Performance Bounds for Planning in Unknown Terrain

Sven Koenig [a], Craig Tovey [a], and Yuri Smirnov [b]

[a] *College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280*
[b] *Akonite, 465 Fairchild Drive, Suite 232, Mountain View, CA 95043*

Abstract

Planning in nondeterministic domains is typically intractable due to the large number of contingencies. Two techniques for speeding up planning in nondeterministic domains are agent-centered search and assumption-based planning. Both techniques interleave planning in deterministic domains with plan execution. They differ in how they make planning deterministic. To determine how suboptimal their plans are, we study two planning methods for robot navigation in initially unknown terrain that have successfully been used on mobile robots but not been analyzed before. The planning methods differ both in the technique they use to speed up planning and in the robot-navigation task they solve. Greedy Mapping uses agent-centered search to map unknown terrain. Dynamic A* uses assumption-based planning to navigate to a given goal location in unknown terrain. When we formalize abstractions of these planning methods on undirected graphs $G = (V, E)$, they turn out to be similar enough that we are able to analyze their travel distance in a unified way. We discover that neither method is optimal in a worst-case sense, by a factor of $\Omega(\log |V| / \log \log |V|)$. We also derive factor $O(\sqrt{|V|})$ upper bounds to show that these methods are not very badly sub-optimal in this sense. These results provide a first step towards explaining the good empirical results that have been reported about Greedy Mapping and Dynamic A* in the experimental literature. More generally, they show how to use tools from graph theory to analyze the plan quality of practical planning methods for nondeterministic domains.

1 Introduction

A domain is nondeterministic if agents are not able to predict with certainty which successor state an action execution will result in. Planning in nondeterministic domains is time-consuming due to the large number of contingencies. Finding optimal plans is often intractable since the agents have to find large

conditional plans to solve the planning tasks. Thus, they need planning techniques that speed up planning by sacrificing the optimality of the resulting plan. In this paper, we study two such planning techniques that apply to nondeterministic domains in general [1].

One technique for speeding up planning in nondeterministic domains is *agent-centered search* [2], which restricts planning to the part of the domain around the current state of the agents (local search), for example, the current locations of mobile robots. The part of the domain around the current state of the agents is the part of the domain that is immediately relevant for them in their current situation because it contains the states that they might soon be in. Agent-centered search methods do not plan all the way from the start state to the goal state. Instead, they determine the local search space, search it, and decide which actions to execute within it. Then, they execute these actions (or only the first action) and repeat the overall process from their new state, until the planning task is solved. They can use techniques tailored to agent-centered search methods [3] or more general techniques from limited rationality and deliberation scheduling [4] to determine how much to plan. In this paper, we study a greedy approach that makes planning fast by planning only in the deterministic part of the nondeterministic domain around the current state of the agents. The agents then execute the plan, the last action of which is nondeterministic. Then, they observe the resulting state and repeat the process from the state that actually resulted from the action execution instead of all states that could have resulted from its execution, until the planning task is solved.

Another planning technique for speeding up planning in nondeterministic domains is *assumption-based planning*, which makes assumptions about the outcomes of action executions. In particular, if the agents make the domain deterministic by deleting all outcomes of each action but one, then planning is fast. The agents then execute the plan. If an action execution results in a deleted outcome, the agents repeat the process from the state that resulted from the action execution, until the planning task is solved.

The two planning techniques can be used to solve planning tasks in nondeterministic domains by interleaving planning in deterministic domains with plan executions. They differ in how they make planning deterministic. The resulting plans are likely not optimal but this is often outweighed by the computational savings gained. The question arises, then, how suboptimal the resulting plans can be. This can only be answered by analyzing the planning techniques in specific domains.

Mobile robots have recently shown impressive autonomous navigation performance in long-term demonstrations. This includes Xavier [5], Martha [6], the Remote Agent [7] and Minerva [8]. Nondeterminism can result from several

causes, for example, actuator uncertainty, sensor uncertainty, and incomplete information about the terrain. In each of these cases, mobile robots cannot predict with certainty which observations they will make after they have moved.

In this paper, we study robot navigation in initially unknown terrain, both acquiring a map (mapping) and navigating to a given goal location (goal-directed navigation). We assume that the nondeterminism results exclusively from the initially unknown terrain. This assumption is, for example, justified outdoors where localization methods (such as global positioning systems) enable mobile robots to compensate sufficiently for actuator and sensor uncertainty. Similarly, our assumption is justified for mobile robots that use robot-navigation architectures where a lower level performs local movement guided by sensory feedback and an upper level performs global navigation [9]. The lower level can then use probabilistic methods [10] to compensate for actuator and sensor uncertainty and our assumption is justified for the upper level.

We study two different planning methods that have successfully been used on mobile robots but not been analyzed before. The planning methods differ both in the technique they use to speed up planning and in the robot-navigation task they solve. Greedy Mapping (GM) [11] uses agent-centered search to map unknown terrain. Dynamic A* (D*) [12–14] uses assumption-based planning to navigate to a given goal location in unknown terrain. Each planning episode of GM and D* requires solving a shortest path problem on a graph with nonnegative edge-weights. Even with a data structure no more sophisticated than a binary heap, such a problem is solved in $O(m \log n)$ for arbitrary graphs, and $O(n \log n)$ for planar graphs [15], where n is the number of vertices and m is the number of edges. Hence, evaluating the plan-execution times and thus the travel distances of the mobile robots is crucial to determining overall performance.

We use tools from graph theory to analyze the travel distance of the mobile robots for both planning methods in a unified framework and compare it to the optimal travel distance. We show that the worst-case travel distance of GM and D* is at least on the order of $\frac{\log |V|}{\log \log |V|} |V|$ edge traversals for graphs with $|V|$ vertices, even if the graphs are planar. It had previously been reported that the travel distance of D* is good in typical mobile robot domains [14,16], but it was unknown whether this robot-navigation method is worst-case optimal, taking into account the lack of initial knowledge about the terrain. Our results establish that its worst-case travel distance is not optimal. However, it is not very badly sub-optimal either. The worst-case optimal travel distance is on the order of $|V|$ while that of GM and D* is at most on the order of $|V|^{3/2}$ edge traversals, which provides a first step towards explaining the good empirical results that have been reported about GM and D* in the experimental literature. More generally, our results show how to use tools from graph theory to analyze the plan quality of practical planning methods

for nondeterministic domains.

2 Robot Navigation in Unknown Terrain

Robot navigation in unknown terrain has been studied in both theoretical robotics and theoretical computer science [17–33]. A good overview is given in [34]. However, empirical robotics researchers have often developed their own planning methods and demonstrated them on mobile robots that solve complex real-world tasks and perform well in practice (such as museum tour guide robots indoors or autonomously navigating high-mobility multi-wheeled vehicles outdoors).

In the following, we assume for clarity that the mobile robots are omni-directional, point-sized, equipped with only a radial short-distance sensor, and capable of error-free motion and sensing. The sensors on-board the mobile robots uniquely identify their location and the adjacent locations and determine whether the adjacent locations contain obstacles. The states of robot-navigation tasks in unknown terrain then correspond to pairs of the current robot location and the currently known portion of the map. Consequently, the state space is exponential in the size of the map. As long as the mobile robot moves within the known part of the map, the state transitions are deterministic. However, when the mobile robot leaves the known part of the map, the state transitions are nondeterministic because the mobile robot cannot predict its observations and thus the resulting map updates with certainty.

Robot-navigation tasks in unknown terrain can be solved with a variety of planning methods for nondeterministic domains. For example, complete and-or graph searches always provide worst-case optimal paths but are completely intractable [1]. In this section, we provide an overview of two robot-navigation methods in unknown terrain that have been used experimentally by different research groups on different mobile robots. Both robot-navigation methods have something in common. The sensors on-board a mobile robot can typically sense the terrain only near its current location, and the mobile robot thus has to interleave planning with movement to be able to sense new parts of the terrain. As the mobile robot moves, it acquires more knowledge about the terrain and consequently reduces its uncertainty about the terrain, which also reduces the number of obstacle configurations that the planner has to consider. This is called sensor-based planning [35]. Interleaving planning and plan execution and using the sensed information for re-planning makes planning for robot navigation in unknown terrain tractable.

- **Greedy Mapping:** The mobile robot has to map an unknown terrain. Greedy Mapping (GM) [11] always moves the mobile robot from its current

location on a shortest path to a closest unvisited location, until it has visited all of the terrain that is reachable from the start location. An example of a resulting robot trajectory is given in Section 3.1. The calculation of the shortest paths can be done efficiently by re-using information from previous planning episodes. GM has been used on a nomad-class tour-guide robot that offered tours to museum visitors [36]. It has also been used indoors on Nomad 150s [11,37] and Super Scouts [38].

- **D*:** The mobile robot has to move to a given goal location in unknown terrain. Dynamic A* (D*) [12–14] always moves the mobile robot from its current location on a shortest presumed unblocked path to the goal location. A presumed unblocked path is a path that does not contain known obstacles. Thus, the mobile robot assumes that unobserved parts of the terrain are clear of obstacles (free-space assumption), an assumption that is very popular in robotics [13,39–41]. D* then traverses the path until it discovers new obstacles. It then repeats the procedure, taking into account all the obstacles that it has learned about. If the mobile robot reaches the goal location, its stops and reports success. If, at any point in time, it fails to find a presumed unblocked path from its current location to the goal location, it stops and reports that the goal location cannot be reached from the start location [12–14,16]. An example of a resulting robot trajectory is given in Section 4.1. D* Lite [42] is functionally equivalent to D* but is simpler. Both D* and D* Lite are functionally equivalent to finding shortest presumed unblocked paths with A* [43] each time new obstacles are discovered but re-calculate the paths much faster than A* by re-using information from previous planning episodes, which is why empirical robotics researchers use them [14]. D* has been used outdoors on an autonomous high-mobility multi-wheeled vehicle that navigated 1,410 meters to the goal location in an unknown area of flat terrain with sparse mounds of slag as well as trees, bushes, rocks, and debris [44]. As a result of this demonstration, D* is now widely used in the DARPA Unmanned Ground Vehicle (UGV) program, for example, on the UGV Demo II vehicles. D* is also being integrated into Mars Rover prototypes, tactical mobile robot prototypes and other military robot prototypes for urban reconnaissance [45–47]. Furthermore, it has been used indoors on Nomad 150s in robot-programming classes to reach goal locations in unknown mazes that were built with three-foot high, forty inch long cardboard walls [41,1]. Finally, the GRAMMPS planner, a mission planner for mobile robot teams, uses D* as a component [48].

Both GM and D* use fast deterministic planning methods. GM performs an agent-centered search in the deterministic part of the nondeterministic state space. Thus, GM does not plan all the way until the domain is mapped and avoids a combinational explosion this way. On the other hand, D* performs assumption-based planning since it assumes that cells with unknown blockage status are unblocked. This corresponds to making the nondeterministic state

space deterministic by assuming that all actions have their intended outcome. Thus, D* plans all the way to the goal but avoids a combinational explosion because its plans do not cover all possible contingencies.

Care must be taken to ensure that agent-centered search methods and assumption-based planning methods make progress towards a solution rather than cycling forever. GM and D* do that by guaranteeing a gain in information between plan executions. After the execution of a plan, both planning methods have either solved the planning task or increased their knowledge of the map. This is so because GM reaches the fringe of the known portion of the map when it executes the planned path and can then update the map. Similarly, D* reaches the goal location if it executes the planned path to completion. The only reason for not executing the planned path to completion is that the path unexpectedly turned out to be blocked which means that D* was able to update the map.

3 GM

In this section, we study the travel distance of GM analytically, using graph-theoretical methods. We model the terrain as graph. Vertices in the graph represent locations in the terrain. Traversing an edge in the graph corresponds to traveling from one location to an adjacent location. We study how the travel distance (measured in edge traversals) depends on the number of vertices of the graph. We use the worst-case travel distance of the mobile robot to measure the plan quality because a small worst-case travel distance guarantees that the mobile robot performs well in all terrains.

We first explain how we model the robot-navigation task of GM as a graph search task and then derive a lower bound on the worst-case travel distance of GM. The lower bound allows us compare GM to other planning methods for the same robot-navigation task.

3.1 GM: Graph

The mobile robot has to map an initially unknown finite (undirected) graph $G = (V, E)$. The mobile robot begins at some designated start vertex. When the mobile robot is at a vertex v, it learns the vertices adjacent to v (that is, the vertices connected to vertex v by an edge), and can identify these vertices when it observes them again at a later point in time. This assumption is realistic, for example, if there is no actuator noise, if the locations look sufficiently different, or if the mobile robot has a global positioning system or

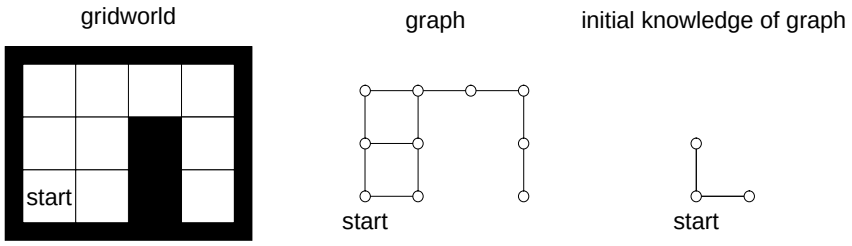

Fig. 1. Example for GM

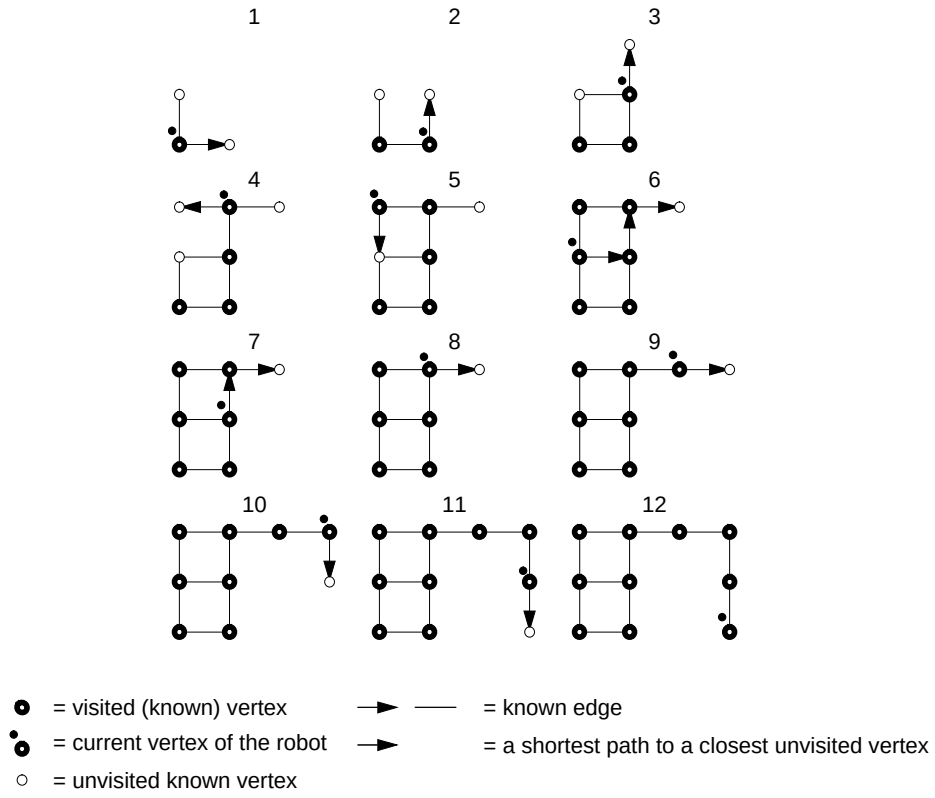

Fig. 2. Possible Trajectory of GM for the Example

some other localization method available. GM always moves the mobile robot on a shortest path from its current vertex to a closest unvisited vertex. It terminates when it knows of no unvisited vertices. GM must terminate, since each time it moves from the current vertex to the closest unvisited vertex, it visits one more vertex, and there are only a finite number of them. At termination, GM has visited all vertices that can be reached from the start location and thus has mapped the connected component of the graph that contains the start location.

Figure 1 shows a simple grid-world and the corresponding graph, assuming that the mobile robot can move to any of the four adjacent cells that are traversable. Figure 2 shows a possible trajectory of GM on the graph. GM is not constrained to working in grid-worlds but can be used on arbitrary graphs, including arbitrary maps.

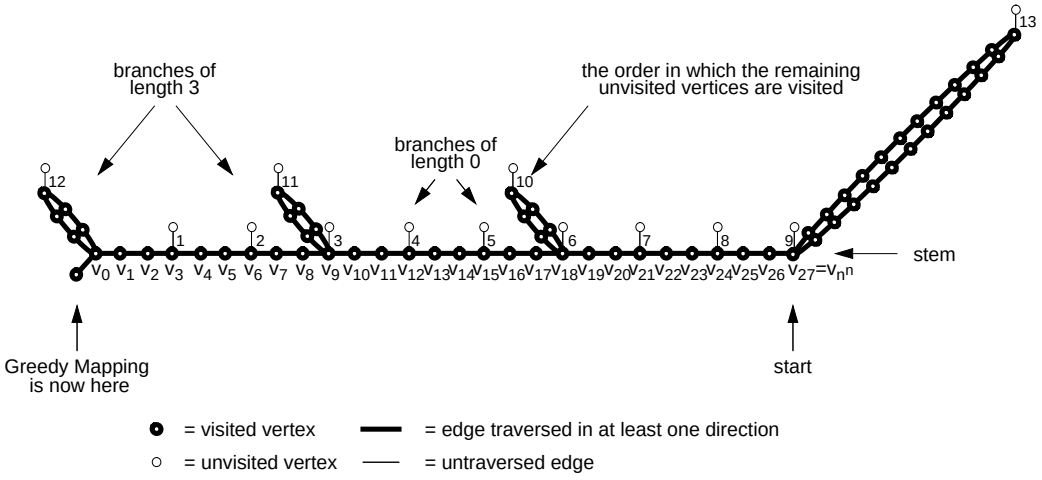

Fig. 3. Planar Graph for $n = 3$

Table 1
Branches of Graph G

number of branches	length of each branch	vertices at which the branches attach to the stem
n^{n-1}	0	$v_n, v_{2n}, v_{3n}, \ldots, v_{n^n}$
n^{n-2}	n	$v_0, v_{n^2}, v_{2n^2}, \ldots, v_{n^n-n^2}$
n^{n-3}	n^2+n	$v_{n^3}, v_{2n^3}, v_{3n^3}, \ldots, v_{n^n}$
n^{n-4}	n^3+n^2+n	$v_0, v_{n^4}, v_{2n^4}, \ldots, v_{n^n-n^4}$
$\ldots$	$\ldots$	$\ldots$

3.2 GM: Lower Bound

A lower bound on the worst-case travel distance of GM can be established by example. In this section, we present a graph $G = (V, E)$ for which the worst-case travel distance of GM is $\Omega(\frac{\log|V|}{\log\log|V|}\,|V|)$ edge traversals [49]. The graph makes GM traverse the same path repeatedly forward and backward, and this travel distance is large compared to the number of edges that are necessary to mislead GM into this trajectory. We have made our example graph planar since maps and other graphs used in robotics often have this property. Notice that the graph provides only a lower bound on the worst-case travel distance of GM as there might exist graphs that result in an even larger worst-case travel distance.

Theorem 1 *The worst-case travel distance of GM is* $\Omega(\frac{\log|V|}{\log\log|V|}\,|V|)$ *edge traversals, even on planar graphs* $G = (V, E)$.

Proof: Consider the planar graph $G = (V, E)$ shown in Figure 3, which is a variation of a graph in [50]. It consists of a stem with several branches. Each branch consists of two parallel paths of the same length that connect the stem

to a single edge. The leaves at the ends of these single edges are crucial to "fooling" GM. When the mobile robot traverses one of the parallel paths, GM might choose to return to the stem along the other path without first exploring the leaf.

We say that the length of a branch is the length of each of its two paths. The stem has length n^n for some integer $n \geq 3$ and consists of the vertices $v_0, v_1, \ldots, v_{n^n}$. For each integer i with $1 \leq i \leq n$ there are n^{n-i} branches of length $\sum_{j=1}^{i-1} n^j$ each (including branches of length zero), see Table 1. These branches attach to the stem at the vertices v_{jn^i} for integers j; if i is even, then $0 \leq j \leq n^{n-i} - 1$, otherwise $1 \leq j \leq n^{n-i}$. There is one additional single edge that attaches to vertex v_0.

The start vertex is v_{n^n}. GM can choose to break ties so as to behave as follows (the graph could be modified to enforce this trajectory no matter how ties are broken): start at vertex v_{n^n}, traverse the whole stem and all branches, but bypass all the leaves at their ends, and then traverse the additional edge attached to vertex v_0, as shown in Figure 3. At this point in time, GM again traverses the whole stem, visiting the leaves of the branches of length 0. It then switches directions and travels along the whole stem in the opposite direction, this time visiting the leaves of the branches of length n, and so forth, switching directions repeatedly. It completes its exploration when it finally visits the leaf of the longest branch.

To summarize, the leaves at the ends of the branches are tried out in the order indicated in Figure 3. The total travel distance is $\Omega(n^{n+1})$ edge traversals since the stem of length n^n is traversed $n+1$ times. To be precise, the total travel distance is $(n^{n+3} + 3n^{n+2} - 8n^{n+1} + 2n^2 - n + 3)/(n^2 - 2n + 1)$ edge traversals. It holds that $|V| = \Theta(n^n)$ since $|V| = (3n^{n+2} - 5n^{n+1} - n^n + n^{n-1} + 2n^2 - 2n + 2)/(n^2 - 2n + 1)$. This implies that $n = \Omega(\frac{\log |V|}{\log \log |V|})$ since

$$\frac{\log(n^n)}{\log \log(n^n)} = \frac{n \log n}{\log n + \log \log n} \leq \frac{n \log n}{\log n} = n.$$

It follows that the total travel distance is $\Omega(n^{n+1}) = \Omega(n\,|V|) = \Omega(\frac{\log |V|}{\log \log |V|}\,|V|)$ edge traversals. ∎

Theorem 1 shows that the worst-case travel distance of GM is super-linear in the number of vertices. However, our lower bound on the worst-case travel distance is not much worse than linear, as Figure 4 shows. We remark for future use that the graphs constructed in the proof are outer-planar, that is, planar with no interior vertices. Our graphs can be adapted to different assumptions as well. For example, we have assumed that the mobile robot can

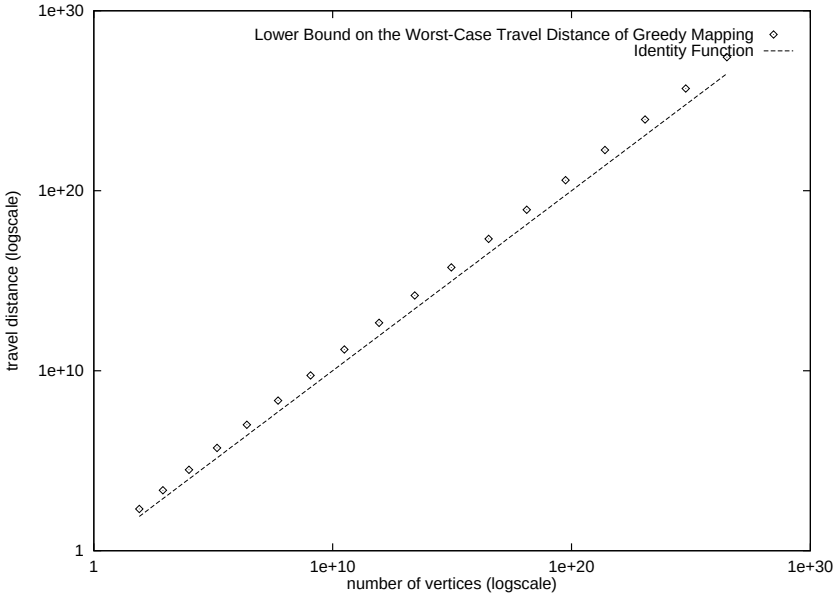

Fig. 4. Lower Bound on the Worst-Case Travel Distance of GM

identify only the vertices adjacent to its current vertex. The graph can easily be adapted to sensors with look-aheads that allow the mobile robot to identify all vertices that are at most x edge traversals away from it (this implies that the mobile robot knows the adjacent vertices of all vertices that are at most $x-1$ edge traversals away from it), by replacing each edge with x consecutive edges that are connected via $x-1$ intermediate vertices.

4 D*

In this section, we explain how we model the robot-navigation task of D* as a graph search task and then derive a lower bound on the worst-case travel distance of D*, using the lower bound on the worst-case travel distance of GM.

4.1 D: Graph*

On a finite (undirected) graph $G = (V, E)$ edge-blocked by B, the mobile robot has to reach a designated goal vertex. We call a graph $G = (V, E)$ *edge-blocked* by $B \subseteq E$ if B is the set of blocked edges (that is, edges that cannot be traversed). The mobile robot begins at some designated start vertex, knowing only the graph but not which edges are blocked. When the mobile robot is at a vertex v, it learns which edges incident to v are blocked. D* always moves the mobile robot from its current vertex along a shortest presumed unblocked path

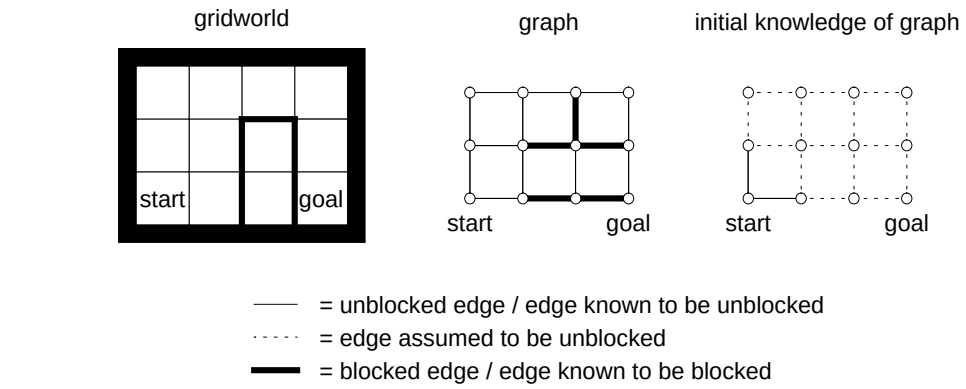

Fig. 5. Example for D*

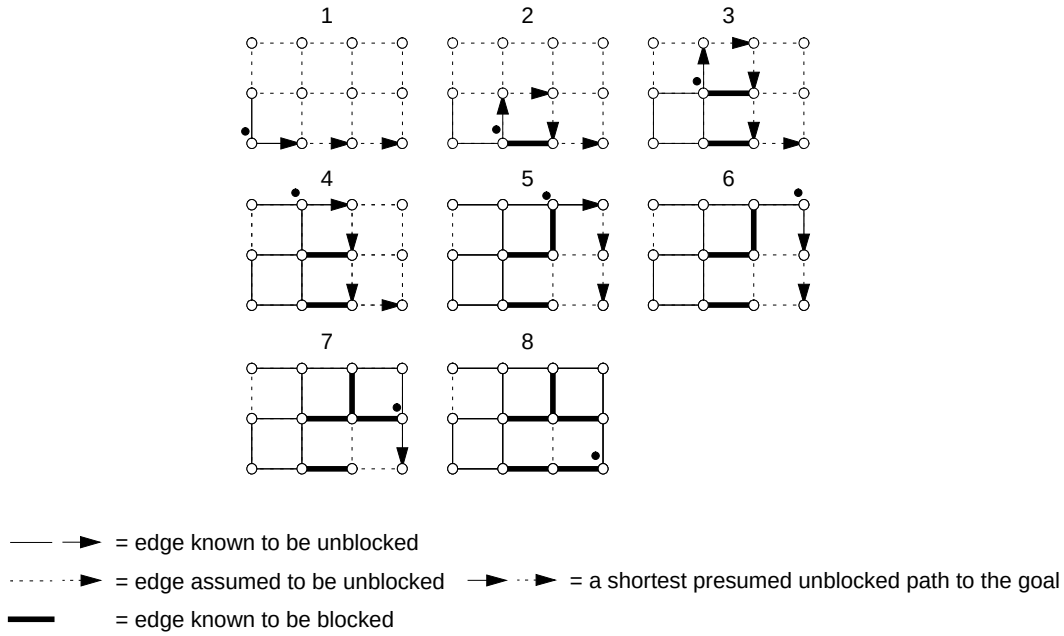

Fig. 6. Possible Trajectory of D* for the Example

to the goal vertex. A presumed unblocked path is one that contains no edges which are known to be blocked. D* terminates when the mobile robot reaches the goal vertex or there are no presumed unblocked paths to the goal vertex, in which case the goal vertex is unreachable from the start vertex since the graph is undirected. D* must terminate since it can either follow a presumed unblocked path to the goal vertex or discovers at least one blocked edge and there are only a finite number of these.

Figure 5 shows a simple grid-world and the corresponding edge-blocked graph, assuming that the mobile robot knows the boundaries of the terrain and its start and goal location, can observe the walls immediately surrounding it in the four main compass directions, and can move to each of the four adjacent cells unless a wall blocks its path. Figure 6 shows a possible trajectory of D* on the graph. Note that cells in the grid-world correspond to vertices in the

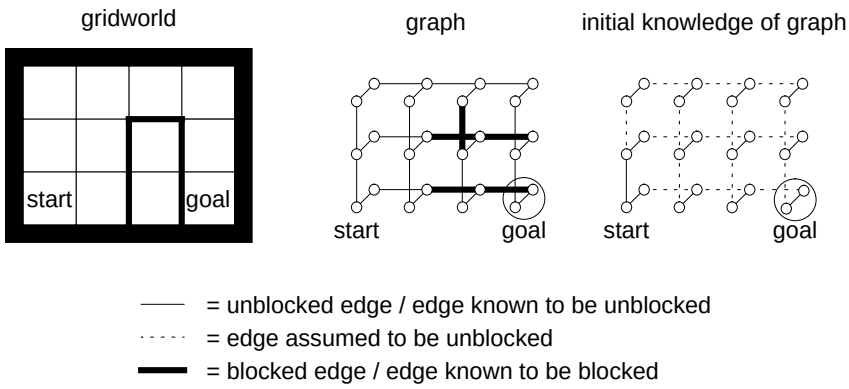

Fig. 7. Another Example for D*

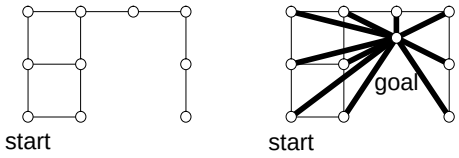

Fig. 8. Transformation of Graph for GM to Graph for D*

graph.

As an example for how to model non-omni-directional mobile robots with limited sensing direction, we consider a mobile robot that can only sense whether there are walls directly in front or behind it and then can either move forward or backward (that is, either north/south or east/west) or turn 90 degrees (say, to the left). Figure 7 shows a grid-world with the corresponding edge-blocked graph, assuming that the mobile robot starts facing north/south and that its orientation in the goal location is unimportant. In this case, the vertices no longer correspond to locations but rather to pairs of locations and orientations, and the vertices that correspond to the different orientations in the goal location can be collapsed to one goal vertex.

D* is not constrained to working in grid-worlds but can be used on arbitrary graphs, including arbitrary maps. D* on edge-blocked graphs has been used on Nomad 150s in robot-programming classes at Stanford University and Carnegie Mellon University [41,1].

4.2 D: Lower Bound*

In this section, we show that the worst-case travel distance of D* is $\Omega(\frac{\log |V'|}{\log \log |V'|} |V'|)$ edge traversals, even on planar edge-blocked graphs $G' = (V', E')$. To derive this lower bound, we demonstrate a close relationship between GM and D*.

Assume that GM operates on some graph G. Construct a super-graph G' of graph G as follows: add a new vertex to graph G and declare it to be the goal vertex. Connect the new vertex by a blocked edge to each of the existing vertices. Leave all other edges of graph G', which are also edges of graph G, as unblocked. Figure 8 gives an example of how a graph G (left) is transformed into a graph G' (right).

We argue that D* traverses exactly the same edges on graph G' that GM traverses on graph G if they start in the same vertex and break ties in the same way. Intuitively, D* knows that a blocked edge is blocked if and only if it has visited the non-goal vertex that the edge is incident to. Thus, the shortest presumed unblocked path to the goal vertex is always a shortest path to a closest unvisited vertex and from there via one edge to the goal vertex. D* traverses this path until it reaches the closest unvisited vertex, only to find out that the edge to the goal vertex is blocked. Thus, D* repeatedly moves from its current vertex to a closest unvisited vertex on graph G' and behaves like GM does on graph G. (For a given trajectory of GM on graph G, we could make the edge unblocked that connects the vertex last visited by GM to the new vertex. This would not change the trajectory of D* on graph G' but would allow the mobile robot to reach the goal vertex.) A lower bound on the worst-case travel distance of D* then follows directly from Theorem 1. The following theorem formalizes this construction.

Theorem 2 *Let $G = (V, E)$ be a graph. Then there exists a graph $G' = (V', E')$ edge-blocked by B' with $V \subset V'$, $|V'| = |V| + 1$, $|B'| = |V|$, and $E = E' - B'$, such that for each trajectory of GM on graph G there is a trajectory of D* with the same number of edge traversals.*

Proof: Construct graph G' as follows. Add a new vertex v_g to graph G, connect it to the existing vertices with one blocked edge each, and declare it the goal vertex. Thus, $V' = V \cup v_g$, $B' = \{(v, v_g) : v \in V\}$, and $E' = E \cup B'$.

Let $t = 0, 1, \ldots$ index the steps (edge traversals) of GM on graph G and D* on graph G'. Let v^t denote the vertex at which GM is after t steps and B^t the edges that it knows about at this point in time. Let $V^t = \cup_{i \leq t} v^i$ denote the vertices GM has visited at least once after t steps. Similarly, let v'^t denote the vertex at which D* is after t steps and B'^t the blocked edges that it knows about at this point in time.

We now show by induction that D* traverses exactly the same edges on graph G' that GM traverses on graph G if they start in the same vertex and break ties in the same way, that is, that there is a valid trajectory of D* on graph G' with $v'^t = v^t$ and $B'^t = \{(v, v_g) : v \in V^t\}$ for all steps t until termination. This is true for $t = 0$. Inductively assume that $v'^i = v^i$ and $B'^i = \{(v, v_g) : v \in V^i\}$ for all steps $i \leq t$. At step t, GM finds a shortest known path P on graph G

from the current vertex v^t to a vertex v not in V^t. Append the edge (v, v_g) to path P to get a path P'. We claim that path P' is a shortest presumed unblocked path from vertex v^t to vertex v_g on graph G'. Path P' is certainly a presumed unblocked path since all edges but its last one are not in B' (and thus not in B'^t) and the last edge, although blocked, is not in B'^t according to the induction assumption. Furthermore, there is no shorter presumed unblocked path from vertex v^t to vertex v_g on graph G'. If there were such a path Q', then its last edge (v', v_g) could not be in B'^t. Thus, v' could not be in V^t. Removing the last edge from Q' therefore would produce a path Q from vertex v^t to a vertex not in V^t on graph G. Path Q would be shorter than path P since path Q' was assumed to be shorter than path P'. GM knows at least a prefix of Q that ends in an unvisited vertex, that is, a vertex not in V^t. This is a contradiction since path P was assumed to be a shortest known path on graph G from vertex v^t to a vertex not in V^t.

We have now shown that D* can traverse the edge on graph G' at step t that GM traverses on graph G at the same step. Thus, $v'^{t+1} = v^{t+1}$. It follows that $V^{t+1} = V^t \cup v^{t+1}$ and $B'^{t+1} = B'^t \cup (v'^{t+1}, v_g) = B'^t \cup (v^{t+1}, v_g) = \{(v, v_g) : v \in V^{t+1}\}$.

Finally, when GM determines that it knows of no unvisited vertices of graph G and thus stops, then D* with the same history of traversed edges determines that there is no potentially traversable path from its current vertex to vertex v_g and thus stops as well. ■

Corollary 1 *The worst-case travel distance of D* on edge-blocked graphs* $G = (V, E)$ *is* $\Omega(\frac{\log |V|}{\log \log |V|} |V|)$ *edge traversals, even on planar edge-blocked graphs.*

Proof: By the proof of Theorem 1, there are graphs $G = (V, E)$ for which the worst-case travel distance of GM is $\Omega(\frac{\log |V|}{\log \log |V|} |V|)$ edge traversals. We have remarked that these graphs are outer-planar, that is, planar with all vertices on the outer face. By the addition of a single new vertex connected to all these vertices, the transformation used in the proof of Theorem 2 therefore constructs planar graphs. These are graphs $G' = (V', E')$ with $|V'| = |V| + 1$ on which D* traverses the same number of edges as GM on graph G, namely $\Omega(\frac{\log |V|}{\log \log |V|} |V|) = \Omega(\frac{\log |V'|}{\log \log |V'|} |V'|)$ edges. ■

5 D*: Upper Bounds

In this section, we continue our theoretical analysis and show that the worst-case travel distance of D* over all graphs is not very badly suboptimal. Because we use Theorem 2 in the contrapositive, the upper bounds will be derived in

the reverse order of the earlier sections: first for D* and then for GM.

Clearly, the worst-case travel distance of D* on an edge-blocked graph $G = (V, E)$ is at most $|V|^2$ edge traversals. To understand why this is so, we use the term *blockage* to mean a blocked edge. A *blockage detection* occurs when the mobile robot observes a blockage it did not know about before. When a blockage detection occurs, the mobile robot detects all blockages incident to its current vertex. The mobile robot always follows a shortest presumed unblocked path to the goal vertex until it either detects a blockage or stops. Each such path has a length of at most $|V| - 1$ edge traversals. A blockage detection can never occur twice at the same vertex, so there can be no more than $|V|$ blockage detections. Thus, the total travel distance can be no longer than $|V|$ paths of length $|V|-1$, each preceding a blockage detection, plus one more path of length $|V|$ leading to the goal vertex. Thus, the worst-case travel distance of D* on an edge-blocked graph is at most $|V|^2$ edge traversals. We now lower this quadratic upper bound by proving that the worst-case travel distance of D* is $O(|V|^{3/2})$ edge traversals. The proof of this tighter upper bound uses the same ideas of blockage detections and of conceptually separating the route of the mobile robot into paths between successive blockage detections that we just used to justify the quadratic upper bound.

Theorem 3 *The worst-case travel distance of D* on edge-blocked graphs $G = (V, E)$ is $O(|V|^{3/2})$ edge traversals.*

Proof: We call a blockage detection at which the mobile robot discovers that it cannot reach the goal vertex a *terminating* blockage detection. All other blockage detections are *nonterminating*. Let $b \leq |V|$ denote the number of nonterminating blockage detections that occur. Let c^i $(0 \leq i \leq b)$ denote the vertex at which the mobile robot is located during the ith blockage detection. (The zeroth blockage detection is considered to occur at the start vertex, just prior to the first edge traversal.) Let L_i $(1 \leq i \leq b)$ denote the number of edge traversals between the $(i-1)$st and the ith blockage detection.

The mobile robot cannot traverse more than $|V|$ edges after the bth blockage detection, because it travels on a shortest presumed unblocked path from c^b to the goal vertex and stops either at the goal vertex or earlier at the terminating blockage detection. Then, the total number of edges traversed by the mobile robot is at most

$$|V| + \sum_{i=1}^{b} L_i. \tag{1}$$

For every vertex $v \in V$, let d_v^i $(0 \leq i \leq b)$ denote the length of a shortest presumed unblocked path from vertex v to the goal vertex directly after the

ith blockage detection. That is, d_v^i is the *presumed distance* from v to the goal vertex, just after the ith blockage detection occurs. We define the value d_v^i as unchanged from the value d_v^{i-1} if D* detects that it cannot reach the goal vertex from vertex v. If the presumed distance from a vertex v to the goal vertex is infinite right from the start, we set $d_v^0 = 0$. Note that $0 \le d_v^i \le |V|-1$ and that d_v^i is nondecreasing in i.

We define $\phi^i = \sum_{v \in V} d_v^i \ge 0$. Note that ϕ^i is nondecreasing in i and $\phi^b \le |V|^2$.

The main idea behind the proof is that the mobile robot "thinks" it is getting closer to the goal vertex with each edge traversal it takes along a presumed unblocked path, but gets "disappointed" by a blockage detection which increases the presumed distances d_v to the goal. The only way for there to be many large L_i is for the mobile robot to get many big disappointments. However, big disappointments increase ϕ a lot, and there is a limit on how large ϕ can get. This forces $\sum_i L_i$ and hence the total travel distance to be small. The proof is somewhat complicated because we use a proxy Δ^i instead of counting the number of edge traversals L_i directly.

We define $\Delta^i = d_{c^i}^i - d_{c^i}^{i-1} \ge 0$ $(1 \le i \le b)$, that is, Δ^i is the amount by which the presumed distance from the mobile robot to the goal vertex increases when the mobile robot is at vertex c^i and detects the ith blockage.

Let D denote the presumed distance from the current vertex of the mobile robot to the goal vertex (the present time will be clear from the context). Consider the motion of the mobile robot from the $(i-1)$st to the ith blockage detection. The mobile robot traverses L_i edges, and D decreases by L_i during these edge traversals. Then, when the mobile robot is at vertex c^i, D increases by Δ^i. Therefore, D increases by a total of $\sum_{i=1}^b (\Delta^i - L_i)$ during the execution of D* from the start to directly after the bth blockage detection. On the other hand, $D = d_{c^0}^0$ at the start and $D = d_{c^b}^b$ directly after the bth blockage detection. Therefore, D increases by a total of $d_{c^b}^b - d_{c^0}^0$ during the execution of D* from the start to directly after the bth blockage detection. Thus, it holds that $d_{c^b}^b - d_{c^0}^0 = \sum_{i=1}^b (\Delta^i - L_i)$. Since $-|V| \le d_{c^b}^b - d_{c^0}^0 \le |V|$, it holds that $-|V| \le \sum_{i=1}^b (\Delta^i - L_i) \le |V|$ and $\sum_{i=1}^b L_i \le |V| + \sum_{i=1}^b \Delta^i$. From the bound (1), it then follows that the total number of edges traversed by the mobile robot is at most

$$|V| + \sum_{i=1}^{b} L_i \le 2|V| + \sum_{i=1}^{b} \Delta^i. \tag{2}$$

This bound is crucial since it relates Δ^i to the actual quantity of interest.

Let x_z denote any vertex at a presumed distance of z edges or less from c^i $(1 \le i \le b)$, where the distance is the length of a shortest presumed unblocked

path directly after the ith blockage detection. Then $d^i_{c^i} \leq z + d^i_{x_z}$ since, according to the triangle inequality, the presumed distance directly after the ith blockage detection from c^i to the goal vertex cannot be larger than the presumed distance from c^i via x_z to the goal vertex. Since blockage detections can only increase the presumed distances, x_z must also have been at a presumed distance of z edges or less from c^i directly after the $(i-1)$st blockage detection. Also, since the graph is undirected, the presumed distance from c^i to x_z equals the presumed distance from x_z to c^i. Then $d^{i-1}_{x_z} \leq z + d^{i-1}_{c^i}$ since, according to the triangle inequality, the presumed distance directly after the $(i-1)$st blockage detection from x_z to the goal vertex cannot be larger than the presumed distance from x_z via c^i to the goal vertex. Putting the two inequalities together, we get $d^i_{x_z} - d^{i-1}_{x_z} \geq (d^i_{c^i} - z) - (d^{i-1}_{c^i} + z) = (d^i_{c^i} - d^{i-1}_{c^i}) - 2z = \Delta^i - 2z$. Since the presumed distances $d^i_{x_z}$ are nondecreasing in i, it also holds that $d^i_{x_z} - d^{i-1}_{x_z} \geq 0$ and thus $d^i_{x_z} - d^{i-1}_{x_z} \geq \max(\Delta^i - 2z, 0)$.

For all $0 \leq z \leq d^i_{c^i}$ $(1 \leq i \leq b)$, there must be at least $z+1$ vertices at a presumed distance of z edges or less from vertex c^i directly after the ith blockage detection, since there exists a presumed unblocked path from vertex c^i to the goal vertex. Let x_z $(0 \leq z \leq d^i_{c^i})$ denote a vertex at a presumed distance of z edges or less from vertex c^i directly after the ith blockage detection, with $x_z \neq x_{z'}$ for $z \neq z'$. Note that $x_0 = c^i$. Since $\Delta^i = d^i_{c^i} - d^{i-1}_{c^i} \leq d^i_{c^i}$, it holds that

$$\begin{aligned}
\phi^i - \phi^{i-1} &\geq \sum_{z=0}^{d^i_{c^i}} (d^i_{x_z} - d^{i-1}_{x_z}) \\
&= \Delta^i + \sum_{z=1}^{d^i_{c^i}} (d^i_{x_z} - d^{i-1}_{x_z}) \\
&\geq \Delta^i + \sum_{z=1}^{d^i_{c^i}} \max(\Delta^i - 2z, 0) \\
&\geq \Delta^i + \sum_{z=1}^{\Delta^i} \max(\Delta^i - 2z, 0) \\
&\geq \frac{(\Delta^i)^2}{4}.
\end{aligned}$$

Summing over i results in

$$\sum_{i=1}^{b} \frac{(\Delta^i)^2}{4} \leq \phi^b - \phi^0 \leq |V|^2.$$

We can now bound $\sum_{i=1}^{b} \Delta^i$.

Let Δ denote the b-dimensional vector $\{\Delta^1, \ldots, \Delta^b\}$. We have just shown that $||\Delta|| \leq 2|V|$, using the usual Euclidean norm. By the Cauchy-Schwartz inequality, $\sum_{i=1}^{b} \Delta^i = \Delta \cdot 1 \leq ||\Delta||\sqrt{b} \leq 2|V|\sqrt{b} \leq 2|V|^{\frac{3}{2}}$ because $b \leq |V|$. Finally, the bound (2) implies that the number of edges traversed by the mobile robot is at most $2|V| + 2|V|^{\frac{3}{2}}$. ■

6 GM: Upper Bounds

In this section, we show that the worst-case travel distance of GM is not very badly suboptimal, making use of the corresponding upper bounds on the worst-case travel distance of D*. It is easy to show that the worst-case travel distance of GM is at most $|V|^2$ edge traversals on graphs $G = (V, E)$. The following theorem uses Theorem 2 to derive a tighter upper bound.

Corollary 2 *The worst-case travel distance of GM is $O(|V|^{3/2})$ edge traversals on graphs $G = (V, E)$.*

Proof by contradiction: Suppose that GM traverses $\omega(|V|^{3/2})$ edges on some graph $G = (V, E)$. Then, by Theorem 2, there exists a graph $G' = (V', E')$ edge-blocked by B' with $|V'| = |V| + 1$, such that D* traverses $\omega(|V|^{3/2})$ edges on G'. However, $\omega(|V|^{3/2}) = \omega(|V'|^{3/2})$, which contradicts Theorem 3. ■

7 Extensions

Our analysis can be extended in various ways. For example, we analyzed D* under the assumption that the edges can be blocked and that the mobile robot can observe the status of all edges incident to its current vertex. However, our analysis also extends to the assumption that the vertices can be blocked and that the mobile robot can observe the status of all vertices adjacent to its current vertex. In the following, we sketch how this can be done for the lower bound on the worst-case travel distance of D*.

On a finite (undirected) graph $G = (V, E)$ vertex-blocked by B, the mobile robot has to reach a designated goal vertex. We call a graph $G = (V, E)$ *vertex-blocked* by $B \subset V$ if B is the set of blocked vertices (that is, vertices that cannot be visited). The mobile robot begins at some designated start vertex, knowing only the graph but not which vertices are blocked. The start vertex is guaranteed to be unblocked. When the mobile robot is at a vertex v, it learns which vertices adjacent to v are blocked. We can use D* unchanged on vertex-blocked graphs, except that a presumed unblocked path is now one

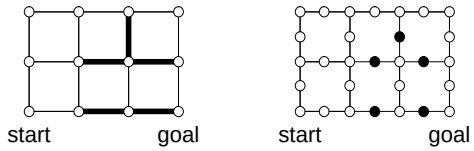

Fig. 9. Graph Transformation

that contains no vertices which are known to be blocked. D* must terminate since it can either follow a presumed unblocked path to the goal vertex or discovers at least one blocked vertex and there are only a finite number of these.

D* on vertex-blocked graphs has been used on NAVLAB II, Carnegie Mellon's unmanned high mobility multi-wheeled vehicle [44]. [1]

To determine a lower bound on the worst-case travel distance of D* on vertex-blocked graphs, we now relate it to D* on edge-blocked graphs. Assume that D* operates on some edge-blocked graph G'. We then derive a vertex-blocked graph G'' from the edge-blocked graph G' by inserting an unblocked vertex into every unblocked edge and a blocked vertex into every blocked edge. Notice that this transformation does not affect planarity. Figure 9 gives an example of how a graph G' (left) is transformed into a graph G'' (right). Dark edges or vertices are blocked. We then claim that D* traverses exactly the edges on graph G'' that correspond to the edges that it traverses on graph G' if it starts in the same vertex and breaks ties in the same way. Intuitively, since D* detects all blocked edges incident to the current vertex on graph G' and all blocked vertices adjacent to the current vertex on graph G'', the information that D* has about the blocked edges on graph G' corresponds exactly to the information that it has about the blocked vertices on graph G''. Furthermore, the transformation doubles the distances of all presumed unblocked paths. Hence, a shortest presumed unblocked path of graph G' always corresponds exactly to a shortest presumed unblocked path of graph G'', and D* behaves on graph G' and graph G'' alike if it starts in the same vertex and breaks ties in the same way. A lower bound on the worst-case travel distance of D* on vertex-blocked graphs then follows directly from Corollary 1. The following theorem formalizes this construction.

[1] This statement is a slight simplification. For example, NAVLAB II smoothed its paths by always considering a small set of navigable arcs. Furthermore, it did not only distinguish between traversable and untraversable cells, but differentiated more fine-grained by assigning traversal costs to them. Details are given in [44]. Our description is a special case of this approach that assumes an omni-directional mobile robot and distinguishes only two traversal costs, one of which is infinite. This does not affect our conclusions, because a robot-navigation method whose travel distance is not worst-case optimal for a special case is not worst-case optimal in general either.

Theorem 4 *Let $G' = (V', E')$ be a graph edge-blocked by B'. Then there exists a graph $G'' = (V'', E'')$ vertex-blocked by B'' with $V' \subseteq V''$, $|V''| = |V'| + |E'|$, $|B''| = |B'|$, and $|E''| = 2|E'|$, such that for each trajectory of D* on graph G' there is a trajectory of D* on graph G'' with the same number of edge traversals. Furthermore, G'' is planar if G' is planar.*

Proof: Construct graph G'' as follows. Create a new vertex v_{ij} for each edge $(v_i, v_j) \in E'$, replace each edge $(v_i, v_j) \in E'$ with two edges (v_i, v_{ij}) and (v_j, v_{ij}), and put $v_{ij} \in B''$ iff $(v_i, v_j) \in B'$. Thus, $B'' = \{v_{ij} : (v_i, v_j) \in B'\}$, $V'' = V' \cup \{v_{ij} : (v_i, v_j) \in E'\}$, and $E'' = \{(v_i, v_{ij}), (v_j, v_{ij}) : (v_i, v_j) \in E'\}$. The theorem then follows directly from the motivating remarks above. ■

Corollary 3 *The worst-case travel distance of D* on vertex-blocked graphs $G'' = (V'', E'')$ is $\Omega(\frac{\log |V''|}{\log \log |V''|} |V''|)$ edge traversals, even on planar vertex-blocked graphs.*

Proof: By Corollary 1, the worst-case travel distance of D* on edge-blocked graphs $G' = (V', E')$ is $\Omega(\frac{\log |V'|}{\log \log |V'|} |V'|)$ edge traversals, even on planar edge-blocked graphs. Since G' is planar, $|E'| = O(|V'|)$. By Theorem 4, there exists a planar vertex-blocked graph $G'' = (V'', E'')$ with $|V''| = |V'| + |E'| = O(|V'|)$ on which D* traverses twice the number of edges as on graph G', namely $2\Omega(\frac{\log |V'|}{\log \log |V'|} |V'|) = \Omega(\frac{\log |V''|}{\log \log |V''|} |V''|)$ edges. ■

An upper bound on the worst-case travel distance of D* on vertex-blocked graphs follows directly from the proof of Theorem 3 since the proof remains valid if we use the term blockage to mean a blocked vertex.

Theorem 5 *The worst-case travel distance of D* on vertex-blocked graphs $G = (V, E)$ is $O(|V|^{3/2})$ edge traversals.*

Proof: Identical to the proof of Theorem 3. ■

8 Discussion of the Results

In the previous sections, we derived upper and lower bounds on the worst-case travel distance of GM and D* as a function of the number of vertices of the graphs. We showed that the worst-case travel distance of GM and D* is $\Omega(\frac{\log |V|}{\log \log |V|} |V|)$ edge traversals, even on planar graphs. Thus, the worst-case travel distance grows faster than linearly in the number of vertices $|V|$.

No robot-navigation method in unknown terrain can be sure to omnisciently follow a best possible path. To establish that the travel distance of GM and

D* is not worst-case optimal, we therefore need to compare it to other robot-navigation methods that can be used in their place. Chronological backtracking is such a method. First, chronological backtracking can be used in place of GM. In this case, chronological backtracking always moves the mobile robot from its current vertex to an adjacent unvisited vertex. If such a vertex does not exist, it leaves the current vertex along the edge with which it was entered for the first time (backtracking). Chronological backtracking terminates when it knows of no unvisited vertices. Second, chronological backtracking can also be used in place of D*. In this case, chronological backtracking always moves the mobile robot from its current vertex to an adjacent unvisited vertex that can be reached via an unblocked edge. If such a vertex does not exist, it leaves the current vertex along the edge with which it was entered for the first time. It terminates when the mobile robot reaches the goal vertex, or reaches the start vertex and has already visited all adjacent unvisited vertices that can be reached via an unblocked edge, in which case the goal vertex is unreachable.

Consequently, chronological backtracking can be used both for mapping unknown terrain (and thus in place of GM) and to move to a given goal location in unknown terrain (and thus in place of D*). Its worst-case travel distance is at most twice the number of vertices since each edge traversal either visits a previously unvisited vertex (which can happen at most once for each vertex) or backtracks from a vertex (which can also happen at most once for each vertex). No uninformed robot-navigation method in unknown terrain can do significantly better than that if the sensor range of the mobile robot is very small. To see that the worst-case travel distance of robot-navigation methods that replace GM is at least twice the number of vertices minus a small constant consider star graphs, for which many edges emanate from the start vertex. The worst-case travel distance for mapping star graphs is approximately twice their number of vertices. To see that the worst-case travel distance of robot-navigation methods that replace D* is at least twice the number of vertices minus a small constant, apply the construction of Theorem 2 to star graphs. This means that chronological backtracking provides a standard that can be used to evaluate the travel distance of GM and D*. Since the worst-case travel distance of these robot-navigation methods is super-linear in the number of vertices but the worst-case travel distance of chronological backtracking is linear in the number of vertices, they are not worst-case optimal. This is a crucial and perhaps surprising insight into the behavior of these robot-navigation methods.

On the other hand, GM and D* have several very attractive features, which explains why empirical robotics researchers use them. Advantages of D*, for example, include: First, its travel distance is often small in realistic terrain, for example, terrain with sparse obstacles. Second, it improves its travel distance with experience if it solves several robot-navigation tasks in the same terrain until it follows a shortest path to the goal location. This is so because the free-

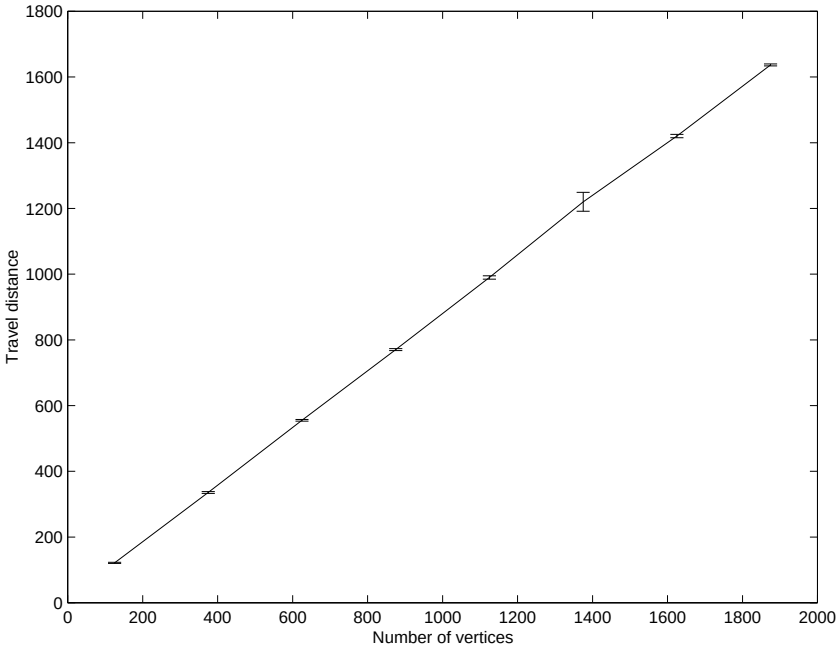

Fig. 10. Average-Case Travel Distance of GM in Mazes of Various Sizes

space assumption encourages the exploration of unknown terrain that might result in the discovery of shortcuts. Third, D* is simple and can be integrated easily into mobile robot architectures since it does not need to have control of the mobile robot at all times. For example, if a mobile robot needs to recharge its batteries, then it might have to preempt D* and move to a known power outlet. Once restarted, D* automatically resumes its operation from the power outlet, instead of returning to the location where its operation was stopped (which could be far away). Fourth, D* can very naturally take advantage of prior partial knowledge about the grid-worlds, such as blocked regions obtained from satellite images or the unblocked edges present in Figure 7), to direct planning. It also uses newly acquired knowledge to direct planning immediately. Fifth, D* can be used by several mobile robots efficiently and cooperatively by sharing their maps, without any changes to the planning method. It may be that chronological backtracking can be modified to acquire some or all of these characteristics, while retaining its other advantages [51]. However, these attempts have so far resulted in impractical robot-navigation methods. Also, if the mobile robot can sense vertices at a distance then it may be possible to map terrain with significantly fewer edge traversals than chronological backtracking.

We showed that the worst-case travel distance of GM and D* is $O(|V|^{3/2})$ edge traversals and thus not very badly suboptimal. Furthermore, this bound holds for all graphs and might be even smaller if the topology of the graphs is restricted, for example, to grids. Empirical robotics researchers are also interested in an average-case analysis of the travel distance of GM and D*. We therefore also performed experiments in maze-like grid-worlds whose corridors

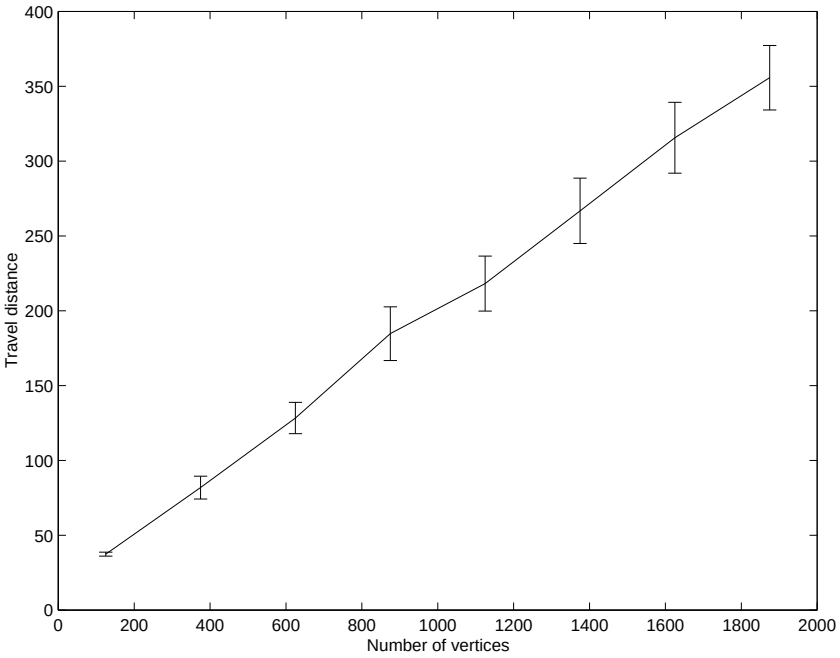

Fig. 11. Average-Case Travel Distance of D* in Mazes of Various Sizes

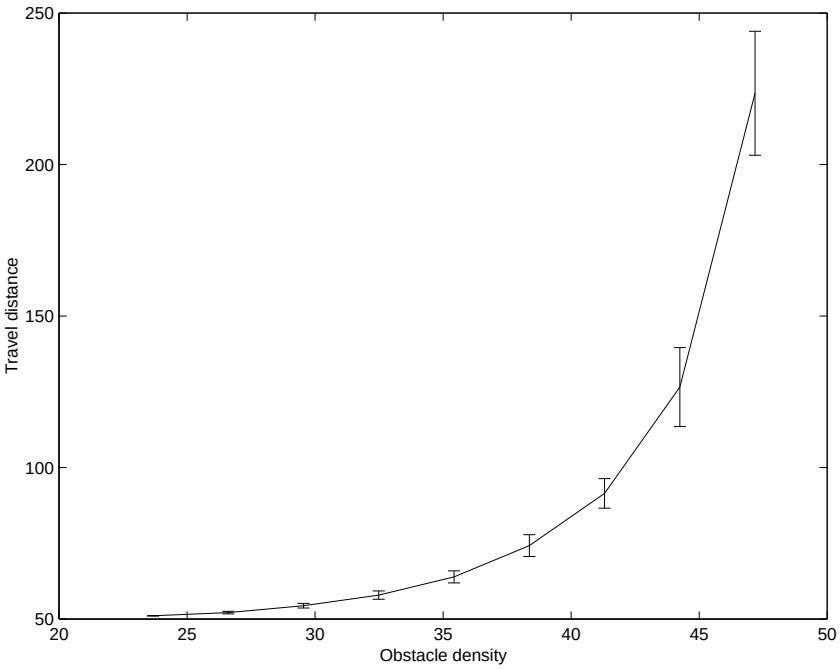

Fig. 12. Average-Case Travel Distance of D* in Mazes of Various Obstacle Densities

were created by a depth-first search. Their height was 25 cells and their width varied from 5 cells (resulting in 61.60 percent of unblocked cells) to 75 cells (resulting in 52.64 percent of unblocked cells), not counting outer walls. The mobile robot started in the upper left cell, always observed which ones of the eight adjacent cells were blocked, and could then move to any one of them

as long as the cell was unblocked. Figure 10 shows that the travel distance of GM (averaged over 20 mazes each) appears to be linear in the number of cells (vertices). The figure also shows the corresponding 95 percent confidence intervals (using a t distribution). Similarly, Figure 11 shows that the travel distance of D* (averaged over 50 mazes each) also appears to be linear in the number of cells, where D* had to move to the lower right cell of the grid-world. Thus, GM and D* are even better in these grid-worlds than our worst-case bounds suggest. Furthermore, the obstacle density is often much smaller in practice than in the mazes that we used for the experiments. Thus, the free-space assumption is often approximately satisfied and D* needs to re-plan only infrequently, which keeps its travel distance small. For example, Figure 12 illustrates how the travel distance of D* (averaged over 50 mazes each) decreases as the obstacle density decreases, using mazes of size 35×35 that were generated as before but afterwards had walls randomly removed.

9 Related Work

Most of the related work is found in the theoretical computer science literature. That work generally differs from ours in that the domains are usually a narrower class of geometrically described graphs, rather than the arbitrary labeled graphs we study here. Also, in most cases where the terrain is initially unknown, the criterion used to assess robot-navigation algorithms is a worst-case ratio with the travel distance of a hypothetical omniscient competitor (competitive ratio criterion), rather than the worst-case path length studied here. In particular, the distance traveled by the mobile robot is compared with the distance traveled by an omniscient mobile robot; the worst (largest) possible value of the ratio of these two distances, with respect to all possible scenarios, is the competitive ratio. Finally, the problems being studied are often similar to but not quite the same as the ones studied here.

The watchman tour problem is, given a map of a region, to find a tour of minimum length from which every part of the region is visible. This problem is similar to the mapping problem considered here, in that total distance traveled is the key measure of quality, but differs greatly in that the terrain and location are known. For simple polygons with n vertices, Carlsson et al. [52] find an optimal tour in time $O(n^3)$, and Chin and Ntafos [53] give an $O(n^4)$ algorithm to minimize tour length if the mobile robot is required to start at a particular location, subsequently improved to $O(n^2)$ by Tan and Hirato [54]. Other work such as [55] considers cases of limited vision.

Exploration, or mapping of unknown terrain, has also received attention in the theoretical computer science literature, but principally with respect to the competitive ratio criterion. Deng et al. [56] found the first algorithm (greedy,

but not equivalent to the algorithm studied here) with O(1) competitive ratio for rectilinear polygons; and Hoffman et al. [57] found one for the more general case of simple polygons. Mapping unweighted unoriented graphs with a minimum number of edge traversals is studied in [26,27] and references therein.

There is also a fairly extensive literature on goal-directed robot navigation, or motion planning, through both known and unknown terrain and with both long-range and short-range (tactile) sensors [58]. For known terrain, the general problem of minimizing distance traveled, often called the Piano Movers problem, is P-space complete [59] if the robot shape is not idealized as a point or space is not discretized, unlike as in our study. This problem has been studied in both discrete and continuous settings. For unknown terrain, most research focuses on the competitive ratio criterion. Blum et al. [60], for example, study motion in rectilinear polygons, and Icking et al. [61] study more restricted domains such as street polygons. The most closely related results to ours are from Lumelsky and Stepanov [62]. They consider a mobile robot in the Euclidean plane, with unknown obstacles, moving to a given goal location. The model differs from ours in some respects: it is continuous and Euclidean, not discrete; the mobile robot only uses local information plus one numerical datum, so it does not develop a partial map of the terrain as it progresses; and the boundary of the region is treated differently. The key measure is total distance traveled, as it is here. They find two provably correct algorithms, and close lower and upper bounds in terms of the sum of the Euclidean distance to the goal location and the sum of the perimeter lengths of the obstacles. There are two other differences worth noting. On the one hand, because the Lumelsky-Stepanov model uses only local information, its algorithms are more readily applied to cases where obstacles are not stationary and move randomly. On the other hand, the algorithms studied here can more readily use additional information, for example a partial map constructed from earlier experience.

10 Conclusions

In this article, we showed how to use tools from graph theory to analyze the plan quality of practical planning methods for nondeterministic domains. Planning in nondeterministic domains is typically intractable due to the large number of contingencies. Two techniques for speeding up planning in nondeterministic domains are agent-centered search and assumption-based planning. To determine how suboptimal their plans are, we studied two planning methods for robot navigation in initially unknown terrain that had not been analyzed before. Greedy Mapping (GM) is mapping with agent-centered search, and Dynamic A* (D*) is goal-directed robot navigation with assumption-based planning. Both robot-navigation methods have been demonstrated by different empirical robotics researchers on mobile robots that solve complex real-world

tasks and have performed well in practice. Both methods use heuristic knowledge to focus planning. Thus, they relate to traditional artificial intelligence search methods. D* for example, could be implemented by interleaving A* searches with plan executions and even uses underestimates of the goal distances to guide planning. Despite these similarities, the behavior of GM and D* is very different from that of traditional artificial intelligence search methods.

We related GM and D* and then analyzed their worst-case travel distances at the same time, using a graph-theoretic framework. We showed that the worst-case travel distance of GM and D* is at least on the order of $\frac{\log |V|}{\log \log |V|} |V|$ edge traversals for graphs with $|V|$ vertices, even if the graphs are planar. Thus, it is not optimal since the worst-case travel distance of chronological backtracking is only on the order of $|V|$ edge traversals. These results point out a significant difference from properties of traditional artificial intelligence search methods. A*, for example, minimizes the number of node expansions (up to tie-breaking) for consistent heuristic estimates compared to any search method that has access to the same heuristic estimates and also finds shortest paths [43]. In this sense, A* achieves its objective with minimal effort. On the other hand, we have shown that D* does not minimize the worst-case travel distance compared to other robot-navigation methods that also do not know the terrain in advance and move the mobile robot from the start location to the goal location. In this sense, D* does not achieve its objective with minimal effort. This is often misunderstood in the literature by interpreting the statement that D* is "a provably optimal and efficient path planning method for sensor-equipped robots" [12] wrongly. This statement does not refer to the travel distance of the mobile robot. It only means that D* always determines a shortest (and thus optimal) path to the goal location given the current knowledge of the mobile robot about its terrain (not taking into account that the knowledge can change during plan execution) and does so efficiently. However, it is also important to understand how good the resulting travel distance of the mobile robot and thus the plan-execution time is, since both the planning and plan-execution time together determine the overall performance of the mobile robot, which provided the motivation for our analysis.

While we have shown that the worst-case travel distance of GM and D* is not optimal, we have also shown that it is not very badly sub-optimal either. The worst-case optimal travel distance is on the order of $|V|$ edge traversals while the worst-case travel distance of GM and D* is at most on the order of $|V|^{3/2}$ edge traversals. These results are a first step towards explaining the good empirical results that have been reported about GM and D* in the experimental literature and providing a theoretical foundation for these robot-navigation methods, that have a variety of advantages over alternative robot-navigation methods.

In the future, we hope to tighten the bounds, perform average-case analyses rather than worst-case analyses, and analyze restricted classes of graphs, such as grids. We also hope to extend our results to domains other than robot-navigation domains and other planning techniques that make planning in nondeterministic domains tractable [63], including variants of probabilistic planning techniques.

Acknowledgments

Sven Koenig and Yuri Smirnov derived some of the previous results on which we are building our work [50]. Maxim Likhachev performed the experiments that we reported in Section 8. Leonard Schulman pointed out a problem in our earlier conference submission [64] that we have corrected in this article. Thanks to Illah Nourbakhsh and Anthony Stentz, two empirical robotics researchers, for their ideas and input. Thanks also to Howie Choset, Geoffrey Gordon, Andrew Moore, Reid Simmons, and Manuela Veloso for early discussions. Aditya Utturwar simplified the proof of Theorem 3 with a Cauchy-Schwartz inequality argument. This research was supported by an NSF award on "Understanding and Improving On-Line Planning Methods" to Sven Koenig and Craig Tovey under contract IIS-0098807. The Intelligent Decision-Making Group is also partly supported by NSF awards to Sven Koenig under contracts IIS-9984827 and ITR/AP-0113881 as well as an IBM faculty partnership award. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, companies or the U.S. government.

References

[1] I. Nourbakhsh, Interleaving Planning and Execution for Autonomous Robots, Kluwer Academic Publishers, 1997.

[2] S. Koenig, Agent-centered search, Artificial Intelligence Magazine 22 (4) (2001) 109–131.

[3] T. Ishida, Moving target search with intelligence, in: Proceedings of the National Conference on Artificial Intelligence, 1992, pp. 525–532.

[4] S. Russell, E. Wefald, Do the Right Thing – Studies in Limited Rationality, MIT Press, 1991.

[5] S. Koenig, R. Simmons, Xavier: A robot navigation architecture based on partially observable Markov decision process models, in: D. Kortenkamp,

R. Bonasso, R. Murphy (Eds.), Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems, MIT Press, 1998, pp. 91–122.

[6] R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert, Multi-robot cooperation in the MARTHA project, IEEE Robotics and Automation Magazine 5 (1) (1998) 36–47.

[7] N. Muscettola, P. Nayak, B. Pell, B. Williams, Remote agent: To boldly go where no AI system has gone before, Artificial Intelligence 103 (1/2) (1998) 5–47.

[8] S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, D. Schulz, MINERVA: A second generation mobile tour-guide robot, in: Proceedings of the International Conference on Robotics and Automation, 1999, pp. 1999–2005.

[9] V. Lumelsky, Algorithmic and complexity issues of robot motion in an uncertain environment, Journal of Complexity 3 (1987) 146–182.

[10] S. Thrun, Probabilistic algorithms in robotics, Artificial Intelligence Magazine 21 (4) (2000) 93–109.

[11] S. Koenig, C. Tovey, W. Halliburton, Greedy mapping of terrain, in: Proceedings of the International Conference on Robotics and Automation, 2001, pp. 3594–3599.

[12] A. Stentz, Optimal and efficient path planning for partially-known environments, in: Proceedings of the International Conference on Robotics and Automation, 1994, pp. 3310–3317.

[13] A. Stentz, The focussed D* algorithm for real-time replanning, in: Proceedings of the International Joint Conference on Artificial Intelligence, 1995, pp. 1652–1659.

[14] A. Stentz, Optimal and efficient path planning for unknown and dynamic environments, International Journal of Robotics and Automation 10 (3) (1995) 89–100.

[15] T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, MIT Press, 1990.

[16] A. Stentz, Best information planning for unknown, uncertain, and changing domains, in: S. Koenig, A. Blum, T. Ishida, R. Korf (Eds.), Proceedings of the AAAI Workshop on On-Line Search, 1997, pp. 110–113, available as AAAI Technical Report WS-97-10.

[17] H. Choset, Sensor-based motion planning: The hierarchical generalized Voronoi graph, Ph.D. thesis, California Institute of Technology, Pasadena (California) (1996).

[18] J. Leonard, H. Durrant-Whyte, I. Cox, Dynamic map building for an autonomous mobile robot, International Journal of Robotics Research 11 (4) (1992) 286–298.

[19] G. Dudek, M. Jenkin, E. Milios, D. Wilkes, Robotic exploration as graph construction, IEEE Transactions on Robotics and Automation 7 (6) (1991) 859–865.

[20] B. Kuipers, Y. Byun, A robust, qualitative method for robot spatial learning, in: Proceedings of the National Conference on Artificial Intelligence, 1988, pp. 774–779.

[21] R. de Almeida, C. Melin, Exploration of unknown environments by a mobile robot, Intelligent Autonomous Systems 2 (1989) 715–725.

[22] C. Choo, J. Smith, N. Nasrabadi, An efficient terrain acquisition algorithm for a mobile robot, in: Proceedings of the International Conference on Robotics and Automation, 1991, pp. 306–311.

[23] G. Dudek, P. Freedman, S. Hadjres, Using local information in a non-local way for mapping graph-like worlds, in: Proceedings of the International Conference on Artificial Intelligence, 1993, pp. 1639–1647.

[24] G. Oriolo, G. Ulivi, M. Vendittelli, Real-time map building and navigation for autonomous robots in unknown environments, IEEE Transactions on Systems, Man, and Cybernetics 28 (3) (1998) 316–333.

[25] N. Rao, Algorithmic framework for learned robot navigation in unknown terrains, IEEE Computer 22 (6) (1989) 37–43.

[26] S. Albers, M. Henzinger, Exploring unknown environments, SIAM Journal on Computing 29 (4) (2000) 1164–1188.

[27] X. Deng, C. Papadimitriou, Exploring an unknown graph, in: Proceedings of the Symposium on Foundations of Computer Science, 1990, pp. 355–361.

[28] S. Kwek, On a simple depth-first search strategy for exploring unknown graphs, in: Proceedings of the Workshop on Algorithms and Data Structures, Vol. 1272 of Lecture Notes in Computer Science, Springer, 1997, pp. 345 –353.

[29] N. Rao, Robot navigation in unknown generalized polygonal terrains using vision sensors, IEEE Transactions on Systems, Man, and Cybernetics 25 (6) (1995) 947–962.

[30] L. Prasad, S. Iyengar, A note on the combinatorial structure of the visibility graph in simple polygons, Theoretical Computer Science 140 (2) (1995) 249–263.

[31] B. Awerbuch, M. Betke, R. Rivest, M. Singh, Piecemeal graph exploration by a mobile robot, Information and Computation 152 (2) (1999) 155 –172.

[32] P. Panaite, A. Pelc, Exploring unknown undirected graphs, Journal of Algorithms 33 (2) (1999) 281 –295.

[33] C. Papadimitriou, M. Yannakakis, Shortest paths without a map, Theoretical Computer Science 84 (1) (1991) 127–150.

[34] N. Rao, S. Hareti, W. Shi, S. Iyengar, Robot navigation in unknown terrains: Introductory survey of non-heuristic algorithms, Tech. Rep. ORNL/TM–12410, Oak Ridge National Laboratory, Oak Ridge (Tennessee) (1993).

[35] H. Choset, J. Burdick, Sensor based planning and nonsmooth analysis, in: Proceedings of the International Conference on Robotics and Automation, 1994, pp. 3034–3041.

[36] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Hennig, T. Hofmann, M. Krell, T. Schmidt, Map learning and high-speed navigation in RHINO, in: D. Kortenkamp, R. Bonasso, R. Murphy (Eds.), Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems, MIT Press, 1998, pp. 21–52.

[37] M. Likhachev, S. Koenig, Incremental replanning for mapping, in: Proceedings of the International Conference on Intelligent Robots and Systems, 2002, pp. 667–672.

[38] L. Romero, E. Morales, E. Sucar, An exploration and navigation approach for indoor mobile robots considering sensor's perceptual limitations, in: Proceedings of the International Conference on Robotics and Automation, 2001, pp. 3092–3097.

[39] A. Zelinsky, A mobile robot exploration algorithm, IEEE Transactions on Robotics and Automation 8 (6) (1992) 707–717.

[40] G. Foux, M. Heymann, A. Bruckstein, Two-dimensional robot navigation among unknown stationary polygonal obstacles, IEEE Transactions on Robotics and Automation 9 (1) (1993) 96–102.

[41] I. Nourbakhsh, M. Genesereth, Assumptive planning and execution: a simple, working robot architecture, Autonomous Robots Journal 3 (1) (1996) 49–67.

[42] S. Koenig, M. Likhachev, D* Lite, in: Proceedings of the National Conference of Artificial Intelligence, 2002, pp. 476–483.

[43] J. Pearl, Heuristics: Intelligent Search Strategies for Computer Problem Solving, Addison-Wesley, 1985.

[44] A. Stentz, M. Hebert, A complete navigation system for goal acquisition in unknown environments, Autonomous Robots 2 (2) (1995) 127–145.

[45] M. Hebert, R. MacLachlan, P. Chang, Experiments with driving modes for urban robots, in: Proceedings of the SPIE Mobile Robots, 1999, pp. 140–149.

[46] L. Matthies, Y. Xiong, R. Hogg, D. Zhu, A. Rankin, B. Kennedy, M. Hebert, R. Maclachlan, C. Won, T. Frost, G. Sukhatme, M. McHenry, S. Goldberg, A portable, autonomous, urban reconnaissance robot, Robotics and Autonomous Systems 40 (2-3) (2002) 163–172.

[47] S. Thayer, B. Digney, M. Diaz, A. Stentz, B. Nabbe, M. Hebert, Distributed robotic mapping of extreme environments, in: Proceedings of the SPIE: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII, Vol. 4195, 2000.

[48] B. Brumitt, A. Stentz, GRAMMPS: a generalized mission planner for multiple mobile robots, in: Proceedings of the International Conference on Robotics and Automation, 1998, pp. 1564–1571.

[49] S. Koenig, Exploring unknown environments with real-time search or reinforcement learning, in: Advances in Neural Information Processing Systems 12, Vol. 11, 1999, pp. 1003–1009.

[50] S. Koenig, Y. Smirnov, Graph learning with a nearest neighbor approach, in: Proceedings of the Conference on Computational Learning Theory, 1996, pp. 19–28.

[51] I. Wagner, M. Lindenbaum, A. Bruckstein, Distributed covering by ant-robots using evaporating traces, IEEE Transactions on Robotics and Automation 15 (5) (1999) 918–933.

[52] S. Carlsson, H. Jonsson, Computing a shortest watchman path in a simple polygon in polynomial time, in: S. Akl, F. Dehne, J. Sack, N. Santoro (Eds.), Proceedings of the Workshop on Algorithms and Data Structures, Vol. 955 of Lecture Notes in Computer Science, Springer, 1995, pp. 122–134.

[53] W. Chin, S. Ntafos, Shortest watchman routes in simple polygons, Discrete and Computational Geometry 6 (1991) 9–31.

[54] X. Tan, T. Hirata, Constructing shortest watchman routes by divide-and-conquer, in: K. Ng, P. Raghavan, N. Balasubramanian, F. Chin (Eds.), Proceedings of the International Symposium on Algorithms and Computation, Vol. 762 of Lecture Notes in Computer Science, Springer, 1993, pp. 68–77.

[55] S. Ntafos, Watchman routes under limited visibility, in: Proceedings of the Canadian Conference on Computational Geometry, 1990, pp. 89–92.

[56] X. Deng, T. Kameda, C. Papadimitriou, How to learn an unknown environment I: the rectilinear case, Journal of the ACM 45 (2) (1998) 215–245.

[57] F. Hoffman, C. Icking, R. Klein, K. Kriegel, A competitive strategy for learning a polygon, in: Proceedings of the Symposium on Discrete Algorithms, 1997, pp. 166–174.

[58] V. Lumelsky, A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica 2 (1987) 403–430.

[59] J. Reif, Complexity of the mover's problem and generalizations, in: Proceedings of the Symposium on Foundations of Computer Science, 1979, pp. 421–427.

[60] A. Blum, P. Raghavan, B. Schieber, Navigating in unfamiliar geometric terrain, SIAM Journal on Computing 26 (1) (1997) 110–137.

[61] C. Icking, R. Klein, E. Langetepe, An optimal competitive strategy for walking in streets, in: C. Meinel, S. Tison (Eds.), Proceedings of the Symposium on Theoretical Aspects of Computer Science, Vol. 1563 of Lecture Notes in Computer Science, Springer, 1999, pp. 110–120.

[62] V. Lumelsky, A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica 2 (1987) 403–430.

[63] S. Koenig, R. Simmons, Real-time search in non-deterministic domains, in: Proceedings of the International Joint Conference on Artificial Intelligence, 1995, pp. 1660–1667.

[64] S. Koenig, Y. Smirnov, Sensor planning with the freespace assumption, in: Proceedings of the International Conference on Robotics and Automation, 1997, pp. 3540–3545.