

# Intensity-Based Navigation with Global Guarantees

Kamilah Taylor · Steven M. Lavelle

the date of receipt and acceptance should be inserted later

**Abstract** This article introduces simple, information-feedback plans that guide a robot through an unknown obstacle course using the sensed information from a single intensity source. The framework is similar to the well-known family of bug algorithms; however, our plans require less sensing information than any others. The robot is unable to access precise information regarding position coordinates, angular coordinates, time, or odometry, but is nevertheless able to navigate itself to a goal among unknown piecewise-analytic obstacles in the plane. The only sensor providing real values is an intensity sensor, which measures the signal strength emanating from the goal. The signal intensity function may or may not be symmetric; the main requirement is that the level sets are concentric images of simple closed curves. Convergence analysis and distance bounds are established for the presented plans. Furthermore, they are experimentally demonstrated using a differential drive robot and an infrared beacon.

## 1 INTRODUCTION

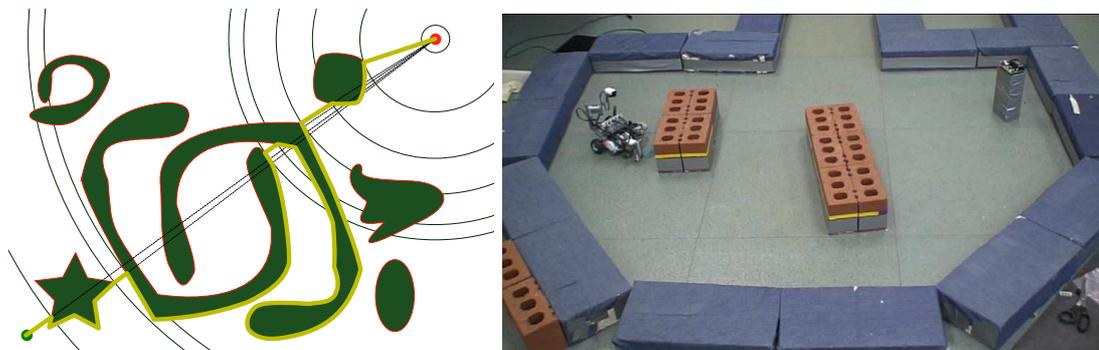


Fig. 1: On the left, a simulated robot starts at the lower-left green dot and moves towards the upper-right red dot while traversing various obstacle boundaries. Some level sets of equal intensity are represented by the circular arcs. On the right, the robot from our experiments is shown in a test environment.

---

K. Taylor  
Department of Computer Science  
University of Illinois  
E-mail: kamilah@gmail.com *Present address:* LinkedIn, Mountain View, CA, USA.

S. M. Lavelle  
Department of Computer Science  
University of Illinois  
E-mail: lavelle@uiuc.edu

We increasingly use devices that rely on all types of signals. Various portions of the radio wave spectrum are used, for example, by submarines, wireless heart monitors, radios, televisions, mobile phones, and bluetooth. Navigation based on traditional sources such as GPS and lasers may obscure other information sources. Imagine the possibilities that emerge once we understand what can be accomplished using non traditional sources. The main objective of this research is to investigate what information from this signal is minimally needed for global navigation. Figure 1 shows examples of both a simulated robot and an experimental robot that executes a strategy based on incrementally maximizing a single intensity while moving among unknown obstacles. The robot is then able to go to the goal, which we call a *tower*, without using the coordinates of its position or of the tower. This could represent a robot operating in a GPS-denied environment, possibly for a search-and-rescue mission.

The remainder of the paper proceeds as follows. Related literature is surveyed in Section 2. Following this, Section 3 introduces a coordinate-free mathematical model for a robot that navigates based on the *intensity* of a *signal* emanating from the goal, the *tower*. The *intensity function* is introduced, which describes how the intensity of the signal varies with distance. This model will prove sufficient for the robot to be guaranteed to reach the goal.

Section 4 presents a solution for the case of a *radially symmetric* intensity function. The resulting plan guarantees that the plan converges, implying that the robot always reaches the tower *if the tower is reachable*, and an upper bound on the total distance traveled is given. It is then shown that the robot with just an intensity and contact sensor cannot *decide* whether the tower is reachable. Section 5 addresses the more general case of an *asymmetric* intensity function. In practice a robot will most likely use the asymmetric plan because there are few real world examples of a perfect symmetric signal. This generalized plan also has a convergence proof, but the looser bound is related to the bound for the well known optimization technique *steepest descent with line searching* [40]. Section 6 presents experimental verification of the plans. Finally, 7 present conclusions and potential directions for future research. This paper is expanded from [54].

## 2 RELATED WORK

Our work is inspired by two main areas of previous literature: 1) Research on using scalar signals or the gradient of signals for navigation, and 2) the design of minimalist robots.

There is a growing amount of work that uses gradient information for navigation. Odor localization focuses on robots that search for a source using chemical sensing [26, 27, 52]. These are largely biologically inspired because moths [35], lobsters [22], and *Escherichia coli* bacteria [51] all use odor localization to locate mates and food. A survey of robot odor localization can be found in [34]. Leaving the animal kingdom, in [23], the gradient of a Wi-Fi signal is approximated using a technique called *wardriving*. A moving vehicle collects received signal strength measurements (RSS), and then the direction of an Access Point (AP) at a measuring point is estimated by calculating the gradient at that point. This gradient is given by the direction of the strongest signal in the neighborhood. The algorithm was experimentally verified, but it is possible that its success is due in part to the experiment being conducted outdoors. Wi-Fi signals suffer from problems such as multi-path fading in indoor environments [24]. Even sound can be used for localization [20, 45, 47, 48]. Robots in [44] use noise maps and the noise distribution, and build a gradient field from the noise map to perform tasks such as avoiding a sound source.

The most common strategy in mobile robotics is to use powerful sensors that can reconstruct a complete map of the environment and perform localization directly in the map frame coordinates [56]. In an outdoor setting, GPS is commonly used for localization. As noted in [57], however, there are cases when robots are not able use GPS for localization, such as in underground mines. These *GPS-denied environments* include most indoor environments [25]. What they found in [1] is that a lack of GPS is just one constraint; other common constraints are unknown environments and even payload limits. Powerful sensing can be used to compensate for these problems; however, there are associated costs, modeling burdens, energy consumption, software, and hardware integration issues. It is important to *understand* what information really necessary for the motion strategy of the robot to enable it to accomplish a predetermined task.

The goal then is clear: We consider robots that can accomplish tasks with as little sensing as possible. Examples of this in robotics literature include *on-line algorithms* [4, 8, 10, 11, 15, 33], *gap navigation trees*

[36, 46, 58], *sensorless manipulation* [12–14, 21], *bug algorithms*, and others such as [2, 53, 59]. All of these aim to reduce sensing requirements.

Among these works, the most closely related line of research to ours is the family of bug algorithms, which have two main modes of movement: Following obstacle boundaries and moving towards the goal. The original bug algorithms [43, 41] propose a minimalist sensing model and a robot navigation algorithm to bring the robot to a specified goal in a 2D environment with unknown smooth obstacles. The work was later extended to include a range sensor, which led to improved bounds on the total distance traveled [42]. Since then there have been several other bug algorithms. In [28, 30], the TangentBug was proposed, which enhanced the sensing model to improve the bound on the length of the path to the goal. In [29], TangentBug was extended to three dimensions. WedgeBug and its relative RoverBug [37–39] restrict the TangentBug sensing model so that it can be applied in an actual planetary rover. A bug algorithm for solving pursuit-evasion was presented in [49]. Bug1 is used as the sub-algorithm for the CBug family [17, 18], which modifies the original algorithm to give it quadratic competitive performance.

The motivation for our paper came from carefully studying the models of previous bug algorithms. Even though these models are aimed at minimizing sensing and mapping requirements, they appear to require some strong, precise information that may not be necessary. For example, the original Bug1 and Bug2 algorithms [41] use: 1) a contact sensor, 2) coordinates of the initial robot position, 3) coordinates of the current robot position, 4) coordinates of the target, and 5) odometry to obtain the distance traveled around an obstacle boundary. This information is evident when studying the particular approach. Bug1 goes around the entire obstacle, calculates the closest leaving point, returns to that point, and then goes in a straight line towards the target. Bug2 calculates an “m-line”, which is a line segment that connects the initial point to the goal point, and always moves on that line unless it is contacted an obstacle. While moving along an obstacle, it follows the boundary until it is once more on the m-line, and then it returns to moving towards the target on the m-line. Thus, it seems that the robot needs a position sensor, a linear odometer, and angular odometer to execute both the Bug1 and Bug2 algorithms. Bug2 would also need to calculate whether the obstacle has intersected the m-line. VisBug’s algorithm [41] is based on the Bug2 algorithm but uses a range sensor to decrease the Bug2 path bound. TangentBug uses a 360° range sensor to avoid following the boundary and instead move a certain distance away from the boundary unless it is unavoidable. WedgeBug is based on TangentBug and uses a more limited 30° to 45° range sensor to minimize the number of sensor readings.

A common theme among the bug algorithms is that they rely on knowing the robot’s exact coordinates. Our motivation comes from a simple observation: The previous bug robots had access to the exact coordinates of every place they visited, and in some cases to the exact distance they traveled. If they had had unlimited memory, it would even have been possible to reconstruct a perfect map of their environments. It seems that it ought to be possible to navigate through an environment without collecting all of this information. We therefore want to determine whether the robot can navigate to a goal without collecting all of this information. Can it reach the goal without having access to its coordinates?

### 3 PROBLEM FORMULATION

Suppose that a point robot (or sufficiently small disc) moves in  $\mathbb{R}^2$  according to a kinematic differential drive model, given by the following standard configuration transition equation

$$\begin{aligned}\dot{x} &= \frac{r}{2}(u_l + u_r) \cos \theta \\ \dot{y} &= \frac{r}{2}(u_l + u_r) \sin \theta \\ \dot{\theta} &= \frac{r}{L}(u_r - u_l).\end{aligned}$$

Here,  $u = (u_r, u_l)$  specifies the two angular wheel velocities. A differential drive robot consists of two independently controlled wheels attached to an axle (see Figure 2). By sending equal power to both identical wheel motors, the robot can move straight or rotate in place; therefore, these are the primitives that are available to our robot.

Let  $\mathcal{O}$  be a disjoint set of *obstacles*, in which each  $O \in \mathcal{O}$  is closed with a connected piecewise-analytic boundary that is finite in length. Furthermore, the obstacles in  $\mathcal{O}$  are pairwise-disjoint. There may be a countably infinite number of obstacles; however, at most a finite number are contained in any fixed disc

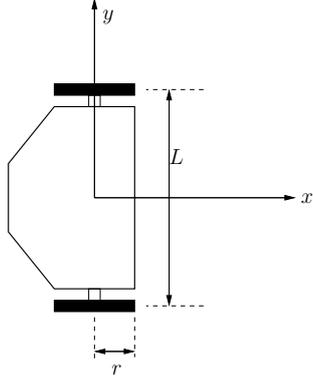


Fig. 2: A generic differential drive robot model.

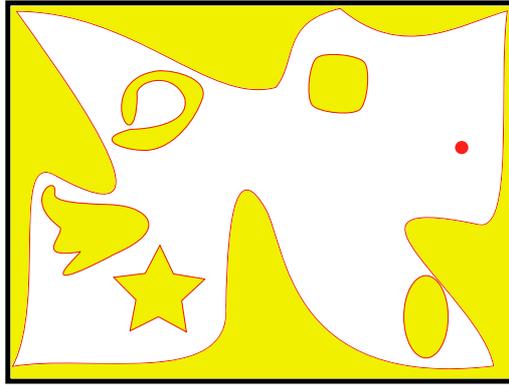


Fig. 3: There may be an outer obstacle  $O_{outer}$ , which has a finite-length boundary curve but extends infinitely outward in all directions.

(this property is called locally finite in [41]). The obstacle set  $\mathcal{O}$  may contain an *outer obstacle*  $O_{outer}$  that is unbounded; all other obstacles are bounded. See Figure 3.

Let  $E$  be the closure of  $\mathbb{R}^2$  minus all  $O \in \mathcal{O}$  and be called the *environment*. Note that the environment is connected and may or may not be bounded.

A point called the *tower* exists at some location  $(x_t, y_t) \in \mathbb{R}^2$ . The tower broadcasts a *signal*, which is modeled as an intensity function over  $\mathbb{R}^2$ . Let  $m$  denote the *signal mapping*  $m : \mathbb{R}^2 \rightarrow [0, 1]$ , in which  $m(p)$  yields the *intensity* at  $p \in E$ , generated from a tower at  $(0, 0) \in \mathbb{R}^2$ . It is assumed that the maximum intensity, 1, is achieved at the tower:  $m(x_t, y_t) = 1$ . If the robot is at  $p$ , then the intensity is translated accordingly as  $m(p - (x_t, y_t))$ . The actual tower location is irrelevant to the calculations because any coordinate can be translated to the actual location without changing the value of the intensity. Thus we will assume without loss of generality that the tower location is  $(0, 0)$ , which reduces the intensity to  $m(p)$ .

For any  $i \in [0, \infty)$ , consider the level sets (or preimages)

$$m^{-1}(i) = \{p \in \mathbb{R}^2 \mid m(p) = i\}. \quad (1)$$

We want to allow intensity functions that are as complicated as those measured in practice from radio signals or other physical sources. An important restriction, however, will be that we allow only one local maximum, which is at the tower. In spite of this, it will be assumed that  $m$  could be any locally Lipschitz, piecewise-analytic function for which  $m^{-1}(i)$  is homeomorphic to a circle, i.e. a topological circle, for every  $i \in (0, 1)$  and  $m^{-1}(1) = \{(0, 0)\}$  (this includes, for example, some polyhedral surfaces). Furthermore, the level sets must be concentric, with  $(0, 0)$  at the center. We make a general position assumption that for the boundary of every  $O \in \mathcal{O}$  and every preimage  $m^{-1}(i)$ , they are either disjoint or intersect in a finite number of places. Let  $M$  denote the set of all intensity functions that satisfy these conditions.

Let  $M_s \subset M$  denote the set of all *radially symmetric* intensity functions. In this case, the level sets form concentric circles in the classical sense (rather than concentric topological circles). As an example,

$$m(p) = \frac{1}{p_x^2 + p_y^2} \quad (2)$$

causes the intensity to decay quadratically with distance, without regard to direction. This corresponds to radiation patterns of isotropic radiators [50]. More generally, if the level sets are not concentric circles, then  $m \in M \setminus M_s$  is called *asymmetric*.

The environment  $E$  and even the signal mapping  $m$  are unknown to the robot. Furthermore, the robot does not even know its own position and orientation. Based on these quantities, a *state space*  $X$  is defined as

$$X \subset SE(2) \times \mathcal{E} \times M \quad (3)$$

in which  $SE(2)$  is the set of all possible robot positions and orientations,  $\mathcal{E}$  is the set of all possible environments, and  $M$  is the set of all possible intensity mappings.

Each sensor available to the robot will be defined as a mapping  $h : X \rightarrow Y$  from the state space  $X$  into an *observation space*  $Y$ . Three main sensors will be considered. First, the *contact sensor* indicates whether the robot is touching the environment boundary  $\partial E$ :

$$h_t(x) = h(p, \theta, E, m) = \begin{cases} 1 & \text{if } p \in \partial E \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The other two sensors obtain information regarding the tower. The *intensity sensor* indicates the strength of the signal from position  $p$ :

$$h_i(x) = h(p, \theta, E, m) = m(p). \quad (5)$$

The robot can use the intensity sensor to determine when it is at the tower, which uniquely occurs when  $h_i(x) = 1$ . However, if the robot does not know the maximum possible intensity, then a ‘‘tower detection sensor’’ can be added; this is avoided for this paper because the two become mathematically equivalent.

For the third sensor, there are two possibilities. The *tower alignment sensor* indicates whether the robot is facing the tower:

$$h_a(x) = \begin{cases} 1 & \text{if } \theta = \text{atan2}(-p) \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Alternatively, the *gradient alignment sensor* indicates whether the robot is facing the direction of steepest ascent of  $m$ :

$$h_a(x) = \begin{cases} 1 & \text{if } (\cos \theta, \sin \theta) \propto \nabla m(p) \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

At nonsmooth points, the gradient  $\nabla$  is assumed to be extended in a standard way from nonsmooth analysis; see [9] in general, and [7] for the use of this in the context of sensor-based planning. In this general case,  $h_a(x) = 1$  if  $(\cos \theta, \sin \theta)$  is proportional to any vector in the *generalized gradient* [9]:

$$\text{co} \left\{ \lim_{i \rightarrow \infty} \nabla m(p_i) : p_i \rightarrow p, m'(p_i) \text{ exists} \right\}, \quad (8)$$

in which the  $p_i$  correspond to any sequence that converges to  $p$ ,  $\text{co}$  denotes the convex hull, and  $m'$  denotes the derivative of  $m$ . Intuitively, this definition gathers up all possible gradients by taking derivatives along all sequences converging to  $p$  for which derivatives exist.

In Section 4,  $m$  is radially symmetric, in which case either alignment sensor can be used because they give the same result. In Section 5, the asymmetric case is handled, and only the gradient alignment sensor is used. The robot has no other sensors, such as global positioning, odometry, or a compass. Therefore, it is unable to obtain precise position or angular coordinates.

Now consider possible actions or *motion primitives* that are given to move the robot. Each motion primitive must terminate on its own using sensor information. The robot is allowed only three motion primitives:

$u_{fwd}$  The robot goes straight forward in the direction it is facing, stopping only if it: 1) contacts the obstacle ( $h_t(x) = 1$ ), 2) hits the tower ( $h_i(x) = 1$ ), or 3) detects a local maximum in intensity along its line of motion.

## PLAN FOR THE SYMMETRIC CASE

1. Let  $i_L = h_i(x)$ .
2. Apply  $u_{ori}$  and then  $u_{fwd}$ .
3. If  $h_i(x) = 1$ , then terminate; the tower was reached.
4. If  $i_L \neq h_i(x)$ , then let  $i_H = h_i(x)$ .
5. Apply  $u_{fol}$ .
6. If  $h_i(x) > i_H$  then go to Step 1.
7. Go to Step 5.

Fig. 4: A solution plan for the case of a radially symmetric intensity function.

- $u_{ori}$  The robot rotates counterclockwise, stopping only when it is aligned with the tower ( $h_a(x) = 1$ ).
- $u_{fol}$  The robot travels around an obstacle boundary counterclockwise, maintaining contact to its left at all times, stopping only when it reaches a local maximum in the intensity.

Details regarding the implementation of these primitives have been abstracted away, especially in the cases of  $u_{fwd}$  and  $u_{fol}$ . Both of these have termination conditions that depend on detecting a local maximum. This could be achieved in practice by sampling the intensity at high frequency and checking the relations  $i_{k-1} > i_{k-2}$  and  $i_{k-1} \geq i_k$ , of the last three intensity observations are  $i_{k-2}$ ,  $i_{k-1}$ , and  $i_k$ . Relaxing the comparison between  $i_{k-1}$  and  $i_k$  to include the possibility of equality allows the robot to detect plateaus in intensity values. This sampling policy could obviously cause the robot to slightly pass the maximum, which could be deemed to be insignificant due to a high sampling rate, or the robot could execute a short reversal motion. Further discussion of the implementation of these primitives can be found in Section 6; however, it is important to point out that some subtle details remain regarding the implementation of the primitives in practice. In this paper, the primitives are given, and seem reasonable under the sensing model.

## 4 THE RADIALY SYMMETRIC CASE

This section considers the case in which the level sets of the intensity function are concentric circles. A plan is presented that incorporates sensor feedback and guarantees the robot will reach the tower after a finite number of primitives have been applied. In this section,  $u_{fwd}$  always terminates when either the tower or boundary is reached; the possibility of a local maximum in intensity arises only in Section 5.

### 4.1 A plan for the robot

Using its motion primitives and enough memory to store two intensity values,  $i_L$  and  $i_H$ , the plan is shown in Figure 4.<sup>1</sup> The intensity  $i_H$  is the intensity observed when the current obstacle was contacted via completion of a  $u_{fwd}$  motion. The intensity  $i_L$  is the value obtained just prior to the execution of  $u_{fwd}$ . This is used in Step 4 to compare with the current intensity  $h_i(x)$  to determine whether  $u_{fwd}$  caused the robot to move. If the robot moved, then a new value for  $i_H$  is stored because the robot moved across the interior of  $E$ . It is assumed that the starting position lies in the interior of  $E$ , which guarantees that  $i_H$  is defined in the first iteration. In each execution of Step 5, the robot moves to another local maximum, and then it tries to leave the boundary in Step 6 if the maximum is greater than  $i_H$ . If after  $u_{ori}$  the robot is facing the boundary, then it cannot make progress, and  $i_L = i_H$ . This indicates that another local maximum must be reached before trying to escape again. Note that our robot cannot follow the Bug1 approach in [41] because the robot is unable to determine whether it has traveled completely around the obstacle.

<sup>1</sup> A technicality regarding the storage of real numbers is avoided here. Of course, real numbers may require unbounded or infinite memory; however, we imagine fixed precision representations. If desired, the theoretical bounds in this paper can be expanded to incorporate floating point precision error.

## 4.2 Convergence

Does this plan actually succeed? The following lemma represents a crucial step in establishing convergence to the tower:

**Lemma 1** *For every obstacle boundary  $\partial O$  and every possible tower location  $\in \mathbb{R}^2 \setminus O$ , there exists at least one intensity local maximum  $p \in \partial O$  for which the disc centered at the tower  $(0, 0)$  with radius  $\|p\|$  is disjoint from the interior of  $O$ .*

**Proof:** Using the general position assumption and  $(0, 0)$ , there are at most a finite number of intensity local maxima along  $\partial O$ . One or more of these may be global maxima. Since intensity increases monotonically as distance decreases, the global maxima are also the points along  $\partial O$  that are closest to  $(0, 0)$ . Let  $p$  denote any one of these and let  $D((0, 0), p)$  be the closed disc centered at  $(0, 0)$  with  $p$  lying on its boundary. All other global maxima must lie on the boundary of this disc. By construction, no other points in  $O$  are closer to  $(0, 0)$  than  $p$ ; hence,  $D$  and the interior of  $O$  are disjoint. ■

Convergence is established in the following proposition:

**Proposition 1 (Convergence)** *The plan in Figure 4 causes the robot to reach the tower after a finite number of steps, regardless of the particular environment  $E$ , initial robot position in the interior of  $E$ , and tower location in  $E$ .*

**Proof:** After executing Step 2 for the first time, either the tower is reached or the robot contacts the boundary of an obstacle. Assuming the latter, Step 4 stores  $i_H$ , the intensity at this boundary point. The main idea of the proof is that the intensity increases monotonically with every subsequent execution of Step 2. Since distance decreases monotonically as intensity increases, the robot arrives at  $(0, 0)$ . Step 6 ensures that  $u_{fwd}$  is attempted only at a point that is closer to  $(0, 0)$  than the point at which the robot arrived at the obstacle boundary (where it recorded  $i_H$ ). It might seem that an infinite loop is possible by failure to satisfy the condition of Step 6 or by the motion being blocked by the obstacle boundary. However, Lemma 1 ensures that it is always possible to leave the obstacle boundary and obtain a higher intensity value. In the worst case, the robot may repeatedly return to the same obstacle boundary  $\partial O$ , but it cannot become trapped. Each time it arrives at  $O$ ,  $i_H$  is larger, and the number of local maxima is finite. A new departure point along  $\partial O$  exists each time due to Lemma 1. Eventually, the robot must leave from a global intensity maximum, and its direction faces the interior of the disc  $D$  from the proof of Lemma 1. Therefore, the robot is not blocked, it increases the intensity, and will never contact  $\partial O$  again. Since this is assured for every obstacle, the robot must eventually arrive at  $(0, 0)$ . ■

Note that in the proof above, the robot cannot necessarily determine whether it is returning to the same obstacle multiple times. It may alternate between several obstacles unknowingly, but this causes no harm!

## 4.3 Bounding the total distance

How far might the robot travel in the worst case to reach the tower? Let  $\ell(p_0, E)$  denote the distance traveled by the robot after executing the plan in Figure 4 from position  $p_0$ . This would be the reading obtained by a perfect odometer, if it had existed. Let  $N$  be the total number of obstacles that intersect a disc of radius  $\|p_0\|$ , centered at  $(0, 0)$ . A local maximum at a point  $p \in \partial O$  is called *unblocked* if the robot can freely move toward the tower from  $p$ , without immediately entering the interior of  $O$ . The following proposition bounds the total distance traveled:

**Proposition 2 (Bounding the Path Length)** *The total distance traveled by the robot satisfies the bound:*

$$\ell(p_0, E) \leq \|p_0\| + \sum_{k=1}^N n_k c_k, \quad (9)$$

in which  $n_k$  is the number of unblocked local maxima along  $O_k$  and  $c_k$  is its perimeter.

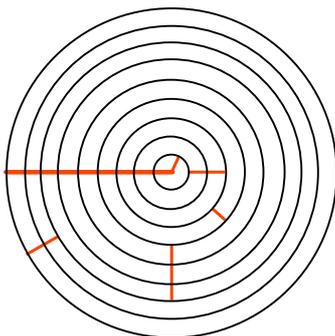


Fig. 5: As part of the proof that bounds the path length, each path segment, which corresponds to the execution of a primitive, is rotated around the circle. Once aligned (shown as a thicker horizontal line heading to the left from the center) then their total length is equivalent to that of a straight path.

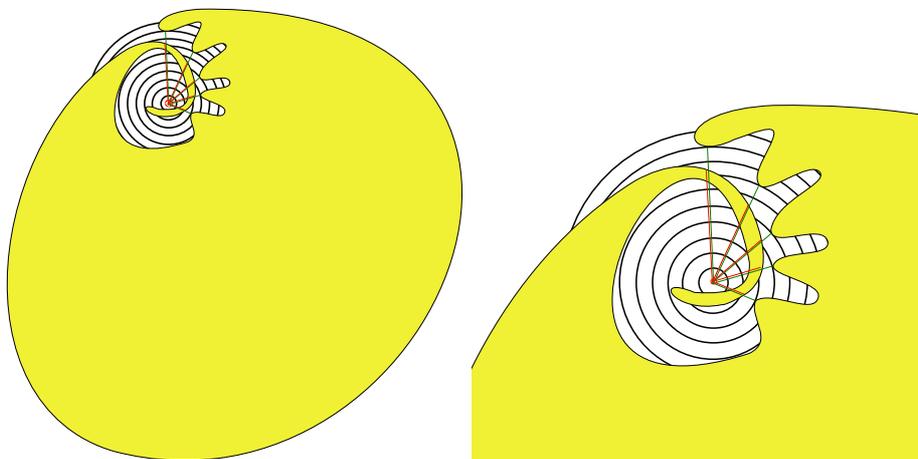


Fig. 6: Worst-case behavior: The robot is repeatedly sent around the obstacle before finally reaching the goal. After each traversal, a new red segment is taken, starting with the nearly vertical segment. The segments are traversed in clockwise order.

**Proof:** The proof proceeds by bounding the total path length due to  $u_{fwd}$  separately from that of  $u_{fol}$ . Let  $\ell_{fwd}$  denote the total distance traveled from all  $u_{fwd}$  executions. If  $u_{fwd}$  is executed only once before reaching  $(0, 0)$ , then clearly  $\ell_{fwd} = \|p_0\|$ . In the more general case, each time  $u_{fwd}$  is applied the path coincides with a line through  $(0, 0)$ . All path segments can be radially rotated, as shown in Figure 5, so that their sum is clearly no larger than  $\|p_0\|$ . This explains the first term in (9).

Now let  $\ell_{fol}$  be the total distance traveled due to all  $u_{fol}$  motions. For a single obstacle  $O_k$  with perimeter  $c_k$ , consider the total number amount of boundary traveling that occurs. The robot never leaves  $O_k$  twice from the same local maximum because the intensity increases monotonically each time  $\partial O_k$  is reached by a  $u_{fwd}$  primitive. This implies that the robot must leave  $O_k$  via the  $u_{fwd}$  primitive no more than  $n_k$  times. Furthermore, the total distance traveled by executing a consecutive sequence of  $u_{fol}$  primitives is always less than  $c_k$ ; otherwise, it would surpass the desired departure point. Therefore, the total distance traveled along  $O_k$  is bounded above by  $n_k c_k$ . Summing over all obstacles yields the bound  $\ell_{fol} \leq \sum_{k=1}^N n_k c_k$ .

Combining the two components yields

$$\ell(p_0, E) = \ell_{fwd} + \ell_{fol} \leq \|p_0\| + \sum_{k=1}^N n_k c_k. \quad (10)$$

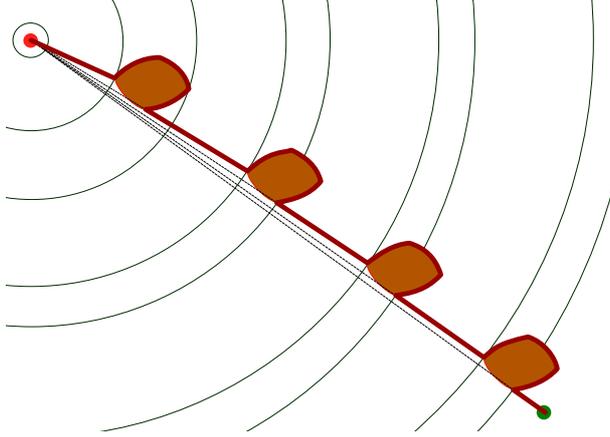


Fig. 7: In the case of convex obstacles, the path length approaches both terms,  $\ell_{fwd}$  and  $\ell_{fol}$ , in the bound.

There are  $n_k$  unblocked local maxima along each obstacle, each of which could cause the robot to traverse nearly all of the perimeter of the same obstacle. ■

Figure 6 shows the worst-case behavior for the  $\ell_{fol}$  term.

Note that if all obstacles are convex, then the second term of (9) can be improved to  $\ell_{fol} \leq \sum_k^N P_k$ . Paths arbitrarily close to this worst-case behavior exist, as shown in Figure 7.

It is interesting that the bound in Proposition 2 is similar to that of Bug2 [41], even though our robot receives much less information. In that case, the bound on  $\ell_{fol}$  is 1/2 of what is obtained in (10).

#### 4.4 Decidability

It has been assumed so far that the tower lies in  $E$ . Suppose that the tower may lie anywhere in  $\mathbb{R}^2$  and the robot must either move to the tower if it exists or declare after a finite number of steps that the tower is unreachable. Not only does the plan in Figure 4 fail to achieve this, the following proposition establishes that the robot cannot generally decide whether  $(0, 0) \in E$ :

**Proposition 3 (Decidability)** *Using its sensors and motion primitives, it is impossible for the robot to determine whether the tower is reachable (in other words whether  $(0, 0) \in E$ ) in any environment.*

**Proof:** See Figure 8. There exists a sequence of rippled disc obstacles for which each has  $k$  intensity maxima. Over the sequence,  $k$  ranges from 1 to any natural number. Since the robot does not know  $E$ , it must repeatedly advance to local maxima in hopes of an opportunity to move to  $(0, 0)$ . Since there could be arbitrarily many maxima and the robot cannot determine whether it has gone completely around the obstacle, it will iterate forever without learning whether  $(0, 0)$  is reachable. ■

The main impediment with the robot deciding when the tower is reachable is that it cannot tell when it returns to the same point along  $\partial O$ . This is discussed more in Section 7.

#### 4.5 Obstacles Without Nonsmooth Points

Suppose that we restrict the obstacle boundaries to be analytic, rather than piecewise-analytic. This implies that every point along  $\partial O$  has a well-defined normal, in the sense from classical calculus. Now remove the tower alignment sensor and convert  $u_{fwd}$  into a new primitive  $u_{nor}$  that always moves toward the tower in the direction of the normal at the robot position in  $\partial O$ . Suppose that the plan in 4 is modified by executing  $u_{nor}$  in Step 2, instead of  $u_{ori}$  followed by  $u_{fwd}$ .

**Proposition 4** *If the boundary of every obstacle is analytic, then the modified plan (which avoids the tower alignment sensor) always succeeds and the path satisfies the bound in Proposition 2.*

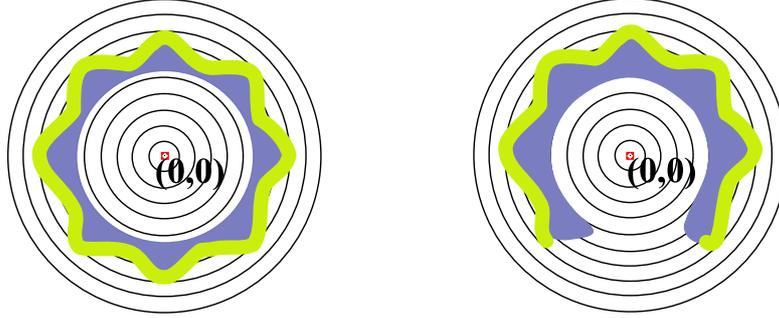


Fig. 8: The environment on the top is one representative from the sequence of obstacles that cause  $k$  local maxima. The thick green curve shows the path taken by the robot. Consider the intensity observed along the way. The robot does not know how many peaks may exist. Furthermore, it cannot determine whether there is a solution beyond the next peak, as would occur for the example on the bottom. Since the environment is unknown, the robot cannot decide whether the tower is reachable.

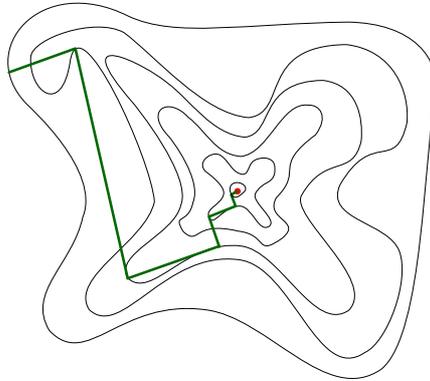


Fig. 9: Robot makes progress towards the target in an environment with asymmetric intensity.

**Proof:** The proof follows from the key observation from classical constrained optimization. Recall that when optimizing an analytic function  $f(x)$  subject to an analytic constraint  $g(x) = 0$ , then local extrema occur only if  $\nabla f(x) = \lambda \nabla g(x)$  for some nonzero scalar constant  $\lambda$  (called a *Lagrange multiplier*). In our context,  $f$  is replaced by the intensity function  $m$ , and  $g$  is replaced by  $\partial O$ . Each time that  $u_{fol}$  terminates due to an extremum, the gradient of the intensity function must be normal to the boundary. Due to radial symmetry, the direction of  $\nabla f(x)$  is always on a line through  $(0, 0)$ . Thus, the robot can move toward the tower by executing  $u_{nor}$ . This produces the same motion that would have been executed by the original plan in Figure 4. Therefore, the convergence proof in Proposition 1 and length bound in Proposition 2 still hold. ■

## 5 THE GENERAL ASYMMETRIC CASE

This section generalizes some of the ideas from Section 4 to the setting of intensity functions that are asymmetric. In this section, the level sets are topologically equivalent to circles, but may take any shape. The intensity function  $m$  is piecewise-analytic with a single maximum at  $(0, 0)$ . The primary trouble caused by this case is that the gradient of the intensity function no longer “points” to the tower. In the symmetric case, the tower alignment sensor (6) and gradient alignment sensor (7) produce the same orientation. In this section, the two sensors generally produce different results. It is assumed here that the gradient alignment sensor is used. Note that a straight-line motion will no longer take the robot to the tower. Fortunately, the robot is able to make progress by relying on the main idea from the classical optimization technique *steepest descent with line searching* (SDLS) [40].

PLAN FOR THE ASYMMETRIC CASE

1. Let  $i_L = h_t(x)$ .
2. Apply  $u_{ori}$  and then  $u_{fwd}$ .
3. If  $h_i(x) = 1$ , then terminate; the tower was reached.
4. If  $i_L \neq h_i(x)$ , then let  $i_H = h_i(x)$ .
5. If  $h_t(x) = 0$ , then go to Step 1.
6. Apply  $u_{fol}$ .
7. If  $h_i(x) > i_H$  then go to Step 1.
8. Go to Step 6.

Fig. 10: A solution plan for the case of an asymmetric intensity function. The difference is that multiple iterations are needed when crossing the interior of  $E$ . This is reflected in Step 5, which did not exist in Figure 4.

The plan from Figure 4 is modified in the present setting to obtain the plan shown in Figure 10. The only real difference is given by the insertion of Step 5. During the execution of  $u_{fwd}$ , the robot may fail to reach the obstacle boundary. Therefore, it must realign itself and move in a new direction. Figure 9 shows a sample path in the asymmetric intensity scenario. This iteration continues until the tower or boundary is reached. If the boundary is reached, then  $u_{fol}$  is applied as in Section 4.

The proof of convergence follows the same general strategy as in Section 5. Recall Lemma 1, which was perfect for ensuring that the robot does not get trapped moving along an obstacle boundary. In the current setting, replace the disc  $D((0,0),p)$  with a *topological disc*,  $B((0,0),p)$ , which is defined as

$$B((0,0),p) = \{p' \in E \mid m(p') \geq m(p)\}. \quad (11)$$

Informally, the topological disc includes all points with intensity greater than or equal to the intensity at  $p$ . Using this definition, the following lemma can be stated, which generalizes Lemma 1 to a topological disc:

**Lemma 2** *For every obstacle boundary  $\partial O$  and every possible tower location, there exists at least one intensity local maximum  $p \in \partial O$  for which the topological disc  $B((0,0),p)$  is disjoint from the interior of  $O$ .*

**Proof:** The argument is similar to the proof of Lemma 2. Using the general position assumption, there are at most a finite number of intensity local maxima along  $\partial O$ . One or more of these may be global maxima. Let  $p$  denote any one of these and let  $B((0,0),p)$  be the corresponding topological disc. All other global maxima must lie on the boundary of  $B((0,0),p)$ . By construction, no other points in  $O$  have an intensity greater than the intensity at  $p$ ; hence,  $B((0,0),p)$  and the interior of  $O$  are disjoint. ■

Using Lemma 2, it is straightforward to establish the convergence of the plan in Figure 10:

**Proposition 5 (Convergence)** *For any  $\epsilon > 0$ , the plan in Figure 10 causes the robot to arrive within  $\epsilon$  distance from the tower after a finite number of steps, regardless of the particular environment  $E$ , initial robot position in the interior of  $E$ , and tower location in  $E$ .*

**Proof:** The proof follows in the same manner as the proof of Proposition 1. In each step, the intensity is guaranteed to increase. The only significant change is that it may take multiple iterations of  $u_{fwd}$  to traverse the interior of  $E$ . Since the robot moves along a line in the direction of the gradient in each step, this is equivalent to SDLS optimization, which is well-known to converge asymptotically [40]. The asymptotic convergence of SDLS is the reason why  $\epsilon$  is used in the proposition; precise convergence to  $(0,0)$  would require an infinite number of steps in general. ■

It is natural to wonder whether a bound can be constructed on the total path length, as established in Proposition 2. In the current setting, the second term of  $\sum_{k=1}^N n_k c_k$  remains as an upper bound on the motions due to  $u_{fol}$ . The first term, however, is complicated by the convergence rate of the SDLS iterations, which depends on the properties of the intensity function  $m$ . For optimization problems,

conjugate gradient descent is usually preferred over SDLS because of its faster convergence rate; however, our robot does not receive enough information to apply the method reliably (higher-order derivatives of the intensity function are needed).

## 6 EXPERIMENTS

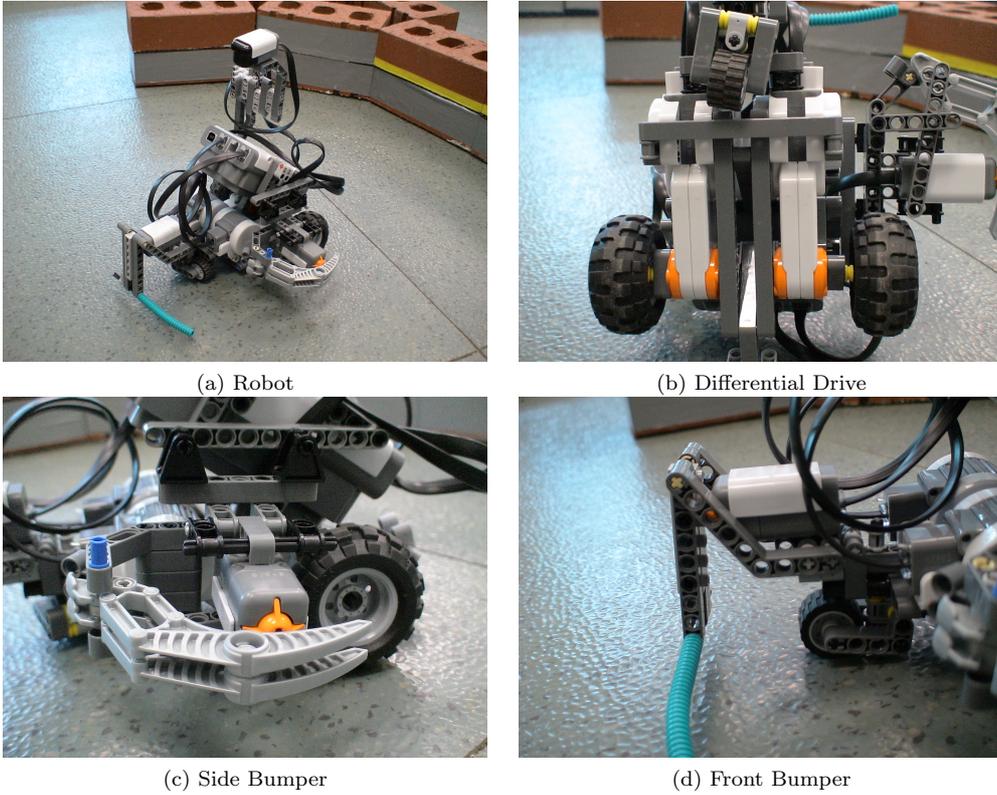


Fig. 11: The modified Lego Mindstorms robot used in our experiments. The main motivating design factor was low cost and simplicity.

We implemented the plan from Figure 10, which is the more general, asymmetric case. The design philosophy was to use simple, cheap components to demonstrate the feasibility of the method.

The first issue was generating an intensity signal that is appropriate for our model. Many signals such as Wi-Fi and magnetic fields were considered before we decided to use an infrared beacon. Wi-Fi proved to be tricky because of the multi-path fading that commonly occurs in indoor environments [24]. The signal intensity at a point is a combination of the direct path from the source and indirect paths due to reflection, refraction and scattering of radio waves, which can lead to multiple maxima instead of a single maximum at the source. In [55] they found that in addition to these problems, it is difficult to use the gradient of a Wi-Fi signal outside of a close range. Alternatively, a magnetic field generated by a charge on a vertical wire has the ideal shape but would need an excessively high voltage to generate a field with a large enough range. An infrared beacon was chosen to represent the tower from Section 3 because it has a long range without needing a large power source, and when the IR signal is modulated it suffers from relatively little electromagnetic interference. More powerful beacons have a range of up to five miles.

To maintain low costs and simple design, we used the Lego NXT Mindstorms system for the robot. The base is a simple differential drive robot (see Figure 11) equipped with two touch sensors and a HiTechnic NXT IRSeeker V2 (NSK1042) infrared sensor (see Figure 12a). The final design for the touch bumpers is the result of several iterations of designs. The IRSeeker has five IR detectors, which provide

regions 1, 3, 5, 7 and 9 in Figure 12b. Regions 2, 4, 6 and 8 are obtained by interpolating between regions, and region "0" is the dead zone.

The tower, the HiTechnic IR Beacon (FTCBCN) (Figure 19a), sends pulse modulated IR signals at a frequency of 1200 Hz. The beacon circuit is essentially a 555 timer and three infrared LEDs, and the frequency of the circuit is determined by the capacitors. It has a range of two to three meters. Since the maximum intensity for an LED is directly above it, we replaced the IR LEDs with longer stemmed LEDs that we could bend at an angle, which is visible in 19b.

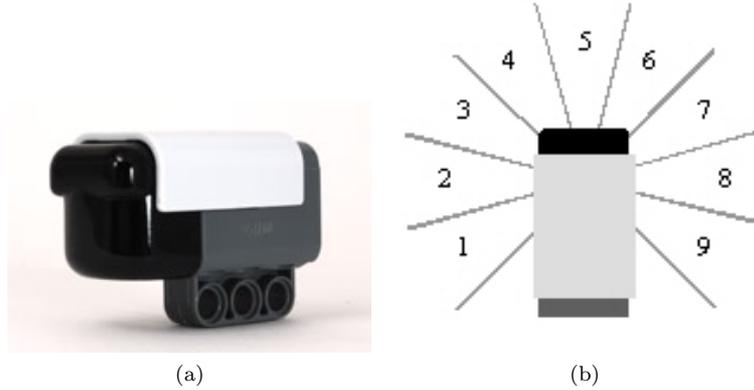


Fig. 12: The HiTechnic Infrared Sensor and a schematic of the 10 IR regions.

The IRSeeker allowed us to implement the  $u_{ori}$  and  $u_{fwd}$  primitives from Section 3. In the implementation, we concluded that the robot is aligned with the tower when the IRSeeker sensors reports an IR signal in region 5. The regions in the sensor are cones rather than lines; therefore, moving towards the tower corresponds to moving towards the tower cones or minus some error. Observe, however, that using the general asymmetric plan 10 has an important implication: An error in sensing or actuation during  $u_{ori}$  or  $u_{fol}$  leads to the robot detecting a maximum and re-orienting itself. The plan is therefore robust to such error; however, it increases execution cost.

The other primitive from Section 3 was  $u_{fol}$ . In this case, the robot must be able to move itself along the wall using the contact sensor. This might be achieved by mounting a horizontal wheel that rolls along the wall and is force controlled. In our experiments we achieved it with two touch bumpers (Figures 11c and 11d). We found that the front and side bumpers needed slightly different designs. The front bumper has a swinging design and was made of more flexible material, whereas the side bumper needed to allow the robot to glide along a wall. The robot also needs to handle corners, which it accomplishes by turning counterclockwise unless the front sensor is on, in which case it turns clockwise.

We tested the plan of Section 5 in a laboratory environment. The boundaries were cloth-covered cinder blocks, the obstacles were smaller horizontal clay bricks, and the beacon was placed on a single vertical brick. Duct tape and electrical tape were used to cover the contact points on the bricks to decrease the friction between the robot and the bricks.

Five different environments were designed to test the robot's tower seeking and boundary following behaviors. Additionally, the start and end positions were varied within particular environments. At

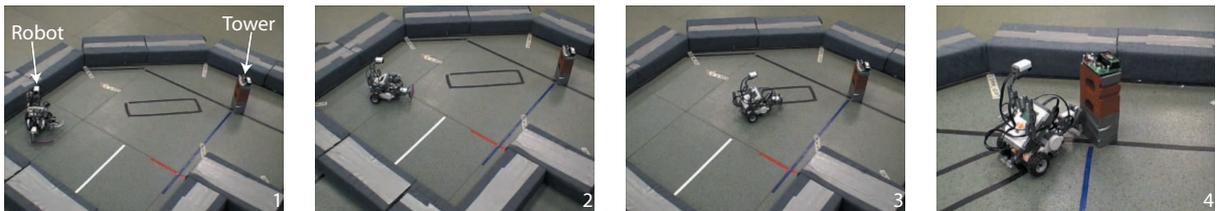


Fig. 13: An experiment that involves no obstructions.

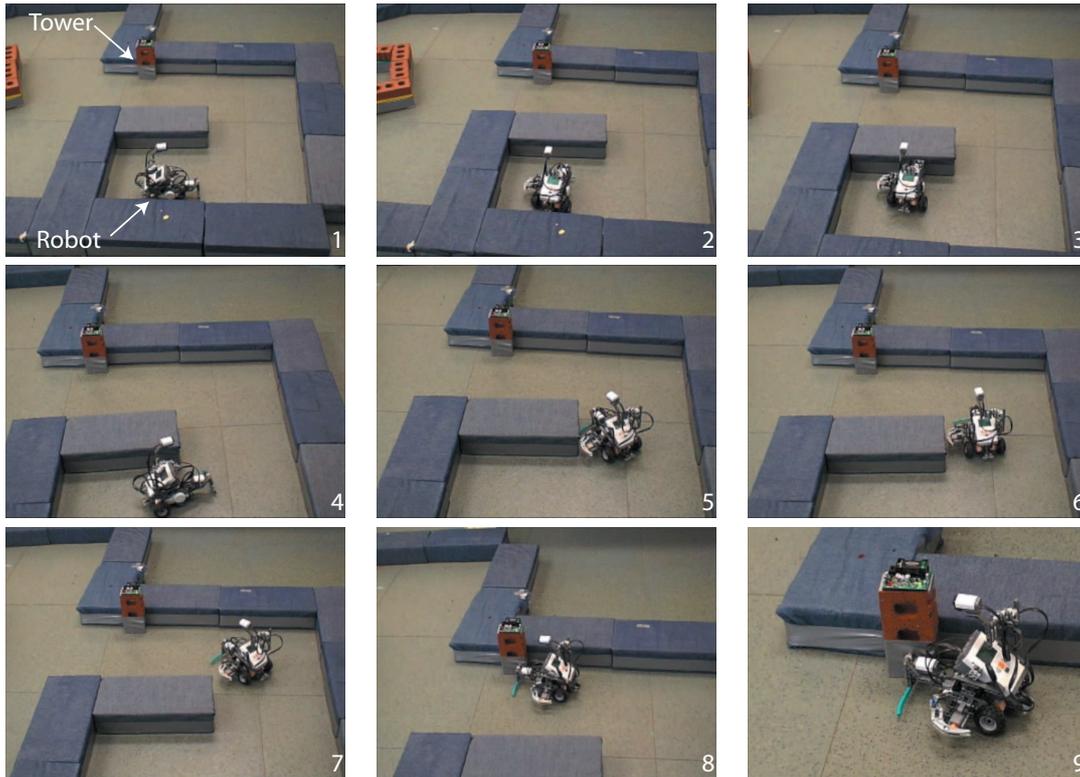


Fig. 14: An experiment in which the outer boundary interferes with navigation.

least five navigation experiments were performed in each environment. Videos of some experiments are available at:

<http://msl.cs.uiuc.edu/intensity/>

The earlier environments, Figures 13 and 14, featured either no obstacles or a single rectangular obstacle and were mainly used for basic testing. Of more interest are the environments in Figures 15- 18. Figure 15 is an experiment that used the star-shaped obstacle, whereas Figure 16 is an experiment with the U-shaped obstacle. These environments were designed to test the robot's wall following capabilities. Although the robot easily managed the star-shaped obstacle, the U-shaped obstacle was more challenging. The series of sharp corners lead to the robot straying away from the boundary, which is visible in frames 10-12 of Figure 16. The robot was always, however, able to return to the boundary. Figures 17 and 18 show experiments that featured multiple inner obstacles, designed to test the robot's ability to switch back and forth between tower seeking mode and boundary seeking mode.

The robot successfully navigated towards the tower in all of the above mentioned experiments. There were, however, some performance issues, related to the IR beacon and wall following. The range and design of the beacon was a significant limitation when we designed test environments. This can easily be fixed by using a more powerful IR beacon. In more general environments, the need for a clear line of sight to the beacon could prove to be problematic. Wall following could also be improved substantially by using more sophisticated methods. Based on these experiments, we are convinced that the primitives from Section 3 and the plans from Sections 4 and 5 are reasonable models in at least some practical settings. It remains to develop robust, efficient, and low-cost implementations of the method.

## 7 DISCUSSION

We introduced a simple robot model and corresponding plans that enable the robot to successfully navigate to an intensity source in an unknown planar environment. Our approach fits well into the well-known family of bug algorithms, and the execution requires less sensing information than previous

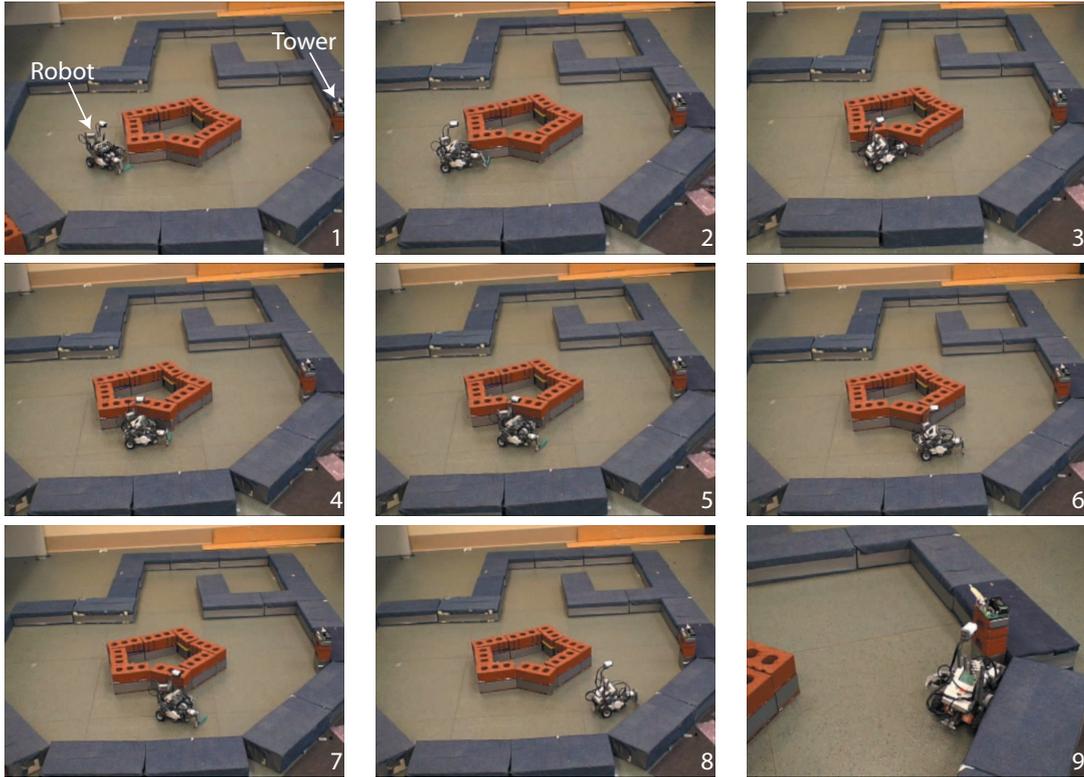


Fig. 15: An experiment in which the environment contains a star shaped obstacle that demonstrates the robot’s ability to handle corners.

approaches. The robot uses a contact sensor, an intensity sensor, and an alignment sensor to achieve the task of reaching a goal, which is the signal source. The robot does not have access to perfect clocks, odometry, or other sensors that would enable it to infer any coordinates in  $\mathbb{R}^2$ , its orientation in  $[0, 2\pi)$ , or its total distance traveled. For a radially symmetric intensity function, we presented a plan that is guaranteed to succeed in a finite number of steps. Also, a bound is provided on the total distance traveled. If the intensity function is asymmetric, the plan is slightly modified, but nevertheless converges. This case seems quite interesting due to its extreme generality. Although the robot is unable to move directly to the goal, its convergence is assured through an approach that is mathematically equivalent to the SDLS algorithm from optimization. This plan was experimentally verified with a simple, cheap robot and an infrared beacon as the tower.

The completed work raises many new questions and issues for future research. It is interesting that the robot accomplishes the task without having sufficient information to determine whether it is returning to the same obstacles. What other tasks can be accomplished in spite of this confusion? What tasks require distinguishability between obstacles? What forms of sensing should be added to give the robot enough information to make such distinctions (e.g., a mathematical pebble)? In another direction, can the plans given in this paper be improved by allowing the robot to alternative between clockwise and counterclockwise directions? Could a randomized approach lead to good expected-case behavior? We look at some of these questions in detail.

As mentioned in Section 4, the main impediment with the robot deciding whether the tower is reachable is that it cannot determine whether it returns to the same point along  $\partial O$ . This is a familiar problem in the searching of unknown mazes [5], graphs [3,16], and polygons [19]. The usual solution is to introduce a *pebble* that serves as a marker. There are many ways to simulate the effect of a pebble, but all of them require additional sensor or actuation capabilities.

One method would be to give the robot a physical pebble, a pebble sensor, and a pebble actuator that allows it to pick up and drop pebbles. This allows the robot to tell if it has returned to the same



Fig. 16: An experiment that involves a more complicated obstacle. This environment contains several sharp corners and polygonal curves.

point on an obstacle but it forces the robot to completely circumnavigate the obstacle to retrieve the pebble, so the plan would need to be amended.

Another solution is to give the robot an unlimited number of pebbles. The trick here is that we will have to be careful about the distinguishability of the pebbles. Choosing to use infinite distinguishable pebbles implies that the sensor can ignore previous pebbles. More significantly, we may have made the robot much more powerful [6]. Since the current plan allows the robot to visit the same obstacle multiple times, indistinguishable pebbles may not be suitable either. Ideally, we would have pebbles that vanish once the robot leaves an obstacle and no longer needs it. It remains to be seen if this would be a reasonable model.

Another natural extension would be to increase the number of towers. We have experimentally verified that it is straightforward to make distinguishable towers from the setup of Section 6. By modifying the circuit to have two  $.01\mu F$  capacitors in parallel, we halved the frequency from 1200 Hz to 600 Hz. See Figure 19c for a side by side comparison.

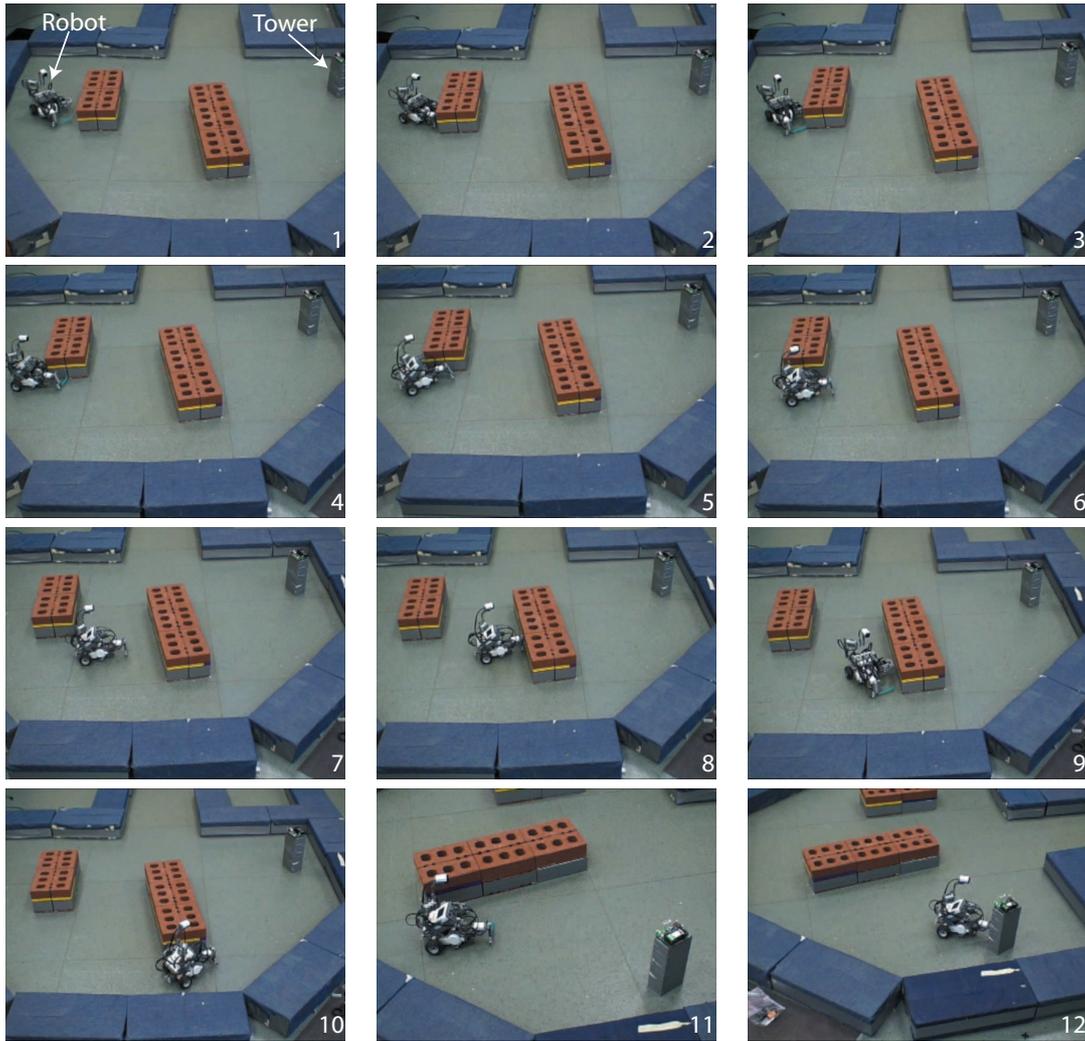


Fig. 17: An experiment in an environment that contains two rectangular obstacles.

As we saw in Section 6, it is possible to have towers with distinguishable signals. What problems disappear with two towers? What new problems arise? With three or more towers, we may not need pebbles for the robot to decide when it has returned to the same point.

With two towers, we can start to map obstacles in *intensity coordinates*. If the robot were to trace the boundary of an obstacle, it would get intensity values from both towers at every point. What do the obstacles look like in this new transformed coordinate system? Another interesting multiple tower-question is: Can the robot calculate or trace the Voronoi diagram of the towers? See [31,32] for approaches to this problem.

Finally, another interesting direction for future research is to develop a framework for competitive ratio-based analysis in this context. Such an approach was taken in [17,18] for a specific bug algorithm model.

#### Acknowledgments

This work was supported in part by the DARPA STOMP program (DSO HR0011-07-1-002), NSF grant 0904501 (IIS Robotics), NSF grant 1035345 (Cyberphysical Systems), and MURI/ONR grant N00014-09-1-1052. We thank Stephen Bond for helpful discussions.

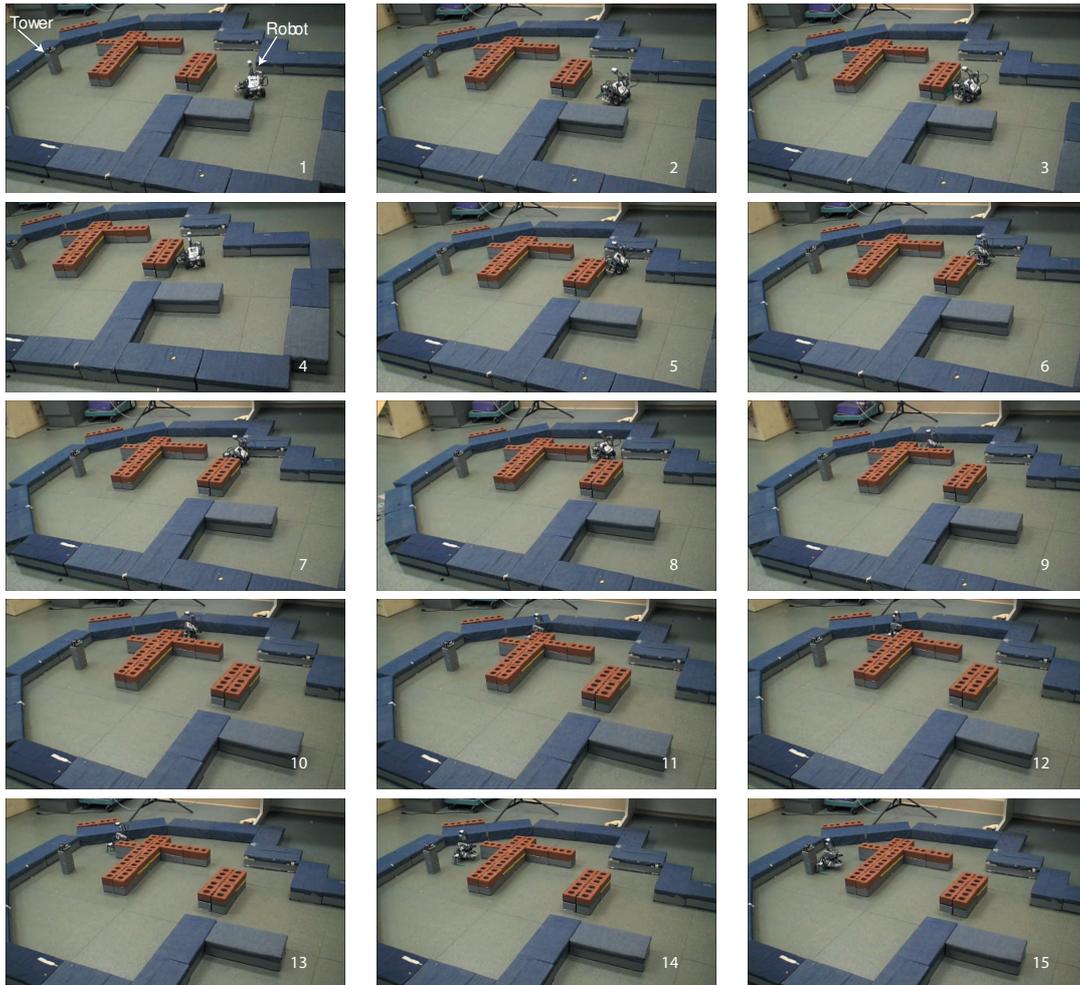
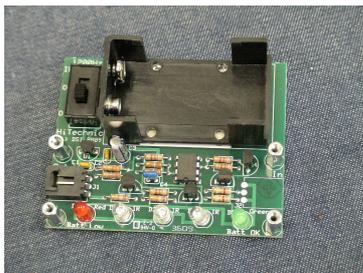


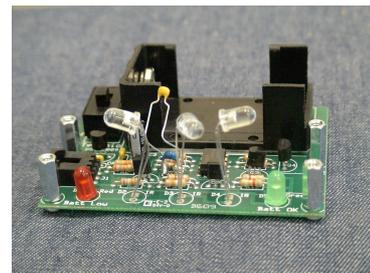
Fig. 18: An experiment in an environment that contains a rectangular obstacle and a cross-shaped obstacle.



(a) The original beacon



(b)



(c) The 600 Hz beacon

Fig. 19: The HiTechnic IR Beacon. Note that the center beacon was used in the experiments, and the 600 Hz beacon has an extra capacitor in parallel.

## References

1. A. Bachrach, R. He, and N. Roy. Autonomous Flight in Unknown Indoor Environments. *International Journal of Micro Air Vehicles*, 1(4):217–228, 2009.
2. R. A. Baeza, J. C. Culberson, and G. J. E. Rawlins. Searching in the plane. *Information and Computation*, 106(2):234–252, 1993.
3. M. A. Bender, A. Fernandez, D. Ron, A. Sahai, and S. Vadhan. The power of a pebble: Exploring and mapping directed graphs. In *Proceedings Annual Symposium on Foundations of Computer Science*, 1998.
4. A. Blum, P. Raghavan, and B. Schieber. Navigating in unfamiliar geometric terrain. In *STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 494–504, New York, NY, USA, 1991. ACM.
5. M. Blum and D. Kozen. On the power of the compass (or, why mazes are easier to search than graphs). In *Proceedings Annual Symposium on Foundations of Computer Science*, pages 132–142, 1978.
6. J. Brunner, M. Mihalák, S. Suri, E. Vicari, and P. Widmayer. Simple robots in polygonal environments: A hierarchy. *Algorithmic Aspects of Wireless Sensor Networks*, pages 111–124, 2008.
7. H. Choset and J. Burdick. Sensor-based exploration: Incremental construction of the hierarchical generalized Voronoi graph. *International Journal of Robotics Research*, 19(2):126–148, 2000.
8. H. Choset, M. La Civita, and J.C. Park. Path planning between two points for a robot experiencing localization error in known and unknown environments. In *FSR'99 Proceedings of the International Conference on Field and Service Robotics*, pages 98–103, August 1999.
9. F. H. Clarke. *Optimization and Nonsmooth Analysis*. Springer-Verlag, Berlin, 1998.
10. A. Datta, C. A. Hipke, and S. Schuierer. Competitive searching in polygons—beyond generalized streets. In J. Staples, P. Eades, N. Katoh, and A. Moffat, editors, *Algorithms and Computation, ISAAC '95*, pages 32–41. Springer-Verlag, Berlin, 1995.
11. X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment. I: the rectilinear case. *Journal of the ACM (JACM)*, 45(2):215–245, 1998.
12. B. R. Donald. The complexity of planar compliant motion planning under uncertainty. In *Proceedings ACM Symposium on Computational Geometry*, pages 309–318, 1988.
13. M. A. Erdmann. On motion planning with uncertainty. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, August 1984.
14. M. A. Erdmann and M. T. Mason. An exploration of sensorless manipulation. *IEEE Transactions on Robotics & Automation*, 4(4):369–379, August 1988.
15. S. P. Fekete, R. Klein, and A. Nüchter. Online searching with an autonomous robot. In *Proceedings Workshop on Algorithmic Foundations of Robotics*, Zeist, The Netherlands, July 2004.
16. P. Fraigniaud, D. Ilcinkas, G. Peer, A. Pelc, and D. Peleg. Graph exploration by a finite automaton. *Theoretical Computer Science*, 345(2-3):331–344, December 2005.
17. Y. Gabriely and E. Rimon. Cbug: A quadratically competitive mobile robot navigation algorithm. *IEEE Transactions on Robotics*, 24(6):1451–1457, 2008.
18. Y. Gabriely and E. Rimon. Competitive disconnection detection in on-line mobile robot navigation. *Algorithmic Foundation of Robotics VII*, pages 253–267, 2008.
19. B. Gfeller, M. Mihalak, S. Suri, E. Vicari, and P. Widmayer. Counting targets with mobile sensors in an unknown environment. In *ALGOSENSORS*, July 2007.
20. L. Girod and D. Estrin. Robust range estimation using acoustic and multimodal sensing. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.
21. K. Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10:201–225, 1993.
22. F.W. Grasso, T.R. Consi, D.C. Mountain, and J. Atema. Biomimetic robot lobster performs chemo-orientation in turbulence using a pair of spatially separated sensors: Progress and challenges. *Robotics and Autonomous Systems*, 30(1):115–131, 2000.
23. D. Han, D. Andersen, M. Kaminsky, K. Papagiannaki, and S. Seshan. Access point localization using local signal strength gradient. *Passive and Active Network Measurement*, pages 99–108, 2009.
24. H. Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7):943–968, 1993.
25. R. He, S. Prentice, and N. Roy. Planning in information space for a quadrotor helicopter in a gps-denied environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008)*, pages 1814–1820, Los Angeles, CA, 2008.
26. H. Ishida, Y. Kagawa, T. Nakamoto, and T. Moriizumi. Odor-source localization in the clean room by an autonomous mobile sensing system. *Sensors and Actuators B: Chemical*, 33(1-3):115–121, 1996.
27. H. Ishida, H. Tanaka, H. Taniguchi, and T. Moriizumi. Mobile robot navigation using vision and olfaction to search for a gas/odor source. *Autonomous Robots*, 20(3):231–238, 2006.
28. I. Kamon, E. Rimon, and E. Rivlin. TangentBug: A Range-Sensor-Based Navigation Algorithm. *The International Journal of Robotics Research*, 17(9):934, 1998.
29. I. Kamon, E. Rimon, and E. Rivlin. Range-sensor based navigation in three dimensions. *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, 1, 1999.
30. I. Kamon and E. Rivlin. Sensory-based motion planning with global proofs. *IEEE Transactions on Robotics & Automation*, 13(6):814–822, December 1997.
31. M. Katsev and S. M. LaValle. Learning the Delaunay triangulation of landmarks from a distance ordering sensor. In *Proceedings IEEE International Conference on Intelligent Robots and Systems*, 2011.
32. J. Kim, F. Zhang, and M. Egerstedt. A provably complete exploration strategy by constructing Voronoi diagrams. *Autonomous Robots*, pages 1–14, August 2010.
33. J. M. Kleinberg. On-line algorithms for robot navigation and server problems. Technical Report MIT/LCS/TR-641, MIT, Cambridge, MA, May 1994.
34. G. Kowadlo and R.A. Russell. Robot odor localization: a taxonomy and survey. *The International Journal of Robotics Research*, 27(8):869, 2008.

35. Y. Kuwana, S. Nagasawa, T. Shimoyama, and R. Kanzaki. Synthesis of the pheromone-oriented behaviour of silkworm moths by a mobile robot with moth antennae as pheromone sensors1. *Biosensors and Bioelectronics*, 14(2):195–202, 1999.
36. Y. Landa and R. Tsai. Visibility of point clouds and exploratory path planning in unknown environments. *Communications in Mathematical Sciences*, 2008.
37. S. L. Laubach and J. W. Burdick. An autonomous sensor-based path-planning for planetary microrovers. In *Proceedings IEEE International Conference on Robotics & Automation*, 1999.
38. S. L. Laubach and J. W. Burdick. Practical autonomous path planner for turn-of-the-century planetary microrovers. *Proceedings of SPIE*, 3525:182, 1999.
39. S. L. Laubach and J. W. Burdick. RoverBug: Long Range Navigation for Mars Rovers. *LECTURE NOTES IN CONTROL AND INFORMATION SCIENCES*, pages 339–348, 1999.
40. D. G. Luenberger. *Introduction to Linear and Nonlinear Programming*. Wiley, New York, 1973.
41. V. J. Lumelsky. *Sensing, Intelligence, Motion: How Robots and Humans Move in an Unstructured World*. Wiley Interscience, 1005.
42. V. J. Lumelsky and T. Skewis. A paradigm for incorporating vision in the robot navigation function. In *Proceedings IEEE International Conference on Robotics & Automation*, pages 734–739, 1988.
43. V. J. Lumelsky and A. A. Stepanov. Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.
44. E. Martinson and R.C. Arkin. Noise maps for acoustically sensitive navigation. In *Proceedings of SPIE*, volume 5609, pages 50–60. Citeseer, 2004.
45. E. Martinson and A. Schultz. Robotic discovery of the auditory scene. In *2007 IEEE International Conference on Robotics and Automation*, pages 435–440, 2007.
46. L. Murphy and P. Newman. Using incomplete online metric maps for topological exploration with the gap navigation tree. In *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008*, pages 2792–2797, 2008.
47. K. Nakadai, K. Hidai, H.G. Okuno, and H. Kitano. Epipolar Geometry Based Sound Localization and Extraction for Humanoid Audition. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Maui, Hawaii, 2001.
48. K. Nakadai, D. Matsuura, H.G. Okuno, and H. Kitano. Applying scattering theory to robot audition system: Robust sound source localization and extraction. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1147–1152. Citeseer, 2003.
49. S. Rajko and S. M. LaValle. A pursuit-evasion bug algorithm. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1954–1960, 2001.
50. S. Ramo, J.R. Whinnery, and T. Van Duzer. *Fields and waves in communication electronics*. Wiley-India, 2009.
51. R.A. Russell. Chemical source location and the robomole project. In *Proceedings of the Australasian Conference on Robotics and Automation*. Citeseer, 2003.
52. R.A. Russell. Robotic location of underground chemical sources. *Robotica*, 22(01):109–115, 2004.
53. S. Suri, E. Vicari, and P. Widmayer. Simple robots with minimal sensing: From local visibility to global geometry. *The International Journal of Robotics Research*, 27(9):1055, 2008.
54. K. Taylor and S. M. LaValle. I-Bug: An intensity-based bug algorithm. In *Proceedings IEEE International Conference on Robotics and Automation*, 2009.
55. O. Tekdas, N. Karnad, and V. Isler. Efficient Strategies for Collecting Data from Wireless Sensor Network Nodes using Mobile Robots. In *ISRR 2009: 14th International Symposium on Robotics Research*, 2009.
56. S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005.
57. S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *IEEE international conference on robotics and automation*, volume 3, pages 4270–4275. Citeseer, 2003.
58. B. Tovar, R. Murrieta-Cid, and S. M. LaValle. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics*, 23(3):506–518, June 2007.
59. A. Yershova, B. Tovar, M. Katsev, R. Ghrist, and S.M. LaValle. Mapping and Pursuit-Evasion Strategies For a Simple Wall-Following Robot. *IEEE Transactions on Robotics*, under review, 2010.