995.

[6] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, page 298, October 1991.

[7] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis*. New York, Wiley, 1973.

[8] S. Iyengar, C. Jorgensen, S. Rao, and C. Weisbin. Robot navigation algorithms using learned spatial graphs. *Robotica*, 4(2):93–100, April-June 1986.

[9] B. Kalyanasundaram and K. Pruhs. A competitive analysis of algorithms for searching unknown scenes. *Computational Geometry: Theory and Applications*, 3:139–155, 1993.

[10] D. J. Kriegman, E. Triendl, and T. O. Binford. Stereo vision and navigation in buildings for mobile robots. *IEEE Trans. on Robotics and Automation*, 5(6):792–803, December 1989.

[11] B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8:47–63, 1991.

[12] B. Kuipers and T. Levitt. Navigation and mapping in large-scale space. *AI Magazine*, 9(2):25–43, 1988.

[13] A. Lazanas and J.-C. Latombe. Landmark-based robot navigation. In *Proc. Am. Assoc. Art. Intell.*, July 1992.

[14] X. Lebegue and J. Aggarwal. Generation of architectural cad models using a mobile robot. In *IEEE Int. Conf. on Robotics and Automation*, page 711. IEEE, 1994.

[15] T. S. Levitt, D. T. Lawton, D. M. Chelberg, and P. C. Nelson. Qualitative navigation. In *Proc. Image Understanding Workshop*, pages 447–465, February 1987.

[16] T. Lozano-Perez. Spatial-planning: A configuration space approach. *IEEE Transactions on Computers*, C-32:108–120, February 1983.

[17] T. Lozano-Perez and M. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, October 1979.

[18] V. Lumelsky and A. Stepanov. Dynamic path planning for a mobile automaton with limited information on the environment. *IEEE Trans. on Automatic Control*, 31(11):1058–63, 1986.

[19] V. J. Lumelsky, S. Mukhopadhyay, and K. Sun. Dynamic path planning in sensor based terrain acquisition. *IEEE Journal of Robotics and Automation*, 6(4):462–472, August 1990.

[20] V. J. Lumelsky and A. A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.

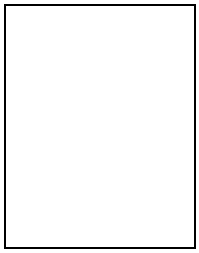
[21] M. Mataric. Integration of representation into goal directed behavior. *IEEE Trans. on Robotics and Automation*, 8(3):304–312, June 1992.

[22] B. Oomen, S. Iyengar, N. Rao, and R. Kashyap. Robot navigation in unknown terrains using learned visibility graphs. part i: The disjoint convex obstacle case. *IEEE Journal of Robotics and Automation*, RA-3(6):672–681, December 1987.

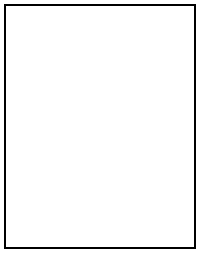
[23] N. Rao and S. Iyengar. Autonomous robot navigation in unknown terrains: Incidental learning and environmental exploration. *IEEE Systems, Man, and Cybernetics*, 20(6), November/December 1990.

[24] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202, February 1985.

[25] Z. Zheng and S. Tsuji. Panoramic representation for route recognition by a mobile robot. *International Journal of Computer Vision*, 9(1):55–76, 1992.



Camillo J. Taylor received an AB degree in electrical computer and systems engineering magna cum laude with highest honors from Harvard University in 1988. He received the MS and PhD degrees in Electrical Engineering from Yale University in 1990 and 1994 respectively. Dr. Taylor is a member of the Harvard chapter of Phi Beta Kappa and was the recipient of the Jamaica Scholarship in 1985. His research interests include recovering geometric information from image data and mobile robotics. He is currently an Assistant Professor with the Computer and Information Science Department at the University of Pennsylvania.



David J. Kriegman graduated *summa cum laude* from Princeton University with a B.S.E. degree in Electrical Engineering and Computer Science in 1983 where he was awarded the Charles Ira Young Award for electrical engineering research. He received the M.S. degree in 1984 and Ph.D. in 1989 in electrical engineering from Stanford University where he studied under a Hertz Foundation Fellowship. Currently, he is an Associate Professor at the Center for Computational Vision and Control in the Departments of Electrical Engineering and Computer Science at Yale University and was awarded a National Science Foundation Young Investigator Award in 1992. His paper on characterizing the set of images of an object under variable illumination received the best paper at the 1996 IEEE Conference on Computer Vision and Pattern Recognition. He has published over sixty papers on object representation and recognition, illumination modeling, geometry of curves and surfaces, structure from motion, robot planning and mobile robot navigation.