SIAM J. Discrete Math.
Vol. 19, No. 2, pp. 431–447

BOUNDS ON THE TRAVEL COST OF A MARS ROVER PROTOTYPE SEARCH HEURISTIC[*]

APURVA MUDGAL[†], CRAIG TOVEY[†], SAM GREENBERG[‡], AND SVEN KOENIG[§]

Abstract. D^* is a greedy heuristic planning method that is widely used in robotics, including several Nomad class robots and the Mars rover prototype, to reach a destination in unknown terrain. We obtain nearly sharp lower and upper bounds of $\Omega(n \log n/ \log\log n)$ and $O(n \log n)$, respectively, on the worst-case total distance traveled by the robot, for the grid graphs on n vertices typically used in robotics applications. For arbitrary graphs we prove an $O(n \log^2 n)$ upper bound.

Key words. robot travel, D^*, grid graph, girth, planar graph, search heuristic, Mars rover, greedy algorithm

AMS subject classifications. 68W40, 68T20

DOI. 10.1137/S089548010444256X

1. Introduction.

D^* is a greedy heuristic planning method that is widely used to direct a robot in a terrain with initially unknown obstacles from given start to given goal coordinates. D^* always moves the robot along a shortest presumed unblocked path from its current coordinates to the goal coordinates, presuming that as-yet-unobserved portions of the terrain have no obstacles. It stops when it has reached the goal coordinates or determined that this is impossible. If movement along the current path is blocked by an obstacle, the shortest presumed unblocked path changes and D^* needs to replan. This can be implemented efficiently [11] and easily [4].

In robotics applications, the continuous terrain is usually discretized into a grid. Robot movement then corresponds to traversal from vertex to adjacent vertex in a grid graph. The graph is known in the sense that the vertices (grid cells) and edges are known. Impassable features of the terrain, which determine the graph's structure, may be known via satellite reconnaissance, prior exploration, or mapping. The graph is unknown in the sense that vertices of the graph may be blocked by debris, crevices, or other obstacles. An obstacle is not known until the robot's sensors detect it, for example, as the robot attempts to move to it.

D^* is also used in other AI applications to reach a desired goal state from an initial starting state [13, 3, 7, 14]. In these applications, and in some terrains such as buildings, the graph may be a Voronoi or other type of graph rather than a grid graph. In all of these applications the vertices can be recognized—in the case of robot movement, by the physical coordinates; in other planning problems by state identifiers.

[*]Received by the editors March 29, 2004; accepted for publication (in revised form) April 15, 2005; published electronically November 4, 2005. This research was partly supported by NSF awards under contracts IIS-9984827, IIS-0098807, and ITR/AP-0113881. 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 and agencies or the U.S. Government.

http://www.siam.org/journals/sidma/19-2/44256.html

[†]College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280 (apurva@cc.gatech.edu, ctovey@cc.gatech.edu).

[‡]School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332-0160 (SamIAm@math.gatech.edu).

[§]Computer Science Department, USC, Los Angeles, CA 90089-0781 (skoenig@usc.edu).

The D^* algorithm has some advantages over depth first search (DFS) in practice, including ease of replanning if the robot is moved to a new location, empirically good average performance, and effective use of partial terrain information [6]. D^* has been used outdoors on an autonomous high-mobility multiwheeled 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 [13]. 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 a Mars rover prototype (according to Anthony Stentz), tactical mobile robot prototypes, and other military robot prototypes for urban reconnaissance [3, 7, 14]. Furthermore, it has been used indoors on Nomad 150 mobile robots in robot-programming classes to reach a goal location in unknown mazes [9, 8]. D^* has also been used as the key method in various robot-navigation software [2, 12].

Given its simple form and many applications it would be quite interesting to know analytically how well D^* performs. The measure by which we assess performance here is the worst-case distance traveled by the robot. We focus on travel distance in the terrain rather than travel planning time because robots move so slowly that the task-completion times are completely dominated by their travel times.

For the rest of the paper, n denotes the number of vertices in the terrain graph $G = (V, E)$. In practice, D^* seems to perform reasonably well and, in many domains, exhibits a performance that is linear in n [6], i.e., the same order as DFS, but it is not known whether this is due to properties of the test terrains or whether the plan-execution times are indeed guaranteed to be good on any terrain. However, in [6] it was also shown that for arbitrary graphs the performance is $\Omega(n \log n/ \log\log n)$. Here we prove the same $\Omega(n \log n/ \log\log n)$ bound for grid graphs. The proof is a considerably modified version of the construction in [6]. This establishes that D^* has superlinear worst-case performance on the class of graphs used in real robotics applications.

The best upper bound on D^* previously known was $O(n^{3/2})$ [5]. We prove an upper bound of $O(n \log n)$ for planar graphs. This leaves only a $\log\log n$ gap, and establishes that D^* is only slightly inferior to DFS in this worst-case performance sense. As mentioned above, D^* is also employed for other applications in which the graph may not possess the grid structure. For arbitrary graphs we prove an upper bound of $O(n \log^2 n)$. Thus D^* has a rather good performance guarantee in general.

In sections 2–4 we assume that the robot has tactile (short-range) sensors. In section 5.1 we extend the results to long-range sensors. In particular, the lower bound applies to any line-of-sight sensor, and the upper bounds apply to all sensor types. In section 5.2 we extend results to the case where both vertices and edges may be blocked.

2. Definitions.

We assume that the robot is equipped with a tactile (short-distance) sensor, omni-directional, point-sized, and capable of error-free motion and sensing. The sensors on board the robot uniquely identify its location. We model the terrain as a 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 in the terrain. We are interested in the quality of the plans determined by D^* as a function of the number of vertices of the graph.

With these assumptions, we can formalize the behavior of D^* as follows. We call a graph $H = (V, E)$ vertex-blocked by $B \subset V$ if B is the set of blocked vertices, vertices that cannot be traversed. On a finite undirected graph $H = (V, E)$ vertex-blocked by

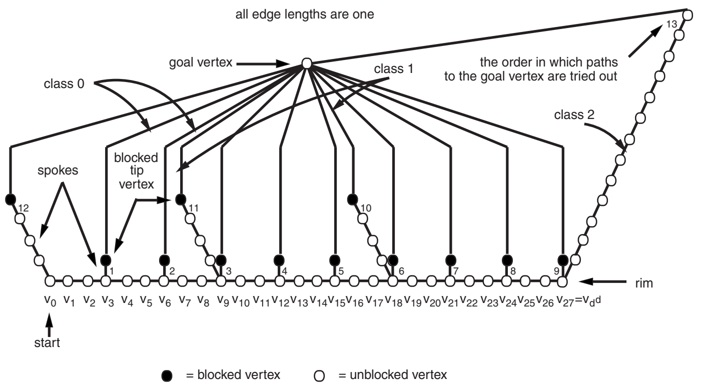
all edge lengths are one
goal vertex
class 0
class 1
the order in which paths to the goal vertex are tried out
class 2
spokes
blocked tip vertex
rim
start
v0 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18 v19 v20 v21 v22 v23 v24 v25 v26 v27=vdd
● = blocked vertex ○ = unblocked vertex

FIG. 1. *Reference* [6]*'s example graph for lower bound.*

B, a robot has to reach a designated goal vertex t from a start vertex s. D^* always moves the robot from its current vertex along a shortest presumed unblocked path to the goal vertex. A presumed unblocked path is one that contains no vertices which are known to be blocked. Initially, the robot has no information about B except that $s \notin B$. If the robot attempts to move to a blocked vertex v, it learns that $v \in B$. D^* then recomputes a new presumed unblocked path to begin the next iteration. D^* terminates when the 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. Additional notation to formalize the information state of the robot is given in section 4.1.

3. D^*: Lower bound on grids.

We now prove a lower bound on the worst-case travel distance of D^* on vertex-blocked grids. First, we review the construction of [10, 6], which employs the key idea of tricking the robot into traversing the same long path back and forth many times. Second, we give an overview of how to transform that example into a grid without losing the key idea. Third, we explain exactly how the grid is constructed. Last, we analyze the worst-case travel distance of D^* on our grid graph, proving the lower bound.

3.1. Making the robot go to and fro.[1]

The analysis of [10, 6] proved that the worst-case travel distance of D^* is $\Omega(\frac{n \log n}{\log \log n})$ steps on vertex-blocked graphs $H = (V, E)$. This lower bound is achieved with graphs of the structure shown in Figure 1. We now sketch the main idea of its construction, but with our own "rim-and-spoke" terminology, in order to introduce our much more complex grid construction.

The graph of Figure 1 consists of a long horizontal path of length d^d (where d is an integer parameter), which we call the "rim," and a set of "spokes" of varying lengths attached to the rim at various vertices. The uppermost "tip" vertex of each spoke is blocked and connected to the goal vertex by an edge. Note that the edges from the tips to the goal are physically unrealistic edges, because they allow the robot

[1]

A charming old bear at the zoo
Could always find something to do
  When tired, you know
  Of the walk to and fro
He'd reverse, and walk fro and to.

to move from any tip to the goal in one step. The possible spoke lengths are $\sum_{i=0}^{h} d^i$ for the nonnegative integers $h = 0 \cdots d-1$. We refer to a spoke of length $\sum_{i=0}^{h} d^i$ as a "class h spoke." Longer spokes are spaced farther apart from each other than short spokes. In particular, the vertices where class h spokes attach to the rim have distance d^{h+1} from each other. Hence, if the robot is at a vertex where a class h spoke attaches to the rim, then it is shorter to go to the goal along the rim to the next class h spoke than it is to go via any class $h+1$ spoke.

In particular, in Figure 1 there are three classes of spokes: 0, 1, and 2. The robot does not know that the shortest unblocked path to the goal from starting vertex v_0 is to traverse the rim to v_{27}, then the long class 2 spoke, and reach the goal vertex. Instead, the robot tries to reach the goal through the shortest presumed unblocked path via the short class 0 spoke at v_3, then the class 0 spoke at v_6, and so on until it tries the class 0 spoke at the right end of the rim, v_{27}. From there, the shortest presumed unblocked path to the goal is via the class 1 spoke at v_{18}. Thus the robot is led to traverse the rim from right to left, checking each class 1 spoke. Finally, the robot traverses the rim a third time, reaching the goal via the class 2 spoke.

In general, the robot starts at vertex v_0; it traverses the rim from left to right, checking the class 0 spokes for a path to the goal vertex; then it returns along the rim from right to left, checking class 1 spokes for a path to the goal vertex, and so on. Each class forces the robot to traverse the rim once. Thus the total travel distance is $\geq d^{d+1}$. A computation shows that there are $O(d^d)$ vertices in the graph, and hence the total travel distance is $\Omega(\frac{n \log n}{\log \log n})$.

3.2. Conceptual overview. We wish to construct a grid that captures the key idea from the previous analysis: to fool the robot into traversing a lengthy rim many times by visiting all the class h spokes before visiting any class $h+1$ spokes. However, the graph topology of the previous analysis cannot directly be embedded into a grid; the goal vertex must be simultaneously adjacent to the ends of many spokes of greatly different lengths, which moreover are placed at great distances from each other. In a grid, on the other hand, each cell is adjacent to at most four cells, and adjacent cells are physically close. We use several ideas to modify the graph topology of the previous construction to be able to embed it into a grid. A conceptual sketch of these ideas is shown in Figure 2.

1. Attach each spoke at a separate vertex to the rim (Figure 2a). This eliminates the problem of a vertex on the rim being adjacent to too many other vertices. As long as longer class spokes are spaced far enough apart, the robot is still fooled into repeatedly traversing the rim.
2. Remove the very short spokes (Figure 2b). We must place the goal vertex at some distance D from the rim, and we thus cannot construct spokes of length less than D. In particular, we only use classes $0.8d$ to $0.9d$ instead of using classes 0 to d.
3. Move the spokes physically closer together, but maintain their distances from each other along the rim. We do this by "squeezing" the rim into an accordion shape (Figure 2c). In particular, the sections of the rim between spokes get bent into long loops, which we call "columns."
4. Redesign the spokes so that they all have the same physical height, while maintaining their original lengths (Figure 2c). In particular, build a pair of blocked walls of the same height, with some space between them. Put a twisty path of the appropriate length in between the walls.
5. Once the spokes are fairly close and of equal height, bend the rim into part of

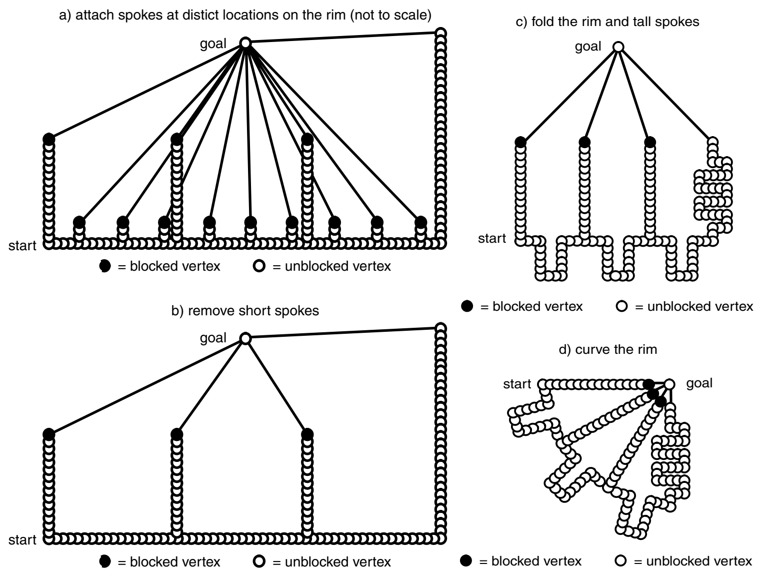
a) attach spokes at distict locations on the rim (not to scale)
goal
start
● = blocked vertex ○ = unblocked vertex
b) remove short spokes
goal
start
● = blocked vertex ○ = unblocked vertex
c) fold the rim and tall spokes
goal
start
● = blocked vertex ○ = unblocked vertex
d) curve the rim
start
goal
● = blocked vertex ○ = unblocked vertex

FIG. 2. *Steps of the transformation.*

a circular arc, pushing the tips of the spokes together towards the goal vertex (Figure 2d). It is not possible to squeeze too many distinct vertices into a small area on a grid, but this problem is solved by blocking the paths to the goal vertex a bit before the goal vertex.

3.3. Construction. Place the goal vertex g at $(0, 0)$. For some sufficiently large integer $z \equiv 0(mod\ 10)$, define the outer rim R such that

$$R := \left\{(x, y) \in \mathbb{Z}_{\geq 0} \times \mathbb{Z}_{\geq 0} : z^{0.7z} \leq x + y \leq z^{0.7z} + 1\right\}.$$

The rim is a long diagonal path from $(0, z^{0.7z})$ to about $(z^{0.7z}, 0)$. Note that the rim is two concentric quarter circles in the "taxicab" metric $\mathcal{L}_1$, so each point in R is within 1 of $z^{0.7z}$ from g. Along the rim, there will be "spoke-base points" and "column-base points" alternating. (Note: to avoid notational clutter, we omit the "floor" operation notation. Here, for example, $z^{0.7z}$ means $\lfloor z^{0.7z} \rfloor$.)

For each $i \in \{0.8z, \ldots, 0.9z\}$, create z^{z-i-1} spokes of class i. A conceptual figure is given in Figure 3. Let $S := \frac{z^{0.2z} - z^{0.1z-1}}{z-1}$ be the number of total spokes. For $i \in \{1, \ldots, S\}$, define the ith spoke-base, b_i, such that

$$b_i := \left(\frac{z^{0.7z}}{S} i + \frac{z^{0.7z}}{2S}, z^{0.7z} - \frac{z^{0.7z}}{S} i - \frac{z^{0.7z}}{2S}\right).$$

Therefore $b_i \in R$.

From each spoke-base, construct a twisty path towards g. Each path has length $2z^j$ for some $j \in \{0.8z, 0.8z + 1, \ldots, 0.9z\}$. We call such a path of length $2z^j$ a "spoke of class j." The graph will contain z^{z-j-1} spokes of class j, for each j.

The taxicab distance between two adjacent spoke bases will be $2z^{0.7z}/S$, but to make the construction's key idea work, these distances must be longer for the robot as it moves in the graph. We increase the graph distances by inserting detour loops into the rim. Lemma 3.1 makes this idea precise.

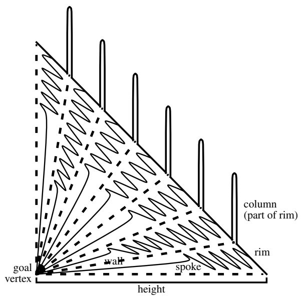

FIG. 3. *Conceptual figure with two spoke classes.*

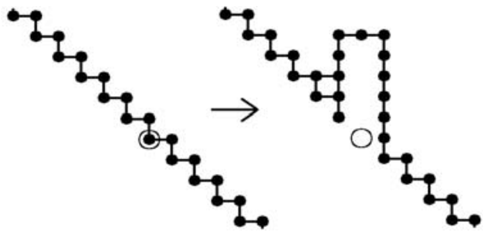

FIG. 4. *A column of height* 3.

LEMMA 3.1. *Given a function $t : \{1, \ldots, S\} \to \mathbb{Z}_{\geq 0}$ such that $t(i) - t(i-1) \geq \frac{2z^{0.7z}}{S+1}$ $\forall i$, it is possible to modify R using only cells above R in the plane, such that $\forall j > i$, traveling along R from b_i to b_j takes between $(t(j)-t(i)-4)$ and $(t(j)-t(i)+4)$ steps.*

Proof. For $i \in \{0, \ldots, S\}$, define the ith column-base, c_i such that

$$c_i := \left(\frac{z^{0.7z}}{S}i, z^{0.7z} - \frac{z^{0.7z}}{S}i\right).$$

Therefore $c_i \in R$. At each of the column-bases, remove the point itself and the point above it from the rim and add two paths traveling upwards, connected at the top. So, if the column-base is at (x, y), remove (x, y) and $(x, y + 1)$ from R and add one path from $(x - 1, y + 2)$ to $(x - 1, y + 3 + h)$ and another path from $(x + 1, y)$ to $(x + 1, y + 3 + h)$, with a connecting point at $(x, y + 3 + h)$. This increases the steps needed to cross this point in the rim by $2h$. We call such a construction a "column of height h." A column of height 3 is illustrated in Figure 4.

We now define an iterative algorithm for building the columns. For a fixed i, assume the previous columns have been built and let D be the current distance from b_1 to b_i. Let $h := \lfloor \frac{t(i)-t(1)-D}{2} \rfloor$ and build a column of height h at c_{i-1}. This ensures that the distance from b_1 to b_i is now $t(i) - t(1)$, up to round-off error. Repeat the process for all later i. (Note that the recalculation from $t(1)$ here prevents accumulation of round-off error.)

The fourth idea is to build each spoke as a twisty path of the appropriate length. Each spoke consists of a wedge of sufficient area. The spokes do not overlap, except in a small unblocked triangular region near the goal vertex, within which all paths are direct.

LEMMA 3.2. *Given a function $l : \{1, \ldots, S\} \to \mathbb{Z}_{\geq 0}$ such that $z^{0.7z} \leq l(i) \leq z^{z-1}$, $\forall i$, it is possible to construct spokes from b_i such that the distance from b_i to t is*

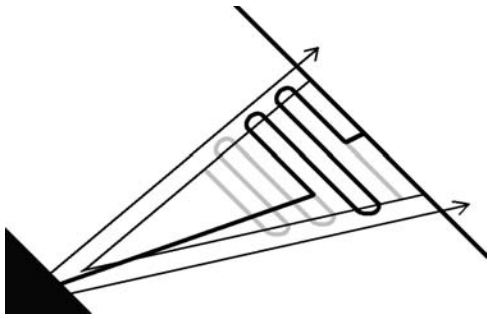

FIG. 5. *The hypothetical path P_i, along with the shortening necessary to set the path length to $l(i)$.*

between $(l(i) - 4)$ and $(l(i) + 4)$.

Proof. We would like to connect each of the spokes to g. However, the max degree of a grid prevents this. Therefore we add a triangle T such that

$$T := \left\{(x, y) : x, y \in \mathbb{Z}_{\geq 0} \bigwedge x + y \leq z^{0.5z}\right\}.$$

We may then simply connect the spokes to T. For each $i \in \{1, \dots, S\}$, define the ith "tip" vertex t_i such that

$$t_i := \left(\frac{z^{0.5z}}{S}i + \frac{z^{0.5z}}{2S}, z^{0.5z} - \frac{z^{0.5z}}{S}i - \frac{z^{0.5z}}{2S}\right).$$

Therefore $t_i \in T$. This will be the point that the ith spoke connects to.

To construct the paths of length $l(i)$ for each i, construct a hypothetical path P_i from b_i of excessive length. Then, when building the actual graph, simply include as much of P_i as necessary before the graph takes a direct path to t_i, as illustrated in Figure 5. The point where the graph ignores P_i and instead switches to a direct path to t_i depends on $l(i)$. We block the cell just prior to t_i on the path.

Building P_i takes a bit of construction. We use Euclidean rays from g to partition the space between R and g into areas A_i and then create many path segments running parallel to R called "levels." P_i runs up and down these levels, traveling back and forth to increase length. We define a space C_i to give room to connect one level to the next without coming close to the rays. This is all illustrated in Figure 5. The triangle in the lower left corner represents a region of unblocked cells, bordered by the t_i. Since all of the twisty paths have become direct paths by the time they reach their t_i, and the blockages occur prior to reaching t_i, the spokes may overlap within this unblocked region.

For each $i \in \{0, \dots, S\}$, define the ith "ray" r_i to be the Euclidean line from $(\frac{z^{0.5z}}{S}i, z^{0.5z} - \frac{z^{0.5z}}{S}i)$ to c_i. Hence r_i goes from T to R. For each $i \in \{1, \dots, S\}$, define the ith area A_i to be the integer points between r_{i-1} and r_i. Define the ith cushion C_i such that

$$C_i := \left\{(x, y) : x, y \in \mathbb{Z}_{\geq 0} \bigwedge d[((x, y), r_i)] \leq 8\right\}.$$

For each $i \in \{1, \dots, z^{0.3z} - 2\}$ and each $j \in \{1 \dots, 0.1z^{0.7z}\}$, define the level $l_{i,j}$ such that

$$l_{i,j} := \left\{(x, y) : z^{0.7z} - 6j \leq x + y \leq z^{0.7z} - 6j + 1\right\} \bigcap (A_i - C_i - C_{i+1}).$$

Use levels $\{l_{i,0}, \ldots, l_{i,0.1z^{0.7z}}\}$ to make P_i, using C_i and C_{i+1} to connect the levels. C_i is large enough to let P_i avoid crossing the ray.

The distance between c_i and c_{i+1} is $\frac{2z^{0.7z}}{S} \geq 2z^{0.3z}$. The levels are parallel and contained in a Euclidean triangle. The smallest is only one tenth of the way to the point, so each level is longer than $z^{0.3z}$. There are $0.1z^{0.7z}$ levels, so the total length of P_i is at least $0.1z^z$.

To build the actual spoke, we define a function $s(p)$ for all points $p \in P_i$, such that $s(p)$ is the distance from b_i to t if we were to shorten P_i at p and take a direct path to t_i from p. (Note that this definition involves the actual distance in the graph, avoiding accumulated round-off error.) Let S_i be the point in P_i that minimizes $|s(S_i) - l(i)|$. For any two points p, p' in the same level, if $d(p, p') = 2$, then $|s(p) - s(p')| = 2$, since $d(p, t) = d(p', t)$. Therefore, if S_i is contained in one of the levels, shortening P_i at S_i gives a spoke within 2 of $l(i)$. If S_i is contained in one of the cushions, we may be able to create an even more precise spoke. For any adjacent $c, c' \in C_i \cap P_i$, $|s(c) - s(c')| \leq 1$, as the path from c to t_i likely passes through c'. Hence, regardless of whether S_i appears in a level or in a cushion, we exceed the precision required by Lemma 3.2. ▯

To finally build our graph, define a function $p[i, j]$ such that

$$p[i, j] := z^{i+1}j + \frac{z^{0.8z+1}}{0.1z + 1}(i - 0.8z).$$

This will be the "position" of the jth spoke of class i. We order the spokes by this position function, so the first spoke is the one with the lowest position, the second is the one with the second lowest position, and so forth.

Put a blocked cell near the end of each spoke except one of class $0.9z$. Hence the robot will be tempted by each of the spokes of class $0.8z$ in turn, following the rim for about z^z steps. The robot will then turn around and travel up the rim, tempted only by the spokes of the next class $0.8z + 1$, again taking about z^z steps, and so on until it reaches the goal via the unblocked spoke of class $0.9z$.

Define the length function l such that $l[k] := 2z^i$, where i is the class of the kth spoke. We use this l with Lemma 3.1.

Define t such that $t[k] = p[i, j]$, where i and j are the coordinates for the kth spoke. We use this t with Lemma 3.2.

3.4. Analysis.

3.4. Analysis. Note: here we prove the lower bound for tactile (short-range) sensors. In section 5.1 we show that the theorem applies to all line-of-sight sensors as well.

THEOREM 3.3. *The worst-case travel distance of D^* on vertex-blocked grids $H = (V, E)$ is $\Omega(\frac{n \log n}{\log \log n})$ steps.*

Proof. The distance the robot must travel to find a spoke of class i and then travel to g is at most $z^{i+1} + 2z^i$ steps. For any $j > i$, simply traveling a spoke of class j will take at least $2z^j \geq 2z^{i+1}$ steps. Hence the robot will walk to the smallest class spoke available, find a blocked cell, and go to the next of that class, traversing the rim.

By the placement of the spokes, notice that $\forall h, h' \in \{0.8z, \ldots, 0.9z\}$, if $h < h'$, then the rightmost h-class spoke is to the right of the rightmost h'-class spoke. Also the leftmost h-class spoke is to the left of the leftmost h'-class spoke. Hence, after visiting every spoke of class i, the robot turns around and finds the first spoke of class $i + 1$. Each time the robot traverses the rim, it goes from the leftmost spoke of class

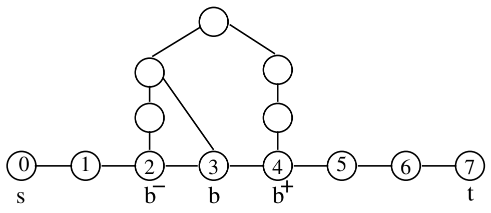

FIG. 6. *When the blockage at b of P_i is detected, $d(2,t)$ increases by at most $d(b^-,b^+)^{H^{i+1}}-2=4$.*

i to the rightmost spoke of class i. This distance is more than $z^z - 2z^{i-1} = \Omega(z^z)$. The total travel distance (just on the rim) is therefore at least $z^{0.1z}\Omega(z^z) = \Omega(z^{z+1})$.

On the other hand, there are $\theta(z^z)$ vertices in the rim (including the columns). There are $O(z^{0.5z})$ vertices in T. In class i there are $O(z^{z-i-1})$ spokes, each with $O(z^i)$ vertices, so each class contains $O(z^{z-1})$ vertices. There are $0.1z$ classes, so there are $O(z^{z-.9})$ vertices in the spokes. Therefore the total number of vertices in the graph is $\theta(z^z)$. If $n = \theta(z^z)$, then $\log n = \theta(z \log z)$ and $\log\log n = \theta(\log z)$. Then the total distance is $\Omega(z^{z+1}) = \Omega(nz) = \Omega(\frac{n \log n}{\log\log n})$. $\square$

4. D^*: Upper bounds.

4.1. Notation. As defined in section 2, the robot knows the graph $H = (V, E)$, the starting location $s \in V$ and a goal vertex $t \in V$. However, it does not know which vertices in V are blocked. D^* travels along a shortest presumed unblocked path to t. If the robot has tactile sensors, it replans whenever it encounters a blocked vertex along its currently planned path. To prepare the way for an extension to long-range sensors in the next section, we analyze here a slightly more general case. We permit the robot to detect a blocked vertex some distance ahead on its planned path. For example, in Figure 6, the robot starting from 0 might travel as far as 2 and then detect blocked vertex 6. Note that an earlier vertex such as 4 might be blocked, but go undetected at this iteration.

We assume that the initial graph $H = (V, E)$ given to the robot is connected with $n = |V|$ vertices (if not, take the component containing the starting vertex). The starting and target vertices are denoted $s, t \in V$, respectively. At the start of the ith iteration of D^*, let v_{i-1} denote the robot's location and $H^i = (V, E_i)$ denote its current information about the environment. E_i is obtained from E by removing all edges incident on vertices that have been found to be blocked. Initially $v_0 = s$ and $H^1 = H$. Let P_i denote the shortest path in H^i from v_{i-1} to t that the robot decides to follow. If i is not the final iteration, let b_i be the vertex found to be blocked by the robot while following P_i. H^{i+1} is obtained from H^i by removing edges incident on b_i. Let b_i^- and b_i^+ denote, respectively, the vertices preceding and following b_i on P_i; see Figure 6. Let v_i be the starting vertex for the next iteration. Clearly v_i either is b_i^- in P_i or precedes b_i^- in P_i and the subpath of P_i between v_i and b_i^- exists in H^{i+1}. Also, the subpath of P_i from b_i^+ to t exists in H^{i+1}.

Let $d(u,v)^H$ denote the shortest distance between vertices u and v in graph H. If u and v are not connected, then $d(u,v)^H = \infty$.

Let $v_0, v_1, \ldots, v_k$ be a run of the method. This captures a run up to ties in shortest viable paths. If the robot reaches t, then $v_k = t$. The total distance traveled by the robot is

$$C = \sum_{i=1}^{k} d(v_{i-1}, v_i)^{H^i}.$$

4.2. Telescoping.

LEMMA 4.1. $C \leq n + \sum_{i=1}^{k-1} d(b_i^-, b_i^+)^{H^{i+1}}$.

Proof. Since v_i lies on the shortest path P_i from v_{i-1} to t in H^i, by the principle of optimality

$$\begin{aligned} C = \sum_{i=1}^{k} d(v_{i-1}, v_i)^{H^i} &= \sum_{i=1}^{k} (d(v_{i-1}, t)^{H^i} - d(v_i, t)^{H^i}) \\ = d(v_0, t)^{H^1} - d(v_k, t)^{H^k} &+ \sum_{i=1}^{k-1} (d(v_i, t)^{H^{i+1}} - d(v_i, t)^{H^i}) \\ &\leq n + \sum_{i=1}^{k-1} (d(v_i, t)^{H^{i+1}} - d(v_i, t)^{H^i}). \end{aligned}$$

This formula has the following intuitive explanation: the robot optimistically thinks that undetected vertices are unblocked. When the robot gets to v_i and detects a blockage, it is set back in the distance it thinks it is from t, by the amount $(d(v_i, t)^{H^{i+1}} - d(v_i, t)^{H^i})$. The sum of these setbacks, plus the initial optimistic distance to t, equals the total distance traveled by the robot.

By the triangle inequality,

$$d(v_i, t)^{H^{i+1}} \leq d(v_i, b_i^-)^{H^{i+1}} + d(b_i^-, b_i^+)^{H^{i+1}} + d(b_i^+, t)^{H^{i+1}}. \tag{4.1}$$

By the principle of optimality, the subpath of P_i from v_i to b_i^- in H^i has length $d(v_i, b_i^-)^{H^i}$, the subpath of P_i from b_i^- to b_i^+ has length $d(b_i^-, b_i^+)^{H^i} = 2$, and the subpath of P_i from b_i^+ to t in H^i has length $d(b_i^+, t)^{H^i}$. Hence,

$$d(v_i, t)^{H^i} = d(v_i, b_i^-)^{H^i} + 2 + d(b_i^+, t)^{H^i}. \tag{4.2}$$

Observe that the first and third of these subpaths exist in H^{i+1}. Only the path of length 2 through b_i between b_i^- and b_i^+ is no longer viable in H^{i+1}. Therefore, $d(v_i, b_i^-)^{H^i} = d(v_i, b_i^-)^{H^{i+1}}$ and $d(b_i^+, t)^{H^i} = d(b_i^+, t)^{H^{i+1}}$. Plugging (4.1) and (4.2) into the bound for C above yields the lemma. ∎

In plain words, the amount of the setback when at v_i cannot be more than the revised distance $d(b_i^-, b_i^+)^{H^{i+1}} - 2$ since the robot could splice in that path to replace the blocked b_i^-, b_i, b_i^+ portion of P_i. Notice that $d(b_i^-, b_i^+)^{H^{i+1}} < \infty$ because the following pairs are all in the same connected component in H^{i+1}: v_i and b_i^-; v_i and t; b_i^+ and t.

4.3. Time reversal and weighted edges.

Define the following function:

CYCLE-WEIGHT(T, S). *Input*: a tree $T = (V, F)$ and an ordered list $S = \{e_k, e_{k-1}, \dots, e_1\}$ of distinct edges from the complete graph on V such that $S \cap F = \phi$. Define the weight w_i of edge $e_i \in S$ to be the length of a shortest cycle that contains e_i in the graph $T_i = (V, F \cup \{e_k, e_{k-1}, \dots, e_i\})$.
Output: $\sum_{i=1}^{k} w_i$.

We next show that $\sum_{i=1}^{k-1} d(b_i^-, b_i^+)^{H^{i+1}} \leq$ CYCLE-WEIGHT(T, S) for a suitably constructed tree T and $S = \{e_i = (b_i^-, b_i) : 1 \leq i \leq k-1\}$.

The basic idea relating the edge weights in CYCLE-WEIGHT to the $d(b_i^-, b_i^+)^{H^{i+1}}$ values can be understood by considering a special case. Suppose H^k is connected except for the isolated vertices $b_1, b_2, \ldots, b_{k-1}$. Reverse the time perspective so that the robot motion *adds* edges, first the edges incident on b_{k-1}, then the edges incident on b_{k-2}, and so on. Pick T to be a spanning tree of the graph $(V, E_k \cup \{(b_1, b_1^+), (b_2, b_2^+), \ldots, (b_{k-1}, b_{k-1}^+)\})$ and S to be $e_i = (b_i^-, b_i) : 1 \leq i \leq k-1$. Then $w_i \geq 2 + d(b_i^-, b_i^+)^{H^{i+1}}$ because any cycle containing (b_i^-, b_i) in T_i must also contain (b_i, b_i^+).

Unfortunately such a simple construction does not work in the general case as multiple connected components may be formed when the edges incident to a blocked vertex are removed. To get around this problem, we define a new sequence of graphs $F_k, F_{k-1}, \ldots, F_1$ as follows:

1. F_k is a spanning forest of H^k.
2. For $1 \leq i \leq k-1$, let C^i be the connected component of H^{i+1} containing b_i^+ and b_i^-. Then F_i is a spanning forest of H^i containing the subgraph $F_{i+1} \bigcup \{(b_i, b_i^+)\}$.

The following lemma follows by induction directly from the definition of F_i.

LEMMA 4.2. *For $1 \leq i \leq k$ and all vertices u and v, F_i is acyclic; $d(u,v)^{F_i} < \infty$ iff $d(u,v)^{H_i} < \infty$; and $d(u,v)^{F_i} \geq d(u,v)^{H^i}$.*

Consider the cycle weight problem with $T = F_1$ and $S = \{e_i = (b_i, b_i^-) : 1 \leq i \leq k-1\}$. The next lemma bounds the cost of our method by CYCLE-WEIGHT(T,S).

LEMMA 4.3. *Let $H^1, H^2, \ldots, H^k$ be a sequence of graphs as defined in section 4.1. Let $T = F_1$ and $S = \{e_i = (b_i^-, b_i) : 1 \leq i \leq k-1\}$. Then $\sum_{i=1}^{k-1} d(b_i^-, b_i^+)^{H^{i+1}} \leq$ CYCLE-WEIGHT(T,S).*

Proof. According to Lemma 4.2, F_{i+1} and H^{i+1} have the same connected components. The subgraph of F_1 induced by C^i is connected since C^i is a component of H^{i+1}. The edges e_j for $i < j < k$ are contained in C^i since $b_j^-, b_j, b_j^+ \in C^i$ for all $i < j < k$. Thus, the graph obtained by contracting all vertices of C^i in T_{i+1} is acyclic. Since T_i is obtained from T_{i+1} by adding e_i, every cycle that contains $e_i = (b_i^-, b_i)$ in T_i must also contain (b_i, b_i^+). Thus, w_i is equal to 2 plus the distance between b_i^- and b_i^+ in the subgraph G' of T_i induced by C^i. But G' is also a subgraph of H^{i+1} and hence it holds that $w_i \geq 2 + d(b_i^-, b_i^+)^{H^{i+1}}$. Consequently, $\sum_{i=1}^{k-1} d(b_i^-, b_i^+)^{H^{i+1}} \leq \sum_{i=1}^{k-1} w_i =$ CYCLE-WEIGHT(T, S). $\square$

4.4. An extremal problem on graphs. We now bound CYCLE-WEIGHT$((V, E), S)$ in terms of $|V|$ and $|S|$. Let $E_w = \{e_i; w_i \geq w\}$ be the set of edges with weight at least w. Recall that the *girth* of a graph is the length of its shortest cycle. Define $\Gamma(n, w)$ (respectively, $\Gamma_P(n, w)$) to denote the maximum number of edges in a graph (respectively, planar graph) with n vertices and a girth of at least w. The following lemma relates E_w and $\Gamma(n, w)$.

LEMMA 4.4. $|E_w| \leq \Gamma(|V|, w) - |V| + 1$ *for all CYCLE-WEIGHT$((V, E), S)$ and all w.*

Proof. Consider the graph $T_w = (V, E \cup E_w)$. We claim that T_w has a girth of at least w. To see this, assume that it does not and thus has a cycle C of length $w' < w$. Since (V, E) is a tree, at least one edge of C must belong to E_w. Consider the edge $e_j \in E_w \cap C$ with the smallest j. Then T_j contains C and thus $w_j \leq w' < w$. On the other hand, $w_j \geq w$ since $e_j \in E_w$, which is a contradiction. Thus, T_w has a girth of at least w. This implies that $\Gamma(|V|, w) \geq |E \cup E_w| = |E| + |E_w| = |V| - 1 + |E_w|$ and the lemma follows. ∎

COROLLARY 4.5. *$|E_w| \leq \Gamma_P(|V|, w) - |V| + 1$ for all CYCLE-WEIGHT$((V,E),S)$ such that $(V, E \cup S)$ is planar, and all w.*

Proof. In the proof of Lemma 4.4, T_w is planar because it is a subgraph of planar graph $(V, E \cup S)$. Hence $\Gamma(|V|, w)$ may be replaced by $\Gamma_P(|V|, w)$. ∎

We now bound CYCLE-WEIGHT$((V,E),S)$ by making use of bounds on $\Gamma(n, w)$, a well studied problem in extremal combinatorics. We first consider the case that the graph $(V, E \cup S)$ is planar.

LEMMA 4.6. *$\Gamma_P(n, w) \leq \frac{wn}{w-2}$ for all n and w.*

Proof. Since the sum of the lengths of all faces of any planar graph $G = (V, E)$ is at most $2|E|$ and every face has length at least w, the number of its faces can be at most $2|E|/w$. The bound of the lemma follows from substituting this relationship in Euler's formula. ∎

Note that the weight of any edge in S is at most $|V|$. Define $E_{w,2w} = \{e_i \in S : w \leq w_i < 2w\}$. Then, by Corollary 4.5 and Lemma 4.6 it holds that

$$\begin{aligned}
\text{CYCLE-WEIGHT}((V,E),S) &\leq \sum_{i=1}^{\log|V|} 2^{i+1}|E_{2^i,2^{i+1}}| \\
&\leq O(|S|) + \sum_{i=3}^{\log|V|} 2^{i+1}|E_{2^i}| \\
&\leq O(|S|) + \sum_{i=3}^{\log|V|} 2^{i+1}(\Gamma_P(|V|, 2^i) - |V| + 1) \\
&\leq O(|S|) + \sum_{i=3}^{\log|V|} 2^{i+1}\left(\frac{2^i|V|}{2^i - 2} - |V| + 1\right) \\
&\leq O(|S|) + \sum_{i=3}^{\log|V|} 2^{i+1}\, 4|V|/2^i \\
&= O(|V| \log|V|).
\end{aligned}$$

The last inequality depends on planarity (so $S = O(|V|)$) and $|V| \geq 6$. We now repeat the analysis for general graphs. In this case, we use a recent result by Alon, Hoory, and Linial [1] that states that any graph $G = (V, E)$ with average degree $d > 2$ has a girth of at most $\log_{d-1}|V|$ [1], resulting in the following lemma.

LEMMA 4.7. *$\Gamma(n, w) \leq n(n^{\frac{1}{w}} + 1)/2$ for all n and w.*

Proof. Consider any graph $G = (V, E)$ with $|V| = n$, $|E| \geq |V| + 1$ and a girth of at least w. Then, its average degree is $d = 2|E|/n > 2$ and thus, according to the result by Alon, Hoory, and Linial [1], $w \leq \log_{2|E|/n-1} n$. Solving this inequality for $|E|$ yields the lemma. ∎

This lemma allows us to bound CYCLE-WEIGHT$((V,E),S)$ for general graphs.

LEMMA 4.8. *$w(|V|(|V|^{\frac{1}{w}} - 1)) = O(|V| \log|V|)$ for $|V| \geq w > \log^2|V|$.*

Proof. Let $n = |V|$ and remove the common factor $|V|$ from the statement of the lemma. The resulting left-hand side defines the function $f(w) = w(n^{\frac{1}{w}} - 1)$. Its derivative is

$$f'(w) = n^{\frac{1}{w}}\left(1 - \frac{\ln n}{w}\right) - 1$$

and its second derivative is

$$f''(w) = \frac{n^{\frac{1}{w}} \ln^2 n}{w^3} > 0.$$

Therefore f is convex (in the range $w > 0$). Hence $\arg\max_{n \geq w \geq \log^2 n} f(w)$ occurs at one of the endpoints of the range, n or $\log^2 n$. We will show that $f(w) = O(\log n)$ for both endpoints.

At $w = n$, let $t = \frac{\ln n}{n} \to 0$ as $n \to \infty$. The Taylor series for e^t around 0 then gives

$$n^{\frac{1}{n}} - 1 = e^{\frac{\ln n}{n}} - 1 = e^t - 1 = \frac{\ln n}{n} + \frac{ln^2 n}{2n^2} + o(n^{-2}) = O\left(\frac{\log n}{n}\right).$$

Thus $f(n) = O(\log n)$.

At $w = \log^2 n$, let $t = \frac{\ln 2}{\log n}$, so

$$\frac{f(w)}{\log n} = \log n (n^{\frac{1}{\log^2 n}}) - 1 = \log n (e^{\frac{\ln n}{\log^2 n}} - 1) = \log n (e^{\frac{\ln 2}{\log n}} - 1) = \frac{\ln 2}{t}(e^t - 1).$$

Again using the Taylor series we get $\frac{f(w)}{\log n} = \ln 2(1 + \frac{t}{2} + \frac{t^2}{6} + \cdots) = \ln 2(1 + o(1)) = O(1)$. $\square$

Using Lemmata 4.8, 4.4, and 4.7, we have

$$\begin{aligned}
\text{CYCLE-WEIGHT}((V,E),S) &= \sum_{i: w_i \leq \log^2 |V|} w_i + \sum_{i: w_i > \log^2 |V|} w_i \\
&\leq |S| \log^2 |V| + \sum_{i = 2 \log\log |V|}^{\log |V|} 2^{i+1} |E_{2^i, 2^{i+1}}| \\
&\leq |S| \log^2 |V| + \sum_{i = 2 \log\log |V|}^{\log |V|} 2^{i+1} |E_{2^i}| \\
&\leq |S| \log^2 |V| + \sum_{i = 2 \log\log |V|}^{\log |V|} 2^{i+1} (\Gamma(|V|, 2^i) - |V| + 1) \\
&= |S| \log^2 |V| + \sum_{i = 2 \log\log |V|}^{\log |V|} 2^{i+1} (|V|(|V|^{\frac{1}{2^i}} - 1)/2 + 1) \\
&= |S| \log^2 |V| + \sum_{i = 2 \log\log |V|}^{\log |V|} O(|V| \log |V|) \\
&= O((|V| + |S|) \log^2 |V|).
\end{aligned}$$

We now state these results as a lemma.

LEMMA 4.9. *CYCLE-WEIGHT*$((V,E),S) = O((|V| + |S|) \log^2 |V|)$. *If the graph* $(V, E \cup S)$ *is planar, CYCLE-WEIGHT*$((V,E),S) = O(|V| \log |V|)$.

4.5. Worst-case travel bound. We are now ready to prove an upper bound on the worst-case travel distance of D^*.

THEOREM 4.10. *For robot sensors as described in section* 4.1*, D^* traverses $O(n \log^2 n)$ edges on connected graphs $G = (V, E)$. It traverses $O(n \log n)$ edges on connected planar graphs $G = (V, E)$.*

Proof. According to Lemmata 4.1 and 4.3, D^* traverses at most $O(n) + \sum_{i=1}^{k-1} d(b_i^-, b_i^+)^{H^{i+1}} \leq O(n) + \text{CYCLE-WEIGHT}((V, E'), S)$ edges, where $|S| < n$ and $(V, E' \cup S)$ is a subgraph of G. According to Lemma 4.9, it holds that $\text{CYCLE-WEIGHT}((V, E'), S) = O((n + |S|) \log^2 n) = O(n \log^2 n)$ and, if G and thus $(V, E' \cup S)$ are planar, $\text{CYCLE-WEIGHT}((V, E'), S) = O(n \log n)$. ▯

5. Extensions.

5.1. Long-range sensors. Both the lower and upper bounds of the previous sections extend to the case of long-range sensors, rather than the tactile sensors we have assumed so far. Many real robots are equipped with sonar, radar, or laser sensors, so it is worthwhile to consider this case. In directions where the view is not blocked by obstacles, these sensors can detect at moderate or even unlimited distances.

The lower bound is easy. Place a little twist in the path P_i just before the blocked vertex of each spoke, so that the blocked vertex cannot be detected until the robot is $O(1)$ vertices away. Therefore Theorem 3.3 applies to robots with long-range field-of-vision sensors.

We now extend the upper bound to the case of long-range sensors. We will not require that the sensors be field-of-vision; they may see around corners, have gaps in their vision, etc. We only require that if the robot attempts to move to vertex $v \in B$ from a vertex adjacent to v, then the robot will detect that $v \in B$. This is a minimal property required for any functioning robot.

THEOREM 5.1. *Suppose that the robot follows the D^* algorithm on graph $H = (V, E)$. Each time the robot attempts to move to an adjacent vertex, it either moves successfully or it detects that the vertex is blocked. After an attempted move (whether successful or not) the robot may detect additional blocked vertices in H. Then the bounds of Theorem* 4.10 *apply.*

Proof. Our proof consists of two parts. Part 1 shows that our bounds apply if the robot detects blocked vertices that are not on the planned path to the target. Part 2 shows that if more than one blocked vertex on the planned path is detected, then there exists a different robot whose movements are the same, but which does not detect more than one blocked vertex on the planned path.

We preface part 1 by stating the very simple ideas hidden in the technical statements. Blocked vertices off the path do not affect the telescoping formula of Lemma 4.1, because, by definition, they do not affect the current path. When we reverse time and add the special edges $e_k, \ldots, e_1$, we add extra edges (those connected to the off-path vertices). Our upper bound is on the length of a smallest cycle containing e_i, so adding extra edges can only make this smaller. Therefore the upper bound, which is computed in Lemma 4.9 as though there were no extra edges, is still valid.

Let $B_i \subset V$ denote the off-path vertices detected as blocked in iteration i. The definitions of v_i and H^i remain the same as in section 4.1, but now H^{i+1} is obtained from H^i by removing all edges incident on b_i or incident on any $b \in B_i$. Lemma 4.1 remains true in this setting because no vertices in B_i are on the path P_i. In particular, the subpaths of P_i from v_i to b_i^- and from b_i^+ to t still exist in H^{i+1}. Intuitively, the blockages B_i contribute to the setback amount suffered by the robot, but this setback is still bounded by the change in distance from b_i^- to b_i^+.

For the associated cycle weight problem, we define a sequence of forests

$F_k, F_{k-1}, \dots, F_1$. As before, F_k is a spanning forest of H_k and F_i is a spanning forest of H_i containing the subgraph $F_{i+1} \bigcup \{(b_i, b_i^+)\}$. It is easy to show that taking $T = F_1$ and $S = \{e_i = (b_i, b_i^-) : 1 \le i \le k-1\}$ satisfies Lemma 4.3. Therefore we have verified part 1.

Based on part 1, the bounds of Theorem 4.10 apply as long as the robot never detects more than one blocked vertex on the current planned path to t. For the second part of the proof, whenever the robot detects more than one such blocked vertex, categorize the detected vertices as follows:

- *off-path:* all vertices not on the current planned (shortest presumed unblocked) path to t.
- *first-path:* the nearest detected blocked vertex on the current planned path to t.
- *more-path:* all other detected blocked vertices on the current planned path to t.

Consider now a fictional robot whose movements have been identical to the real one, and which until the present step has detected the same set of blocked vertices. Now, however, our fictional robot only detects the off-path vertices and first-path vertex. It replans the shortest presumed unblocked path to t, moves zero steps, and then considers detecting the more-path vertices (more-path with respect to the original plan, not the new plan). It detects all of those which are off the newly replanned path. It can also detect one vertex on the new path, if there is one. If more than one of these are on the new path, it recategorizes them with respect to the new path and repeats the procedure.

This procedure must terminate, because each replan strictly decreases the number of more-path vertices. At termination, the fictional robot has performed precisely the same set of physical movements as has the real robot, and it has detected the same set of blocked vertices. The fictional robot has never detected more than one blocked vertex on its current planned path. The desired bounds therefore apply to both it and the real robot. $\square$

5.2. Blocked edges. Another natural extension is when in addition to blocked vertices $B \subset V$, some edges $B' \subset E$ might also be blocked. This can be reduced to the vertex blocking case by adding a new vertex v_e in the middle of every edge $e \in E$. Blocking of e then corresponds to blocking of vertex v_e in the tranformed graph. We consider two cases.

First, assume that the robot does not expend travel cost to detect an incident blocked edge. Then if the robot encounters a blocked edge (u, v) while going from u to v, it can sense all other edges emanating from u to check which ones are blocked at zero additional cost. Thus the robot will stop in at most n iterations. To bound the travel cost, let (b_i^-, b_i^+) be the edge found blocked by the robot in iteration i. Lemma 4.1 remains true in this setting as the subpaths of P_i from v_i to b_i^- and from b_i^+ to t still exist in H^{i+1}. For the associated cycle weight problem, H^{i+1} is now obtained from H^i by removing all edges found blocked by the robot in iteration i. Define the sequence $F_k, F_{k-1}, \dots, F_1$ by taking F_k a spanning forest of H^k and F_i a spanning forest of H^i containing the subgraph F_{i+1}. Similar arguments show that $T = F_1, S = \{(b_i^-, b_i^+) : 1 \le i \le k-1\}$ satisfies Lemma 4.3. By arguments for long-range sensors above, the bounds in Theorem 4.10 also hold when the robot detects a combination of blocked vertices and edges in each iteration.

Next we assume that the robot must traverse an edge in order to detect edge blockage. In this case detecting blocked e in the original graph corresponds to traveling to vertex v_e in the transformed graph. However, the number of vertices in the transformed graph is $|V| + |E|$ and Theorem 4.10 gives an $O(|E| \log |E|)$ upper bound

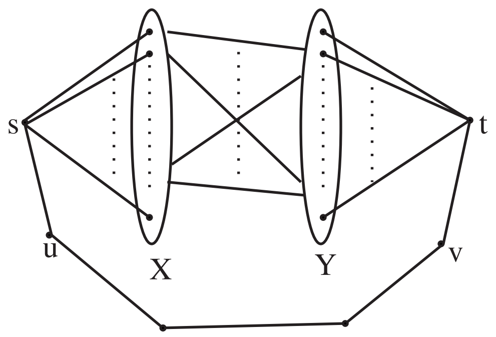

FIG. 7. *Lower bound example for blocked edges.*

for planar graphs and $O(|E| \log^2 |E|)$ upper bound for general graphs. For planar graphs this is still $O(n \log n)$ since $|E| = O(n)$. We next show a lower bound of $\Omega(|E|)$ for D^* on general graphs. Thus our bounds leave a $O(\log^2 |E|) = O(\log^2 n)$ gap.

Consider the graph $H = (\{s,t\} \bigcup X \bigcup Y, E)$ as shown in Figure 7 where $|X| = |Y| = \frac{n}{2}$. Assume that all edges $E' \subset E$ between X and Y are blocked without the knowledge of the robot. Imagine a little twist towards the end of each edge $e \in E'$, so the robot has to travel to the twist to find out whether e is blocked. Now consider running D^* with start vertex s and target vertex t. As long as there exists a "presumed unblocked" edge $(x,y) \in E'$ at the start of iteration i, the robot has a length 2 path $v_{i-1} - y - t$ or a length 4 path $v_{i-1} - s - x - y - t$ available to it. Therefore the robot will not take the length 6 path $v_{i-1} - s - u - \cdots - v - t$ until iteration $|E'| + 1$. In each preceding iteration, the robot will travel on edge (x,y) till the twist near y, find it to be blocked, and then come back to x. Therefore its travel cost is at least $\Omega(|E'|) = \Omega(|E|)$ steps on H.

6. Conclusions. The popular robot-navigation method that we have analyzed in this paper, D^*, is appealingly simple and easy to implement from a robotics point of view and appealingly complicated to analyze from a mathematical point of view. Our results, likewise, are satisfying in two ways. First, our tighter upper bounds on worst-case travel distances guarantee that D^* cannot perform badly under any circumstances. Second, the gap between the best known lower and upper bounds is now quite small, namely $O(\log \log n)$ for planar graphs, and $O(\log n \log \log n)$ on arbitrary graphs.

REFERENCES

[1] N. ALON, S. HOORY, AND N. LINIAL, *The Moore bound for irregular graphs*, Graphs Combin., 18 (2002), pp. 53–57.

[2] B. BRUMITT AND A. STENTZ, *GRAMMPS: A generalized mission planner for multiple mobile robots*, in Proceedings of the International Conference on Robotics and Automation, Leuven, Belgium, 1998.

[3] M. HEBERT, R. MCLACHLAN, AND P. CHANG, *Experiments with driving modes for urban robots*, in Proceedings of the SPIE Mobile Robots, Boston, 1999.

[4] S. KOENIG AND M. LIKHACHEV, *Improved fast replanning for robot navigation in unknown terrain*, in Proceedings of the International Conference on Robotics and Automation, Washington, D.C., 2002, pp. 968–975.

[5] S. KOENIG, C. TOVEY, AND W. HALLIBURTON, *Greedy mapping of terrain*, in Proceedings of the International Conference on Robotics and Automation, Seoul, Korea, 2001, pp. 3594–3599.

[6] S. Koenig, C. Tovey, and Y. Smirnov, *Performance bounds for planning in unknown terrain. Planning with uncertainty and incomplete information*, Artificial Intelligence, 147 (2003), pp. 253–279.

[7] L. Matthies, Y. Xiong, R. Hogg, D. Zhu, A. Rankin, B. Kennedy, M. Hebert, R. Maclachlan, C. Won, T. Frost, G. Sukhatme, M. McHenry, and S. Goldberg, *A portable, autonomous, urban reconnaissance robot*, in Proceedings of the International Conference on Intelligent Autonomous Systems, Paris, France, 2000.

[8] I. Nourbakhsh, *Interleaving Planning and Execution for Autonomous Robots*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

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

[10] Y. Smirnov, *Hybrid Algorithms for On-Line Search and Combinatorial Optimization Problems*, Tech. report CMU-CS-97-171, Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1997; available online from http://www.cs.cmu.edu/afs/cs/cmu.edu/user/smir/www/home/html.

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

[12] A. Stentz, *CD*: A real-time resolution optimal re-planner for globally constrained problems*, in Proceedings of the National Conference on Artificial Intelligence, Edmonton, Alberta, 2002, pp. 605–612.

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

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