# Leaving an Unknown Maze Using an Error-Prone Compass*

Tom Kamphans, Elmar Langetepe

University of Bonn
Institute of Computer Science I
Römerstraße 164
53117 Bonn, Germany
{kamphans,langetep}@cs.uni-bonn.de

**Abstract**

Imagine you are trapped in a maze of caves. All you have is an old rusty compass and barely enough light to read it. How much inaccuracy can you allow to ensure that you can leave the maze? Or you are walking in a maze of hedgerows that are on the whole orthogonal. Can you find the exit by counting left and right turns?

We consider the problem of escaping from an unknown polygonal maze under limited resources and errors in inputs and motion. It is well known that the Pledge algorithm always finds a path out of an unknown maze without any means of orientation—provided that such a path exists—but it relies on fact that inputs and motions are correct.

Keywords: computational geometry, online algorithms, motion planning, autonomous mobile robots, maze leaving, Pledge algorithm.

## 1 Introduction

Online motion planning in unknown environments is theoretically well understood and practically solved in many settings. During the last decade many different objectives where discussed under several robot models.

For instance, the task of searching for a target using a touch sensor and a compass to the goal was first considered by Lumelsky and Stepanov [26]. They invented the BUG algorithm. Many variants of this algorithm were

*A preliminary version was presented at WAOA '03 [18].

discussed and analyzed afterwards; see, for example, Rajko and La Valle [32], and Kamon and Rivlin [17]. A BUG-like algorithm was also used for the Mars-Rover project, see Laubach [22]. The correctness of the algorithms is proven under idealistic assumptions whereas the practical relevance is always tested empirically.

Moreover, the task of exploring an unknown environment has attracted theoretical and practical attention. Deng et al. [10] studied the case of a robot equipped with a vision system that maps an unknown, simple, rectangular environment. They developed a strategy that guarantees for an explorer with correct vision and motion that the agent's path is never longer than $\sqrt{2}$ times the optimal path computed with full information in advance. Using the same idealistic assumptions Hoffmann et al. [15] presented a 26.5-competitive algorithm for exploring the interior of a simple polygon. On the other hand, the task of building a map of an unknown environment is solved practically and analyzed empirically in the field of robotics; see, for example, Batalin and Sukhatme [4] or Yamauchi et al. [36]. For a general overview on theoretical online motion planning problems there are some surveys [33, 34, 5, 28, 16].

Theoretical correctness results and performance guarantees from labyrinth theory or computational geometry often suffer from idealistic assumptions that cannot be fulfilled in a practical setting. For example, it is usually assumed that agents are error free. On the other hand, practitioners often measure correctness results and performance guarantees statistically (e.g. [3, 4, 20, 21, 24, 36]). So it might be useful to investigate how online algorithms with idealistic assumptions behave if those assumptions cannot be fulfilled. More precisely, can we incorporate assumptions of errors in sensors and motion directly into the theoretical analysis?

There are basically three approaches to deal with errors: The first objective is to *reduce errors*; either by reducing odometry errors (e.g., Chong and Kleeman [8], Borenstein and Feng [6]) or by avoiding faulty data by using more reliable input (e.g., preferring angular measures over distance measures; see Lumelsky and Tiwari [27], Angluin et al. [2], Demaine et al. [9], or LaValle et al. [23]). Dudek et al. [11] presented an exploration strategy for a group of robots, where the moving robot uses the other robots as landmarks. The second approach is to *tolerate errors* and show that the strategy is robust under certain types of errors (e.g., Noborio et al. [29, 30, 31], López-Ortiz and Schuierer [25]). Another method is to *detect errors and react appropriately* (e.g., Byrne et al. [7], Zelinsky [37], or Stentz [35]).

We use the second approach to consider the task of leaving an unknown maze without the ability of leaving landmarks or memorizing the visited

parts of the environment. This task can be solved using the well-known Pledge algorithm [1, 33, 19]. The Pledge algorithm assumes that the searcher is able to move along a straight line between the obstacles and to follow an obstacle boundary while counting its turning angles. In the idealized setting the searcher is error free. It was shown by Abelson and diSessa [1] that an error-free searcher will escape from a polygonal maze by applying the Pledge algorithm provided that an escape route exists. See also Hemmerling [14] for a similar proof. Lumelsky and Tiwari [27] showed that the Pledge algorithm is a special case of a BUG algorithm with azimuth input.

The searcher may be, for example, a caver equipped a compass, a walker in a maze of hedgerows using a set square or—if the hedgerows are on the whole orthogonal—counting left and right turns, or a robot measuring turns by odometry.

Gritzmann [12] remarked that it would be interesting to know how the Pledge algorithm behaves if the idealistic assumptions cannot be fulfilled. Of course, if the searcher can make arbitrarily big mistakes there are always environments in which the searcher is hopelessly trapped. But what if the searcher's errors are small enough? Can we guarantee success if the measuring errors are bounded? We investigate which conditions must hold to ensure that a searcher can leave an unknown maze with a Pledge-like algorithm.

The paper is organized as follows. In Section 2 we specify the searcher's task and the model of the searcher and its environment. We recapitulate the Pledge algorithm, and point out possible causes for failure in a non-ideal world. With this insight, in Section 3 we give a set of conditions on the searcher's path that ensures that a searcher will escape from an unknown maze if its path fulfills these conditions. The conditions on the searcher's path give a convenient framework that we apply in Section 4 to show error bounds for different types of searchers. Our main result is that a searcher equipped with a compass with reasonable small errors is able to leave an unknown maze using the Pledge algorithm; see Corollary 5. Further, we show results for a searcher with exact free motion and bounded angle measuring error, as well as for a searcher in an 'almost rectangular' environment.

## 2 Preliminaries

We are given a maze with polygonal obstacles in the plane. We assume that the searcher is able to recognize a wall (i.e., an obstacle boundary), to

follow a wall in a specified direction (w.l.o.g. we assume that the searcher uses the *left-hand rule*; that is, the searcher keeps the obstacle boundary on its left side), and to count the turning angles. This general view allows us to analyze the error-prone Pledge algorithm independent of any hardware realization. For example, the searcher may count its turning angles using a compass or a set square, by counting left and right turns, or by odometric data. We consider some of these settings in Section 4.

Other abilities for orientation and navigation are not required; particularly, it is not necessary that the searcher stores a map of its environment or uses any landmarks.

**Algorithm 1:** Pledge

$\omega := 0$
**REPEAT**
  **REPEAT**
    Move in direction $\omega$ in the free space
  **UNTIL** Searcher hits an obstacle
  **REPEAT**
    Follow the wall using the *left-hand rule*
    Count the overall turning angle in $\omega$
  **UNTIL** Angle counter $\omega = 0$
**UNTIL** Searcher is outside the maze

The task of leaving an unknown maze can be solved using the well-known Pledge algorithm, see Algorithm 1,[1] which performs only two types of movements: Either the searcher follows an obstacle boundary while counting its turning angles, or the searcher moves in a fixed direction through the space between the obstacles. The latter task always starts at vertices of the obstacles when the angle counter reaches a predefined value. We assume that the searcher recognizes—somehow or other—that it left the maze.

An example of the searcher's path using an error-free Pledge algorithm is given in Figure 1. The angle counting technique is illustrated for the second obstacle: After the searcher hits the obstacle in $p_2$ it turns $-\frac{\pi}{2}$ to follow the wall. In $p_3$ the searcher turns $+\frac{\pi}{4}$ to follow the next wall. Finally, in $p_4$ it turns $+\frac{\pi}{4}$ again until the angle counter reaches zero and the searcher leaves the obstacle. Observe that the searcher does not leave the obstacle in $p_1$,

[1] For an implementation of the Pledge algorithm with error-prone angle counting see [13].

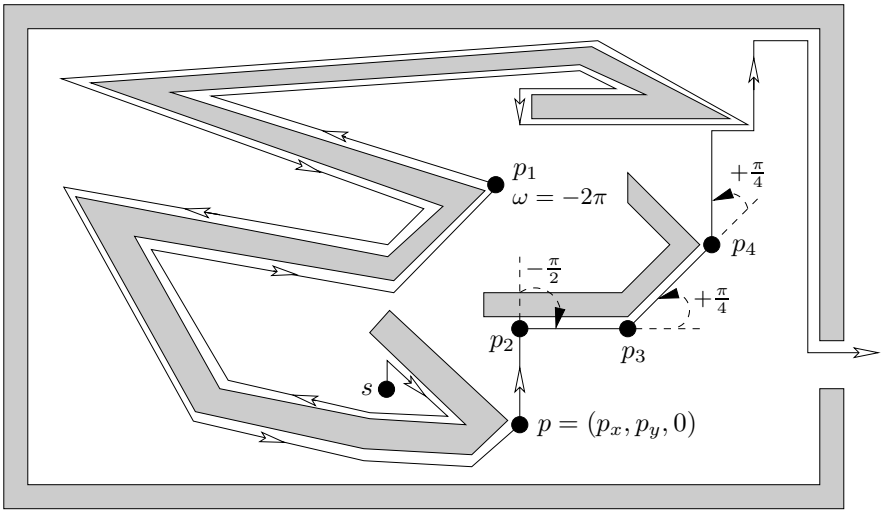



Figure 1: A path generated by the Pledge algorithm.

because its angle counter is $-2\pi$ instead of zero.

In the following, we state a set of conditions on the searcher's path that are sufficient for a successful application of the Pledge algorithm even if the searcher's sensors and motions are erroneous.[2] For convenience, we assume that the searcher is point shaped. Therefore, the parts of the searcher's path that map to a movement along a wall are line segments on the boundaries of obstacles.

The current location of the searcher is given by its *position* and its *heading*. Thus, the path, $\Pi$, of the searcher is a subspace of the configuration space $\mathcal{C} = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$, and can be described as a parameterized curve $\Pi(t) = (P(t), \varphi(t))$, where $P(t) = (X(t), Y(t))$ denotes the position at time $t$ and $\varphi(t)$ the heading. Observe that $\varphi(t)$ is the sum of all turns the searcher has made so far. For example, after the searcher has made two full counterclockwise turns the heading $\varphi(t)$ is equal to $4\pi$ instead of zero.

Note that there is an important difference between the searcher's heading and its *angle-counter value*, $\omega(t)$. While the heading is always correct, the value of the angle counter may be faulty due to measurement errors. Thus, there may be a discrepancy between the heading and the angle counter.

**Definition:** In the following, the term *heading*, denoted by $\varphi(t)$, always refers to the correct orientation as measured by an external observer. In

[2]Note that the searcher is not aware of making any errors; it always believes that its movement and angle counting is correct.

contrast, the searchers internal—and perhaps faulty—orientation is always referred as the *angle-counter value*, $\omega(t)$.

To classify the possible positions in the workspace, we divide the space of positions, $\mathcal{P} = \mathbb{R} \times \mathbb{R}$, into three subspaces: First, the space of forbidden configurations, $\mathcal{C}_{\text{forb}}$, the union of the interior of all obstacles. Second, the space of semi-free configurations, $\mathcal{C}_{\text{semi-free}}$, that is the union of the boundaries of the obstacles, and last the free configurations, $\mathcal{C}_{\text{free}}$, where $P(t)$ is neither inside nor on the boundary of an obstacle.

If a searcher hits a point $P(h_i)$ in $\mathcal{C}_{\text{semi-free}}$ after a movement through $\mathcal{C}_{\text{free}}$, we will call $h_i$ a *hit point*. If a searcher leaves $\mathcal{C}_{\text{semi-free}}$ and enters the free space at $P(\ell_i)$, we call $\ell_i$ a *leave point*.[3] With respect to the paths that are produced by the Pledge algorithm we assume that every leave point belongs to a vertex of an obstacle.

## 3 Sufficient Conditions

In this section we develop a set of conditions that are sufficient to guarantee that a searcher will find the exit of the given maze. We do not expect that a searcher plans its path according to these conditions. Instead, the conditions developed in this section provide a convenient framework for the correctness proof of maze-leaving strategies for different kinds of searchers. We apply this framework in the next section.

First, let us give a motivation for our set of sufficient conditions for escaping from the maze. The Pledge algorithms uses two types of movements: Moving along a straight line in the free space, and moving along an obstacle boundary counting the turning angles. Both types of movements may be error prone: Either the searcher cannot follow its initial direction during the movement in the free space, or the turning angles are not measured exactly and the searcher leaves the obstacle earlier or later than expected. Thus, we can distinguish between two sources of errors. Each of them leads to a condition, and both conditions together ensure the correctness of an error-prone agent using the Pledge algorithm. To ensure that a searcher can escape from the maze we want to avoid infinite cycles in the searcher's path.

To establish the set of conditions, we first observe that already small counting errors along the boundary of obstacles or deviations in the free

[3] We prefer the terms *hit point* and *leave point* although $h_i$ and $\ell_i$ are parameters of the curve. Just keep in mind that $h_i$ and $\ell_i$ are points in time while $P(h_i)$ and $P(\ell_i)$ are points in the plane.

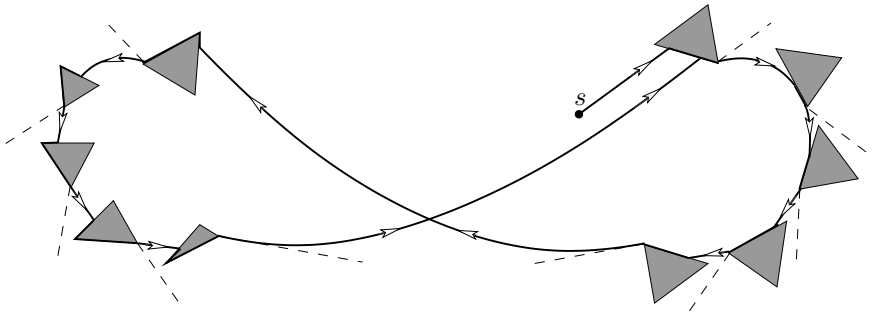



Figure 2: Small errors along each boundary can sum up to an infinite cycle.

space can sum up to a big mistake and lead to an infinite cycle. Figure 2 shows an example; the dashed lines show the correct leaving direction with respect to the direction in the most recent hit point. On the path from one obstacle to another the searcher drifts a little bit away from the correct direction and ends up in an infinite loop. Obviously, infinite cycles would be inhibited, if the path between two obstacles would stay in a wedge around the initial direction. In fact, this wedge can be as large as a half plane; that is, it is sufficient if the path between two obstacles does not leave the half plane that is defined by a line, $L_{\ell_k}$, through the leave point $\ell_k$. We have to ensure that the lines $L_\ell$ are parallel for all leave points $\ell$. To prevent such infinite loops, we require that for any two points in the free space, the difference in the headings should be lower than $\pi$; that is, $\forall t_1, t_2 : P(t_1), P(t_2) \in \mathcal{C}_{\text{free}} \Rightarrow \varphi(t_1) - \varphi(t_2) < \pi$. We refer to this requirement as the *free-space condition*. Note that the bound on the error is cumulative; that is, the difference between the headings in two points must be lower than $\pi$ even if there are a lot of turns between the two points.

Unfortunately, the free-space condition is not sufficient. Figure 3 shows two examples where the searcher's path has an infinite cycle although the free-space condition is fulfilled. In both cases, the searcher starts at $P(0)$, meets an obstacle at $t = h_i$, and misses the first possible leave point at $t_1$ because of some errors in the angle counter. The searcher leaves the obstacle at another vertex at $t_2$. The searcher in Figure 3(i) hits the same obstacle again in the hit point $h_k$. In Figure 3(ii) the searcher hits another obstacle, leaves this one at $t = \ell_{k-1}$, because—for some reason or other—the faulty angle counter $\omega(t)$ gets zero in $\ell_{k-1}$. Then it hits the first obstacle again at $t = h_k$.[4] We see that missing a leave point may lead to an infinite cycle.

[4]Note that the free space condition is not violated in this figure: the headings between

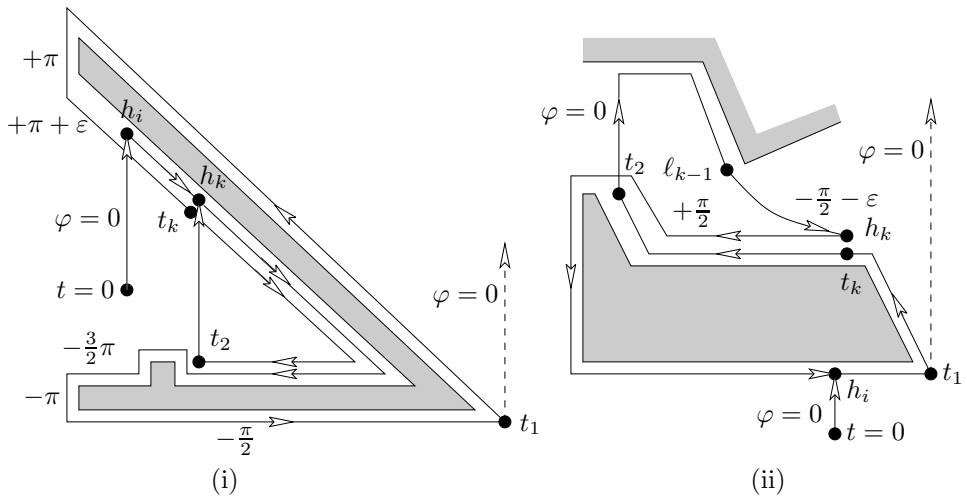



Figure 3: Missing a leave point can lead to an infinite cycle.

In both figures, $P(h_k)$ is visited twice, the first time at $t_k$ and a second time at $h_k$. In Figure 3(i), the heading $\varphi(t_k)$ is slightly larger than $\pi$, in Figure 3(ii) $+\frac{\pi}{2}$. Note that the angle counter at this time is smaller than zero, because the searcher still follows an obstacle boundary.

Observe that the searcher in Figure 3(ii) makes a second mistake: The heading $\varphi(h_k)$ in the hit point $h_k$ is not zero as it should be according to the Pledge algorithm. This mistake may occur because the preceding obstacle was left too early or too late, or the path between the obstacles is not a straight line segment, or it may be a combination of both reasons. However, the heading $\varphi(h_k)$ is equal to $-\frac{\pi}{2} - \varepsilon$ and both deviations (i.e., the faulty angle counter values in $t_k$ and $h_k$) sum up to an error that is slightly larger than $\pi$, too.

Note that the problem is not related to a second visit of a single point; the path generated by an error-free Pledge algorithm may have a lot of such repeated visits of the same point (imagine a searcher starting in the center of a spiral). Instead, the reason is that the heading in $t_k$ is too large compared to the heading in the hit point.

These observations lead to the conjecture that $\varphi(t_k) - \varphi(h_k) < \pi$ should hold whenever the searcher hits an obstacle at time $h_k$ and $P(h_k)$ was already visited or will be visited again at another time $t_k$. We refer to this condition

$\ell_{k-1}$ and $h_k$ are never smaller than $-\pi + \varepsilon$, every other heading in the free space is 0. Thus, $\varphi(t_1) - \varphi(t_2) < \pi$ is fulfilled for every $t_1, t_2 \in \mathcal{C}_{\text{free}}$.

as the *obstacle condition*.

Additionally, we want our searcher to resemble a searcher using the Pledge algorithm. Thus, we define a direction to circle obstacles. Further, the value of the angle counter changes only in vertices, so there is no need to leave an obstacle anywhere else.

**Theorem 1** *A searcher using the Pledge algorithm will escape from an unknown maze (provided that it is possible to leave the maze from the searchers start point), if its path,* $\Pi$*, fulfills the following conditions:*

1. *The searcher circles an obstacle using the* left-hand rule.
2. *Every leave point belongs to a vertex of an obstacle.*
3. $\forall t_1, t_2 : P(t_1), P(t_2) \in \mathcal{C}_{free} \Rightarrow \varphi(t_1) - \varphi(t_2) < \pi$ *(free-space condition).*
4. $\forall h_i, t : h_i$ is a hit point $\wedge\, P(t) = P(h_i) \Rightarrow \varphi(t) - \varphi(h_i) < \pi$ *(obstacle condition).*

Obviously, an error-free searcher using the Pledge algorithm fulfills these conditions: Its heading in the free space is always zero; thus, the free-space condition is satisfied. Moreover, the angle counter is always less than or equal to zero when moving along an obstacle, so the obstacle condition is fulfilled, too.

To prove Theorem 1 we use two important properties of the searcher's path that we show in the following three lemmata. The first property is that paths fulfilling the conditions in Theorem 1 have no crossings. Informally, a crossing occurs if it is not possible to remove the intersection by separating the path segments. See Figure 4: In (i) we can move the dashed part of the path a little bit upward or downward, but in any case the dashed part will intersect the other part. In contrast, in (ii) there is no intersection if we move the dashed part a little bit downward. Thus, these parts of the path just touch each other, but have no crossing. Note that a path that fulfills the conditions in Theorem 1 can touch itself.

**Lemma 2** *A path that fulfills the conditions in Theorem 1 cannot cross itself.*

**Proof.** Let us assume that the path, $\Pi$, crosses itself. Consider the *first* crossing of $\Pi$; that is, there are two parameters, $t_1$ and $t_2$ with $t_1 < t_2$, such that a crossing occurs at $t_2$ (i.e., $P(t_1) = P(t_2)$) and no crossing exists

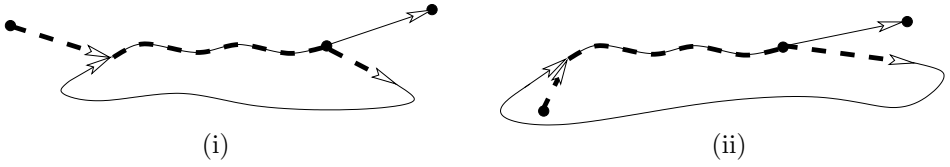



Figure 4: The difference between (i) a crossing and (ii) a touch.

before $t_2$. If a crossing happens in the free space, $\Pi$ obviously violates the free-space condition. Thus, we assume that the crossing occurs in $\mathcal{C}_{\text{semi-free}}$. The loop in $\Pi$ between $t_1$ and $t_2$ makes either a full counterclockwise or a full clockwise turn.

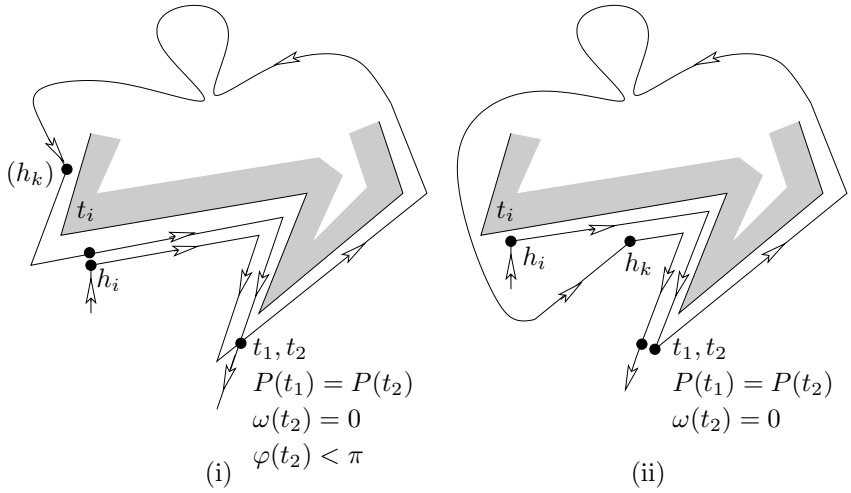



Figure 5: (i) A counterclockwise loop and a crossing, (ii) no crossing.

Let us consider the case of a counterclockwise loop, see Figure 5(i). The path hits an obstacle at $h_i$, makes a counterclockwise turn, meets $P(h_i)$ at $t_i$ again, and has a crossing at $t_2 \geq t_i > h_i$. Note that there is no crossing if the loop between $t_1$ and $t_2$ does not meet the point $P(h_i)$; see Figure 5(ii).

W.l.o.g. we assume $\varphi(h_i) = 0$. The loop may or may not leave the obstacle;[5] however, we reach $t_i$ with the heading $\varphi(t_i) + (-\gamma) = 2\pi$, because the loop has no further crossings. At $h_i$ the searcher turns clockwise with angle $\gamma$ to follow the obstacle boundary, so $-\pi < \gamma < 0$ must hold. Hence,

[5]Figure 5 shows the case that the searcher leaves the obstacle, so there is another hit point, $h_k$. If the searches does not leave the obstacle, there is no hit point between $t_1$ and $t_2$.

we have $\varphi(t_i) = 2\pi + \gamma > \pi$, and the obstacle condition is violated.

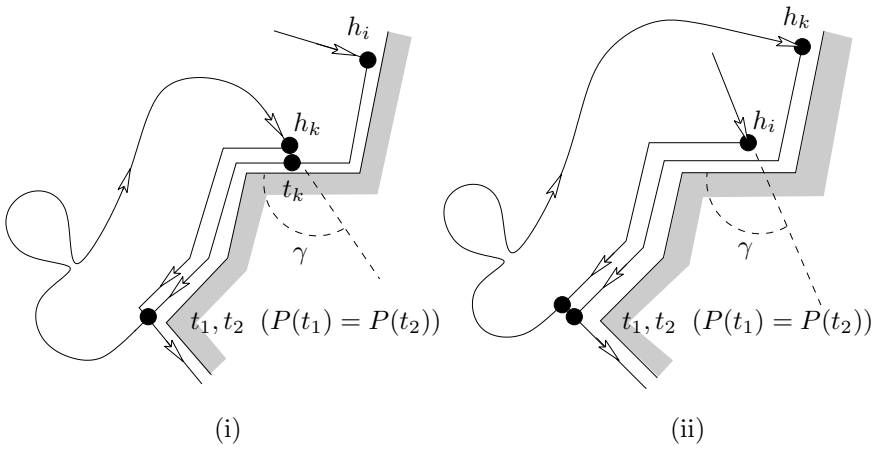



Figure 6: (i) A clockwise loop and a crossing, (ii) no crossing.

Now we look at a clockwise turn. The path hits an obstacle at $h_i$, and follows the obstacle for some time before it leaves the obstacle. Eventually, it returns to the obstacle at another hit point $h_k > h_i$ and has a crossing at $t_2$, see Figure 6(i). The point $P(h_k)$ has to be met before at $t_k$ with $h_i < t_k < t_1$; otherwise, the path only touches itself and there is no crossing at $t_2$, see Figure 6(ii).

Let $\varphi(h_k^+)$ denote the heading immediately after the searcher has turned in $h_k$; that is, $h_k^+ := h_k + \varepsilon$ such that $\varphi(h_k^+) = \varphi(h_k) + \gamma$ holds. Again, we have $-\pi < \gamma < 0$. On the other hand, the path has made a full clockwise turn between $t_k$ and $h_k^+$; thus, $\varphi(h_k^+) = \varphi(t_k) - 2\pi$. Altogether, we have

$$\varphi(t_k) - \varphi(h_k) = \varphi(h_k^+) + 2\pi - \varphi(h_k^+) + \gamma > \pi$$

and—again—the obstacle condition is violated. It follows that a first crossing cannot exist; thus, the path cannot cross itself. □

**Lemma 3** *A path that fulfills the conditions in Theorem 1 hits every edge in the environment at most once.*

**Proof.** Let us assume that the path Π hits an edge $e$ more than once: After a first hit at $h_i$ the searcher moves around and hits $e$ again at $h_k$, see Figure 7. At time $h_i$ and $h_k$ the searcher turns clockwise to follow the edge $e$, therefore $-\pi < \gamma_i, \gamma_k < 0$ holds. Let $\varphi(h_i^+)$ and $\varphi(h_k^+)$ be defined as in the proof of Lemma 2. W. l. o. g. we assume $\varphi(h_i^+) = 0$. Because the path in

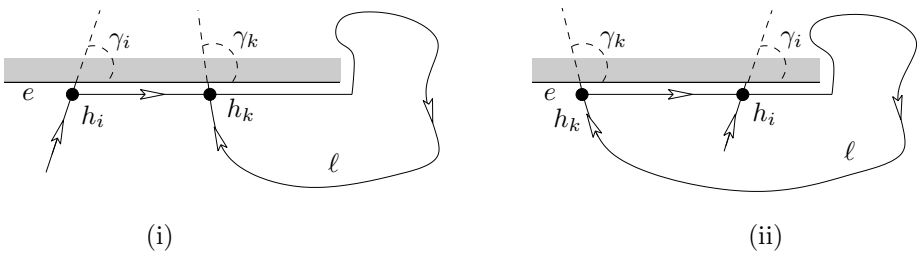



Figure 7: A path that hits an edge twice.

$h_i^+$ and $h_k^+$ follows the same edge $e$, the headings $\varphi(h_i^+)$ and $\varphi(h_k^+)$ must be the same modulo $2\pi$; thus, there must be a $j \in \mathbb{Z}$ such that $\varphi(h_k^+) = 2j\pi$ holds.

For $j \neq 0$ with $\varphi(h_i) = -\gamma_i$ and $\varphi(h_k) = \varphi(h_k^+) - \gamma_k$ follows that $|\varphi(h_k) - \varphi(h_i)| = |2j\pi - \gamma_k + \gamma_i| > \pi$ holds. Thus, at least one of $\varphi(h_k) - \varphi(h_i)$ or $\varphi(h_i) - \varphi(h_k)$ is greater than $\pi$, and the free-space condition is violated.

Therefore, we can assume that $j = 0$ and $\varphi(h_k^+) = 0$ holds. Consider the part of $\Pi$ between the first and the second visit of $P(h_k)$—in a situation as shown in Figure 7(i)—, or the path between the consecutive visits of $P(h_i)$ as shown in Figure 7(ii). If this loop, $\ell$, has no crossings, then $\ell$ is a Jordan curve and the searcher makes a $\pm 2\pi$ turn in $\ell$. Thus, $\varphi(h_k^+)$ is equal to $\pm 2\pi$, in contradiction to our assumption. Hence, the path between the two visits of $e$ must have at least one crossing. But this contradicts Lemma 2. □

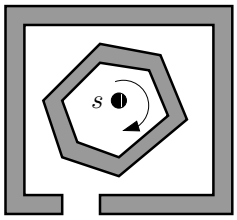



Figure 8: A searcher that is trapped in a courtyard.

**Lemma 4** *If a searcher does not leave an obstacle although it fulfills the conditions in Theorem 1 then it is trapped in a courtyard.*

**Proof.** A searcher that does not leave an obstacle circles the obstacle on and on. By Lemma 2 the path along the obstacle has no crossing; thus, for every full turn the heading either increases or decreases by $2\pi$. If the

heading increases for every turn by $2\pi$, the searcher will eventually meet the position of the most recent hit point[6] with a heading that is much larger than it was in the hit point. Thus, the obstacle condition will eventually be violated. If the heading decreases for every turn, the searcher follows the *inner* wall of an obstacle; thus, it is trapped in a courtyard as shown in Figure 8.[7] □

Finally, with the three lemmata we are able to show that the conditions in Theorem 1 are sufficient to solve our problem:

**Proof.** (Theorem 1) Let us assume that is possible to leave the maze from the searcher's start point. A searcher that fulfills the conditions in Theorem 1 always leaves an obstacle by Lemma 4. Further, by Lemma 3 such a searcher hits every edge in the environment at most once. Altogether, at the latest after every edge is visited the searcher escapes because it cannot hit any further edge. □

[6]There is always a most recent hit point when the searcher follows an obstacle. This is obvious if the start point is in the free space. If the start point is on the boundary of an obstacle, we define $t = 0$ as the first hit point.

[7]Note that a searcher using the Pledge algorithm has no chance to recognize that it is trapped!

# 4 Applications

In this section we consider some applications of Theorem 1 to different types of searchers.

## 4.1 Searcher Using an Error-Prone Compass

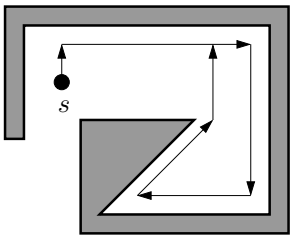



Figure 9: A strategy that leaves an obstacle as soon as the compass points to *north* can be trapped.

Now, let us return to our compass-equipped searcher. The Pledge algorithm deals only with relative turning angles and does not use the compass' ability to give an absolute direction. However, this is no weak point of the Pledge algorithm, because any strategy has to track at least the number of full turns: A strategy that deals only with absolute directions and decides, for example, to leave an obstacle as soon as the compass points to *north* (i.e., $\omega \bmod 2\pi = 0$ holds) can be trapped, see Figure 9. Thus, a searcher equipped with a compass does not extend the model that is assumed for a searcher using the Pledge algorithm. Note that such a searcher no longer needs an angle counter, $\omega(t)$. Instead, it needs a counter, $c(t)$, for the number of full turns. The compass readings provide $\omega(t) \bmod 2\pi$, $c(t)$ provides $\omega(t) \operatorname{div} 2\pi$.

Theorem 1 can be easily applied to determine the maximal error in the compass readings. Amazingly, the searcher can escape even if the errors in the compass readings are quite large! We have to require only that the error *in every reading* is below $\pm\frac{\pi}{2}$. We can afford such a big error, because the errors in the angle counting do not accumulate; every compass reading yields a value that is accurate with respect to the real heading of the searcher (i.e., the compass readings provide $\varphi(t) \bmod 2\pi$ with accuracy $\pm\frac{\pi}{2}$)!

**Corollary 5** *A searcher equipped with a compass will find the exit from an unknown maze using a simple Pledge-like strategy if the error in a single compass reading is below $\frac{\pi}{2}$ (provided that it is possible to leave the maze).*

**Proof.** If the compass has a measuring error less than $\frac{\pi}{2}$ it is easy to ensure that the heading of the searcher in the free space remains in an interval of $]-\frac{\pi}{2}, \frac{\pi}{2}[$. This guarantees the free-space condition at hand. Additionally, at every hit point $h_i$ we have $\varphi(h_i) \in ]-\frac{\pi}{2}, \frac{\pi}{2}[$.

For the detection of leave points, we have to track the number of full turns of the searcher in addition to the compass reading that points us to our preferred escape direction (w.l.o.g. $0°$). It is easy to detect $+2\pi$ or $-2\pi$ turns along the walls—we simply observe the full turns of the compass needle—, although again erroneous within a range of $]-\frac{\pi}{2}, \frac{\pi}{2}[$ due to the compass inaccuracy. Therefore, the difference between the angle counter $\omega(t)$ and the heading $\varphi(t)$ is in the range $]-\frac{\pi}{2}, \frac{\pi}{2}[$. This yields $\varphi(t) < \omega(t)+\frac{\pi}{2}$. But $\omega(t)$ is never greater than zero, because the angle counter gets negative in a hit point and the searcher leaves the obstacle as soon as $\omega(t) = 0$ is fulfilled, so we have $\varphi(t) < \frac{\pi}{2}$. Thus, while the searcher follows a wall its heading is always less than $\frac{\pi}{2}$. Further, we have $\varphi(h_i) \in ]-\frac{\pi}{2}, \frac{\pi}{2}[$ as above. Altogether, the obstacle condition is fulfilled. □

### 4.2 Searcher with Exact Free Motion

In this section, we assume a searcher that is able to move along a straight path between obstacles correctly, or the deviations on a straight path are negligible; that is, the searcher always hits the same edge as an error-free agent using the Pledge algorithm. Only the angle counter of the searcher is inaccurate in some way. Let $\Delta(t)$ denote the absolute value of the difference between the searcher's heading and its angle counter (i.e., $\Delta(t) := |\varphi(t) - \omega(t)|$).

**Lemma 6** *If the searcher is able to move correctly along a straight path in the free space and ensures that $\Delta(t) < \pi$ holds for all $t$, then it is able to escape from an unknown maze using the Pledge algorithm.*

**Proof.** We have to show that our new condition does not violate the conditions in Theorem 1.

First, let us assume that the free-space condition is not met, so there must be two points $t_1, t_2$ in the free space where $\varphi(t_1) - \varphi(t_2) \geq \pi$ holds. W.l.o.g. we assume $\varphi(t_2) = 0$. Because the searcher correctly moves along a straight line in the free space, the headings at the leave point, the following hit point, and all points between them must be the same. Thus, there must be a leave point $\ell_k$ with $\varphi(\ell_k) = \varphi(t_1) \geq \pi$. But the searcher leaves an obstacle only, if its angle counter has reached zero (i.e., $\omega(\ell_k) = 0$). Thus, $\Delta(\ell_k) = |\varphi(\ell_k) - \omega(\ell_k)| \geq \pi$ holds.

Second, we assume that the obstacle condition is violated. Then there must be an obstacle with a hit point $h_k$ and another point $t_k$ with $P(t_k) = P(h_k)$, such that $\varphi(t_k) - \varphi(h_k) > \pi$ holds. W.l.o.g. let $\varphi(h_k) = 0$, then $\varphi(t_k) \geq \pi$ holds. The angle counter, $\omega(t)$, cannot be greater than zero, because the searcher leaves the obstacle as soon as the counter becomes zero. From $\omega(t_k) \leq 0$ we conclude $\Delta(t_k) = |\varphi(t_k) - \omega(t_k)| \geq \pi$. □

If the searcher escapes and the conditions in Theorem 1 are satisfied, the searcher visits every vertex at most once between two consecutive hit points. Thus, with Lemma 3 the searchers visits at most $n^2$ vertices, where $n$ is the total number of vertices in the environment. Now, Let $\beta_i$ denote the difference between the real angle (the nominal value) and the measured angle at the $i$th turn, and let the searcher's maximal error be $\beta_{\max} := \max \beta_i$. With Lemma 6 we have:

**Corollary 7** *A searcher that is able to move correctly along a straight path in the free space and guarantees $|\beta_{\max}| < \frac{\pi}{n^2}$, where $n$ is the total number of vertices in the environment, is able to escape from an unknown maze with the Pledge algorithm (provided that it is possible to leave the maze).*

### 4.3 Searcher Moving in (Pseudo-) Orthogonal Scenes

In this section, we observe the Pledge algorithm in a special case of scenes—orthogonal scenes—, but allow them to be inaccurate. A scene is called *orthogonal*, if every polygon in the scene is orthogonal; that is, the polygon edges meet with internal angles of either $\frac{\pi}{2}$ or $\frac{3}{2}\pi$.[8]

W.l.o.g. we assume that the polygon edges are axis parallel. It is easy to see that we can simplify the Pledge algorithm in orthogonal scenes. We have only two types of vertices: Vertices with an outer angle of $\frac{\pi}{2}$, and vertices with an outer angle of $\frac{3}{2}\pi$. It is sufficient to keep track of the number vertices of both types that the searcher passes by. Thus, the angle counting amounts to count $-1$ for a $\frac{3}{2}\pi$ vertex and $+1$ for a $\frac{\pi}{2}$ vertex. Further, we count $+1$ for the hit point; see Figure 10(i).

Now, we assume that the environment is more or less orthogonal; we allow small deviations from the strict axis-parallelism. To quantify these deviations, we use the angle of edges in a tilted position. More precisely, we define the *divergence*, $\mathrm{div}(e)$, of an edge $e = (v, w)$ as the smallest angle between $e$ and an axis-parallel line through $v$ or $w$, see Figure 10(ii).

[8] Polygons with this property are also called *rectilinear*.

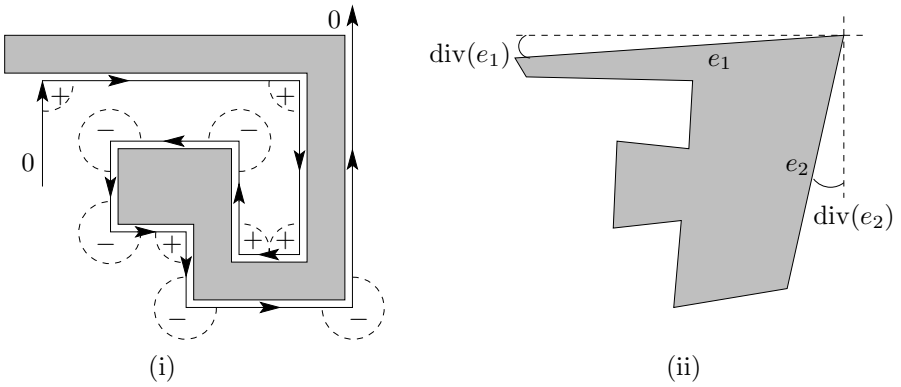



Figure 10: (i) Angle counting for an orthogonal polygon, (ii) pseudo-orthogonal polygon and divergence div.

**Definition 8** *A simple polygon, $P$, is called* pseudo orthogonal *if*

$$\#\textit{convex vertices of } P \;=\; \#\textit{reflex vertices of } P + 4$$

*holds.*[9] *We call $P$ $\delta$-*pseudo orthogonal *if $P$ also satisfies*

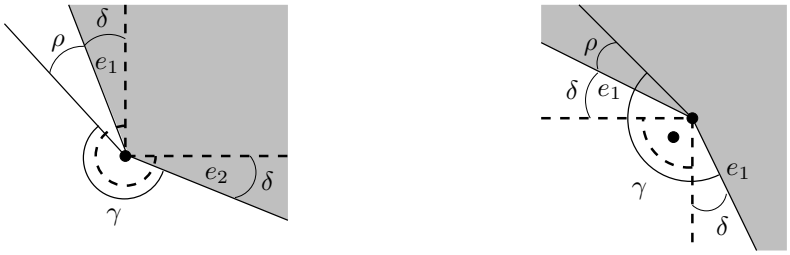
$$\operatorname{div}(P) := \max_{e \in P} \operatorname{div}(e) \leq \delta\,.$$

*A set of polygons, $\mathcal{P}$, is called ($\delta$-)pseudo-orthogonal scene if every $P_i \in \mathcal{P}$ is ($\delta$-)pseudo orthogonal.*

The searcher's angle counter may be a second source of errors. We assume that the searcher is able to measure angles with an accuracy of $\rho$; that is, the difference between the nominal value and the measured value is smaller than $\rho$. Now, we are interested in upper bounds for $\delta$ and $\rho$ that guarantee a successful application of the simplified Pledge algorithm to a $\delta$-pseudo-orthogonal scene.

First, we have to guarantee that the searcher is able to distinguish convex and reflex vertices correctly. Taking the worst case into account, we assume that both edges adjacent to the current vertex deviate with the maximal angle $\delta$ and the searcher's measuring error is maximal, too; see Figure 11. To ensure a correct classification, we require that the measured outer angle, $\gamma$, is greater than $\pi$ for a convex vertex, and smaller than $\pi$ for a reflex

[9] A *convex* (*reflex*) vertex of a simple polygon is a vertex with an inner angle smaller (greater) than $\pi$.



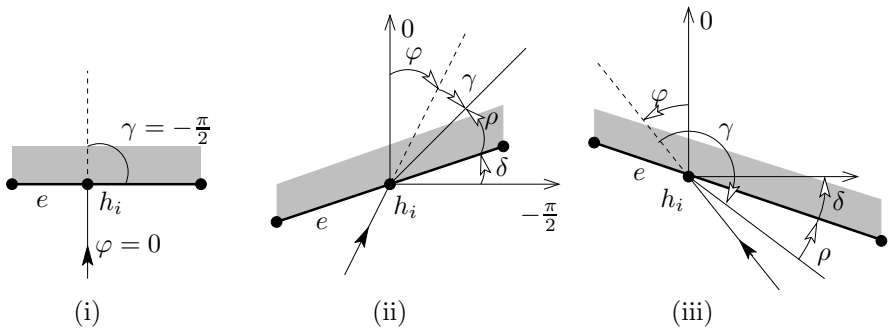

convex vertex reflex vertex

Figure 11: Maximal deviation between the outer measured angle ($\gamma$) and the outer nominal value (dashed) for a convex and a reflex vertex.

vertex. Thus, we have to ensure

$$\frac{3}{2}\pi - 2\delta - \rho > \pi \quad \Leftrightarrow 2\delta + \rho < \frac{\pi}{2} \qquad \text{and} \qquad \frac{\pi}{2} + 2\delta + \rho < \pi \quad \Leftrightarrow 2\delta + \rho < \frac{\pi}{2}\,.$$

On the other hand, the searcher should be able to maintain its initial directions in the free space; see Algorithm 1. To establish bounds for the maximal deviation from the initial direction, we consider the headings in the hit points. In the error-free case, the searcher hits only horizontal edges, but in a $\delta$-pseudo-orthogonal scene the searcher has to determine whether an edge is classified as *horizontal* or as *vertical*. If the searcher hits a *horizontal* edge, it has to surround an obstacle using the simplified angle-counting procedure, whereas a *vertical* edge can be ignored; that is, the searcher just slides along this edge.




(i) (ii) (iii)

Figure 12: The searcher hits a horizontal edge (i) error-free case, (ii) small absolute value for $\gamma$, (iii) large absolute value for $\gamma$.

As in the preceding case, let $\gamma$ denote the *measured* angle that the searcher turns in a hit point to follow a wall. Considering the divergence in the edges, it is reasonable that the searcher assumes an edge to be *horizontal*, if $-\frac{1}{4}\pi > \gamma > -\frac{3}{4}\pi$ holds, and *vertical* otherwise. Figure 12(ii) and (iii) show the worst cases for $\gamma$. The searcher hits an edge $e$ in $h_i$ with the heading $\varphi = \varphi(h_i)$. In Figure 12(ii) the deviations $\varphi, \delta$ and $\rho$ make the absolute value of $\gamma$ as small as possible, in (iii) as large as possible. To ensure that $\gamma \in ]-\frac{1}{4}\pi, -\frac{3}{4}\pi[$ holds, we have to restrict the heading, $\varphi$, in the hit point. From Figure 12(ii) we get

$$\gamma = -\frac{\pi}{2} - \varphi + \delta + \rho < -\frac{\pi}{4} \;\Leftrightarrow\; -\frac{\pi}{4} + \delta + \rho < \varphi\,,$$

and Figure 12(iii) yields

$$\gamma = -\Big(\frac{\pi}{2} + \varphi + \delta + \rho\Big) > -\frac{3}{4}\pi \;\Leftrightarrow\; \frac{\pi}{4} - \delta - \rho > \varphi\,.$$

Thus, the searcher detects a *horizontal* edge correctly, if $\varphi(h_i) \in ]-\frac{\pi}{4} + \delta + \rho, \frac{\pi}{4} - \delta - \rho[$ holds. Thus, we require $\delta + \rho < \frac{\pi}{4}$, which includes the preceding condition $2\delta + \rho < \frac{\pi}{2}$.

In an orthogonal scene, the searcher can rely on the vertical edges to determine the correct heading for a movement in the free space. In our case, the searcher leaves an obstacle moving along a *vertical* edge, too, but this means that the heading in a leave point is in the range $[-\delta, +\delta]$. Considering the range for the heading in the (next) hit point, we conclude that the deviation in the free space with respect to the heading in the leave point has to be smaller than $\frac{\pi}{4} - 2\delta - \rho$. Altogether, we have

**Corollary 9** *Let an unknown $\delta$-pseudo-orthogonal scene be given, and let us assume that it is possible to escape from this maze. If the searcher is able to measure angles within an accuracy of $\rho$ with $\delta + \rho < \frac{\pi}{4}$, and in free space does not deviate by more than $\frac{\pi}{4} - 2\delta - \rho$ from the direction in which it left the last obstacle, the searcher is able to escape using the Pledge algorithm with the simplified angle counting.*

# 5 Conclusion

We considered the Pledge algorithm under errors in sensors and motion and established sufficient requirements for the searcher to leave an unknown maze. In particular, we showed that a searcher equipped with a simple compass will fulfill this task, we gave upper bounds for errors in pseudo-rectilinear mazes and for searchers with correct straight motion.

## Acknowledgements

We would like to thank Peter Gritzmann for pointing out the problem, Rolf Klein for helpful discussions, and the anonymous referees for a lot of valuable comments that helped to improve the presentation of this paper.

## References

[1] H. Abelson and A. A. diSessa. *Turtle Geometry*. MIT Press, Cambridge, 1980.

[2] D. Angluin, J. Westbrook, and W. Zhu. Robot navigation with distance queries. *Siam J. Comput.*, 30(1):110–144, 2000.

[3] M. Batalin and G. S. Sukhatme. Coverage, exploration and deployment by a mobile robot and communication network. *Telecommunication Systems Journal*, 26:181–196, 2004.

[4] M. A. Batalin and G. S. Sukhatme. Efficient exploration without localization. In *Proc. IEEE Internat. Conf. Robot. Autom.*, 2003.

[5] P. Berman. On-line searching and navigation. In A. Fiat and G. Woeginger, editors, *Online Algorithms: The State of the Art*, volume 1442 of *Lecture Notes Comput. Sci.*, pages 232–241. Springer, 1998.

[6] J. Borenstein and L. Feng. UMBmark: A benchmark test for measuring odometry errors in mobile robots. In *Proc. 1995 SPIE Conf. Mobile Robots*, pages 113–124, 1995.

[7] J. Byrne, R. Jarvis, S. Yuta, and A. Zelinsky. Planning paths of complete coverage of an unstructured environment by a mobile robots. In *Internat. Conf. Adv. Robotics*, pages 553–538, 1993.

[8] K. S. Chong and L. Kleeman. Accurate odometry and error modelling for a mobile robot. In *IEEE Internat. Conf. Robot. Automat.*, pages 2783–2788, 1997.

[9] E. Demaine, A. López-Ortiz, and I. Munro. Robot localization without depth perception. In *Proc. 8th Scand. Workshop Algorithm Theory*, volume 2368 of *Lecture Notes Comput. Sci.*, pages 249–259, 2002.

[10] X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment I: The rectilinear case. *J. ACM*, 45(2):215–245, 1998.

[11] G. Dudek, E. Milios, and I. M. Rekleitis. Multi-robot collaboration for robust exploration. *Ann. Math. Artif. Intell.*, 31:7–40, 2001.

[12] P. Gritzmann. Personal communication via Rolf Klein. 2001.

[13] U. Handel, T. Kamphans, E. Langetepe, and W. Meiswinkel. Polyrobot — an environment for simulating strategies for robot navigation in polygonal scenes. Java Applet, 2002. http://www.geometrylab.de/Polyrobot/.

[14] A. Hemmerling. *Labyrinth Problems: Labyrinth-Searching Abilities of Automata*. B. G. Teubner, Leipzig, 1989.

[15] F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem. *SIAM J. Comput.*, 31:577–600, 2001.

[16] C. Icking, T. Kamphans, R. Klein, and E. Langetepe. On the competitive complexity of navigation tasks. In H. Bunke, H. I. Christensen, G. D. Hager, and R. Klein, editors, *Sensor Based Intelligent Robots*, volume 2238 of *Lecture Notes Comput. Sci.*, pages 245–258, Berlin, 2002. Springer.

[17] I. Kamon and E. Rivlin. Sensory-based motion planning with global proofs. *IEEE Trans. Robot. Autom.*, 13:814–822, 1997.

[18] T. Kamphans and E. Langetepe. The Pledge algorithm reconsidered under errors in sensors and motion. In *Proc. of the 1th Workshop on Approximation and Online Algorithms*, volume 2909 of *Lecture Notes Comput. Sci.*, pages 165–178. Springer, 2003.

[19] R. Klein. *Algorithmische Geometrie*. Springer, Heidelberg, 2nd edition, 2005.

[20] B. J. Kuipers and Y.-T. Byun. A robust qualitative method for spatial learning in unknown environments. In *Proc. 7th Nat. Conf. Artif. Intell.* Morgan Kaufman, 1988.

[21] B. J. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *J. Robot. Auton. Syst.*, 8:47–63, 1991.

[22] S. L. Laubach. *Theory and Experiments in Autonomous Sensor-Based Motion Planning with Applications for Flight Planetary Microrovers*. Ph.D. thesis, California Institute of Technology, Pasadena, CA, 1999.

[23] S. M. LaValle, S. Rajko, and S. Sachs. Visibility-based pursuit-evasion in an unknown planar environment. *Internat. J. Robot. Res.*, 23:3–26, 2004.

[24] D. Lee and M. Recce. Quantitative evaluation of the exploration strategies of a mobile robot. *Internat. J. Robot. Res.*, 16(4):413–447, 1997.

[25] A. López-Ortiz and S. Schuierer. Simple, efficient and robust strategies to traverse streets. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 217–222, 1995.

[26] 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.

[27] V. J. Lumelsky and S. Tiwari. An algorithm for maze searching with azimuth input. In *Proc. 1994 Internat. Conf. Robot. Automat.*, pages 111–116, 1994.

[28] J. S. B. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 633–701. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[29] H. Noborio and T. Yoshioka. On the sensor-based navigation under position and orientation errors. *J. Automatisierungstechnik*, 48:281–288, 2000.

[30] H. Noborio, T. Yoshioka, and T. Hamaguchi. On-line deadlock-free path-planning algorithms by means of a sensor-feedback tracing. In *Proc. IEEE Int. Conf. Systems, Man, Cybernetics*, pages 1291–1296, 1995.

[31] H. Noborio, T. Yoshioka, and T. Hamaguchi. On-line deadlock-free path-planning algorithms in the presence of a dead reckoning error. In *Proc. IEEE Int. Conf. Systems, Man, Cybernetics*, pages 483–488, 1995.

[32] S. Rajko and S. M. La Valle. A pursuit-evasion bug algorithm. In *Proc. IEEE Conf. Robotics Automat.*, 2001.

[33] N. S. V. Rao, S. Kareti, W. Shi, and S. S. Iyengar. Robot navigation in unknown terrains: introductory survey of non-heuristic algorithms.

Technical Report ORNL/TM-12410, Oak Ridge National Laboratory, 1993.

[34] M. Sharir. Algorithmic motion planning. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 40, pages 733–754. CRC Press LLC, Boca Raton, FL, 1997.

[35] A. Stentz. Optimal and efficient path planning for unknown and dynamic environments. Technical Report CMU-RI-TR-93-20, Carnegie Mellon University Robotics Institute, August 1993.

[36] B. Yamauchi, A. Schultz, and W. Adams. Mobile robot exploration and map-building with continuous localization. In *Proc. IEEE Internat. Conf. Robot. Autom.*, 1998.

[37] A. Zelinsky. A mobile robot exploration algorithm. *IEEE Transact. Robot. Automat.*, 8(6):707–717, 1992.