The Pledge Algorithm Reconsidered under Errors in Sensors and Motion

Tom Kamphans and Elmar Langetepe

University of Bonn
Institute of Computer Science I
Römerstraße 164
53117 Bonn
Germany
{kamphans,langetep}@cs.uni-bonn.de
http://web.cs.uni-bonn.de/I/agklein.html

Abstract. We consider the problem of escaping from an unknown polygonal maze under limited resources and under errors in sensors 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. The pledge algorithm sums up the turning angles along the boundary of the obstacles and plans its way by using a single counter and no further information. The correctness proof makes use of the fact that motions in the free space and the measurement of turning angles can be done exactly. We consider the case that the free motion of the robot and the computation of turning angles are erroneous.
This work is a first attempt to incorporate error assumptions into theoretically sound proofs of online motion planning algorithms.

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 [12]. They invented the BUG algorithm and many variants of this algorithm were discussed and analyzed afterwards, for example see Rajko and La Valle [14] and Kamon and Rivlin [10]. A BUG-like algorithm was also used for the Mars-Rover project, see Laubach [11]. The correctness of the algorithms are 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. [4] studied the case of a robot equipped with a vision system that maps an unknown, simple, rectangular environment. They developed a strategy and prove that under correct vision and motion the corresponding path of the robot is never longer than $\sqrt{2}$ times the optimal path computed under full information in advance. The strategy is called

R. Solis-Oba and K. Jansen (Eds.): WAOA 2003, LNCS 2909, pp. 165–178, 2004.

$\sqrt{2}$-*competitive* in this model. Under the same idealistic assumptions Hoffmann et. al. [8] presented and analyzed an 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 e. g. Batalin and Sukhatme [2] or Yamauchi et. al. [17]. For a general overview of theoretical online motion planning problems see the surveys [15, 16, 3, 13, 9].

Theoretical correctness results and performance guarantees from labyrinth theory or computational geometry suffer from idealistic assumptions, therefore in the worst case a correct implementation is impossible. On the other hand practioners analyze correctness results and performance guarantees mainly statistically. 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?

As a first approach we consider a simple and theoretically sound online strategy, the well-known pledge algorithm, see Abelson and diSessa [1] and Hemmerling [7]. Given an unknown polygonal maze, the robot has to leave the maze, using nothing else than a touch sensor, the ability to measure its turning angles and a very limited amount of memory. The robot is not able to create a map or to set landmarks. The pledge algorithm assumes that the robot is able to move a straight line between the obstacles and to count its turning angles correctly. Gritzmann [5] remarked that it would be interesting to know how the pledge algorithm behaves, if those assumptions cannot be fulfilled. Of course, if the robot can make arbitrary big mistakes, there are always environments, in which the robot is hopelessly trapped. But what if the robot's errors are small enough? Are there upper bounds for measuring errors? We investigate which conditions must hold to ensure that a robot can leave an unknown maze with a pledge-like algorithm.

The paper is organized as follows. In Sect. 2 we specify the robot's task and the model of the robot and its environment. We recapitulate the pledge algorithm, and analyze the sources for errors in this algorithm. With this insight, in Sect. 3 we define a class of curves in the robot's environment that fulfill a set of conditions. Further we show that a robot will escape from an unknown maze, if its path somehow *follows* a curve from that class. Last, in Sect. 5 we discuss the impact of our results on the design of a robot. Our main result is, that a robot equipped with a compass with reasonable small errors is able to leave an unknown maze using the pledge algorithm; see Sect. 4.1. Further we show results for a robot with exact free motion and bounded angle measuring error and for a robot in an 'almost rectangular' environment.

2 Preliminaries

Let us assume that a maze with polygonal shaped obstacles in the plane is given. The robot is able to recognize and follow a wall in a specified direction (w. l. o. g.

counter-clockwise) and to count the turning angles. How these abilities can be realized depends on the hardware of a specific robot. For example, the turning angles may be counted by odometry or by measuring angles along the walls with sensors. Other abilities for orientation and navigation are not required, especially it is not necessary that the robot can build a map of its environment.

The task of leaving an unknown maze can be solved using the well-known pledge algorithm, see algorithm 1, which performs only two types of movements: Either the robot follows the wall of an obstacle and counts the turning angles, or the robot moves through the free space between the obstacles in a fixed direction. The latter task always starts at vertices of the obstacles when the angle-counter reaches a pre-defined value. As soon as the robot leaves the maze it recieves a signal of success.

Algorithm 1: Pledge

REPEAT
  $\omega = 0$
  REPEAT
    Move in direction ω in the free space
  UNTIL Robot hits an obstacle
  REPEAT
    Follow the wall in counter-clockwise direction
    Count the overall turning angle in ω
  UNTIL Angle Counter $\omega = 0$
UNTIL Robot is outside the maze

Both types of movements in the pledge algorithm may be afflicted with errors. Either the turning angles are not measured exactly and the robot leaves the obstacle earlier or later than expected, or the robot cannot follow its initial direction during the movement in the free space. In the idealised setting the robot is error-free and it was shown by Abelson and diSessa [1] and Hemmerling [7] that in this case a robot will escape from a polygonal maze by performing the pledge algorithm provided that there is such a solution. An example of the robot's path using an error-free pledge algorithm is given in Fig. 1. The angle counting technique is illustrated for the second obstacle: After the robot hits the obstacle in p_2 it turns about $-\frac{\pi}{2}$ to follow the wall. In p_3 the robot turns about $+\frac{\pi}{4}$ to follow the next wall. Finally, in p_4 it turns about $+\frac{\pi}{4}$ again until the angle counter reaches zero and the robot leaves the obstacle. Observe that the robot does not leave the obstacle in p_1, since its angle counter is -2π instead of zero.

In the following we define a class $\mathcal{K}$ of curves in the robot's workspace. The curves in $\mathcal{K}$ represent possible paths that lead to an exit, even if the robot's sensors and motions are erroneous. For convenience, we assume that the robot is point-shaped, but this is no restriction as long as the robot is smaller than the distances between obstacles. So the parts of a curve that map to a movement along a wall, are line segments on the boundaries of obstacles. To escape from an unknown maze, the robot's strategy is not required to calculate a path

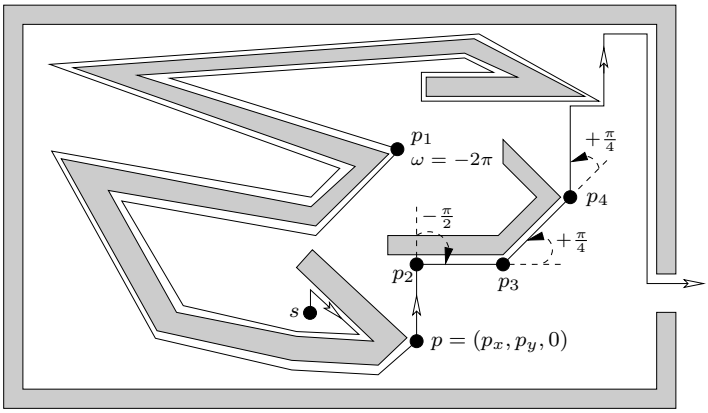

Fig. 1. The path of the pledge algorithm.

that matches a curve in $\mathcal{K}$ exactly, rather it is sufficient that the robot's path orientates essentially at a curve in $\mathcal{K}$. For example the robot may follow a wall in a certain distance, because it is not point-shaped, or it may follow a wall in a zig-zag manner. See Sect. 4 for more details.

Since every point on the curve represents a position as well as a heading, the curve is a subspace of the workspace $\mathcal{C} = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ and a point will be described as $C(t) = (P(t), \varphi(t))$, where $P(t) = (X(t), Y(t))$ denotes the position at time t and $\varphi(t)$ the *heading*. Note, that $\varphi(t)$ is the sum of all turns the curve has made so far, so if the curve has made two full turns counterclockwise then $\varphi(t)$ equals 4π instead of zero.

In order 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 half-free configurations, $\mathcal{C}_{\text{half}}$, that is the union of the boundaries of the obstacles, and last the free configurations, $\mathcal{C}_{\text{free}}$, where $P(t) \notin P_i$ for all obstacles P_i holds.

If a curve hits a point $p = P(H_i)$ in $\mathcal{C}_{\text{half}}$ after a movement through $\mathcal{C}_{\text{free}}$, we will call H_i a *hit point*. If the curve leaves $\mathcal{C}_{\text{half}}$ and enters the free space at $q = P(L_i)$, we call L_i a *leave point*. With respect to the pledge algorithm we assume that every leave point belongs to a vertex of an obstacle.

3 Sufficient Conditions

Let us assume that it is possible to leave a given unknown maze. Within this section we establish a set of sufficient conditions for escaping from the maze. The robot follows—somehow—a curve C and to ensure that the robot can escape we want to avoid infinite cycles in the curve.

There are mainly two sources of errors in the pledge algorithm: one concerns the motion in the free space and the other the motion along the boundaries of the obstacles. The first type of error occurs, if the robot is not able to move a

straight line in the free space, the latter, if the robot's angle counter is in some way inaccurate. Both types lead to a set of sufficient conditions that ensures the correctness of an error-afflicted pledge algorithm.

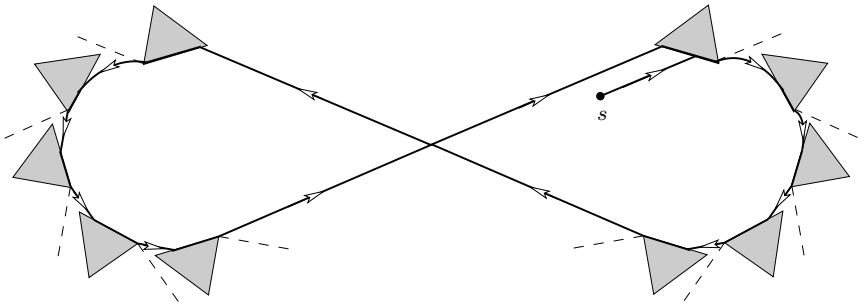

Fig. 2. Small errors along each boundary can sum up to a cycle.

To establish the set of conditions, we first observe that already small counting errors along the boundary of obstacles or deviations in the free 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 hit point. Between the obstacles the robot drifts a little bit away from the correct direction and ends up in an infinite loop. This would happen even if the curve would not be allowed to make a full 2π turn in the free space. Obviously, cycles would be inhibited, if the curve between two obstacles would stay in a wedge around the initial direction. In fact, this wedge can be as large as a half space! This leads to the condition that for any two points in the free space, the difference in the headings should be lower than π. We will refer to this as the *free space condition*.

Unfortunately the free space condition is not sufficient. Figure 3 shows two examples, where C has a cycle, although the free space condition is fulfilled. In both cases the curve starts in s, meets an obstacle in H_i, misses the first possible leave point p and leaves the obstacle at another vertex q. The curve in Fig. 3(i) hits the same obstacle again in H_k. In Fig. 3(ii) the curve hits another obstacle, leaves this one in L_k and hits the first obstacle again in H_k. In both cases $P(H_k)$ is visited two times, at H_k and t_k. In the first case, the heading in t_k is slightly larger than π, in the second case $+\frac{\pi}{2}$. Observe that the curve in Fig. 3(ii) represents a second mistake: the heading in the hit point H_k is not zero, as it should be according to the pledge algorithm. This may occur if the last obstacle was left too early or the path between the obstacles is not a straight line segment or this may be a combination of both reasons. However, the heading in H_k is $-\frac{\pi}{2} - \varepsilon$ and both mistakes sum up to an error that is slightly larger than π, too.

Note, that the problem is not related to a second visit of a single point. The curve of the error-free pledge algorithm, may have many self-hits. It is rather

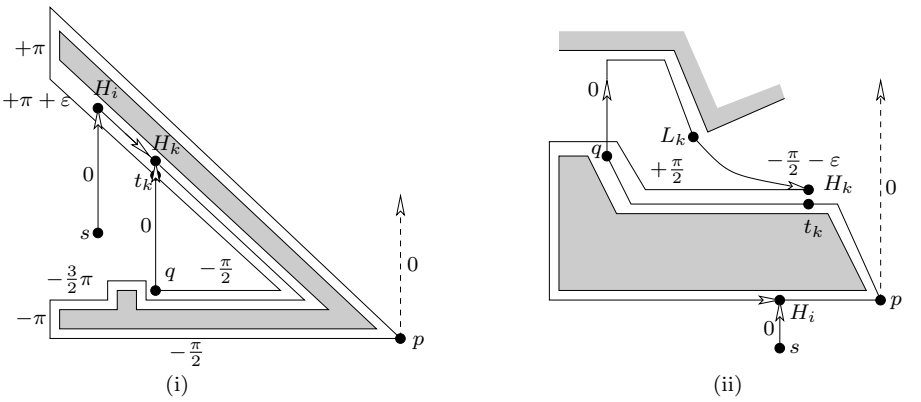

Fig. 3. Missing a leave point can lead to a cycle.

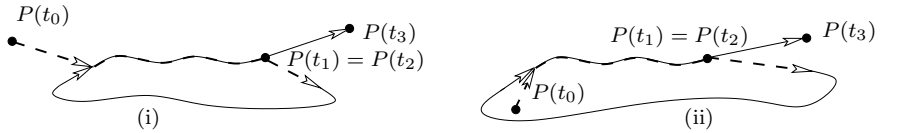

Fig. 4. The difference between a crossing (i) and a touch (ii) at t_2.

the fact that the heading in t_k is quasi overwinded in terms of the heading in the hit point.

Those observations lead to the conjecture that whenever the curve hits an obstacle at H_k and there exists a point t_k with $P(H_k) = P(t_k)$, then $\varphi(t_k) - \varphi(H_k) < \pi$ should hold. We will refer to this as the *obstacle condition*. In the rest of this chapter, we will show that both conditions are sufficient.

Definition 1. *Let $\mathcal{K}$ be the class of curves in $\mathcal{C}_{free} \cup \mathcal{C}_{half}$ that satisfy the following conditions:*

(i) The curve circles an obstacle in a counter-clockwise direction.
(ii) Every leave point belongs to a vertex of an obstacle.
(iii) $\forall t_1, t_2 \in C : P(t_1), P(t_2) \in \mathcal{C}_{free} \Rightarrow |\varphi(t_1) - \varphi(t_2)| < \pi$ *(Free space condition).*
(iv) $\forall H_i, t \in C : P(t) = P(H_i) \Rightarrow \varphi(t) - \varphi(H_i) < \pi$ (Obstacle condition).

Obviously, the curve of the error-free pledge algorithm is a curve in $\mathcal{K}$. The curves in $\mathcal{K}$ have two important properties that will be shown in the following two lemmata.

Lemma 2. *A curve $C \in \mathcal{K}$ cannot cross itself.*

Note, that a curve of $\mathcal{K}$ can touch itself, see Fig. 4.

Proof. Let us assume, C crosses itself. Consider the *first* crossing of C, i. e., there are two parameters t_1 and t_2 with $t_1 < t_2$ and $P(t_1) = P(t_2)$, where a crossing

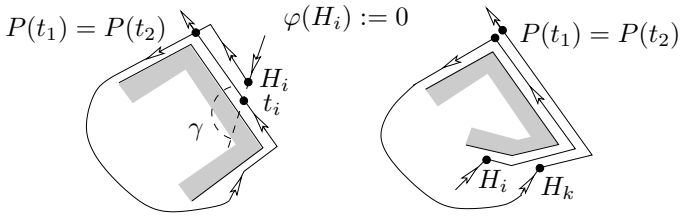

Fig. 5. A counterclockwise turn and a crossing respectively no crossing.

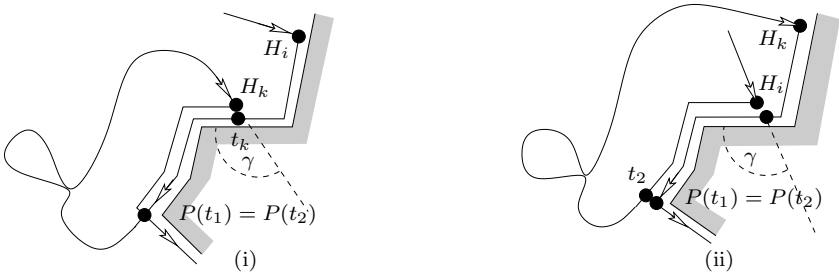

Fig. 6. A clockwise turn and a crossing respectively no crossing.

occurs in $P(t_2)$ and no crossings exist before t_2. The curve C will make either a counterclockwise or a clockwise loop between t_1 and t_2. If this first crossing happens in the free space, the curve will violate the free space condition, so the first crossing could occur only in $\mathcal{C}_{\text{half}}$.

Let us consider the case of a counterclockwise loop, see Fig. 5(i). The curve hits an obstacle at H_i, makes a counterclockwise turn, meets $P(H_i)$ for t_i again and has a crossing for $t_2 > t_i > H_i$. Note, that there is no crossing, if the point $P(H_i)$ is not met between t_1 and t_2, see Fig. 5(ii).

W. l. o. g. we assume $\varphi(H_i) = 0$. The loop may leave the obstacle or not, however, we reach t_i with the heading $\varphi(t_i) + (-\gamma) = 2\pi$. At $P(H_i)$ the robot turns clockwise with angle γ to follow the obstacle boundary, so $-\pi < \gamma < 0$ must hold. Hence $\varphi(t_i)$ is greater than π and the obstacle condition is violated.

Now we look at a clockwise turn. The curve hits an obstacle at H_i, follows the obstacle and 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 Fig. 6(i). The point $P(H_k)$ has to be met before at t_k between H_i and t_1. Otherwise the curve only touches itself and there is no crossing at t_2, see Fig. 6(ii).

Let $\varphi(H_k^+)$ denote the heading immediately after the robot has turned in H_k, thus $\varphi(H_k^+) = \varphi(H_k) + \gamma$. As above we have $-\pi < \gamma < 0$. On the other hand, the curve has made a full clockwise turn between t_k and H_k^+, therefore $\varphi(H_k^+) = \varphi(t_k) - 2\pi$. Finally, the obstacle condition has to be fulfilled, too:

$$\begin{aligned} & \varphi(t_k) - \varphi(H_k) < \pi \\ \Leftrightarrow\ & \varphi(H_k^+) + 2\pi - \varphi(H_k) = \varphi(H_k) + \gamma + 2\pi - \varphi(H_k) < \pi \\ \Leftrightarrow\ & \gamma < -\pi \end{aligned}$$

It follows that the first crossing cannot exist, and—by induction—the curve cannot cross itself. □

Lemma 3. *A curve $C \in \mathcal{K}$ will hit every edge in the environment at most once.*

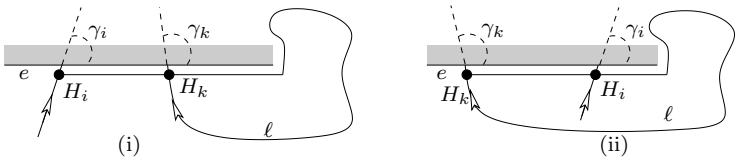

Fig. 7. A curve that hits an edge twice.

Proof. Let us assume, the curve C will hit an edge e more than once: after a first hit at H_i the robot moves around and hits e again at H_k, see Fig. 7. At $P(H_i)$ respectively $P(H_k)$ the robot turns clockwise to follow the edge e, therefore $-\pi < \gamma_i, \gamma_k < 0$. Let $\varphi(H_i^+)$ and $\varphi(H_k^+)$ be defined as in the proof of Lemma 2. W. l. o. g. we assume $\varphi(H_i^+) = 0$. Since the curve in H_i^+ and H_k^+ follows the same edge e, the headings $\varphi(H_i^+)$ and $\varphi(H_k^+)$ must be mod 2π the same, i. e., $\varphi(H_k^+) = 2k\pi, k \in \mathbb{Z}$. For $k \neq 0$, with $\varphi(H_i) = -\gamma_i$ and $\varphi(H_k) = \varphi(H_k^+) - \gamma_k$ it follows that $|\varphi(H_k) - \varphi(H_i)| = |2k\pi - \gamma_k + \gamma_i| > \pi$ and the free space condition would be violated.

Therefore, we can assume $k = 0$ and $\varphi(H_k^+) = 0$. Consider the part of C between the first and the second visit of $P(H_k)$ (in a situation as shown in Fig. 7(i)) respectively the curve between the consecutive visits of $P(H_i)$ (in a situation as shown in Fig. 7(ii)). If this loop, ℓ, has no crossings, then ℓ is a jordan curve and C makes a $\pm 2\pi$ turn in ℓ. Thus $\varphi(H_k^+)$ equals $\pm 2\pi$, in contradiction to our assumption. Hence the curve between the two visits of e must have at least one crossing. But this contradicts to Lemma 2. □

Finally, with Lemma 2 and Lemma 3 we are able to show that the conditions from Definition 1 are sufficient to solve our problem.

Theorem 4. *A robot, whose path follows a curve $C \in \mathcal{K}$, will escape from an unknown maze, if this is possible at all.*

Proof. A curve that meets the conditions from Definition 1 will hit every edge in the environment at most once. Therefore the robot will either escape at the latest after it has visited all edges, or it will be trapped, because there is no escape from the maze. □

4 Applications

Within this section we discuss the practical relevance of Theorem 4. What consequences does it have for the design of a robot that should be able to leave an unknown maze?

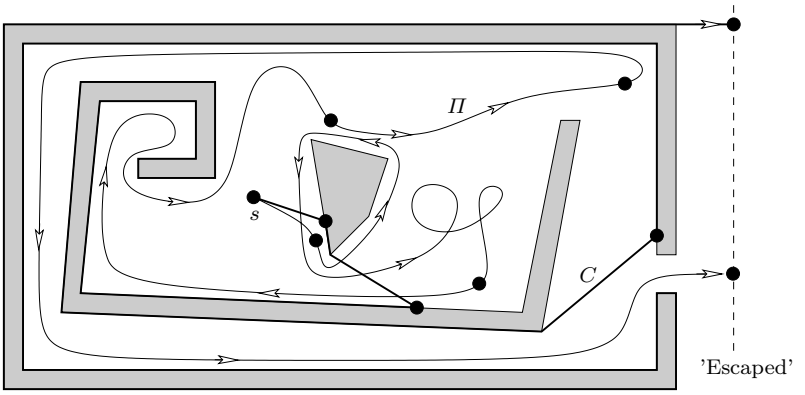

Fig. 8. A curve and a related robot's path.

Still we have to discuss, what it means that the robot follows a curve $C \in \mathcal{K}$. In fact, the deviation from the robot's path, Π, and C can be as large as shown in Fig. 8. The path Π might be produced by an arbitrary strategy, C shows the corresponding curve. The dots on Π represent the hit points in terms of the robot's strategy, i. e., the strategy switches from a move-through-free-space behavior to a go-around-obstacle behavior. Amazingly it is sufficient that the hit points met by C and Π refer to the same edges—including an imaginary 'final edge' which marks the successful escape from the maze. The sequence of hit points is a kind of Ariadne's Thread that leads the robot out of the maze. Between the hit points the robot can do whatever it wants, it can even hit some other edges.

More precisely let $\mathcal{H}_C$ be the sequence of edges hit by the curve C, and $\mathcal{H}_\Pi$ the sequence of edges that are related to a hit point of the robot, then the following holds:

Corollary 5. *A robot escapes from an unkown maze, if there exists a curve $C \in \mathcal{K}$, such that $\mathcal{H}_C$ is a subsequence of $\mathcal{H}_\Pi$.*

4.1 A Simple Compass Is Sufficient

Let us now assume that the robot is equipped with an error afflicted compass. Additionally, the robot should be able to follow walls and walk in the free space until it *detects* hit points. Let the robot measure its turning angles by using compass values. How this is implemented will depend on the specific robot. Note, that just (even exact) compass readings without measuring turning angles is not sufficient, because only directions mod 2π can be detected in this case.

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

On the other side it can be assumed that it is easy to detect 2π or -2π turns along the walls, although again error afflicted within a range of $]-\frac{\pi}{2}, \frac{\pi}{2}[$, due to the compass inaccuracy. Therefore we can assume that the total measured turning angle along the walls is always within a bound of $]-\frac{\pi}{2}, \frac{\pi}{2}[$ away from the exact turning angle. So while the robot follows a wall being at position $t \in C$ we have $\varphi(t) < \frac{\pi}{2}$. Altogether, the obstacle condition is fulfilled.

Corollary 6. *Let an unknown maze be given and let the robot be equipped with an error afflicted compass. Let us assume that it is possible to escape from the maze. If the accuracy of the compass is not worse than $\frac{\pi}{2}$, a simple pledge-like strategy leads to an exit of the unknown maze.*

4.2 Pseudo Rectilinear Scenes

Now we assume that the environment has more or less axis-parallel walls. To be more precise we introduce the notion of (α, β) *pseudo-rectilinear polygons* and specify the robot's ability with respect to α and β.

A *convex (*concave*) corner* of a simple polygon P is a vertex of P with outer angle greater (smaller) than π. A simple polygon P is called *pseudo-rectilinear* iff $|\{p \,|\, p \text{ is a convex corner of } P\}| = |\{p \,|\, p \text{ is a concave corner of } P\}| + 4$.

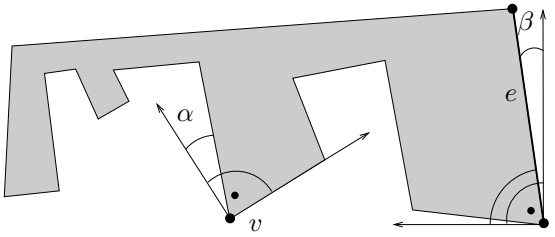

Fig. 9. A pseudo-rectilinear polygon with $\text{anglediv}(v) = \alpha$ and $\text{edgediv}(e) = \beta$.

In Fig. 9 there is an example of a pseudo-rectilinear polygon. There are several ways to measure the quality of rectilinearity of a pseudo-rectilinear polygon. Our aim is that the robot should be able to count the turns along obstacles by considering the number of passed convex and concave vertices. Therefore it is important to detect whether a vertex is convex or not which gives rise to the following definitions.

For a vertex v let $\angle(v)$ denote the outer angle at v. We define the *angle-divergence* at a single corner as follows (see Fig. 9):

$$\text{anglediv}(v) := \begin{cases} \left|\angle(v) - \frac{\pi}{2}\right| & : v \text{ is a concave corner} \\ \left|\angle(v) - \frac{3\pi}{2}\right| & : v \text{ is a convex corner} \end{cases}$$

For a pseudo-rectilinear polygon P we define the angle-divergence $\text{angledivergence}(P)$ of P by

$$\text{angledivergence}(P) := \max_{v \in P} \text{angledivergence}(v)\,.$$

On the other hand the robot should be able to leave an obstacle along a wall in 'horizontal' or 'vertical' direction. For this purpose we have to consider the divergence of the edges from the X-axis respectively Y-axis.

For an edge $e = vw$ let $\text{edgediv}(e)$ denote the smallest angle between e and an axis-parallel line through v or w (see Fig. 9). For a pseudo-rectilinear polygon P we define the *edge-divergence* $\text{edgediv}(P)$ of P by

$$\text{edgediv}(P) := \max_{e \in P} \text{edgediv}(e)\,.$$

A pseudo-rectilinear polygon P is called (α, β) *pseudo-rectilinear* iff $\text{angledivergence}(P) \leq \alpha$ and $\text{edgediv}(P) \leq \beta$ holds.

We assume that the robot measures angles at corners respectively between its heading and a wall within an accuracy of δ, that is, the absolute difference of the actual angle and the measured angle is smaller than δ. In order to guarantee that convex and concave corners are detected correctly it obviously suffices to require that $\alpha + \delta < \frac{\pi}{2}$, see Fig. 10.

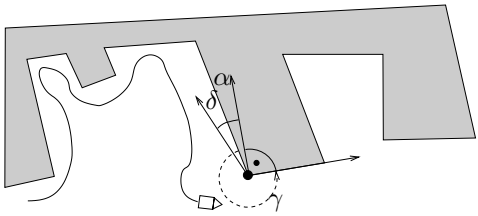

Fig. 10. Detecting concave and convex corners under the presence of errors α and δ. Obviously, $\alpha + \delta$ should not exceed $\frac{\pi}{2}$.

W. l. o. g. we assume that the robot's predefined direction is north, i. e., $\frac{\pi}{2}$ in the mathematical sense. The correctness of the movement in the free space will depend on β and δ, we have to require that the next hit point is appropriate. First, we require that the robot's path between a leave point and a hit point should be monotone with respect to the leaving direction. On the other hand it is important that the robot hits a horizontal edge at the end of its free space movement, thus it will be able to go on counting correctly. Note, that between two hit points the robot can hit and detect vertical walls. In this case we let the robot simply slide along them.

We want to make sure that the robot is able to detect whether an edge is 'horizontal' or 'vertical'. Naturally, if the measured angle between the robot's heading and an edge is smaller than $\frac{3}{4}\pi$ and greater than $\frac{\pi}{4}$, it is reasonable that the robot assumes that the edge is horizontal.

In this sense it suffices to guarantee that the robots heading in the free space lies in the range $\left]\frac{\pi}{4}+\beta+\delta, \frac{3}{4}\pi-\beta-\delta\right[$. For example in Fig. 11 the robot leaves the obstacle at point p and meets edge e with heading ϕ. Since e should be detected as 'vertical' the angle $\gamma-\delta$ should be greater than $\frac{3}{4}\pi$. We have $\gamma=\pi-\left(\pi-\left(\frac{\pi}{2}-\beta\right)-\phi\right)$ and therefore we require $(\gamma-\delta)=\frac{\pi}{2}-(\beta+\delta)+\phi>\frac{3}{4}\pi$ which in turn is equivalent to $\phi>\frac{\pi}{4}+\beta+\delta$. With similar arguments we get the upper bound of the range $\left]\frac{\pi}{4}+\beta+\delta, \frac{3}{4}\pi-\beta-\delta\right[$.

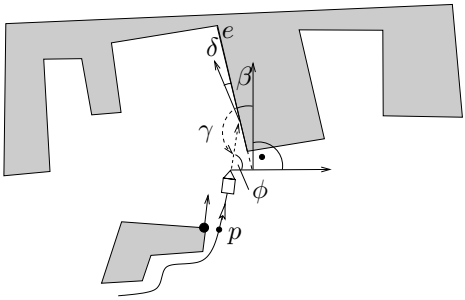

Fig. 11. Detecting 'horizontal' and 'vertical' edges under the presence of errors β and δ.

We assume that the robot counts convex and concave corners rather than counting angles at obstacles. With respect to the leaving direction at a leave point the robot always can start with a heading in the range $\left[\frac{\pi}{2}-\beta, \frac{\pi}{2}+\beta\right]$. Altogether, according to the range above, it suffices to require that the robot should be able to guarantee this heading up to an angle $\frac{\pi}{4}-\delta-2\beta\geq 0$. Obviously, $2\beta\geq\alpha$ holds. Therefore the condition $\alpha+\delta<\frac{\pi}{2}$ is already fulfilled, if $\frac{\pi}{4}-\delta-2\beta\geq 0$ holds.

Corollary 7. *Let an unknown maze with a set of (α,β) pseudo-rectilinear polygons be given. Let the robot be able to measure angles within an accuracy of δ. Let us assume that it is possible to escape from the maze.*

If the robot is able to guarantee its heading in the free space up to an angle $\frac{\pi}{4}-\delta-2\beta$, a simple pledge-like strategy lead to an exit of the unknown maze.

4.3 Exact Free Motion

Now let us assume that the robot is able to move a straight path between obstacles, just its angle counter is inaccurate in some way. Let β_i denote the difference between the real angle and the measured angle at the ith turn and n the number of vertices in the environment.

Lemma 8. *If the robot is able to move a straight path in the free space and ensures that $\left|\sum_{i=k}^{\ell}\beta_i\right|<\pi$ holds for all $k\leq\ell\leq m$, where m denotes the number*

of turns the robot has made so far, then it will be able to escape from an unknown maze.

Proof. The new condition implies that the absolute value of the measuring error that is the difference between the robot's heading and its angle counter, never exceeds π. Now we have to show that our new condition does not violate the conditions from Definition 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 define $\varphi(t_1) = 0$. Since the robot moves 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. So there must be a leave point L_k with $|\varphi(L_k)| = |\varphi(t_2)| \geq \pi$. But the robot leaves an obstacle only, if its angle counter has reached zero, so the absolute value of the measuring error in L_k is at least π.

Second we assume, the obstacle condition would be violated. Then there must be an obstacle P_i 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. Let w. l. o. g. $\varphi(H_k) = 0$, then $\varphi(t_k) \geq \pi$ holds, but the angle counter can't be greater then zero, since the robot leaves the obstacle as soon as the counter becomes zero. So the difference between the robot's heading and its angle counter must be at least π. □

Now, let $\beta_{\max} := \max \beta_i$. With Lemma 3 it is easy to see that the robot visits at most n^2 vertices, and we have the following result:

Corollary 9. *A robot that guarantees* $|\beta_{\max}| < \frac{\pi}{n^2}$ *is able to escape.*

5 Conclusion

We considered the pledge algorithm under errors in sensors and motion and established sufficient requirements for the robot for leaving an unknown maze. Especially, we showed, that a robot equipped with a simple compass will fulfill this task, we gave upper bounds for errors in pseudo rectilinear mazes and for robots with correct motion. Furthermore, in Corollary 5 we gave a framework for proving the correctness of arbitrary strategies for leaving mazes.

Within a simulation environment we have incorporated several sources of errors into an implementation of the pledge algorithm, see [6]. Next, we are going to implement the pledge algorithm for a Pioneer 2 AT Robot and will further observe its behavior on pseudo rectilinear mazes and other error afflicted settings. We are quite sure that on smooth terrains our robot will be able to fulfill the required conditions.

Acknowledgement

We would like to thank Peter Gritzmann for pointing out the problem and Rolf Klein for helpful discussions.

References

1. H. Abelson and A. A. diSessa. *Turtle Geometry*. MIT Press, Cambridge, 1980.
2. M. A. Batalin and G. S. Sukhatme. Efficient exploration without localization. In *Proc. IEEE Internat. Conf. Robot. Autom.*, 2003.
3. P. Berman. On-line searching and navigation. In A. Fiat and G. Woeginger, editors, *Competitive Analysis of Algorithms*. Springer-Verlag, 1998.
4. X. Deng, T. Kameda, and C. Papadimitriou. How to learn an unknown environment I: The rectilinear case. *J. ACM*, 45(2):215–245, 1998.
5. P. Gritzmann. Personal communication via Rolf Klein.
6. 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://web.informatik.uni-bonn.de/I/GeomLab/Polyrobot/.
7. A. Hemmerling. *Labyrinth Problems: Labyrinth-Searching Abilities of Automata*. B. G. Teubner, Leipzig, 1989.
8. F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem. *SIAM J. Comput.*, 31:577–600, 2001.
9. C. Icking, T. Kamphans, R. Klein, and E. Langetepe. On the competitive complexity of navigation tasks. In H. B. Hendrik I. Christensen, Gregroy D. Hager and R. Klein, editors, *Sensor Based Intelligent Robots*, volume 2238 of *LNCS*, pages 245–258, Berlin, 2002. Springer.
10. I. Kamon and E. Rivlin. Sensory-based motion planning with global proofs. In *Proc. 13th IEEE Internat. Conf. on Robotics and Automation*, pages 814–822, 1997.
11. 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.
12. 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.
13. 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.
14. S. Rajko and S. M. La Valle. A pursuit-evasion bug algorithm. In *Proc. IEEE Conf. Robotics Automat.*, 2001.
15. 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.
16. 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.
17. B. Yamauchi, A. Schultz, and W. Adams. Mobile robot exploration and map-building with continuous localization. In *Proc. IEEE Internat. Conf. Robot. Autom.*, 1998.