A NEAR-TIGHT APPROXIMATION ALGORITHM FOR THE ROBOT LOCALIZATION PROBLEM

SVEN KOENIG†, JOSEPH S. B. MITCHELL‡, APURVA MUDGAL§, AND CRAIG TOVEY§

Abstract. Localization is a fundamental problem in robotics. The “kidnapped robot” possesses a compass and map of its environment; it must determine its location at a minimum cost of travel distance. The problem is NP-hard [G. Dudek, K. Romanik, and S. Whitesides, SIAM J. Comput., 27 (1998), pp. 583–604] even to minimize within factor $c \log n$ [C. Tovey and S. Koenig, Proceedings of the National Conference on Artificial Intelligence, Austin, TX, 2000, pp. 819–824], where $n$ is the map size. No approximation algorithm has been known. We give an $O(\log^3 n)$-factor algorithm. The key idea is to plan travel in a “majority-rule” map, which eliminates uncertainty and permits a link to the $\frac{1}{2}$-Group Steiner (not Group Steiner) problem. The approximation factor is not far from optimal: we prove a $c \log^{2-\epsilon} n$ lower bound, assuming $NP \not\subseteq ZTIME(n^{polylog(n)})$, for the grid graphs commonly used in practice. We also extend the algorithm to polygonal maps by discretizing the problem using novel geometric techniques.

Key words. robotics, kidnapped robot, localization, approximation algorithm, hardness of approximation, computational geometry

AMS subject classifications. 68T37, 68T40, 68W25, 68U05

DOI. 10.1137/070682885

1. Introduction. Consider the following problem: a mobile robot is placed at an unknown position in an environment for which it has a map $E$. The robot constructs a map $E'$ of its local environment by going to different places and sensing the environment from there. It rules out positions whose local environment does not agree with map $E'$ until it infers the unique position where it was originally located. The objective is to complete this task by traveling the minimum possible distance. This is known as the kidnapped robot or localization problem [13, 32].

1.1. Motivation. In general, robots must localize when they are switched on because they may have been moved while switched off. Also, the control systems guiding a robot gradually accumulate error due to mechanical drift and sensor noise [15]. Thus, it is necessary to localize from time to time to verify the actual position of the robot in the map, and then apply corrections. In this context, localization eliminates the need for complex and expensive position-guidance systems, such as radio beacons [7, 13], to be installed in buildings or streets with tall buildings, where three satellites are not in view and so GPS is not effective. For situations in which such systems cannot be built, such as a Mars rover (see [30]), localization is the only possibility.
1.2. Model. We study localization within two well-studied two-dimensional models: models based on grid graphs and models based on polygons. A grid graph $G$ is a finite rectangular region consisting of a union of unit square cells, as shown in Figure 1.1(a). Each cell can be either blocked or traversable. In the grid graph model, a robot is always in exactly one traversable cell. It starts in a traversable cell and can move in a single step to any neighboring traversable cell, to its north, south, east, or west. Tactile sensors allow the robot to determine the states (blocked/traversable) of its four neighboring cells. In the polygonal model [48, 13], the environment is a polygon $P$ and the robot occupies exactly one point $p \in P$. The robot is equipped with a range finder, a device that emanates a beam (laser or sonic) and determines distance to the first point of contact with $P$’s boundary in that direction. The robot sends out a series of beams spaced at regular angular intervals about its position, measuring the distance to the boundary at each of these angles. The points of contact are then joined together to obtain a visibility polygon $V$ (see Figure 1.1(b)). We use $n$ to denote the combinatorial size of the map: for grid graphs $n$ is the number of cells in $G$, and for a polygonal model $n$ is the number of vertices in the map polygon $P$.

Following Dudek, Roamanik, and Whitesides [17], we view the localization problem as having two phases: hypothesis generation and hypothesis elimination. The first phase is to determine the set $H$ of hypothetical locations, or hypotheses, that are consistent with the sensing data obtained by the robot from its initial location (see Figure 1.2). The second phase is to determine which location $h \in H$ is the true location of the robot. (The second phase is unnecessary if $|H| = 1$.) For the grid graph model, $H$ is simply the set of all traversable cells, and the localization problem focuses on the second phase. For the polygonal model, Guibas, Motwani, and Raghavan [23] provide an algorithm that generates the set of at most $n$ hypotheses, using the visibility polygon observed by the robot in its initial location. Thus, we focus here on the hypothesis elimination problem.

By a strategy $S$ we mean the hypothesis elimination routine employed in the robot’s computer. We measure the effectiveness of a strategy based on its worst-case performance. For strategy $S$, let $W(h, S)$ be the distance traveled to localize if the robot is placed at hypothesis $h \in H$. Then the cost $W(S)$ of strategy $S$ is defined to be the maximum distance, $W(S) = \max_{h \in H} W(h, S)$, traveled for any starting position $h$. An optimal strategy $S^*$ has cost $W(S^*) = \min_{S \in S} W(S)$, where $S$ denotes the set of all possible localization strategies. $\text{OPT}(G, H)$ denotes the cost of an optimal strategy, where $G$ is the map and $H$ is the set of hypotheses. We say that a strategy is $\alpha$-approximate if its cost is at most $\alpha \cdot \text{OPT}(G, H)$.
1.3. Previous work. Despite the considerable attention it has received in the robotics literature (e.g., [13, 32, 40, 45, 48]), localization has been the subject of relatively little theoretical work. Guibas, Motwani, and Raghavan [23] show how to preprocess the polygon \( P \) so that the set of hypotheses \( H \) consistent with a single observation \( V \) can be returned quickly. Their algorithm preprocesses \( P \) in \( O(n^5) \) time and space and generates hypotheses in \( O(m + \log n + k) \) time, where \( m \) is the number of vertices in the observed visibility polygon \( V \), and \( k = |H| \) is the number of hypotheses generated. (Note that \( k \leq n \), and, in fact, \( k \) is at most the number of reflex vertices of \( P \).)

Kleinberg [27] was the first to give interactive strategies for the hypothesis elimination problem. He measures the performance of his strategies using the competitive ratio criterion, in contrast with our worst-case criterion. The competitive ratio compares the distance traveled by a robot following a strategy to that traveled by an omniscient verifier, i.e., a robot that has a priori knowledge of its position \( h \in H \) and probes the environment just to verify this information. The distance traveled by an omniscient verifier at hypothesis \( h \) is exactly \( \min_{S \in S} W(h, S) \), and an \( \alpha \)-competitive strategy enables a robot initially located at hypothesis \( h \) to travel distance at most \( \alpha \cdot \min_{S \in S} W(h, S) \) prior to completing localization.

In Kleinberg’s model the environment is a geometric tree, \( G(V, E) \), where \( V \) is a set of points in \( \mathbb{R}^d \) and \( E \) is a set of line segments whose endpoints all lie in \( V \). The edges do not intersect except at \( V \) and do not form cycles. The robot occupies a point on one of the edges and is capable of moving along an edge in either direction. Kleinberg further assumes that the only information available to the robot is the orientation of all edges incident at its current position \( p \in E \). He gives an \( O(n^{2/3}) \)-competitive algorithm on geometric trees having bounded degree, and he gives an \( \Omega(\sqrt{n}) \) lower bound. He also gives an \( O(n \sqrt{\log \log n}) \)-competitive algorithm for a geometric model consisting of a packing of rectangles (obstacles) in the plane, with no two rectangles “stuck together” (i.e., two rectangles can nearly touch, but there remains a traversable gap between them) and each rectangle having at least unit width. In section 5.4, we give an \( O(\log^3 n) \)-approximate strategy not just for geometric trees, but for geometric graphs in any Euclidean space \( \mathbb{R}^d \).

Dudek, Romanik, and Whitesides [17] consider the problem of designing competitive strategies for the polygonal model; however, they assume that the robot can compute only the visibility skeleton \( V^*(p) \), which is an approximation of visibility polygon \( V(p) \). The visibility skeleton \( V^*(p) \) (see [23]) is a contraction of \( V(p) \), consisting of only those vertices in \( V(p) \) that can be certified to be vertices of \( P \). For this model, they give a greedy \( 2(k - 1) \)-competitive strategy minimum distance localiza-
tion (MDL) for hypothesis elimination, where $k = |H|$ is the number of hypotheses. They also show that there are polygons $P$ and sets of hypotheses $H$ for which the best strategy is $2(k - 1)$-competitive. We believe that this line of work stands closest to ours in both geometric and algorithmic structure. We refer the reader to the bibliographic note at the end of section 3 for a discussion of the recent work on this strategy as well as a comparison with our results.

Dudek, Romanik, and Whitesides were also the first to study the localization problem from the worst-case perspective, which they describe as the height of a localizing decision tree. They prove that computing an optimal localizing decision tree (i.e., an optimal worst-case strategy) is NP-hard by a reduction from the abstract decision tree problem [25]. Tovey and Koenig [46] show that it is NP-hard even to find a $c \cdot \log n$-approximate strategy, both for grid graphs and for polygons, using a reduction from the set cover problem [28]. Schuierer [44] proposes a technique that uses geometric overlay trees to reduce the running time of Dudek, Romanik, and Whitesides greedy strategy. His technique, along with a careful choice of data structures, allows the robot to localize in computation time $O(kn \log n)$ and space $O(kn)$.

Brown and Donald [5] describe algorithms for localization that allow for uncertainty in the measurements of range sensors. Fox, Burgard, and Thrun [21] use Markov localization to deduce the position of the robot from sensor data. In their work, global localization is achieved as a side effect of robot movement, and the length of the localizing trajectory relative to the optimum is not considered. In Markov localization and related approaches, localization and action are viewed in a compound setting; the effects of various actions are interpreted probabilistically and the robot is able to predict the belief states ensuing from various actions. Long-range path planning using these approaches remains problematic because of the large state space involved.

The motivation for competitive algorithms comes from theoretical work of a similar flavor in robot navigation in unknown environments. The objective of the robot is to navigate from a point $s$ to a target $t$ while avoiding obstacles/walls in the scene, which are not known to the robot a priori, but which the robot learns by encountering them. The goal is seek to minimize the competitive ratio of the distance traveled by the robot to the length of the shortest obstacle-free path from $s$ to $t$. Papadimitriou and Yannakakis [37] gave the first such results, achieving a competitive ratio of 1.5 (which they show is the best possible) in the case that obstacles are unit squares. They, along with Eades, Lin, and Wormald [18] also give a lower bound of $\Omega(\sqrt{n})$ on the competitive ratio in the case that $t$ is an infinite wall and the obstacles are axis-aligned rectangles. Baeza-Yates, Culberson, and Rawlins [1] introduce the technique of spiral search, with which they obtain a $(9 + o(1))$-competitive algorithm for finding a point on a line and a 13.81-competitive algorithm to search for a line at distance $n$ from the origin. A restricted spiral search in a geometric tree forms the first part of Kleinberg’s localization algorithm. Blum, Raghavan, and Schieber [6] use a variant of the spiral search technique to give a strategy that matches the $\Omega(\sqrt{n})$ lower bound for navigating between two points among axis-aligned rectangular obstacles. The navigation problem has also been studied in the polygonal model, for which Klein [26] gives a lower bound of $\sqrt{2}$ on the competitive ratio and gives a 5.72-competitive algorithm for a subclass known as street polygons. Later, Kleinberg [29] improved the ratio to $2\sqrt{2}$, and Datta and Icking [14] gave a 9.06-competitive algorithm for the broader class of generalized streets.

There are advantages to considering worst-case cost over the competitive ratio for the localization problem. In online navigation problems, the map is not known, and hence the informational assumption of competitive analysis holds for the robot.
But in the localization problem the map is given a priori to the robot. Hence the information available to the robot is precisely what is needed for standard worst-case analysis. A competitive analysis assumes there is too little information available to the robot, and too much to the omniscient verifier, than is realistic. From a practical standpoint, worst-case analysis better matches the roboticist’s concerns with guaranteed rapid localization, rather than with comparisons against a nonexistent omniscient verifier. From a theoretical standpoint, it admits an \( O(\log^3 n) \)-approximation algorithm; in contrast, it is NP-hard to obtain a strategy with competitive ratio \( o(\sqrt{n}) \) in polygons [17].

1.4. Group Steiner problem. The Group Steiner problem is the following.

(\textbf{Rooted}) \textbf{Group Steiner problem.} Given a weighted graph \( G = (V, E) \) with \( k \) groups of vertices \( g_1, g_2, \ldots, g_k \subset V \), find a minimum-weight tree that contains at least one vertex from each group. There is a distinguished vertex \( r \) (the root vertex) that must be included in the tree.

The Group Steiner problem generalizes both minimum Steiner tree and set cover problems. For purposes of our algorithm, we need a variant called the \( \frac{1}{2} \)-Group Steiner problem [19], in which the goal is to find a minimum-weight tree that contains vertices from at least half of the groups.

An \( O(\log^2 n) \) algorithm for Group Steiner on trees is given by Garg, Konjevod, and Ravi [22]. They first solve a linear programming relaxation to get a fractional solution and then use an innovative randomized rounding scheme. A modification of the algorithm, by Even, Kosarz, and Slany [19], yields an \( O(\log n) \)-approximation for the \( \frac{1}{2} \)-Group Steiner problem on trees. For general graphs, one can first probabilistically approximate the graph by a tree, using a result of Fakcharoenphol, Rao, and Talwar [20] (which is a recent improvement to Bartal [3]), losing an \( O(\log n) \) factor in the process. Then the algorithm of Garg, Konjevod, and Ravi [22] is applied to the resulting tree, giving an \( O(\log^3 n) \)-approximation for Group Steiner and an \( O(\log^2 n) \)-approximation for the \( \frac{1}{2} \)-Group Steiner problem.

\textsc{Theorem 1.1} (see [22, 19, 20]). \textit{There exists an} \( O(\log^2 n) \)-approximation algorithm \( A \) \textit{for the rooted} \( \frac{1}{2} \)-\textit{Group Steiner problem that runs in randomized polynomial time.}

The running time of this algorithm is high, and hence the computation time of the robot will be large. As the approximation algorithm is used only as a black box, we will denote the running time by the (polynomial) \( \mathcal{P}(n') \) and instead concentrate on reducing the size \( n' \) of the instance. However, if we are willing to trade off between running time and approximation factor, there are much faster algorithms available. Bateman et al. [4] give a \( \sqrt{k} \ln k \)-approximation algorithm that runs in \( O(nk^2 \log k) \) time. Their algorithm is based on the fact that there exists a Group Steiner tree of depth 2 rooted at \( r \) with cost within \( \sqrt{k} \) of optimal. By adapting their algorithm to the \( \frac{1}{2} \)-Group Steiner problem, we get an \( O(\sqrt{n} \log^2 n) \)-approximation strategy for localization on grid graphs with computation time \( O(n^3 \log^2 n) \) (the best previous factor was \( \Omega(n) \)). A more smooth trade-off can be obtained by using the algorithm of Charikar et al. [9] for the Directed Steiner tree problem (which includes the \( \frac{1}{2} \)-Group Steiner problem as a special case), yielding an \( i(i - 1)k^{(1/i)} \)-approximation with running time \( O(n^i k^{2i}) \). For any \( \epsilon > 0 \), the robot localizes by traveling distance within factor \( O(n^{\epsilon^2} \cdot \log n) \) of the optimal and spending computation time \( O(n^{\epsilon^2} \log^2 n) \). We hope that future work on the \( \frac{1}{2} \)-Group Steiner problem will lead to algorithms with better running times.
Chekuri and Pál [10] have recently described a $O(\log^2 n)$-factor quasi-polynomial-time algorithm for the Group Steiner problem. Since the algorithm involves set cover style arguments, this gives an $O(\log n)$ algorithm for the $\frac{1}{2}$-Group Steiner problem by stopping it when it covers half the groups. Thus our approximation algorithm is optimal if we allow for quasi-polynomial time.

The problem they solve is the submodular orienteering problem (SOP). Here each subset $X \subseteq V$ of a directed graph $G(V, E)$ has a reward function $f(X)$ which satisfies the submodular property. The objective is to construct a walk with maximum given length $B$ such that the subset of vertices $V' \subseteq V$ covered by the walk has maximum reward $f(V')$. Their algorithm is reminiscent of Savitch’s algorithm: the algorithm guesses the middle node of the optimal walk and then recurses two times. However, here the second recursive call is dependent on the output of the first recursive call (i.e., the subset of vertices covered by it), unlike in Savitch’s algorithm where the two calls are independent. We add that the question of a polynomial-time $O(\log^2 n)$ algorithm for Group Steiner is still open.

1.5. Our results. The main contribution of this paper is a polynomial-time strategy, repeated half localization (RHL), which localizes the robot with travel distance within a factor $O(\log^2 n)$ of that of an optimal strategy; more precisely, the approximation factor is $O(\log^2 n \log k)$, where $k = |H| \leq n$ is the number of hypotheses. The key algorithmic idea is to plan travel in a “majority-rule” map, which eliminates uncertainty and permits a link to the $\frac{1}{2}$-Group Steiner (not Group Steiner) problem. Section 2 describes the strategy for the commonly used grid graph model. Section 3 extends the algorithm to robots with line-of-sight (i.e., range finder) sensors in polygons. In section 4, we give a $\log^2 - \epsilon n$ approximation lower bound, assuming $NP \not\subseteq ZTIME(n^{polylog(n)})$, for both grid graphs and the polygonal model. Section 5 sketches extensions of our strategy to a wide variety of models: robots without compasses, limited-range sensors, polygons with holes, geometric trees, and three-dimensional environments. In section 6 we show that a variant of our strategy which does not return to the origin after each half-localize step performs very poorly. In section 7 we summarize and comment on some open problems.

The basic framework of the strategy is to break localization into a sequence of half-localize steps.

HALF-LOCALIZE ($G, H$): Devise a strategy by which the robot can correctly eliminate at least half of the hypotheses in $H$. The robot should travel a (worst-case) distance as small as possible to achieve this. HALF-OPT($G, H$) denotes the cost of the optimal strategy.

Intuitively it might appear that an $O(\log^2 n)$ algorithm for half-localization should be a by-product of our $O(\log^3 n)$ localization strategy and not vice versa. As an example of this, consider the $\frac{1}{4}$-set cover problem, in which the objective is to cover half the elements at minimum cost. There is a constant factor approximation for this, and it is obtained by stopping the $O(\log n)$ greedy algorithm for set cover as soon as we cover half the elements. (Another example is the algorithm for $\frac{1}{2}$-Group Steiner [19], which is obtained by stopping the rounding scheme of [22] as soon as the tree covers half the groups.)

However, half-localize seems to play a more fundamental role in our context. We briefly discuss only the simpler grid graph case here. We construct a majority-rule map, in which each cell is blocked or unblocked depending on what the majority of the current hypotheses in $H$ assert. This majority-rule map permits three interrelated simplifications. If the robot tries to follow a route within the majority-rule map but
makes a minority observation (one inconsistent with at least half of the hypotheses), then the robot has half-localized. This permits a plan to be a path rather than a decision tree. Distances in the real environment are uncertain, but distances on the majority-rule map are fixed. This permits us to model half-localization as a Steiner-type problem on a graph, although we are not able to model localization as such. Finally, there is an essential equivalence between optimally half-localizing and halving paths on the majority-rule map (see section 2.3).


2.1. Preliminaries. During half-localization the robot makes observations from different positions in its environment (grid graph $G$) to make a larger and larger local map $G'$, until there is exactly one hypothesis in $H$ that is consistent with $G'$. We say that a hypothesis $h \in H$ is active if the robot’s local map is consistent with it being located at $h$. We denote the set of active hypotheses by $H'$.

We distinguish between the absolute (global) position of the robot in the grid graph $G$ and its relative (local) position in $G'$ by using Greek letters for the latter (whenever possible). Let $\gamma_0$ denote the initial position of the robot with respect to the local map $G'$. We call $\gamma_0$ the origin, and denote any other position in $G'$ by a pair of coordinates. Coordinate $\gamma = (x, y)$ denotes the cell in $G'$ lying $x$ units to the east and $y$ units to the north of $\gamma_0$. Negative values of $x, y$ denote west and south, respectively. Thus a robot at coordinate $\gamma \in G'$ will be located at position $p_0 + \gamma$ in grid graph $G$, where $p_0 \in G$ is its initial position. The robot can keep track of its local coordinates by taking successive readings on the compass and odometer (we assume error-free motion and sensing during localization). At any point of time, the robot is sure of its local coordinates but knows its global position only up to cells in $H' + \gamma$.

Suppose the robot makes an observation when at coordinate $\gamma \in G'$. The outcome depends on its starting location $h \in H$. If the robot started from hypothesis $h$, the observation will be the same as that by a robot located at $h + \gamma$ in $G$. We denote this observation by $O(h, \gamma)$ and call it the opinion of $h$ about $\gamma$. If $h + \gamma$ is blocked, we set $O(h, \gamma) = \emptyset$. The hypothesis partition $H(\gamma)$ is a partition of the set of hypotheses according to the following equivalence relation: $h_1 \sim h_2$ if and only if $O(h_1, \gamma) = O(h_2, \gamma)$. $Maj(\gamma)$ denotes the largest size class of $H(\gamma)$. The “majority opinion” at $\gamma$ is the opinion common to the plurality of hypotheses $h \in Maj(\gamma)$. Note it may occur that $|Maj(\gamma)| < \frac{1}{2}|H|$. The lemmas that follow are valid in this case because the robot immediately half-localizes. Since there are two choices, blocked or traversable, for each of the four neighboring cells of $\gamma$, an observation $o$ can be written $o \in \{b, t\}^{4} \cup \{\emptyset\}$, and we let $G(\gamma, o)$ denote the class of $H(\gamma)$ with opinion $o$ at $\gamma$.

2.2. The majority-rule map. We now describe the majority-rule map $G_{maj}$, a data structure central to our half-localization algorithm.

Definition 2.1. The majority-rule map $G_{maj}$ is a local map in which each cell is blocked or traversable according to what the majority of hypotheses have to say about it (in case of a tie, we consider the cell to be traversable). The majority-rule map also includes the hypothesis partitions for all local coordinates.

In other words, a cell $\gamma \in G_{maj}$ is blocked if and only if $|Maj(\gamma)| > \frac{1}{2}|H|$ and $V(Maj(\gamma)) = \emptyset$. If $G$ is an $l \times m$ grid graph, the majority-rule map has size bounded by $(2l - 1) \times (2m - 1)$, since the absolute values of $x$- and $y$-coordinates for any hypothesis are at most $(l - 1)$ and $(m - 1)$, respectively. Clearly, $G_{maj}$ requires space $4nk$ (there are at most $4n$ cells, and we need to store the partition for each cell) and can be computed in time $O(nk)$. Figure 2.1 shows the majority-rule map for
grid graph $G$ and $H = \{h_1, h_2, h_3, h_4\}$. The black region is unreachable by the robot for any starting hypothesis. The hypothesis partition is constant within each of the regions $R_0$, $R_1$, $R_2$, $R_3$, $R_4$, $R_5$, and $R_6$. $R_5$ and $R_6$ lie outside the grid graph for three different hypotheses and are blocked. Thus the only traversable regions are $R_0$, $R_1$, $R_2$, $R_3$, and $R_4$, with $\text{Maj}(R_0) = \{h_1, h_2, h_3, h_4\}$, $\text{Maj}(R_1) = \{h_2, h_3, h_4\}$, $\text{Maj}(R_2) = \{h_1, h_2, h_3\}$, $\text{Maj}(R_3) = \{h_3, h_4\}$, and $\text{Maj}(R_4) = \{h_1, h_2\}$.

2.3. Halving paths. We now define the notion of a halving path in the majority-rule map.

**Definition 2.2.** A halving path is a (possibly self-intersecting) path $C = (\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_m)$ in the majority-rule map satisfying $|\bigcap_{i=0}^{m} \text{Maj}(\gamma_i)| \leq \frac{1}{2}|H|$.

The next two lemmas show an essential equivalence between half-localization strategies and halving paths.

**Lemma 2.3.** Let $C$ be a halving path. There exists a strategy $S(C)$ for half-localizing the robot with travel cost at most $|C|$.

**Proof.** Let $C = (\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_m)$, where $\gamma_{i+1}$ is a neighbor of $\gamma_i$ in $G_{\text{maj}}$. A description of strategy $S(C)$ is as follows (see Algorithm 1): the robot traces path $C$ from its initial position, taking observation $o_i$ at each new coordinate $\gamma_i$. If the robot finds that the next coordinate is blocked, it stops. We next show that this will half-localize the robot correctly.

After observation $o_i$, the robot keeps only those hypotheses whose opinion at $\gamma_i$ is $o_i$. Thus, it updates $H'$ (the set of active hypotheses) correctly. We show that $S(C)$ reduces the set of hypotheses by half. If the robot finds that the cell at coordinate $\gamma_i$ is blocked, it localizes to a set of size at most $|G(\gamma_i, \emptyset)| \leq \frac{1}{2}|H|$ (since $\gamma_i \in G_{\text{maj}}$). If observation $o_i$ is different from the majority opinion at $\gamma_i$, $H' \subseteq G(\gamma_i, o_i)$, which has size at most $\frac{1}{2}|H|$. Thus the robot reaches $\gamma_m$ if and only if for each $\gamma_i$, $0 \leq i \leq m-1$, $o_i$ is the majority opinion at $\gamma_i$. Now there are two cases: if $o_m$ is different from the majority opinion, the robot half-localizes; otherwise $H' = \bigcap_{i=0}^{m} \text{Maj}(\gamma_i)$, which is again at most $\frac{1}{2}|H|$ (since $C$ is a halving path).

In Figure 2.1, the halving path $C_1 = (\gamma_0, \gamma_1, \gamma_2)$ satisfies $|\bigcap_{i=0}^{2} \text{Maj}(\gamma_i)| = |\{h_2, h_3\}| \leq \frac{1}{2}|H|$. The path $(\gamma_0, \gamma_3)$ is an optimal halving path, with $|\text{Maj}(\gamma_0) \cap \text{Maj}(\gamma_3)| = |\{h_1, h_2\}| \leq \frac{1}{2}|H|$. Note that we did not include intermediate cells in the description,
ROBOT LOCALIZATION PROBLEM

Data: Grid graph $G$, set of hypotheses $H$ and a halving path $(\gamma_0, \gamma_1, \ldots, \gamma_m) \in G_{maj}$.

Result: The robot half-localizes in at most $m$ steps.

Initialize $H' = H$

for $i = 0$ to $m - 1$ do

\begin{verbatim}
begin
  Make observation $o_i$ at coordinate $\gamma_i$
  Update $H' = H' \cup G(p_i, o_i)$. Stop if $|H'| \leq \frac{1}{2}|H|$
  Move to coordinate $\gamma_{i+1}$
end
\end{verbatim}

Make observation $o_m$ at $\gamma_m$; Update $H' = H' \cap G(\gamma_m, o_m)$. Stop.

Algorithm 1: Strategy $S(C)$.

assuming that the robot uses any shortest path in the majority-rule map to go from $\gamma_i$ to $\gamma_{i+1}$. The behavior of a robot following strategy $S(C_1)$ will be as follows. If it was placed at $h_1$, it will hit a wall at $\gamma_1$ and stop with $H' = \{h_1\}$. If it was placed at $h_4$, it will see a wall at $\gamma_2$ and stop with $H' = \{h_4\}$. If it was placed at either $h_2$ or $h_3$, it will make majority observations at both $\gamma_1, \gamma_2$ and half-localize to the set $\{h_2, h_3\}$ of hypotheses.

The next lemma shows that any half-localization strategy $S$ has an associated halving path with length at most $W(S)$ (compare this with localization, for which strategies are decision trees [17] and hence hard to compute).

Lemma 2.4. Let $S$ be a strategy for half-localization. There exists a halving path $C(S)$ of length at most $W(S)$, the cost of the strategy $S$.

Proof. Consider a robot guided by $S$ that stops as soon as it half-localizes. Let $C(S) = (\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_m)$ be the maximum length path traced by the robot in its local map $G'$ for any starting position in $H$. Let $H_i$ denote the set of active hypotheses just after the robot makes an observation at coordinate $\gamma_i$. For $0 \leq i < m$, $|H_i| > \frac{1}{2}|H|$, since otherwise the robot would have stopped at $\gamma_i$. Each coordinate $\gamma_i$ is unblocked for at least $|H_i| > \frac{1}{2}|H|$ hypotheses, and hence $C(S)$ lies in the majority-rule map $G_{maj}$.

We claim that $I = \bigcap_{i=0}^{m} Maj(\gamma_i)$ is of size at most $\frac{1}{2}|H|$. Consider a robot initially located at some $h \in I$. Guided by $S$, the robot will follow path $C(S)$ and make the majority observation $o_i$ at all coordinates $\gamma_i$ (since $I \subset Maj(\gamma_i)$). But then $|\bigcap_{i=0}^{m} Maj(\gamma_i)| = |H_m| \leq \frac{1}{2}|H|$, and hence $C(S)$ is a halving path.

2.4. Computing halving paths. Let $C^*_H$ denote an optimal halving path for the set of hypotheses $H = \{h_1, h_2, \ldots, h_k\}$. We approximate the problem of computing an optimal halving path by reducing it to an instance $I_{G,H}$ of the $\frac{1}{2}$-Group Steiner problem.

The reduction is a restatement of the problem in terms of groups: we take $V$ as the set of traversable coordinates in the majority-rule map. The weight of edge $(\gamma, \gamma')$ is the length of the shortest path joining cells $\gamma$ and $\gamma'$ in $G_{maj}$. Origin $\gamma_0$ is taken as the root vertex. We make $k$ groups, one for each hypothesis $h_i \in H$. Group $g_i$ is the set of all coordinates $\gamma \in V$ such that $h_i$ does not share the majority opinion at $\gamma$, i.e., $h_i \notin Maj(\gamma)$. Thus a tree $T$ covers $k'$ groups if and only if $\bigcap_{i \in T} Maj(x)$ has size $k - k'$. In particular, $T$ covers at least half the groups if and only if $|\bigcap_{i \in T} Maj(\gamma)| \leq \frac{1}{2}|H|$. In particular, every halving path is a $\frac{1}{2}$-Group Steiner tree.

Lemma 2.5. There exists an $O(\log^2 n)$-approximation algorithm for computing an optimal halving path.

Proof. Let $T$ be the tree output by algorithm $A$ (see Theorem 1.1) on instance $I_{G,H}$. Then, the weight of $T$ is at most $O(\log^2 n) \cdot w(T^*)$, where $T^*$ is an optimal
\(1\)-Group Steiner tree. Let \(C\) be the path of length at most \(2 \cdot w(T)\) traced by a depth-first search on \(T\) starting from the origin. \(C\) is a halving path since \(|\bigcap_{y \in C} Maj(x)| = |\bigcap_{y \in T} Maj(x)| \leq \frac{1}{2}|H|\). Since an optimal halving path \(C^*_H\) covers half the groups, \(w(T^*) \leq |C^*_H|\). Therefore, \(|C| \leq O(\log^2 n) \cdot |C^*_H|\). \(\square\)

2.5. Strategy RHL. The overall strategy is as follows (see Algorithm 2). In each half-localize phase, the robot computes a near-optimal halving path \(C\) and then traces \(C\) to reduce the set of (active) hypotheses by half. It retracts \(C\) to move back to its initial position and proceeds with the next phase. We now bound the approximation factor and computation time of strategy RHL.

**Algorithm 2: Strategy RHL (repeated half localization).**

**Data:** Grid graph \(G\), the set of hypotheses \(H\)

**Result:** The robot localizes to its initial position \(h \in H\)

while \(|H| > 1\) do

- Compute the majority-rule map \(G_{maj}\)
- Make instance \(I_G,H\) of \(1\)-Group Steiner problem
- Solve \(I_G,H\) to compute a halving path \(C\) (Lemma 2.5)
- Half-localize by strategy \(S(C)\) (Lemma 2.4)
- Move back to the starting location

end

**Theorem 2.6.** A robot guided by strategy RHL (Algorithm 2) correctly determines its initial position \(h \in H\) by traveling at most \(O(\log^2 n \log k) \cdot OPT(G, H)\) distance, where \(k = |H|\) and \(n\) is the size of \(G\). Further, the computation time of the robot is polynomial in \(n\).

**Proof.** Since the number of active hypotheses reduces by at least half after each phase, the robot localizes in \(m \leq \log |H|\) phases. Let \(H_i\) denote the set of active hypotheses at the start of the \(i\)th phase. By Lemma 2.5, the distance traveled by the robot in the \(i\)th phase is at most \(O(\log^2 n \log k) \cdot |C^*_H|\). By Lemma 2.4, \(|C^*_H| \leq \text{HALF-OPT}(G, H_i) \leq \text{OPT}(G, H_i)\), where the last inequality follows from the fact that any localization plan also reduces the set of hypotheses by half. Therefore, the distance traveled by the robot in each phase is at most \(O(\log^2 n) \cdot \text{OPT}(G, H)\). Since there are \(O(\log k)\) phases, the total worst-case travel distance is \(O(\log^2 n \log k) \cdot \text{OPT}(G, H)\). Since instance \(I_H\) can be constructed in \(O(nk)\) time, the computation time is at most \(O(P(nk) \log n)\), where \(P()\) (a polynomial) is the time taken by the approximation algorithm for \(1\)-Group Steiner (see section 1.4). \(\square\)

The above theorem shows the performance ratio for a robot with very weak sensors; the robot can only “see” four neighboring cells. We note that all of the theorems of this section hold for robots on grid graphs with other kinds of sensors such as range-finders or sonar. An interesting feature of our strategy is that it is well suited to handling the problem of accumulation of errors caused by successive motion in the estimates of orientation, distance, and velocity by the robot’s odometer. This is because after each half-localize phase the robot always returns to the origin, which it can use to recalibrate its sensors [17].

3. Polygonal model. In this section, we extend our algorithm to polygons. We focus here on the case of simple polygons; in section 5 we discuss the extension to the case of polygons with holes. The outline of the algorithm is the same: the robot works in phases, in each phase reducing the set of hypotheses by half. However, since the
robot moves continuously, local coordinates γ lie in the Euclidean plane \( \mathbb{R}^2 \) (for grid graphs, i.e., the visibility polygon observed by a robot at position \( h + \gamma \in P \). If the point \( h + \gamma \) lies outside \( P \), we take \( \mathcal{O}(h, \gamma) = \emptyset \). For a coordinate \( \gamma \in \mathbb{R}^2 \), the hypothesis partition \( \mathcal{H}(\gamma) \) partitions hypotheses in \( H \) according to their opinions \( \mathcal{O}(h, \gamma) \). The majority-rule map denotes the subset of coordinates that lie inside \( P \) for the majority of hypotheses. In section 3.1, we will show that the majority-rule map is a polygon with holes of size \( O(k^2n^2) \) and that this bound is tight in the worst case. We let \( P_{maj} \) denote the connected component of the majority-rule map that contains the origin \( \gamma_0 \), and often we refer to \( P_{maj} \) simply as the majority-rule map, since \( P_{maj} \) is the component of interest to us.

In the polygonal model, a halving path \( \mathcal{C} \) is a curve in the majority-rule map with one endpoint at the origin \( \gamma_0 \) such that \( |\bigcap_{x \in \mathcal{C}} \operatorname{Maj}(x)| \leq \frac{1}{2}|H| \). (Parameter \( x \) varies over the continuum of coordinates along the path \( \mathcal{C} \).) It is straightforward to extend Lemmas 2.3 and 2.4 to the case of polygons with this new definition. A shortest path \( \text{Path}(\gamma, \gamma') \) between any two coordinates \( \gamma, \gamma' \in P_{maj} \) is piecewise linear with bend points at vertices (this includes the vertices of holes) of \( P_{maj} \). Hence, we can specify a halving path by a sequence \( (\gamma_0, \gamma_1, \ldots, \gamma_m) \), where \( \bigcap_{i=0}^{m} \operatorname{Maj}(\gamma_i) \leq \frac{1}{2}|H| \), and a shortest path \( \text{Path}(\gamma_i, \gamma_{i+1}) \) is used to go from \( \gamma_i \) to \( \gamma_{i+1} \).

Since \( P_{maj} \) consists of an infinite number of points, one cannot compute an approximation to the optimal halving path \( \mathcal{C}^*_H \) by reducing it to a \( \frac{1}{2} \)-Group Steiner problem on a finite number of coordinates, as in section 2.4 for the case of grids. Instead, we discretize the problem to a finite, polynomial-size set of coordinates \( H \). We first calculate the boundaries of groups \( \mathcal{G}_i \) (i.e., coordinates \( \gamma \) such that \( h_i \notin \operatorname{Maj}(\gamma) \)), which are polygons \( K_i \subset P_{maj} \) with holes (see section 3.2). Hence the robot just needs to visit the boundary of at least half of the \( K_i \)'s. In section 3.4, we describe how to select a special set of discrete points on the boundary of the \( K_i \)'s so that a halving path of length at most 2 times that of optimal passes through these points. Next, we construct the instance \( \mathcal{I}_{P,H} \) of the \( \frac{1}{2} \)-Group Steiner problem on the finite set of coordinates \( Q_H \), as we did for the case of grid graphs in section 3.5. Finally, in section 3.6 we combine all of the ingredients above to get an \( O(\log^2 n \log k) \)-approximation algorithm for the polygonal model.

### 3.1. Computing the majority-rule map.

The boundary of the majority-rule map can be constructed as follows. Let \( P_i \) denote a translation-congruent copy of the map polygon with hypothesis \( h_i \) at the origin \( \gamma_0 \). Clearly, coordinate \( \gamma \) is traversable for hypothesis \( h_i \) if and only if it lies inside polygon \( P_i \). The overlay of all of these polygons, \( \text{Overlay}(P_1, P_2, \ldots, P_k) \), partitions the plane into polygonal regions, known as cells. Each cell \( C \) either lies completely inside copy \( P_i \) or lies completely outside it. The majority-rule map is formed by the union of all cells \( C \) that lie inside \( P \) for the majority of hypotheses (equivalently, for the majority of \( P_i \)'s). By this construction, the majority-rule map is a union of polygons (possibly with holes). The next lemma gives a tight bound on its worst-case complexity.

**Lemma 3.1.** Let \( A_1, A_2, \ldots, A_k \) be \( k \) polygons (possibly with holes), each containing the origin and each with \( O(n) \) vertices. Then, the face, \( A_{maj} \), containing the origin in the majority-rule map they define has \( O(k^2n^2) \) vertices and can be constructed in time \( O(k^2n^2) \). Furthermore, the upper bound of \( O(k^2n^2) \) on the number of vertices is tight, even if the \( A_i \)'s are translates of the same simple polygon.
Proof. The upper bound is immediate, since the set of \( O(kn) \) line segments that constitute the edges of the \( k \) polygons define an arrangement having at most \( O(k^2n^2) \) vertices in total. The lower bound is illustrated in Figure 3.1. The time to construct \( A_{maj} \) follows from the fact that an arrangement of \( m \) segments in the plane can be constructed in time \( O(m^2) \), and, within this same time bound, the faces of the majority-rule map can be identified, after which the face containing the origin can be constructed by breadth-first search in the dual graph of the arrangement. In fact, using the algorithm of Balaban [2], the arrangement can be constructed in output-sensitive time.

![Fig. 3.1. An example with complexity \( \Omega(k^2n^2) \) of the majority-rule map obtained by overlaying translates of copies of a simple polygon. The solid dots denote the set of hypotheses.](image)

The above lemma also bounds the complexity of \( P_{maj} \) and shows that it is tight, since it arises from the majority-rule map associated with translates of \( P \).

3.2. Computing the group boundaries. Following the definition in section 2.4, the group \( g_i \) is defined to be the set of all coordinates \( \gamma \in P_{maj} \) such that \( h_i \) does not have the majority opinion at \( \gamma \), i.e., \( h_i \notin Maj(\gamma) \). The complement \( \bar{g}_i \) of \( g_i \) is the set of points \( \bar{g}_i = P_{maj} \setminus g_i \) not in \( g_i \).

Consider a hypothesis \( h_j \) (\( h_j \neq h_i \)), and let \( F_{ij} \) denote the face in \( Overlay(P_i, P_j) \) that contains the origin, \( \gamma_0 \) (see Figure 3.2). First, we note the following.

**Lemma 3.2.** The face \( F_{ij} \) has at most \( 2n \) edges.

*Proof.* Consider an edge \( e \) of \( F_{ij} \). If \( e \) is a subsegment of both \( P_i \) and \( P_j \), then
one of its endpoints must be a vertex \( v \) of \( P_i \) or \( P_j \), and we can “charge” \( e \) to that vertex. If \( e \) is a subsegment of \( P_i \) but not of \( P_j \), then it forms a chord of \( P_j \) and can be charged off to the vertex of \( P_j \) it occludes. Since each vertex is charged at most once, \( F_{ij} \) has at most \( 2n \) edges.

Each of the \( O(n) \) edges \( e \subset \partial F_{ij} \) is of one of three types: (i) \( e \) lies on the boundary of \( P_i \) but not of \( P_j \); (ii) \( e \) lies on the boundary of \( P_j \) but not of \( P_i \); or (iii) \( e \) lies on the boundary of both \( P_i \) and \( P_j \). A robot can distinguish between \( h_i \) and \( h_j \) if and only if the robot sees an edge \( e \) of type (i) or (ii).

![Figure 3.2](image)

**Fig. 3.2.** Top: A polygon \( P \) with two hypotheses \( h_1 \) and \( h_2 \). Bottom: The overlap of \( P_1 \) and \( P_2 \), with the face \( F_{12} \) containing \( \gamma_0 \) highlighted.

If \( \gamma_0 \) sees any edge of type (i) or type (ii), then the robot can distinguish between \( h_i \) and \( h_j \) without moving from the origin \( \gamma_0 \). Thus, assume that all edges of \( F_{ij} \) that are visible from \( \gamma_0 \) are of type (iii). Let \( e \) be an edge of \( F_{ij} \) of type (i) or of type (ii). The set \( VP(e) \) of points of \( F_{ij} \) that are visible to some point of \( e \) is a simple polygon (the *visibility polygon* of \( e \)) within \( F_{ij} \), which we know, by assumption, does not include point \( \gamma_0 \). There is a chord of \( F_{ij} \), \( w(e) \), that lies on the boundary of \( VP(e) \), separating \( e \) from \( \gamma_0 \). The line segment \( w(e) \) is often called a *window* (see 3.3).

Consider now the arrangement of the \( O(n) \) boundary edges of \( F_{ij} \) together with the set of all \( O(n) \) windows \( w(e) \) for edges \( e \) of type (i) or (ii). Let \( G_{ij} \) denote the face in this arrangement that contains the origin \( \gamma_0 \). Since \( G_{ij} \) is a face in an arrangement of chords of a simple polygon, it is a simple polygon having linear \( (O(n)) \) complexity. (No chord can contribute more than once to the face.) Note too that \( G_{ij} \subset F_{ij} \subset P_i \)
and that the boundary of $G_{ij}$ is of two types: (a) a polygonal chain of type (iii) edges, which is on the common boundary, $\partial P_i \cap \partial P_j$, of $P_i$ and $P_j$, or (b) a window-chain consisting of a convex polygonal chain comprised of subsegments of windows. A window-chain of $G_{ij}$ separates $\gamma_0$ from one or more edges of $F_{ij}$ of type (i) or type (ii). The next lemma follows from the definition of $G_{ij}$.

**Lemma 3.3.** A robot can distinguish between hypothesis $h_i$ and hypothesis $h_j$ if and only if it visits a window-chain on the boundary, $\partial G_{ij}$, of $G_{ij}$.

**Proof.** Each window $w(e)$ cuts off the part of polygon $VP(e)$ from which $e$ of type (i) or (ii) is visible. In other words, as soon as the robot crosses $w(e)$, it can use its sensors to check whether $e$ exists or not, and hence will be able to distinguish $h_i$ from $h_j$. Since $G_{ij}$ is what remains after all visibility polygons $VP(e)$ of edges of type (i) or (ii) have been chopped off, it satisfies the lemma (see Figure 3.3).

In other words, $G_{ij}$ is the connected component of coordinates including the origin $\gamma_0$ for which $\mathcal{O}(h_i, \gamma) = \mathcal{O}(h_j, \gamma)$; i.e., the opinions of $h_i$ and $h_j$ are the same. Next we use the $G_{ij}$’s to construct the complement, $\bar{g}_i$, of group $g_i$.

Let $K_i$ be the face containing $\gamma_0$ in the majority-rule map of the $k - 1$ polygons $G_{ij}$ for $j \neq i$. Thus, the boundary of $K_i$ consists of polygonal chains on the boundary of $P_i$ and polygonal chains comprised of segments and subsegments of the window-chains that appear on the boundaries of the polygons $G_{ij}$. We refer to $\partial K_i \setminus \partial P_i$ as the window-boundary of $K_i$.

It is clear that $K_i \subseteq P_{maj}$, since each point of $K_i$ lies within a majority of the polygons $G_{ij}$, and therefore of the polygons $P_i$.

**Lemma 3.4.** $K_i$ is a connected component of the set $\bar{g}_i$. A robot initially located at hypothesis $h_i$ will half-localize if and only if it travels to the window-boundary of $K_i$.

**Proof.** We first show that $K_i \subseteq P_{maj}$. Let $I$ denote the set of $k - 1$ indices $\{1 \ldots n\} \setminus i$. Consider any coordinate $\gamma \in K_i$. Let $I' \subseteq I$ denote the set of indices $j$ such that $\gamma \in G_{ij}$. Any coordinate inside $G_{ij}$ clearly belongs to both polygons $P_i$ and $P_j$. Hence, $\gamma$ is inside polygon $P$ for at least $|I'| + 1 \geq \left\lceil \frac{n}{k} \right\rceil + 1 \geq \left\lceil \frac{k}{2} \right\rceil$ hypotheses. Thus, $K_i \subseteq P_{maj}$. Further, the opinions $\mathcal{O}(h_i, \gamma) = \mathcal{O}(h_j, \gamma)$ are the same for any $j \in I'$. Thus, the majority opinion at $\gamma$ is the same as $\mathcal{O}(h_i, \gamma)$ and hence $h_i \in Maj(\gamma)$.

For the second statement, note that if the robot crosses the boundary of $K_i$, it will lie outside at least half of the $k - 1$ sets $G_{i1}, G_{i2}, \ldots, G_{ik}$ and hence by making

![Figure 3.3](image-url)
an observation will be able to distinguish \( h_i \) from at least \( \lceil \frac{k}{2} \rceil \) hypotheses. In the worst case (if the robot is initially at \( h_i \)), we will be left with at most \( k - \lceil \frac{k}{2} \rceil \leq \lceil \frac{k}{2} \rceil \) hypotheses, and hence the robot will half-localize. (Note that the set of hypotheses remaining can be one more than that required for half-localization; however, the number of iterations remains \( O(\log k) \), and hence the approximation factor is unchanged.)  

\[ \text{Lemma 3.5. } K_i \text{ has } O(nk^{4/3} \alpha(n) \log^{2/3} k) \text{ edges, where } \alpha(\cdot) \text{ denotes the inverse Ackermann function.} \]

\[ \text{Proof. } \] First note that the boundary of \( K_i \) that is not part of the window-boundary has complexity \( O(n) \), since it is a boundary shared with \( P_i \). Thus, it suffices to bound the complexity of the window-boundary of \( K_i \).

Each of the \( O(nk) \) edges of the window-chains of the regions \( G_{ij} \) can be mapped to a (finite length) curve in a “polar geodesic” coordinate system defined by the family of all shortest (geodesic) paths within \( P_i \) from \( \gamma_0 \) to points \( t \in \partial P_i \) on the boundary. Then, we appeal to the fact that the \( k \)-level in an arrangement of a set of \( m \) pseudo-segments has complexity \( O(mk^{1/\alpha}(m/k) \log^{2/3} k) \) \[8\]. Since we have \( m = O(nk) \), the total complexity of \( K_i \) is \( O(nk^{4/3} \alpha(n) \log^{2/3} k) \).

(We expect that the complexity of \( K_i \) is \( O(nk^{4/3} \alpha(n)) \), which is the complexity of the \( k \)-level in an arrangement of (straight) line segments.) 

Each region \( K_i = \hat{g}_i \) shares one or more polygonal chains on its window-boundary with the boundary of set \( g_i \). In order to half-localize, the robot needs to visit at least half of the groups \( g_i \). Thus, the robot needs to visit at least half of the window-boundaries of the \( K_i \)'s (i.e., at least half of the sets \( \partial K_i \cap \partial g_i \)), each of which consists of \( O(nk^{4/3} \alpha(n) \log^{2/3} k) \) edges that lie within the majority map \( P_{maj} \). Our goal is to find a path within \( P_{maj} \) that visits at least half of the sets \( \partial K_i \cap \partial g_i \).

\[ \text{3.3. Comparison with visibility skeleton. } \] In section 3.2 we construct cells of the majority-rule map which distinguish between hypotheses according to their visibility polygons. On the other hand, the previous constructions of Guibas, Motwani, and Raghavan [23] and Dudek et al. [17] decompose the plane according to an approximation called the visibility skeleton. We now show that an algorithm using visibility skeletons can perform much worse than one using visibility polygons.

Intuitively, a visibility skeleton is a contraction of the visibility polygon \( V \) so that the skeleton boundary consists of only those vertices that can be certified to be the vertices of \( V \). The main loss in information is as follows: there may be a partial edge in the visibility polygon whose endpoints are blocked by two reflex vertices. The visibility skeleton remembers the “slope” of the line containing this edge but not its visible distance and length.

Figure 3.4 describes the advantage gained by describing decompositions with respect to visibility polygons. The north-south corridors \( N_1 \) and \( N_2 \) are very long compared to the east-west corridors \( E_1 \) and \( E_2 \). Edges \( e_1 \) and \( e_2 \) have the same slopes, but edge \( e_1 \) is “nearer” than edge \( e_2 \). The robot is located at one of the two hypotheses \( h_1 \) and \( h_2 \).

A robot using visibility polygons will localize as soon as it enters the north-south corridor. This is because the distance and length of partial edge \( e_1 \) for a robot located at the start of \( N_1 \) will be smaller than that of partial edge \( e_2 \) for a robot located at the start of \( N_2 \).

On the other hand, a robot using visibility skeletons will need to go up its northern corridor until it finds a new vertex. The earliest such vertex is \( v_2 \) for hypothesis \( h_2 \). Therefore, the robot will go up the northern corridor until it reaches the window.
formed by \( v_2 \). If the robot sees \( v_2 \), it concludes that it is at hypothesis \( h_2 \), otherwise it localizes to \( h_1 \).

Thus our algorithm performs considerably better if we use visibility polygons instead of visibility skeletons.

3.4. The set of coordinates \( \mathcal{Q}_H \). In order to solve our half-localization problem, we define a discrete set \( \mathcal{Q}_H \) of points on the edges of \( \partial K_i \cap \partial g_i \) and then solve an instance of the \( \frac{1}{2} \)-Group Steiner problem on the corresponding point set.

We first note that there does not exist a polynomial-size set of coordinates \( \mathcal{Q}_H \) such that every optimal path that half-localizes has bend points in \( \mathcal{Q}_H \). In particular, in Figure 3.5 we illustrate that there can be an exponential number of distinct points at which an optimal path visits a given subset, \( S \), of a sequence of segments, “reflecting” off of each segment, according to the usual local optimality condition. In particular, there are \( 2k + 1 \) line segments, arranged in two parallel rows of \( k \) segments each. Let \( l_0, l_2, \ldots, l_{2k-2} \) denote the line segments in the top row and \( l_1, l_2, \ldots, l_{2k-1} \) the line segments in the bottom row. The origin \( \gamma_0 \) is located symmetrically to the left of \( l_0 \) and \( l_1 \). The remaining line segment \( l_{2k} \) is placed opposite the origin on the other side of the rows. Let \( S \subseteq [0, 2k-1] \) denote a subset of the line segments forming the two rows. Let \( C_S \) denote the shortest length path visiting segments in \( S \) in increasing order of index and ending at segment \( l_{2k} \). Then one can show that the \( 2^\Omega(k) \) spanning paths contain an exponential number of distinct reflection points. Figure 3.5 shows this for the case \( k = 2 \). We now describe the construction of the discrete set \( \mathcal{Q}_H \) that we use for our approximation. Consider an optimal halving path \( C^*_H \subset P_{maj} \), which visits at least \( \lceil \frac{k}{2} \rceil \) of the sets \( \partial K_i \cap \partial g_i \).

Let \( r^* \) be the (geodesic) radius of the smallest geodesic disk, centered on \( \gamma_0 \), that contains \( C^*_H \). Here, “geodesic” refers to shortest path distance within the majority-rule map \( P_{maj} \) (a geodesic will be a piecewise linear curve). Let \( r_{min} \) be the (geodesic) radius of the smallest geodesic disk, centered on \( \gamma_0 \), that intersects at least \( \lceil \frac{k}{2} \rceil \) of the boundaries \( \partial g_i \). Clearly, \( r^* \geq r_{min} \). Further, we know that the length of \( C^*_H \) is at most \( k \cdot r_{min} \), since one possible halving path stays within the geodesic disk, \( D_0 \), of radius \( r_{min} \) centered at \( \gamma_0 \), and travels at most distance \( 2r_{min} \) between any two consecutive groups visited by the path (just go via \( \gamma_0 \), using geodesic paths to get to and from
γ₀. Note too that it is easy to compute \( r_{\text{min}} \) by computing the shortest path map with respect to source \( γ₀ \) within \( P_{\text{maj}} \); see [33].

Consider the sequence of radii \( r_{\text{min}}, 2r_{\text{min}}, 4r_{\text{min}}, \ldots, 2^{\lceil \log k \rceil} r_{\text{min}} \). Note that \( r^* \in [2^i r_{\text{min}}, 2^{i+1} r_{\text{min}}] \) for some choice of \( i' \) among the \( O(\log k) \) possibilities in the sequence. For each choice of \( i' \), we consider the axis-aligned square (this square is not with respect to geodesic distance), centered at \( γ₀ \), of side length \( 2 \cdot 2^i r_{\text{min}} \), and decompose the square into a \( k \)-by-\( k \) grid of subsquares using \( k-1 \) evenly spaced horizontal/vertical lines. For each segment \( σ \) that is an edge of some \( ∂Kᵢ \cap ∂gᵣ \), we mark on \( σ \) the crossing points (if any) where \( σ \) crosses a grid line (i.e., where \( σ \) crosses between subsquares). This results in at most \( 2k-2 \) marked points along \( σ \), for each choice of \( i' \), so \( O(k \log k) \) marked points in total along \( σ \).

We let \( Q_H \) be the union of the set of all marked points for all edges on the boundaries \( ∂Kᵢ \cap ∂gᵣ \), together with the endpoints of these edges. Since there are \( k \) sets \( ∂Kᵢ \cap ∂gᵣ \), each with \( O(nk^{1/3} \alpha(n) \log^{5/3} k) \) edges/vertices, and we place \( O(k \log k) \) marked points per edge, this yields a total of \( O(nk^{10/3} \alpha(n) \log^{5/3} k) \) points in \( Q_H \).

(Note that this bound is nearly linear in \( n \), and one may expect that, in practice, \( k << n \).

**Lemma 3.6.** Suppose that an optimal halving path \( C_H^* \) visits \( ∂g₁, ∂g₂, \ldots, ∂gₘ \), with \( m = \lceil k/2 \rceil \), and let edge \( eᵢ \subset ∂gᵣ \) be the first edge of \( gᵣ \) visited along \( C_H^* \) (after leaving \( γ₀ \)). Then there exists a piecewise-linear halving path \( C = (γ₀, γ₁, \ldots, γₘ) \) of length at most \( 2 \cdot |C_H^*| \) such that \( γᵢ \in Q_H \), and the shortest (geodesic) path in \( P_{\text{maj}} \) is used to go from \( γᵢ \) to \( γᵢ₊₁ \).

**Proof.** Let \( pᵢ \in eᵢ \) be the first point where \( C_H^* \) visits \( ∂gᵣ \). Let \( r^* \) be the geodesic radius of the smallest geodesic disk \( D₀ \) (within \( P_{\text{maj}} \)) centered at \( γ₀ \) that contains \( C_H^* \); let \( i' \) be such that \( r^* \in [2^{i'} r_{\text{min}}, 2^{i'+1} r_{\text{min}}] \). Then, we know that each segment \( eᵢ \) intersects \( D₀ \) and therefore also intersects the axis-aligned square of side length \( 2 \cdot 2^{i'+1} r_{\text{min}} \) centered at \( γ₀ \). Thus, within distance \( (1/k)2^{i'+1} r_{\text{min}} \) of \( pᵢ \) along the line segment containing \( eᵢ \) there is a marked point \( γᵢ \) of \( Q_H \) associated with the corresponding grid partition into subsquares; in case the endpoint of \( eᵢ \) is encountered along the segment before the marked point, we redefine \( γᵢ \) to be this endpoint. We can modify the path to go through each \( γᵢ \) (this is possible by sliding the endpoint continuously along the edge to the coordinate in \( Q_H \)), adding distance \( (1/k)r^* \) per \( i \). In total, the cost of these detours is \( k \cdot (1/k)r^* = r^* \), thus proving the claim. \( \square \)

**3.5. Reduction to \( \frac{3}{4} \)-Group Steiner.** We formulate now the instance of the \( \frac{3}{4} \)-Group Steiner problem that we need to solve for half-localization.

**INSTANCE \( \mathcal{I}_{P_H} \).** Take \( G \) as the complete graph on \( Q_H \). Define the cost of an edge \( (γᵢ, γᵢ') \) to be the length of a shortest path joining \( γᵢ, γᵢ' \) in the majority-rule map \( P_{\text{maj}} \). Take the root as the origin \( γ₀ \). Make \( k \) groups of points of \( Q_H \) corresponding to the sets \( g₁, g₂, \ldots, gₖ \).

As in section 2.4, a tree \( T \) covers \( k' \) groups if and only if \( \bigcap_{γᵢ \in T} \text{Maj}(γᵢ) \) has size
In particular, \( T \) covers at least half the groups if and only if \( \left| \bigcap_{\gamma \in T} \text{Maj}(\gamma) \right| \leq \frac{1}{2} |H| \). Also every halving path gives a \( \frac{1}{2} \)-Group Steiner tree of the same cost. It is easy to extend Lemma 2.5 to this case, given that a halving path of cost within twice of optimal passes through points in \( Q_H \) (by Lemma 3.6).

3.6. Putting everything together. The overall strategy for polygons is as follows (see Algorithm 3). Theorem 3.7 bounds the approximation factor and computation time of strategy RHL.

**Data:** Map polygon \( P \), the set of hypotheses \( H \)

**Result:** The robot localizes to its initial position \( h \in H \) while \( |H| > 1 \)

begin
Compute the polygons \( G_{ij} \) for each pair of hypotheses, \( h_i \) and \( h_j \)
Compute the polygons \( q_h \)
Compute the majority-rule map \( P_{maj} \)
Compute the set of coordinates \( Q_H \)
Make instance \( I_{P,H} \) of \( \frac{1}{2} \)-Group Steiner problem
Solve \( I_{P,H} \) to compute a halving path \( C \subset P_{maj} \) (Lemma 2.5)
Half-localize by tracing \( C \) and making observations at coordinates in \( Q_H \)
Move back to the starting location
end

**Algorithm 3:** Strategy RHL for polygons.

**Theorem 3.7.** A robot guided by strategy RHL (Algorithm 3) correctly determines its initial position \( h \in H \) by traveling at most distance \( O(\log^2 n \log k) \cdot \text{OPT}(P,H) \), where \( k = |H| \) and \( n \) is the number of vertices in polygon \( P \). Further, the computation time of the robot is polynomial in \( n \) and \( k \).

**Proof.** By Lemma 3.6, an optimal halving path on the coordinates \( Q_H \) is of length \( O(\text{OPT}(P,H)) \). Since the number of vertices in \( Q_H \) is polynomial (bounded by \( O(\alpha(n) \log^{5/3} k) \)), an \( O(\log^2 n) \)-approximate halving path can be computed in polynomial time by using algorithm \( A \) (see Theorem 1.1). Since there are \( \log |H| \) phases, this gives an \( O(\log^2 n \log |H|) \)-factor strategy.

**Bibliographic note.** We now compare previous work based on the greedy strategy of Dudek, Romanik, and Whitesides [17] with our own algorithms. The greedy strategy MDL always goes to the nearest informative point at each iteration.

For the grid graph model, a robot following MDL first computes a unanimous-rule map i.e., the connected component \( O \) of all grid cells which are traversable for all hypotheses. A cell at the boundary of \( O \) is blocked if it is blocked relative to at least one hypothesis.

Strategy MDL visits the nearest blocked cell in \( O \) and makes an observation. It updates the set of hypotheses using this observation and then retraces its path back to the origin. We repeat this until we localize.

Clearly, each iteration removes at least one hypothesis, so there will be at most \( k \) such iterations. Further, the travel cost in each iteration is less than the optimal verification tour, which is itself less than the optimal strategy. This gives an \( O(k) \)-competitive algorithm. The same analysis holds for the approximation algorithm.

Note that the majority-rule map allows for the removal of at least half the hypotheses, whereas a robot using the unanimous-rule map may remove just one hypothesis in each iteration. This allows for the significantly better approximation factor of strategy RHL.
To extend their algorithm to the polygonal model, Dudek, Romanik, and Whitesides compute $O$ by taking the intersection of shifted copies $P_1, P_2, \ldots, P_k$ of the polygon with respect to different hypotheses. A robot has to check the boundary of $O$ to get new information. However, a robot may check whether an edge $e \in O$ exists by going to the boundary of its window $w(e)$ inside $O$. Therefore, we take the intersection of the regions formed by cutting off $O$ at the various edge windows $w(e_1), w(e_2), \ldots, w(e_m)$. We call this restricted region $U$. The robot then needs to visit the nearest point on the boundary of $U$ to get new information.

We refer the reader to the paper by Rao, Dudek, and Whitesides [38, 39] for the above construction as well as randomized variants of MDL.

4. Inapproximability. We now show an $\Omega(\log^{2-\epsilon})$ lower bound for localization by a reduction from the hardness of the Group Steiner problem.

4.1. Hardness of Group Steiner. A tree is said to be of arity $d$ if each non-leaf vertex has exactly $d$ children. A rooted tree has height $H$ if all of its leaves are at distance $H$ from the root. As usual, the level of a vertex is its distance from the root; the root itself is at level 0, and there are $H + 1$ levels.

Definition 4.1 (see [3]). A hierarchically well-separated tree (HST) is defined to be a rooted, weighted tree in which (i) all leaves are at the same distance from the root, and (ii) the weight of each edge is exactly $\frac{1}{\tau}$ times the weight of its parent edge, where $\tau \geq 1$ is any desired constant.

To prove the lower bound, we use the recent result of Halperin and Krauthgamer [24] which establishes $\Omega(\log^{2-\epsilon} n)$ hardness for the Group Steiner problem on HSTs. The next theorem, extracted from their proof, states their result in a detailed form suited to our purpose.

Theorem 4.2 (see [24]). Let $L$ be any NP-complete language. Then there exist a constant $c_0$ and an algorithm $A$ that, given an instance $I$ and a sufficiently large constant $\alpha$, produces in expected running time $O(|I|^{\alpha \log(|I|)})$ an instance $I' = (T, r, G)$ ($r$ is also the root of $T$) of the Group Steiner problem such that the following hold.

1. For some $m \leq |I|^c_0$, $T$ is an HST with height $H = (\log m)^\alpha$, arity $d = m^{O(\log m)}$, and $\tau = m^{\log m}$. Further, each group $g \in G$ is a subset of the leaves of $T$, and there are $k = m^{O((\log m)^{\alpha + 1})}$ groups.
2. If $I \in L$, then there is a (rooted) tree $T' \subseteq T$ of weight $(\log m)^\alpha$ covering all groups.
3. If $I \notin L$, then every tree $T' \subseteq T$ covering all groups has weight $\Omega((\log m)^{3\alpha + 2})$.

4.2. Reduction. The next theorem describes the reduction to an instance of the localization problem.

Theorem 4.3. Let $L$ be any NP-complete language. Then there exist a constant $c_0$ and an algorithm $A'$ that, given an instance $I$ and a sufficiently large constant $\alpha$, produces in expected running time $O(|I|^{\alpha \log(|I|)})$ an instance $I'' = (G, H)$ of the robot localization problem on grid graphs such that the following hold.

1. For some $m \leq |I|^c_0$, $G$ has $N = m^{O((\log m)^{\alpha + 1})}$ cells and $H$ has $m^{O((\log m)^{\alpha + 1})}$ hypotheses.
2. For some $\beta = m^{O((\log m)^{\alpha + 1})}$,
   (a) if $I \in L$, then there exists a localization plan with worst-case cost $O(\beta \cdot (\log m)^\alpha)$, and
   (b) if $I \notin L$, every localization plan has cost $\Omega(\beta \cdot (\log m)^{3\alpha + 2})$.

Proof. Let $I' = (T(V, E), r, G)$ be the instance of Group Steiner on HSTs obtained by running algorithm $A$ on $I$ (see Theorem 4.2 above). Let $d, H$, and $\tau$ denote the
 arity, height, and weight factor of HST $T$, and let $k$ denote the number of groups in $G$. $G$ consists of $k + 1$ (disjoint) copies $B_0, B_1, \ldots, B_k$ of grid graph $B$, where $B$ is an “embedding” of HST $T$ respecting the weights on its edges.

The embedding $B$ is best described inductively. Let $B(v)$ denote the embedding of the subtree rooted at vertex $v \in T$. Cell $c_v$ at the southwest corner of each $B(v)$ corresponds to vertex $v$. For a leaf $l$, $B(l)$ is a $3 \times (\lceil \log k \rceil + 5)$ rectangle with a single traversable cell $c_l$ at its southwest corner (Figure 4.1(b)). The reason for adding blocked space to $c_l$ will be clear later, when we use it to add “signatures” to leaf $l$. For a nonleaf vertex $v$, $B(v)$ is formed by combining the embeddings of the subtrees rooted at its $d$ children $v_1, v_2, \ldots, v_d$ (see Figure 4.1(a)). $B(v_1), B(v_2), \ldots, B(v_d)$ are positioned along the top edge of $B(v)$ separated by north-south walls of width 1. There is an east-west corridor $ew$ running along the bottom edge of $B(v)$. Cell $c_{v_0}$ is connected to this corridor by a north-south corridor $ns$, which corresponds to edge $vv_0 \in T$. We make the length of $ns_{v_0}$ proportional to the weight of $vv_0$: if $v$ is at level $h$, $|ns_{v_0}| = \beta \cdot \frac{1}{\tau^h}$, where $\beta$ is a scaling factor to be chosen later. Finally, $B = B(r)$, where $r$ is the root of $T$.

Let $a_h, b_h$ be the length and breadth of the grid required to embed the subtree rooted at a level $h$ vertex $v \in T$. To see that the tree “fits”, observe that $B(v)$ fits into an $a_h \times b_h$ rectangle, where $a_h = d \cdot a_{h+1} + (d - 1)$ and $b_h = b_{h+1} + \frac{\beta}{\tau^h}$. Hence $b_h = (\lceil \log k \rceil + 5) + \beta \cdot \sum_{\alpha=h}^{H-1} \frac{1}{\tau^\alpha}$, and by induction one can show that $a_h = 4 \cdot d^{H-h} - 1$.

Let $w_{xy}$ denote the weight of the path connecting $x, y \in T$. Let $P_{uv}$ be the unique path connecting cells $c_u$ and $c_v$ in $B$. We show that choosing $\beta = 5d^H \cdot \tau^H$ makes $B$ an embedding of $T$ in the following sense: for all vertices $x, y \in T$, $\beta \cdot w_{xy} \leq |P_{xy}| \leq 2\beta \cdot w_{xy}$. First observe that the length of any north-south corridor $ns_u$ is now at least $5d^H$ while any east-west corridor is less than $4d^H$. Therefore, $|ew_x| \leq |ns_u|$ for all $u, v \in T$. We charge the distances traveled along east-west corridors to the north-south corridors immediately preceding it. First assume that $x$ is the parent of $y$. Then $P_{xy}$ consists of the north-south hallway $ns_y$ along with the portion of $ew_x$ connecting $c_x$ to $ns_y$. Clearly, $\beta \cdot w_{xy} = |ns_y| \leq |P_{xy}| \leq |ns_y| + |ew_x| \leq 2\beta \cdot w_{xy}$. Next consider the case when $x, y$ are siblings with common parent $z$. $P_{xy}$ consists of north-south corridors $ns_x, ns_y$ along with the portion of $ew_z$ connecting them. Hence, $\beta \cdot w_{xy} = \beta \cdot (w_{xz} + w_{zy}) = |ns_x| + |ns_y| \leq |P_{xy}| \leq |ns_x| + |ns_y| + |ew_z| \leq 2\beta \cdot w_{xz} + \beta \cdot w_{zy} \leq 2\beta \cdot w_{xy}$. For general $x, y$, let $c_{z_0=x}, c_{z_1}, \ldots, c_{z_m=y}$ be the cells corresponding to vertices of $T$, in the order they occur along path $P_{uv}$. By the construction of $B$, we know that for each $i$ either (i) $z_{i+1}$ is a parent of $z_i$ or vice-
versa, or (ii) $z_i, z_{i+1}$ are siblings. Therefore, $\beta \cdot w_{z_i, z_{i+1}} \leq |P_{z_i, z_{i+1}}| \leq 2\beta \cdot w_{z_i, z_{i+1}}$. Since $|P_{xy}| = \sum |P_{z_i, z_{i+1}}|$, the length of $P_{uv}$ is within a factor of 2 of $\beta \cdot \sum w_{z_i, z_{i+1}} = \beta \cdot w_{xy}$.

Let $g_1, g_2, \ldots, g_k$ be the $k$ groups in $G$. We make $k+1$ copies $B_0, B_1, B_2, \ldots, B_k$ of embedding $B$. $B_i$’s are the same except for distinguishing “signatures” at some leaf copy and contains no signatures. For $i > 0$, $B_i$ is formed by adding signature $s_i$ (a binary encoding of $i$) to every leaf block $B(l)$ of $B$ such that $l \in g_i$ (Figure 4.1(b)). To add $s_i$, first cell $c_l$ is extended to a north-south corridor along the left edge of $B(l)$. Then a set of log $k$ eastern “alcoves” encoding $i$ in binary are placed along the eastern edge: the $j$th alcove from the top is blocked if and only if the $j$th bit in the binary form is 0. A robot located at $c_l$ can read the value of $i$ by going north and sensing the alcoves to its right for blockage.

Let $x = 2 \cdot a_0 \cdot b_0$. Grid graph $G$ is an $(x + a_0) \times ((k+1) \cdot b_0 + k - 1)$ rectangle formed by connecting group blocks $\{B_i\}_i$, as shown in Figure 4.2. $B_0, B_1, B_2, \ldots, B_k$ are placed along the right edge of $G$ separated by east-west walls of width 1. A north-south corridor $NS$ of width 1 runs alongside the left edge of $G$. The southwest cell of each block $B_i$ is connected to this corridor by an east-west corridor $EW_i$ of length $x$. The set of hypotheses $H$ equals $\{h_0, h_1, \ldots, h_k\}$, where $h_i$ denotes the cell at the southwest corner of block $B_i$. Substituting values of $k, H, d, \tau$ as given by Theorem 4.2, we get $\beta = 5d_2^\alpha \cdot \tau H = mO((\log m)^{n+1})$, $|G| = O(a_0 b_0^2 k) = n2^O((\log m)^{n+1})$, and $|H| = k = m2^O((\log m)^{n+1})$, where $m \leq |T|^{20}$. We complete the proof by showing that the optimal localization plans for $T'' = (G, H)$ in the “yes” ($\mathcal{I} \in L$) and “no” ($\mathcal{I} \notin L$) cases differ by a factor of $\Omega((\log m)^{2n+2})$.

“Yes” case. Suppose $\mathcal{I} \in L$. By Theorem 4.2, there exists a tree $T' \subseteq T$ of weight $(\log m)^n$, which covers all groups in $G$. As all groups $g \in G$ consist of leaves of $T$, without loss of generality every root to leaf path in $T'$ ends at a leaf of $T$. Let $l_0, l_1, \ldots, l_{t-1}$ be the leaves of $T'$ in the order they are visited by a depth-first search from the root. Consider the following plan: read the signatures at leaf blocks $B(l_0), B(l_1), \ldots, B(l_{t-1})$ in that order. As soon as a nonzero signature $s_{t_0}$, $t_0 > 0$ is read, localize to $h_{t_0}$. Otherwise, localize to $h_0$. To prove correctness, assume the robot was placed (without its knowledge) at hypothesis $h_{t_0}$. If $t_0 = 0$, the robot will read zero signatures at all leaf blocks and correctly localize to $h_0$. Suppose $t_0 > 0$. Since $T'$ covers all groups, group $g_{t_0}$ contains at least one leaf vertex from $T'$. The robot will read signature $s_{t_0}$ at the first such
vertex in the sequence \(l_0, l_1, \ldots, l_{i-1}\) and localize to \(h_{i0}\).

The total travel cost of the robot is \(|P_{r, l_0}| + \sum_{i=0}^{t-2} |P_{l_i, l_{i+1}}| \leq 2\beta(w_{rl_0} + \sum_{i=0}^{t-2} w_{l_i l_{i+1}}) \leq 2\beta \cdot w(T) = O(\beta \cdot (\log m)^{\alpha+2})\). We neglect the cost of reading signatures at \(l_i\), as it is \(O(t \cdot \log k) = O(d^2 \log k) \leq \beta\).

"No" case. Suppose \(I \notin L\). Assume that we have found a localization plan with cost \(o(C \cdot (\log m)^{3\alpha+2})\). The number of movements for the plan is no larger than the length of an east-west hallway \(EW_i\). Now assume the robot starts at cell \(h_0\). Thus, it cannot visit a different east-west hallway and, as part of the localization, must determine that no leaf block in its group block has a nonzero signature. Let \(B(l_0), B(l_1), \ldots, B(l_{t-1})\) be all the leaf blocks, in the order they are visited by the robot. The collection of groups that these leaves cover must equal \(G\), for otherwise the robot could not distinguish between hypotheses \(h_0\) and \(h_i\) for the groups \(g_i\) not covered by them.

Let \(T'\) be the Group Steiner tree formed by taking the union of paths connecting \(r\) to \(l_0\) and \(l_i\) to \(l_{i+1}\) for \(0 \leq i \leq t-2\). By Theorem 4.2, the weight of \(T\) is \(\Omega((\log m)^{2\alpha+2})\). Therefore, the cost of the localization plan is at least \(|P_{r, l_0}| + \sum_{i=0}^{t-2} |P_{l_i, l_{i+1}}| \geq \beta \cdot (w_{rl_0} + \sum_{i=0}^{t-2} w_{l_i l_{i+1}}) \geq \beta \cdot w(T') = \Omega(\beta \cdot (\log m)^{3\alpha+2})\).

**Corollary 4.4.** For every fixed \(\epsilon > 0\), the robot localization problem cannot be approximated within ratio \(\log^{2-\epsilon} N\) on grid graphs of size \(N\) unless \(NP \subseteq ZTIME((n^{\text{polylog}(n)})\).

**Proof.** Apply the algorithm in Theorem 4.3 with \(\alpha = 2 \cdot (\frac{1}{\epsilon} - 1)\). The logarithm of the size of grid graph \(G\) is \(\log N = O((\log m)^{\alpha+2})\), where \(m \leq n^{\alpha}\). The optimum localization plans in the "yes" and "no" cases differ by a factor of \(\Omega((\log m)^{2\alpha+2}) = \Omega((\log N)^{2-\epsilon})\).

**Corollary 4.5.** For every fixed \(\epsilon > 0\), the robot localization problem cannot be approximated within ratio \(\log^{2-\epsilon} N\) on polygons with \(N\) vertices unless \(NP \subseteq ZTIME((n^{\text{polylog}(n)})\).

**Proof.** The grid graph \(G\) in Theorem 4.3 above can be viewed as a polygon \(P\) with at most \(N\) vertices. Let \(h'_i\) denote the center of the cell \(h_i\) in \(G\). Consider the localization problem on \(P\) with hypotheses set \(H' = \{h'_0, h'_1, \ldots, h'_k\}\). The optimal localization plan in the "yes" case has cost \(O(\beta \cdot (\log m)^{\alpha})\), as a robot with a range finder can only do better. However, when \(I \notin L\), a robot with a range finder may read the signatures from a distance and localize at lesser cost. To rule this out, put small "twists" in polygon \(P\) just before every signature. Thus the robot cannot read the signatures at a distance and therefore will travel at least \(\Omega((\log m)^{2\alpha+2})\) distance, as in Theorem 4.2 above. The "yes" and "no" cases differ by \(\Omega((\log m)^{\alpha+2})\), and the bound follows by choosing \(\alpha = 2 \cdot (1 - \frac{1}{\epsilon})\).

We note that a lower bound for Group Steiner can be extended to a similar lower bound for localization on grid graphs. The main idea is the same as above: take a hard instance \((G, r, G)\) of Group Steiner on grid graphs. Suppose \(G\) is an \(m \times n\) grid graph, and there are \(k = |G|\) groups. We make a map \(G'\) that consists of \(k\) disjoint copies \(G_1, G_2, \ldots, G_k\) of \(G\). Each copy \(G_i\) is a scaled up (by a factor of \(\beta\)) version of \(G\). Thus, each cell of \(G\) corresponds to a \(\beta \times \beta\) block in \(G_i\). As before, we choose the scaling factor large enough so that the distance between signatures is much larger than their size. A good choice is \(\beta = k\). Initially, the robot is placed at the center of block corresponding to \(r\) in one of the \(G_i\)'s.

In order to localize, the robot has to find the index of its component and, hence, must visit a set of blocks that covers all of the groups. This path can be translated
to a Group Steiner tree of proportional cost (divided by $\beta$) in $G$ (since $\beta$ is much larger than $\log k$). Conversely, we can convert any Group Steiner tree in $G$ into a path by doing depth-first search and then using that path in the scaled grid $G'$ as a localization plan. It is easy to see that this extends the same hardness factor to localization on grids.

Thus, it seems that further improvement (in either the lower bound or upper bound) in the approximation factor of our algorithm can come only after progress on the Group Steiner problem on grid graphs.

5. Extensions to other models. Here we sketch some extensions of our algorithm.

5.1. Robot without compass. If the robot does not possess a compass but has no actuator uncertainty with respect to changes in orientation, the lower bound remains valid. For the algorithm, redefine a hypothesis to be a (location, orientation) pair. For grid graphs, with four axis-parallel orientations per cell, the size of the set $H$ of possible hypotheses remains $O(n)$, and the algorithm extends naturally as the robot operates on the majority-rule map relative to (location, orientation) pairs in $H$.

For polygons there are at most $n$ distinct embeddings, corresponding to rotations, of the visibility polygon $\mathcal{V}(h_i)$ for each choice of $h_i$. This follows since any one edge of $\mathcal{V}(h_i)$ that is not collinear with $h_i$ (as is the case for “shadow edges” or “windows” of $\mathcal{V}(h_i)$) must fall on one of the $n$ edges of $P$ in any candidate pose. Thus, $H$ consists of at most $n$ different poses, $(h_i, \theta_j)$, each specified as a (location, orientation) pair. For each pose $(h_i, \theta_j)$, we construct a copy $P_{i,j}$ of the map polygon $P$. $P_{i,j}$ is formed by first translating $P$ so that $h_i$ coincides with the origin, and then rotating it about $h_i$ so that direction $\theta_j$ points to the north. The majority-rule map and Algorithm 3 are then directly applied to the polygons $P_{i,j}$, as in the translation-only case.

5.2. The limited-range version. Practical sensors have a limited range, $D$, beyond which the noise levels are too high to give reliable measurements [31]. Our algorithm for grids already assumes limited range of visibility, since we assume that the robot senses only the immediate neighboring grid cell; this can readily be extended to allow the robot to sense all cells within grid graph distance $D$. Our algorithm for localization in polygons can also be extended to the limited-range case, as we now describe.

In order to distinguish between hypothesis $h_i$ and hypothesis $h_j$, the robot must get within distance $D$ of an edge of type (i) or type (ii) in the polygon $F_{i,j}$. If $\gamma_0$ sees (within distance $D$) any point on an edge of type (i) or type (ii), then the robot can distinguish between $h_i$ and $h_j$ without moving from the origin $\gamma_0$. Thus, assume that all edges of $F_{i,j}$ that are visible (within distance $D$) from $\gamma_0$ are of type (iii). Let $e$ be an edge of $F_{i,j}$ of type (i) or of type (ii). Assuming an unobstructed space, the set of points within distance $D$ from some point of $e$ is a region bounded by straight edges and circular arcs (of radius $D$). The portion of the boundary of $VP(D)(e)$ that separates $\gamma_0$ from $e$ defines the window, $w(e)$, of $e$; it consists now of $O(1)$ curves (straight segments and radius-$D$ arcs), instead of a single chord, as in the $D = \infty$ case. We now define $G^{(D)}_{i,j}$ to be the face containing the origin $\gamma_0$ in the arrangement of the $O(n)$ boundary edges of $F_{i,j}$ together with the set of all $O(n)$ windows $w(e)$ for edges $e$ of type (i) or (ii). Again, as in Lemma 3.3, we have that a robot can distinguish between hypothesis $h_i$ and hypothesis $h_j$ if and only if it visits the boundary, $\partial G^{(D)}_{i,j}$. This allows us to define the majority-rule map regions $g_i$, the discrete point set $Q_H$, and the half-localization algorithm as before. The only technical difference is the
presence of straight segments and (fixed radius) circular arcs in the arrangements; this does not affect the polynomiality or the correctness of the algorithm.

5.3. Polygons with holes. In the case that the map polygon $P$ is a polygonal domain with holes $H_1, H_2, \ldots, H_m$, our method still applies, but the complexity of the structures increases. Let $n$ be the total number of vertices in the polygon (including the holes).

First, the polygons $F_{ij}$ are now polygons with holes of complexity $O(n)$. A single edge can now have as many as $O(n)$ windows (one for each hole). The $G_{ij}$'s are formed as before by taking the intersection of the regions chopped off by these windows; each has complexity $O(n^2)$ since each window-edge can occur only once. Finally, the majority-rule map's $K_i$'s are also polygons with holes; each is formed from $G_{ij}$, $j \neq i$, and hence has complexity $O(k^2 n^4)$. The specification of the discrete points $Q_H$ applies to the case of multiply connected domains, and the argument of Lemma 3.6 applies as well. And hence the set of coordinates now has complexity $O(k^3 n^4)$. These bounds can be improved somewhat when the holes are convex.

From the above section, it is clear that the same framework works for a robot with limited range sensors inside a polygonal map with holes.

5.4. Geometric trees. As described before, a geometric tree $G = (V,E)$ is a tree with $V$, a set of points in $\mathbb{R}^d$, and $E$, a set of nonintersecting line segments whose endpoints all lie in $V$ [27]. An $O(n^2)$-competitive localization strategy for bounded-degree geometric trees was given by Kleinberg [27]. His strategy is $\Omega(n)$-competitive for trees with arbitrary degree.

Our approach gives an $O(\log^2 n \log k)$ algorithm for any geometric graph $G = (V,E)$ in the plane, not just trees. First, we can assume that the robot begins at some vertex of $G$, since the robot can initially perform a two-way spiral search to reach the closest vertex, while traveling at almost 9 times the cost of the optimal strategy [1, 27]. The set of hypotheses is now of size $k = O(n)$ and consists only of vertices. We make $k = |H|$ translation-congruent copies of $G$, with the $i$th copy $G_i$ having $h_i$ at the origin. To construct the majority-rule map, we overlay the copies $G_i$, $1 \leq i \leq k$, and form the arrangement $D(G,H)$ of line segments in $\bigcup G_i$. Each edge in the arrangement has the same hypothesis partition, and hence the robot gains new information only by visiting new edges. Note that several edges may be collinear, since new points are added by translation. Next, we construct the majority-rule map $G_{maj}$ by finding the set of all half-traversable edges reachable from the origin $\gamma_0$. As the robot can visit an edge only through one of its endpoints, an $O(\log^2 n)$-factor halving path can be found by solving the $\frac{1}{2}$-Group Steiner problem on vertices of $G_{maj}$. Since $D(G,H)$ is formed by the intersection of $k \cdot |E|$ edges, it has complexity at most $O(k^2 |E|^2)$ and can be computed in $O(k^2 |E|^2)$ time by standard methods [2]. Since at least one endpoint of every edge in the majority-rule map $G_{maj}$ is a vertex $v \in \bigcup G_i$, $G_{maj}$ has $O(n|E|)$ edges. Hence the computation time of the robot is $P(n|E|) \cdot \log n$, where $P(\cdot)$ is the running time for approximating the $\frac{1}{2}$-Group Steiner problem. Since a grid graph is a geometric graph in $\mathbb{R}^2$, Corollary 4.4 gives an $\Omega(\log^2 - \epsilon n)$ lower bound.

5.5. Three-dimensional grid graphs. Finally, we consider a three-dimensional grid graph $G$, which can be used to model buildings or offices with several floors. The majority-rule map $G_{maj}$ is a local map in which each cell is blocked or unblocked based on what the majority of hypotheses have to say about it. If $G$ is an $l_1 \times l_2 \times l_3$ cuboid, the majority-rule map has size $(2l_1 - 1) \times (2l_2 - 1) \times (2l_3 - 1)$, since
the absolute values the $x$-, $y$-, and $z$-coordinates for each hypothesis are $l_1 - 1$, $l_2 - 1$, and $l_3 - 1$, respectively. Hence $G_{\text{maj}}$ requires space at most $8n$ and can be computed in $O(nk)$ time. By making one vertex for each cell in $G_{\text{maj}}$, we solve the $\frac{k}{2}$ Group Steiner problem $I_{G,H}$ of size $O(n)$. The performance ratio remains $O(\log^2 n \log k)$, and the running time is $\mathcal{O}(8n) \log n$.

6. Is returning to the origin necessary? In this section, we show that RHL performs very poorly if we do not stipulate that the robot return to its starting position after each half-localize step. In section 6.1, we construct a grid graph $G$ and a set of hypotheses $H$ such that a robot not returning to the origin travels distance $(k - 1 - \epsilon) \cdot OPT(G,H)$, where $k = |H|$ is the number of hypotheses and $\epsilon > 0$ is an arbitrarily small constant. In section 6.2, we show that our lower bound is tight by proving that a robot not returning to the origin always localizes in at most $(k - 1) \cdot OPT(G,H)$ steps.

6.1. Lower bound. The grid graph $G$ for the lower bound is reminiscent of the Group Steiner tree construction. Let $k$ be the number of hypotheses and $x$ be an integer greater than or equal to 3.

The building block $B$ consists of two orthogonal corridors meeting at a corner where the robot is located (see Figure 6.1(a)). The northern corridor has length $x + 1 + \log k$, and the eastern corridor has length $(k - 1) \cdot x$.

![Fig. 6.1. (a) Block $B_y$, where $y = 10 \ldots 0$. (b) The northern corridor is bent at 3 units.](image)

We make $k$ copies $B_0, B_1, \ldots, B_{k-1}$ of block $B$. $B_y$’s are the same except for distinguishing “alcoves” along their northern and eastern corridors.

We now describe the construction of $B_y$. A set of $\log k$ alcoves encoding $y$ in binary are added along the western edge of the northern corridor (see Figure 6.1(a)). The $j$th alcove from the bottom is blocked if and only if the $j$th bit in the binary encoding of $y$ is 0.

In addition to this, we add $\log k$ alcoves encoding $y$ in binary along the eastern corridor. The $i$th alcove is placed at distance $(2^i - 1)x$ and is blocked if and only if the $i$th bit in the binary encoding of $y$ is 0.

Observe that $B_y$ fits into an $a \times b$ rectangle, where $a = 2 + x \cdot (2\log k - 1) = 2 + x \cdot (k - 1)$ and $b = x + \log k$. One can further reduce the height of $B_y$ by bending the northern corridor as shown in Figure 6.1(b). After this reduction, each block fits into a $(2 + x \cdot (k - 1)) \times 5$ rectangle.

Grid graph $G$ is a $(2 + x \cdot k + \log k) \times (6k - 1)$ rectangle formed by connecting blocks $\{B_y\}_y$ as shown in Figure 6.2. $B_0, B_1, \ldots, B_{k-1}$ are placed along the left edge of $G$ separated by east-west walls of width 1. A north-south corridor of width 1 runs alongside the right edge of $G$. The southwest cell of each block is connected to this corridor by an east-west corridor of length $x + \log k$. This distance is chosen so that a
robot located inside a block \( B_y \) never goes outside it to half-localize. Finally, the set of hypotheses \( H \) equals \( \{ h_0, h_1, \ldots, h_{k-1} \} \), where \( h_y \) denotes the cell at the intersection of the two orthogonal corridors in \( B_y \).

**Theorem 6.1.** Let \( G \) be the \( (2 + x \cdot k + \log k) \times (6k - 1) \) grid graph as constructed above and \( H = \{ h_0, h_1, \ldots, h_{k-1} \} \). Then a robot which computes the optimal half-localization strategy but does not return to the origin travels to the end of the eastern corridor of its block \( B_y \) before it localizes.

**Proof.** Consider a robot located at \( h_y \), where \( 0 \leq y \leq k - 1 \). To find its location the robot needs to find all bits in the binary representation of \( y \). To half-localize, it suffices to read one new bit in each phase.

The robot can read either the first alcove on the northern corridor or the first alcove on the eastern corridor. Since the alcove on the eastern corridor is nearer by one grid cell, the robot moves \( x \) units east and “reads” the first alcove.

Suppose that the alcove is blocked, i.e., the first bit in the binary encoding of \( y \) is 0 (the case when it is 1 is similar). To read the next bit, either the robot can read the second alcove on the eastern corridor at cost \( 2x \), or it can go back to the origin and then read the second alcove on the northern corridor at cost \( 2x + 2 \). Since the former is optimal, the robot moves \( 2x \) units west to read the second alcove on the eastern corridor.

In general, at the start of the \( i \)th phase \( (i \geq 2) \) the robot has read the first \( i - 1 \) bits of the binary encoding of \( y \) and is located at alcove \( i - 1 \) on the eastern corridor. Either it can move \( 2^{i-1}x \) steps to the west and check the \( i \)th alcove on the eastern corridor, or it can go back to read the \( i \)th alcove at the northern corridor at cost \( 2^{i-1}x + i \). The optimal half-localization plan consists of going west to read the \( i \)th alcove on the eastern corridor.

Thus in each half-localize phase the robot goes west to read the next alcove on its eastern corridor. The robot will localize after it has gone until the end of the eastern corridor and checked the last alcove. The total distance traveled by the robot is \( (k - 1) \cdot x \). \( \square \)

**Corollary 6.2.** For every fixed \( \epsilon > 0 \), there is a grid graph \( G \) and a set of hypotheses \( H \) such that a robot following RHL without returning to the origin travels at least \( (k - 1 - \epsilon) \cdot OPT(G, H) \) distance before determining its location \( h \in H \).

**Proof.** Take the grid graph \( G \) and the set of hypotheses \( H \) constructed above. By Theorem 6.1 a robot not returning to the origin travels distance \( (k - 1) \cdot x \) to the end of the eastern corridor to determine its initial position \( h_y \in H \).

The optimal localization strategy consists of going \( x \) units up the northern corridor.
and then reading the log $k$ bit signature. It has cost $OPT = x + \log k$.

The approximation factor is $\frac{(k-1)x}{x+\log k}$. If we take $x = \frac{(k-1)\log k}{e}$, this is at least $k - 1 - \epsilon$. \[ \]

The next corollary shows that the lower bound in terms of the size of the grid graph is $\Omega(\sqrt{n})$.

**Corollary 6.3.** There is a grid graph $G$ and a set of hypotheses $H$ such that a robot following RHL without returning to the origin travels distance $\Omega(\sqrt{n}) \cdot OPT$, where $n = |G|$ is the size of the grid graph.

**Proof.** Take $x = 3$ in Theorem 6.1. The grid graph now has size $n = (2 + 3k + \log k) \times (6k - 1) = \theta(k^2)$. The optimal localization strategy has cost $3 + \log k$. The robot travels distance $3(k - 1)$. The approximation factor is $\frac{3(k-1)}{3+\log k}$. Since $k = \theta(\sqrt{n})$, this is $\Omega(\sqrt{n})$. \[ \]

### 6.2. Upper bound

We now show that a robot following strategy RHL without returning to the origin localizes in at most $(k-1) \cdot OPT$ steps.

**Theorem 6.4.** Consider a robot which computes the optimal half-localization strategy in each phase but does not return to the origin after each phase. Then it travels distance at most $(k-1) \cdot OPT$ before determining its initial position $h \in H$, where $k = |H|$ is the number of hypotheses and OPT is the cost of the optimal localization plan.

**Proof.** Let $P_{i-1}$ denote the path traced by the robot relative to the origin before the start of the $i$th half-localize phase. Let $Q_i$ denote the path traced by the robot during the $i$th phase. Then we have that $P_i = P_{i-1} \circ Q_i$ is the concatenation of $P_{i-1}$ followed by $Q_i$.

Since the robot always chooses the optimal half-localization strategy, the length of $Q_i$ is less than or equal to any half-localization strategy for phase $i$. One such strategy makes the robot retrace the path $P_{i-1}$ back to the origin and then run the optimal localization plan until the robot half-localizes. This has cost at most $|P_{i-1}| + OPT$, and hence we have that $|Q_i| \leq |P_{i-1}| + OPT$.

Therefore, we get that $|P_i| = |P_{i-1} \circ Q_i| = |P_{i-1}| + |Q_i| \leq |P_{i-1}| + (|P_{i-1}| + OPT) = 2 \cdot |P_{i-1}| + OPT$.

Since $|P_0| = 0$, we see that $|P_i| \leq (2^i - 1) \cdot OPT$. As the robot localizes in at most $m \leq \log k$ half-localize steps, the distance traveled by $P_m$ is at most $(2^m - 1) \cdot OPT \leq (\Theta(k) \cdot OPT = (k-1) \cdot OPT$. \[ \]

### 6.3. Discussion

This feature may become a problem in probabilistic environments where the robot may incur noise by returning back to the origin, or if the robot gets trapped in a small corner from which it is hard to get out. However, we still feel that our algorithm makes sense, due to the large decrease in uncertainty brought about by each half-localize step. If the robot motion is sufficiently correct, this decrease should more than offset the noise incurred by coming back to the origin. Further, the robot does not need to return to the origin “exactly.” Rather, it suffices that the robot be present within a small distance of the origin with high probability, as this will allow for near-optimal behavior in the next step. If continuous sensing and updating while returning back is allowed, the robot should perform reliably with small corrections.

The robot may get trapped in a corner, but in maps with “signatures” (such as those we constructed above, as well as those in the NP-hardness construction of Dudek, Romanik, and Whitesides [17]), this may be the only way to localize efficiently.
In fact, it seems that only in such highly-replicated environments do such localization strategies make sense.

Further, the task of localization is just a prelude to the robot performing other tasks, such as going to a particular location. This new location may lie anywhere in the map, so the robot will not lose by coming back to its starting place.

Finally, it seems that there is no way to bypass this return-to-origin constraint, as not allowing the robot to return to the origin leads to exceptionally bad performance. In fact, we claim that no reasonable algorithm for localization can be found unless we stipulate that the robot returns to the starting location after each half-localize step.

7. Conclusion and open problems. The main ideas of this paper are half-localization and the majority-rule map, which permit us to eliminate half the hypotheses in each step. Earlier strategies for localization could eliminate only $O(1)$ hypotheses in each step, thus leading to $\Omega(n)$-approximations for general models. There is a $\log n$ factor gap between the upper and lower bounds; it appears that this gap can be closed only by progress on the Group Steiner problem in grid graphs (and also those given by Euclidean shortest path metrics inside a constrained region).

An appealing feature of our algorithm is its wide adaptability over a variety of robot models: the only issue is to devise algorithms for computing the majority-rule map and the set of coordinates for the model at hand. We believe that the majority-rule map will play an important part in other robot navigation problems.

While our algorithms for localization in polygons have been restricted to two dimensions, we expect that the results can be extended to three-dimensional polyhedral domains $P$ in which the robot moves inside $P$ and sends out a series of beams spaced at small solid angles over the sphere and joins them to compute the visibility polyhedron $V(p)$. Modern three-dimensional range finders allow one to estimate the visibility polyhedron from the robot [49].

In this paper, we do not address models with sensor noise, imperfections in the robot’s map, and odometer errors. While sensor noise can be easily accommodated [5], devising a good strategy for a model with odometer errors remains a major open problem. This not only entails redefining what we mean by localization but also requires devising strategies that balance the need for resolving global position with the need for removing local pose estimation errors.

Acknowledgment. We thank the anonymous referees for valuable comments that helped us improve the paper.

REFERENCES

ROBOT LOCALIZATION PROBLEM


