A New Competitive Strategy for Reaching the Kernel of an Unknown Polygon

Leonidas Palios

University of Ioannina, GR-45110 Ioannina, Greece
palios@cs.uoi.gr

Abstract. We consider the following motion planning problem for a point robot inside a simple polygon P: starting from an arbitrary point s of P, the robot aims at reaching the closest point t of P from where the entire polygon P can be seen; the robot does not have complete knowledge of P but is equipped with a 360-degree vision system that helps it "see" its surrounding space. We are interested in a competitive path planning algorithm, i.e., one that produces a path whose length does not exceed a constant c times the length of the shortest off-line path (in this case, $c \times \text{distance}(s, t)$); the constant c is called the competitive factor. In this paper, we present a new strategy that achieves a competitive factor of ∼3.126, improving over a 4.14-competitive strategy of Icking and Klein and a 3.829-competitive strategy of Lee et al. Our strategy possesses two additional advantages: first, the first point reached from where the entire polygon P is seen is precisely the closest such point to the starting position s, and second, all the points of the path are directly determined in terms of s and of polygon vertices, which implies that an actual robot following the strategy is not expected to deviate much from its course due to numerical error. The competitiveness analysis is based on properties of the class of curves with increasing chords.

Keywords: Motion planning, competitive algorithm, kernel, simple polygon, curve with increasing chords.

1 Introduction

The field of robot motion planning has received considerable attention during the 1980s, but research intensified in the late 1980s when technological advances allowed the autonomous function of robots. This, along with the need for autonomous robots to undertake tasks that may be dangerous for humans (areas polluted by chemicals, space exploration, etc.), led to a number of results pertaining to motion planning problems in unknown or partially known environments (see [4] for a survey). The general motion planning problem for an autonomous robot involves devising a strategy which can help the robot to get to a destination point in an environment which is being "discovered" by means of a vision system (or tactile sensing in some early work). Most motion planning problems are being modeled as two-dimensional problems where the robot is a point moving inside or around polygonal shapes. This is not really restrictive, as real-world

M.M. Halldórsson (Ed.): SWAT 2000, LNCS 1851, pp. 367–382, 2000.

problems can be reduced to this formulation by means of transformations of the geometric boundaries of the objects in the robot's world (Minkowski sum, etc).

Of course, one is interested in having strategies which guarantee that the path traveled by the robot up to its destination is no more than a constant times the length of the shortest path if the environment was completely known. Such strategies are called *competitive* [14], and the ratio of the length of the actual path traveled over the length of the shortest path is called the *competitive factor*. In other words, the competitive strategies guarantee that the effort expended is not far from the optimal. Research results have indicated that finding competitive strategies for different motion planning problems exhibits varying degrees of difficulty (from obtaining constant competitive solutions to proving that finding a competitive solution is P-SPACE complete; see [1], [12]).

In this paper, we consider the problem of planning the path of a robot inside a polygon from any given starting position to a point from where the entire polygon can be seen; in fact, the closest such point to the starting position is sought. The robot is equipped with a 360-degree vision system. This is the problem of reaching the *kernel* of a polygon, and is what a mechanical guard is called to solve in order to position itself so that it watches its territory. The problem has been considered by Icking and Klein [5] who described a strategy to reach the closest point of the kernel at a competitive factor of $\sim$5.48; a tighter analysis by Lee and Chwa [8] showed that the strategy is $\sim$4.14-competitive. Icking and Klein also showed that no competitive factor less than $\sqrt{2}$ can be achieved. A different strategy with a competitive factor of $\sim$3.829 was later described by Lee et al. [9], while López-Ortiz and Schuierer [10] improved the lower bound to $\sim$1.48. López-Ortiz and Schuierer also noted that the competitive factor of [5] is not guaranteed for negative instances (i.e., when the polygon has empty kernel) and described a strategy that is guaranteed to work even in this case at a competitive factor of $\sim$46.35.

Our work contributes a new strategy for reaching the kernel of an unknown polygon P with nonempty kernel which achieves a competitive factor of $\sim$3.126. The path consists of line segments and circular arcs whose total number is linear in the size of P. Our strategy is designed so that the robot walks into the kernel at precisely the point that is closest to the starting position; additionally, it has the advantage that any point of the course is determined by the starting position of the robot and vertices of P, and therefore an actual robot following the strategy is not expected to deviate much from its course due to accumulated numerical errors. The competitiveness analysis is based on properties of the class of curves with increasing chords [13]. Experimental results suggest that the strategy performs better than the theoretical competitive factor. (A similar strategy has been used in [7] for motion planning in a street-polygon.)

The paper is structured as follows. In Section 2 we review the terminology that we use throughout the paper, and in Section 3 we outline our strategy and state some of the properties of the resulting path. In Section 4 we establish the competitive factor of the strategy, and in Section 5 we conclude with final remarks and open questions.

2 Terminology

A *simple polygon* is the region enclosed by a single closed non-self-intersecting polygonal line; thus, a simple polygon does not have "holes" in it. The set of all points p of a simple polygon P such that the line segment that connects p with any other point of P lies entirely in P is called the *kernel* of the polygon. If we define the *inner halfplane* of an edge as the closed halfplane which is defined by the edge and contains all the points of P in a sufficiently small neighborhood of the edge's midpoint, then the kernel of P is equal to the intersection of the inner halfplanes of all the edges of P and is therefore convex.

We will follow the terminology of Icking and Klein [5]; we briefly summarize it in this paragraph. From its starting position s, the robot probably does not see parts of the polygon P in which it stands; if the robot sees all of P, then s belongs to the kernel and the robot need not move. The hidden portions of the polygon are called *caves*. Each cave is adjacent to a reflex vertex of P, whose very existence creates the cave; these reflex vertices are called *constraint vertices* (Figure 1). A cave (associated with a constraint vertex v) is characterized as either *left* if it lies to the left of the directed line $\overrightarrow{sv}$, or *right* otherwise. By extension, we say that a vertex is a *left constraint vertex* if it is a constraint vertex associated with a left cave, and similarly for a *right constraint vertex*. In Figure 1, the vertices v and w are left constraint vertices, and the shaded regions next to them are the associated caves; the vertex u is a right constraint vertex. For each of the constraint vertices v, we define its *inner halfplane* with respect to the current position p as the closed halfplane which is delimited by the line $\overline{pv}$ and does not contain the corresponding cave.

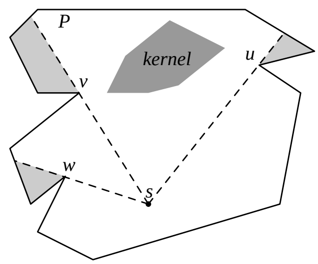

Figure 1

From its starting position, the robot may detect zero or more left caves and zero or more right caves. If the robot sees at least one left cave, the following lemma holds (see [11] for a proof).

Lemma 1. *Suppose that from its starting position s in a simple polygon P the robot detects one or more left caves next to the constraint vertices $l_1, \ldots, l_k$ ($k \geq 1$). Suppose further that no left constraint vertex exists such that the closure of the complement of its inner halfplane contains all the left constraint vertices. Then, the kernel of P is empty.*

A similar lemma holds for the right constraint vertices. Therefore, if the conditions of Lemma 1 hold, we need do nothing, since the polygon has empty kernel. Otherwise, there is a left constraint vertex such that the closure of the complement of its inner halfplane contains all the left constraint vertices and it is unique (if there are more than one vertices collinear with s then we choose the one farthest away from s); we call this vertex *maximal left constraint vertex*. In Figure 1, v is the maximal left constraint vertex. In a similar fashion, we have the *maximal right constraint vertex.* It can be proven that in a polygon with nonempty kernel, the left and right constraint vertices are not "intermixed" and this is why in papers on this problem which assume polygons with nonempty kernel, figures show the left and the right constraint vertices all gathered on the left and on the right of the polygon boundary respectively.

Crucial in the analysis of our strategy is the notion of a *curve with increasing chords*; a curve has increasing chords if $|ad| \geq |bc|$ for any four points a, b, c, d lying on the curve in that order ($|pq|$ denotes the length of the line segment connecting p and q). For a plane curve with increasing chords, Rote proved that

Lemma 2. *[13] The length of a plane curve with increasing chords connecting two points a and b does not exceed $\frac{2\pi}{3}$ times the length of the line segment connecting a and b.*

We close this section with a well known geometric fact and another lemma.

Fact 1. *Consider a circle with diameter ab. Then, the angle $\widehat{apb}$ of the triangle with vertices a, b, and p is less than, equal to, or greater than $\pi/2$ if p lies outside, on the boundary, or inside the circle, respectively.*

Lemma 3. *Let C_1 be a connected non-self-intersecting curve which does not intersect the line segment connecting its endpoints a and b, and C_2 a convex polygonal line with the same endpoints which lies in the region enclosed by C_1 and the line segment ab. Then, the length of C_2 does not exceed the length of C_1.*

Angle Notation: Since three points define two angles (which sum up to 2π), in the following, the notation $\widehat{abc}$ (where a, b, c are three non-collinear points) is meant to indicate the smallest of the two corresponding angles.

3 The Strategy

The basic motivation behind our strategy stems from the study of the simplest case, i.e., a single reflex vertex v whose incident edges are not both visible from the starting position s. Since the robot does not know the direction of the invisible edge e incident upon v, it does not know where the closest point t of the kernel might be. However, in all cases, t belongs to the semicircle with diameter sv, assuming that the semicircle lies in the polygon P (Figure 2). So, it seems a good idea to follow this semicircle.

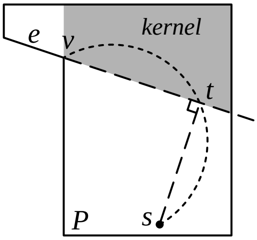

Figure 2

Our strategy is based on this idea. Thus, the path of the robot consists of circular arcs and line segments; each circular arc belongs to a circle with diameter sp, where s is the starting position and p is a constraint vertex. This strategy makes the robot reach the kernel at its closest point to s.[1]

[1] It must be noted that this strategy is not optimal for the simple case of a single reflex vertex; it yields a worst-case competitive factor of $\pi/2 \simeq 1.57$. See [6], for a proof that the optimal competitive factor is $\sim$1.212, and for a strategy achieving it.

We first consider the one-sided case, where there are only left or only right caves; our strategy for the general case consists of applying the one-sided case strategy twice, first for the left caves until we see them all, and then for the right caves (if needed).

3.1 The One-Sided Case

Without loss of generality, we consider the case where there are only left caves (the case where we have only right caves is similar). Until the robot sees all the left caves, there exist left constraint vertices and among them a maximal left constraint vertex, which may change as the robot moves. Initially, the robot finds the maximal left constraint vertex v_0 as seen from the starting position s and starts following the semicircle with diameter sv_0. The two fundamental cases that characterize the robot's path are:

1. *A new maximal constraint vertex u is discovered.* Then, the robot will start following the semicircle with diameter su (Figure 3: point a). Interestingly, the current location of the robot belongs to both semicircles.
2. *The cave next to the currently maximal constraint vertex u becomes visible.* This implies that the second edge e incident upon u has become visible as well. Then, the robot at its current position, say, b, finds the new maximal constraint vertex. If no such vertex exists, then the entire polygon is visible and the robot has achieved its goal. If such a vertex exists —let it be v— and v is a constraint vertex just seen for the first time (for example, if v is the other endpoint of e), then we execute the previous case. The remaining possibility is if v is a constraint vertex that has already been seen, in which case the robot walks along the line segment bu trying to reach (if possible) the semicircle with diameter sv (Figure 3: points b and c).
   Note that it may be the case that the robot has to reach the currently maximal constraint vertex u in order to see the cave next to u. (This can only happen if u is the maximal left constraint vertex v_0 seen from s.) In this case, if there exists a new maximal constraint vertex w, w has to be a constraint vertex just discovered, for otherwise the polygon has empty kernel. The robot at u lies on or outside the semicircle with diameter sw, and it will try to walk along the line su away from s in an attempt to see the cave next to w.

The above two cases do not take into account the fact that the robot may take advantage of what it has seen. Clearly, the kernel of the polygon P is a subset of the inner halfplanes of the edges of P and of the inner halfplanes of the constraint vertices. Since the robot seeks to locate the kernel, it seems reasonable that it should not leave the inner halfplane of any of the polygon edges or constraint vertices which it sees or has seen. To be able to do that, the robot maintains the *free polygon* which is the subset of P in which the robot may walk. Initially, the free polygon is the intersection of the inner halfplanes of the visible edges and the visible constraint vertices from the starting point s. As

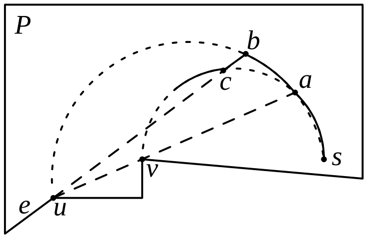

Figure 3

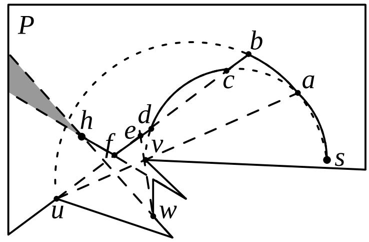

Figure 4

a new edge or a new constraint vertex becomes visible, the robot updates its free polygon by intersecting it with the corresponding inner halfplane. By requiring that the robot maintains the free polygon up to date and remains in it, we ensure that the portion of the polygon seen by the robot never decreases; at the same time, the free polygon keeps shrinking and when the robot reaches the kernel, the free polygon is precisely the kernel of P. Additionally, a left (resp., right) constraint vertex will remain so until both its incident edges become visible; it will not turn into a right (resp., left) constraint vertex, which might happen if the robot zig-zagged inside P.

At any time during its trip, the robot lies at a point, say, p, on the boundary of the current free polygon and it can only walk in the free polygon, that is, in the wedge delimited by the lines supporting the free polygon edges that are incident upon p. Since the free polygon is defined as the intersection of halfplanes, the opening angle of this wedge does not exceed π. Because the line supporting the edge to the left of p (with respect to the robot's motion towards the interior of the free polygon) bounds the current free polygon from the left, we call it a *left-bounding* line; similarly, the line supporting the edge to the right of p is a *right-bounding* line.

The following two cases complete the path planning strategy of the robot.

3. *The robot's intended course leads or lies outside the free polygon.* Then the robot walks along the boundary of the free polygon as close to the intended course as possible. In terms of left- and right-bounding lines, the robot walks along the left-bounding (right-bounding, respectively) line of the current free polygon if and only if the intended course leads to the left (right, respectively) of the free polygon.
4. *An edge that was not visible becomes visible.* Then, the robot updates the free polygon by intersecting it with the inner halfplane of that edge. Note that this case has to be executed in case 2.

An example is shown in Figure 4. It is important to note that the ending point h lies on the line supporting the edge which was seen last. Another important observation pertains to the way the value of the angle $\widehat{psv_0}$ behaves, where p denotes the current position of the robot on its way from s to h, and v_0 is the maximal left constraint vertex as observed from s. In the most general case, the following behavior of the angle $\widehat{psv_0}$ is exhibited: it is initially $\pi/2$, then it decreases, potentially reaching 0 but not decreasing below 0 (sub-path from s to f in Figure 4), and then it increases (sub-path from f to h). (Note that the robot may walk along sv_0.) However, two special cases may arise: first, the value of $\widehat{psv_0}$ is always decreasing from s to h (for example, consider the case that the caves of both v and w of Figure 4 were visible at f), and second, the value of $\widehat{psv_0}$ is always non-decreasing. The latter case may occur if, due to clipping, the left-bounding line of the free polygon is farther to the right from the semicircle with diameter sv_0; in this case, the robot will not follow any of the semicircles defined by s and the maximal left constraint vertices.

Lemma 4. *Suppose that the angle $\widehat{psv_0}$ decreases and then increases, reaching its minimum value when the robot is at the point x. Then,*
(i) x is either on or outside the corresponding semicircle,
(ii) the part of the robot's path past x lies outside the semicircle defined by s and the currently maximal constraint vertex.

3.2 The General Case

Our strategy for the general case consists of applying the one-sided strategy twice, first for the left caves and then for the right caves. Suppose that the robot is at point h, when it finally sees all the left caves. Then, the robot finds the maximal right constraint vertex u and updates its free polygon by intersecting it with the inner halfplane of u at h. The robot's intention is to walk along the semicircle C_{su} with diameter su; however, it has to reach C_{su} first. To do this, the robot tries to walk along the line hu towards the semicircle; by walking in this direction, the robot does neither gain nor lose visibility of the cave next to u. Of course, this course is subject to clipping about the free polygon; so, if the path along hu towards C_{su} leads outside the free polygon, the robot follows left-bounding lines if h is inside C_{su} and right-bounding lines if h is outside C_{su}.

The final path consists of two sub-paths, one from s to h and the other from h to the final point t, each similar to the path shown in Figure 4. That is, each one of them consists of a number of clipped circular arcs and line segments (cases 1 and 2 of Section 3.1), potentially followed by one or more line segments that result from clipping whenever the corresponding semicircles fall outside the free polygon (Figures 5-7 show examples of paths). Our observation in Section 3.1 about the behavior of the values of the angle $\widehat{psv_0}$ (where p is the robot's current position and v_0 is the maximal left constraint vertex as observed from s) is extended and implies that, in the most general case, $\widehat{psv_0}$ is initially $\pi/2$, then decreases, potentially reaching 0 but not decreasing below 0, then it starts increasing assuming values up to $\widehat{u_0sv_0}$ (where u_0 is the maximal right constraint vertex as observed from s), and then it may start decreasing again up to 0.

3.3 Simulating the Strategy

The obvious way to simulate a motion strategy involves starting at the predetermined starting position and executing small steps applying the rules of the strategy. This method has the obvious disadvantage that a good approximation of the robot's path requires a large number of steps which may lead to increased execution time and large errors resulting from accumulated numerical errors at each step.

A second approach is to split the given polygon P into regions in each of which the robot follows the same curve. Clearly, we will have to split P about the lines supporting the polygon edges incident upon reflex vertices. Moreover, we need to split P about lines that connect pairs of (left or right) constraint vertices that consecutively become maximal. To do that, we find the tree of shortest paths inside P from s to all the reflex vertices and we split P about the lines supporting the edges of this tree as well. Then, the robot can traverse any of the resulting regions in one computational step; the only computation in each region involves finding the points of intersection of the path with the region boundary. This method involves fewer steps compared to the previous one but it requires computing the partition of the polygon about the above mentioned lines; the total number of these lines is linear in the number n of polygon vertices. Building the partition requires $O(n^2)$ space and it can be done incrementally in $O(n^2)$ time in a fashion similar to the incremental construction of an arrangement of lines; see [3] and [2]. The free polygon is maintained by turning on or off a bit associated with each region.

3.4 Path Properties

It is interesting to observe that every point of the robot's path belongs either to a semicircle defined by the starting point and a vertex of the polygon P (a maximal constraint vertex) or to the line supporting an edge of P. This guarantees that an actual robot following our strategy is not expected to deviate from the intended course, as opposed to other strategies where this is possible because the motion of the robot is dependent on the current position. For example, in Icking and Klein's strategy, the robot follows the bisector of an angle with apex the current position; but then, due to accumulated numerical error, the robot may deviate substantially from the expected course.

Additionally, the following lemmata establish two important properties of the robot's path (proofs can be found in [11]).

Lemma 5. *The path resulting from the application of the above described strategy reaches the kernel of the polygon at the kernel's point that is closest to the starting point s.*

Lemma 6. *The path that the robot follows in accordance with our strategy consists of $O(n)$ line segments or circular arcs, where n is the number of vertices of the polygon P.*

4 Competitiveness Analysis

In order to compute the competitive factor of our strategy, we need to compute the worst-case ratio of the length of the path resulting from the application of our strategy over the length of the line segment connecting the starting point s to the ending point t. Obviously, the worst case scenario involves double application of the one-sided case. Our analysis relies on computing the competitive factor of an "augmented" path (we ignore (most of) the clipping) whose length is no less than the length of the actual path traveled.

Before we describe the "augmentation" procedure, we review the important stops in the robot's path and define the l-path and r-path which will be used to augment the path. The robot first applies the one-sided strategy trying to see all the left caves; let h be the final point during this phase, that is, the point from where all the left caves are visible. Then, the robot applies the one-sided strategy again, for the right caves this time. As mentioned in Section 3.2, the angle $\widehat{psv_0}$ (defined by the current position p of the robot, the starting position s, and the maximal left constraint vertex v_0 observed from s) decreases, then it may increase and finally it may decrease again; let x and y be the turning points where these changes of monotonicity occur (if the robot walks along the line sx or sy, we let x and y be the closest such points to s). Note that x may coincide with h or may be before or after h along the robot's path; y may coincide with t, although this is not true in the most general case. Moreover, as mentioned earlier, the point h lies on the line supporting the polygon edge that just became visible at h; let l_h be that line. Then, l_h is a right-bounding line of the free polygon at h. Similarly, the ending point t lies on the line l_t supporting the edge that became visible last, and l_t is a left-bounding line of the free polygon at t.

We define the *l-path* as the path that the robot would follow if it only applied cases 1 and 2 of Section 3.1 from its starting position s until it either saw all the left caves or reached the line sx, whichever came first; in the former case, we extend the l-path by adding a line segment along the left-bounding line of the free polygon from the l-path's final point to the point of intersection with sx. Because clipping is ignored, this left-bounding line supports a polygon edge next to a maximal left constraint vertex; this edge is not necessarily the edge that became visible last. As a summary, the l-path consists of a sequence of circular arcs (arcs sa, bc of Figure 3) occasionally separated by a line segment along a line supporting an initially invisible polygon edge (segment bc of Figure 3). We define the *r-path* similarly: this is the path that the robot would follow if it only applied cases 1 and 2 of Section 3.1 starting from s until it either saw all the right caves or reached the line sy; again, if the robot has seen all the right caves before it reached the line sy, we extend the r-path accordingly. We finally define the *l-region* as the closed region bounded by the l-path and the line sx; similarly, the *r-region* is the closed region bounded by the r-path and the line sy. We note that:

Observation 1. *The point h from which all the left caves are finally visible does not belong to the interior of the l-region. Similarly, the final point t from which the entire polygon is visible does not belong to the interior of the r-region.*

The robot tries to follow the l-path and the r-path if possible, or otherwise stay as close to them as possible. On its course from the starting point s to h (the case is similar for the part from h to the ending point t), it follows (parts of) the l-path, may move outside the l-region due to clipping about a left-bounding line (when the l-path leads farther left than the left boundary of the free polygon), or may move inside the l-region due to clipping about a right-bounding line (when the l-path leads farther right than the right boundary of the free polygon). In general, the robot may move in and out of the l-region several times; after it has moved in, it may walk along several different right-bounding lines (tracing a convex curve inside the l-region), whereas after it has moved out, it may follow several different left-bounding lines (tracing a concave curve outside the l-region). It is important to observe:

Observation 2. *The robot never follows a left-bounding line right after a right-bounding line (or vice versa) except at the point h where it sees all the left caves.*

The observation follows from the fact that the robot tries to stay as close to the corresponding semicircle as it can and if this is farther left (right, respectively) than the left (right, respectively) boundary of the free polygon, the robot will keep following the left (right, respectively) boundary of the free polygon until it reaches it, if ever.

4.1 Augmenting the Robot's Path

Now we are ready to see how the actual robot's path is being augmented; we will also define the points x' and y' which will be crucial in partitioning the augmented path into curves with increasing chords. We concentrate on the most general case in which $x \neq s$ (i.e., the angle $\widehat{psv_0}$ starts by decreasing) and $x \neq t$; the special cases where $x = s$ or $x = t$ yield smaller competitive factors (see [11]). Note that y may or may not coincide with t.

1. *the part of the robot's path from s to x:* We recall that x may be either on the l-path or outside the l-region; in the latter case and if additionally h coincides with or is reached after x, then the robot has been walking along left-bounding lines from the last point of its course on the l-path up to x. Recall also that h is either on the l-path or outside the l-region (Observation 1); if it is outside the l-region, then again the robot has been walking along left-bounding lines. In all cases where the robot walks along left-bounding lines after it leaves the l-region (no matter whether h is reached before or after x), the sub-path from s to x is augmented by considering the entire l-path, followed by a line segment from the final point of the l-path to x along sx (Figure 5); this includes as a special case the case where x belongs to the l-path. It remains to consider the cases where the robot walks along right-bounding lines. There are two cases to consider: first, h belongs to the l-path, x is reached after h, and the robot walks along a right-bounding line past h towards x, and second, h is outside the l-region, x is reached after h, and the robot walks along a right-bounding line past h towards x; both cases imply $x = t$ and yield smaller competitive factors (see [11]).

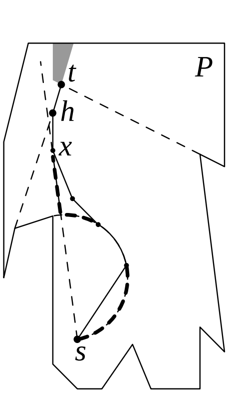

Figure 5

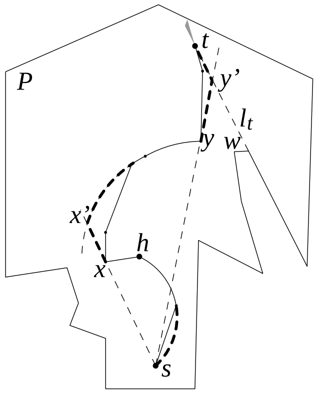

Figure 6

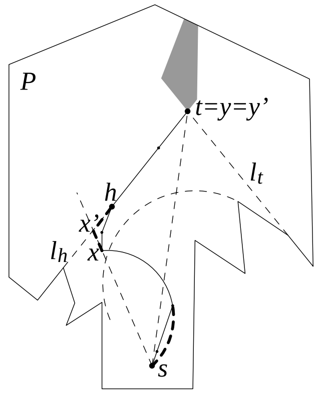

Figure 7

2. *the part of the robot's path from x to y:* We distinguish two cases depending on whether the robot walks along left- or right-bounding lines past x.
(i) the robot walks along a left-bounding line past x. If the point h is before x or coincides with x, then x must belong to the r-region for the robot to follow a left-bounding line past x. We let x' be the point of intersection of sx with the r-path, and we augment the sub-path from x to y by considering the line segment xx', followed by the r-path up to its intersection with the line sy, followed by the line segment from that point to y (Figure 6). If the point h is after x, then past h the robot may walk along a left- or a right-bounding line depending on whether h belongs to the r-region or not. Let q be the point of intersection of the lines sx and l_h. If h is outside the r-region, or if h belongs to the r-region but q does not, we set $x' = q$ and we augment the path by considering the line segment xx' (along sx), followed by a line segment along l_h from x' to the point of intersection with the r-path, followed by a line segment from that point to y along sy (Figure 7). If both q and h belong to the r-region, then we let x' be the point of intersection of the line sx with the r-path, and the sub-path from x to y is augmented by considering the line segment xx' (along sx), followed by the r-path from x' to its final point on the line sy, followed by the line segment from that point to y (along sy); the situation is similar to the one depicted in Figure 6.
(ii) the robot walks along a right-bounding line past x. Then, h cannot be before x, for, if h were reached before x, the robot must have been walking along right-bounding lines from h to x; this implies that $x = t$, a contradiction to the continuation of the path past x. Moreover, h cannot be after x either; if h were reached after x, then h would be outside the l-region and the robot would be walking along left-bounding lines from x to h. Therefore, $h = x$, and we set $x' = h$. Additionally, h lies outside the r-region (otherwise, the robot would not be following a right-bounding line past x). Let q be the point of intersection of l_h with the r-path (if l_h intersects a line segment of the r-path, then q is the point of intersection of l_h with the immediately

following semicircle); if the line l_h does not intersect the r-path, we let q be the point of intersection of l_h and sy. Then, the sub-path from x to y is augmented by considering the line segment xq along l_h, potentially followed by the r-path from q to its intersection with the line sy (if q does not belong to the line sy), followed by the line segment from that point to y.

3. *the part of the robot's path from y to the final point t:* If $y = t$, then we set $y' = y = t$. If $y \neq t$, then the path past y lies outside the corresponding semicircles (Lemma 4) and the robot on its way to t walks along right-bounding lines only. So, this part of the actual path is augmented by considering the polygonal line formed by the segments yy' and $y't$, where y' is the point of intersection of the lines sy and l_t (Figure 6).

It is important to observe that the augmented path does not cross itself. Moreover, the augmented path proceeds along or to the left of the left-bounding lines that the robot follows, and along or to the right of the right-bounding lines, thus enclosing the actual robot's path. Therefore, we have:

Observation 3. *The path traveled by the robot and the augmented path have the same endpoints.*

Observation 4. *The path traveled by the robot can be produced by clipping the augmented path about the edges of a (shrinking) convex polygon.*

4.2 The Competitive Factor

With respect to the points x' and y', the augmented path can be seen as the concatenation of three sub-paths, one from s to x', one from x' to y', and one from y' to the final point t. The sub-path from s to x' consists of circular arcs occasionally separated by a line segment along a line supporting an initially invisible polygon edge (cases 1 and 2 of Section 3.1), potentially ending with a line segment along the line sx. The sub-path from x' to y' consists mainly of arcs and line segments (in accordance with cases 1 and 2 of Section 3.1) as well, but may begin with a line segment along l_h, and may end with a line segment along the line sy; the sub-path may degenerate into a two-segment polygonal line, one along l_h and the other along sy. Finally, the sub-path from y' to t is simply a line segment. See Figures 6-7. More importantly, the following lemmata hold.

Lemma 7. *The (counterclockwise) angle $\widehat{sx'y'}$ is at least equal to $\pi/2$.*

Proof. The definition of the point x' in the case 2 of the preceding section suggests that we need to consider two cases. First, suppose x' is the point of intersection of sx with the r-path (if x is inside or on the boundary of the r-region). Then, x' lies on the semicircle of the currently maximal right constraint vertex (case 1 of Section 3.1), or on the line supporting an edge incident upon a right-constraint vertex which was initially invisible and became visible (case 2 of Section 3.1); in the latter case, x' lies inside the semicircle associated with the right constraint vertex. In either case, if w is the right constraint vertex, then the angle $\widehat{sx'w}$ is at least equal to $\pi/2$. Moreover, the point y and (a fortiori) the point y' lie on or to the left of the directed line $\overrightarrow{x'w}$ (see Figure 6). Therefore, $\widehat{sx'y'} \geq \widehat{sx'w}$, and the lemma follows.

Suppose now that x' is the point of intersection of sx with l_h: this case occurs if h is reached after x, or $x' = h$ and h lies outside the r-region. In either case, x' coincides with or is farther away from s than x; since x is on the boundary or outside the l-region (Lemma 4), x' lies on or outside the semicircle of the last maximal left-constraint vertex, say, v. Then, $\widehat{sx'v} \leq \pi/2$ (Fact 1). The lemma follows from the fact that y and (a fortiori) y' belong to the inner halfplane of l_h and thus the angle $\widehat{sx'y'}$ is at least equal to $\pi - \widehat{sx'v}$ (see Figure 7). ∎

Similarly,

Lemma 8. *If $y \neq t$, the (clockwise) angle $\widehat{sy't}$ is at least equal to $\pi/2$.*

Lemma 9. *The sub-path of the augmented path from s to x' is a curve with increasing chords.*

Proof. For any point p (other than s), we define the quadrants A_p, B_p, C_p and D_p at p as the four closed quadrants determined by the line sp and its perpendicular at p: the quadrant A_p is the quadrant that contains s and lies to the right of the directed line $\overrightarrow{sp}$, while the other quadrants B_p, C_p and D_p follow quadrant A_p in counterclockwise order around p. We first prove that for any point p of this sub-path, the part of the augmented path from s to p belongs to the closed quadrant A_p of p, while the part of the path from p to x' belongs to the closed quadrant C_p of p. One needs to consider the different cases for p: on a circular arc, at the intersection of two arcs, at the intersection of an arc and a line segment, on a line segment. This follows from the fact that for any point q of a semicircle with diameter ab, the angle $\widehat{aqb}$ is equal to $\pi/2$ (see Fact 1). (Figure 8 gives some examples for illustration purposes; the crosses indicate the lines delimiting the quadrants.) Next, we consider 4 points a, b, c and d in that order along the augmented path. We draw the corresponding quadrants for the points b and c and draw the two lines l_b and l_c perpendicular to bc that pass by b and c respectively (Figure 9). Since c belongs to the quadrant C_b of b, l_b lies in the closure of the wedge defined by the quadrants B_b and D_b of b. Similarly, since b belongs to the quadrant A_c of c, l_c lies in the closure of the wedge defined by the quadrants B_c and D_c of c. Moreover, the point a lies in the quadrant A_b of b, that is, to the left of l_b. Similarly, the point d lies in the quadrant C_c of c, that is, to the right of l_c. Therefore, the length of ad is no less than the perpendicular distance of l_b and l_c, which by construction is equal to bc. ∎

In a similar fashion, although with a little more effort because the sub-path of the augmented path between x' and y' may begin with a line segment, we can prove:

Lemma 10. *The sub-path of the augmented path from x' to y' is a curve with increasing chords.*

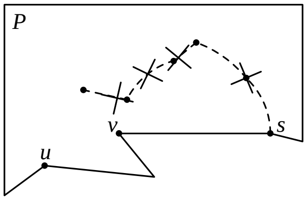

Figure 8

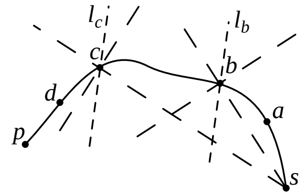

Figure 9

From the above, we conclude

Theorem 1. *Our strategy has a competitive factor of* $\sqrt{2(2\pi/3)^2+1} \simeq 3.126$.

Proof. Clearly, the length of the actual path traveled by the robot is no more than the length of the augmented path, as clipping with convex polygonal lines or curves leads to reduced path length (Observation 4 and Lemma 3). So, an upper bound on the ratio of the length of the augmented path over the length of the line segment st readily implies an upper bound on the competitive factor that we seek. Figure 10 shows the skeleton of the augmented path in the worst case; the angles $\alpha = \widehat{sx'y'}$ and $\beta = \widehat{sy't}$ are at least equal to $\pi/2$ (Lemmata 7 and 8). Let us denote by $|\widetilde{pq}|$ the length of the path from p to q as opposed to $|pq|$ which denotes the length of the line segment pq. Then the competitive factor r is

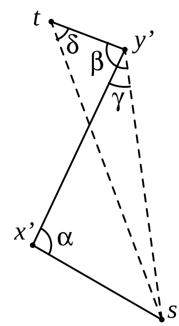

Figure 10

$$r \;=\; \frac{|\widetilde{sx'}| + |\widetilde{x'y'}| + |y't|}{|st|} \;\leq\; \frac{\frac{2\pi}{3}|sx'| + \frac{2\pi}{3}|x'y'| + |y't|}{|st|},$$

since the augmented sub-paths from s to x' and from x' to y' are curves with increasing chords (Lemmata 9 and 10) and therefore their lengths are not more than $2\pi/3$ times the lengths of the line segments sx' and $x'y'$ respectively (Lemma 2). If we apply the law of sines in the triangles $sx'y'$ and $sy't$, factor out the length $|sy'|$, and maximize using partial derivatives, we find

$$r \;\leq\; \frac{\frac{2\pi}{3}\,\frac{\sin\gamma + \sin(\alpha+\gamma)}{\sin\alpha} + \frac{\sin(\beta+\delta)}{\sin\delta}}{\frac{\sin\beta}{\sin\delta}} \;\leq\; \frac{\frac{2\pi}{3}\,\sqrt{2} + \frac{\sin(\beta+\delta)}{\sin\delta}}{\frac{\sin\beta}{\sin\delta}} \;\leq\; \sqrt{2(2\pi/3)^2+1}.$$

where $\pi/2 \leq \alpha < \pi$, $\pi/2 \leq \beta < \pi$, $0 < \gamma < \pi - \alpha$ and $0 < \delta < \pi - \beta$: the term $\frac{\sin\gamma+\sin(\alpha+\gamma)}{\sin\alpha}$ is decreasing as α increases and is thus maximized for $\alpha = \pi/2$ and $\gamma = \pi/4$; similarly, the overall fraction is maximized for $\beta = \pi/2$. ∎

5 Concluding Remarks – Open Problems

We presented a strategy which enables a point robot to reach the point t of the kernel that is closest to the starting point s, and guarantees that the length of the path traveled is not longer than 3.126 times the length of the line segment st (that is, 3.126 times the shortest possible off-line path). Our strategy has the interesting feature that the robot reaches the kernel at precisely the closest point t. We note that the above competitive factor cannot be guaranteed when the polygon has empty kernel (in such cases, the competitive factor is defined as the ratio of the length of the path that a strategy imposes over the length of the shortest path which establishes that the kernel is empty), and this holds for all strategies where a point of the polygon seen by the robot never ceases to be in the robot's visible region thereafter (enforced by means of the free polygon in this work, and by means of the gaining and keeping wedges in [5] and [9]).

Experimental results seem to suggest that the actual competitive factor is smaller than the theoretical competitive factor of 3.126. If true, it would be interesting to come up with tighter theoretical bounds on the competitive factor of our strategy. Of course, the ultimate open question is to invent strategies with smaller competitive factors which will close the gap between the current upper bound of $\sim$3.126 and the lower bound of $\sim$1.48. To this effect, perhaps ideas like the ones in [6] may be of help.

Finally, better competitive solutions are needed for other motion planning problems in unknown environments. López-Ortiz and Schuierer [10] have addressed two interesting problems in this class: finding out whether a given polygon is star-shaped (i.e., it has non-empty kernel), and locating a target (to be recognized when seen) in a polygon with non-empty kernel. The currently best competitive factor for the first problem is 46.35. The currently best competitive factor for the second problem is 12.72 and is coupled with a lower bound of 9.

Acknowledgements

I would like to thank Vassilios Karaiskos whose program (implementing our strategy) helped to produce several of the included figures.

References

1. A. Blum, P. Raghavan, and B. Schieber, "Navigating in unfamiliar Geometric Terrain," *Proc. 23th ACM Symposium on Theory of Computing* (1991), 494–504.
2. H. Edelsbrunner and L.J. Guibas, "Topologically sweeping an arrangement," *Journal of Computer and Systems Science* 38 (1989), 165–194. Corrigendum in 42 (1991), 249–251.
3. H. Edelsbrunner, J. O'Rourke, and R. Seidel, "Constructing arrangements of lines and hyperplanes with applications," *SIAM Journal on Computing* 15(2) (1986), 341–363.
4. Y.K. Hwang and N. Ahuja, "Gross Motion Planning – A Survey," *ACM Computing Surveys*, Vol. 24, No. 3 (1992), 219–291.
5. C. Icking and R. Klein, "Searching for the Kernel of a Polygon — A Competitive Strategy," *Proc. 11th ACM Symp. on Computational Geometry* (1995), 258–266.
6. C. Icking, R. Klein, and L. Ma, "An Optimal Competitive Strategy for Looking around a Corner," *Proc. 5th Canadian Conference on Computational Geometry* (1993), 443–448.

7. C. Icking, A. Lopez-Ortiz, S. Schuierer, and I. Semrau, "Going Home through an Unknown Street," *Technical Report 228*, Dept of Computer Science, FernUniversität Hagen, Germany, 1998.
8. J.-H. Lee and K.-Y. Chwa, "Tight Analysis of a Self-Approaching Strategy for the online Kernel-Search Problem" *Information Processing Letters* 69 (1999), 1–52.
9. J.-H. Lee, C.-S. Shin, J.-H. Kim, S.Y. Shin, and K.-Y. Chwa, "New Competitive Strategies for Searching in Unknown Star-Shaped Polygons," *Proc. 13th ACM Symposium on Computational Geometry* (1997), 427–429.
10. A. López-Ortiz and S. Schuierer, "Position-independent near optimal Searching and on-line Recognition in Star Polygons," *Proc. 5th International Workshop on Algorithms and Data Structures – WADS* (1997), 284–296.
11. L. Palios, "A New Competitive Strategy for Reaching the Kernel of an Unknown Polygon," *Technical Report* (1999), Dept. of Computer Science, Univ. of Ioannina.
12. C. Papadimitriou and M. Yannakakis, "Shortest paths without a map," *Theoretical Computer Science* 84 (1991), 127–150.
13. G. Rote, "Curves with increasing chords," *Mathematical Proceedings of the Cambridge Philosophical Society* 115 (1994), 1–12.
14. D. Sleator and R.E. Tarjan, "Amortized efficiency of list update and paging rules," *Communications of the ACM* 28 (1985), 202–208.