Optimal Strategies to Track and Capture a Predictable Target

Alon Efrat Héctor H. González-Baños† Stephen G. Kobourov Lingeshwaran Palaniappan

Department of Computer Science
University of Arizona, Tucson, AZ 85721
e-mail: {alon, kobourov, lingesh}@cs.arizona.edu

† Honda Research Institute USA, Inc.
Mountain View, California 94041
e-mail: hhg@honda-ri.com

***Abstract*— We present an $O(n \log^{1+\varepsilon} n)$-time algorithm for computing the optimal robot motion that maintains line-of-sight visibility between a target moving inside a polygon with n vertices which may contain holes. The motion is optimal for the tracking robot (the observer) in the sense that the target either remains visible for the longest possible time, or it is captured by the observer in the minimum time when feasible. Thus, the algorithm maximizes the minimum time-to-escape. Our algorithm assumes that the target moves along a known path. Thus, it is an off-line algorithm. Our theoretical results for the algorithm's runtime assume that the target is moving along a shortest path from its source to its destination. This assumption, however is not required to prove the optimality of the computed solution, hence the algorithm remains correct for the general case.**

I. Introduction

In this paper, we consider the problem of keeping a target visible to a mobile observer for the longest possible time. Variations of this tracking problem arise in different applications such as visual servoing [1], [2], computer assisted surgery [3] and visibility-based planning of sensor control strategies [4].

A planning problem for maintaining line-of-sight visibility between two agents is considered in [5]. A dynamic programming approach generates the motions for a mobile robot (the observer) that tracks a moving target. If the target moves predictably, an optimal tracking motion can be computed off-line. If the target's motion is unpredictable, on-line strategies are used instead [6], [7]. Although on-line techniques are more desirable in a robotics context, off-line strategies remain useful in applications involving graphical simulations, supervision of industrial robots (which often follow known paths), and surveillance of vehicles constrained to move in road-maps.

Most off-line algorithms reported in the literature are not computationally efficient, requiring several seconds to compute even simple scenarios. In [8], a problem similar to the off-line tracking problem is modeled in the configuration-time space relative to the target. Their work takes practical considerations for on-line use into account and reduces planning time to a few seconds. But like [5], the work in [8] discretizes the space in order to approximate the optimal path. This process is costly, although heuristic techniques can speed the computation.

A related problem is sensor placement, previously addressed in [9] and [10] for visual tracking and exploration. Several works exists in regards to this problem. For example, the work in [11] defines a measure of motion observability based on the relationship between differential changes in the robot position and the corresponding differential changes in the observed visual features.

In this paper, we consider the problem of maintaining visibility inside a polygonal region P which may contain holes. The observer loses the target as soon as the line of sight between them is broken. The observer is modeled as a holonomic robot with a maximal speed of v_{obs}, fitted with an omnidirectional sensor of unlimited range. Target detection and robot localization are assumed to be perfect. The target is initially in view and moves along a known path until a time t_{stop}.

We present an $O(n \log^{1+\varepsilon} n)$ algorithm for finding the optimal path π^* given an initial observer position M_0. Here $\varepsilon > 0$ is an arbitrarily small constant, and n is the number of vertices of P. The path is optimal in the sense that it maximizes the time t_{esc} when the observer first loses the target. If the target is never lost, then π^* minimizes the distance between the observer and the target at every time $t < t_{stop}$. If this distance ever becomes 0 the target is said to be *captured*.

Although it is not required by our algorithm, if the target follows the shortest path to its destination the presentation becomes simpler and it is easier to obtain a bound on the running time. However, the algorithm computes an optimal path even when this assumption does not hold.

II. Problem Formulation

The 2-D target-tracking problem consists in computing the motion of an observer such that a target moving inside a planar workspace remains in view. We follow a similar notation to the one used in [6], which in turn follows from the standard notation used in motion planning [12].

The observer and the target move in a bounded Euclidean subspace $\mathcal{W} \subset \Re^2$ (the workspace). The free configuration spaces for the observer and the target are denoted by $\mathcal{C}^o$ and $\mathcal{C}^t$, respectively. The observer is assumed to be a point, therefore $\mathcal{C}^o = \mathcal{W}$. The configuration space of the target is also assumed to be two-dimensional, but the target is forbidden from moving arbitrarily close to an obstacle. In order to simplify our discussion, we represent the target as a small square with constant orientation.

Our work does not address system dynamics, therefore the state space $\mathcal{X}$ is equal to the Cartesian product $\mathcal{C}^o \times \mathcal{C}^t$. Define $q(t) \in \mathcal{C}^o$ as the observer's configuration at time t.

Let f be the transition equation for the observer: $\dot{q}(t) = f(q, u)$, where u is the vector of control inputs at t. We assume that $|\dot{q}(t)|$ is bounded by v_{obs}. For non-holonomic robots, it is useful to think of u as the pair (v, ω), where v is the speed of the observer and ω its change in bearing. Again, $|v|$ is bounded by v_{obs}. In this paper, we assume that steering is instantaneous. Although is sometimes possible to neglect steering delays, its inclusion in our framework is a topic for future work.

We represent the workspace as a polygon (possibly with holes). Therefore, from now on we use P instead of $\mathcal{W}$ to emphasize our choice of representation.

Visibility Model The observer is assumed to be fitted with an idealized omnidirectional sensor. Hence, a target is visible *iff* the line-of-sight between the target and the observer is un-obstructed.

Let $\mathcal{V}(q) \subseteq P$ be the set of locations from which the target is visible to an observer located at q. This set is the *visibility region* at q. An important concept in the formulation is that of the *visibility sweeping line* ($\ell(t)$), defined as the chord (a segment fully contained in P connecting two points on ∂P) passing through the target and a reflex vertex of P. At any time t, the observer must be in one of the half-planes bounded by $\ell(t)$ in order to see the target; see Figure 1. Although there could be $\Theta(n)$ visibility lines, we later show that at any given time the observer's path is influenced by at most two of these.

Target Predictability and Optimal Paths The target is supposed to be *predictable*. That is, the configuration of the target $q^t(t) \in \mathcal{C}^t$ is known for all time $t < t_{stop}$, where t_{stop} is the target *stopping time*. Therefore, our algorithm computes an *off-line* strategy. It is important to note that although the target is predictable, it may be unavoidable for the observer to lose the target for a particular choice of initial conditions and target velocities. In this scenario, a useful notion is that of *escape time* (t_{esc}) —the time when the observer first loses the target.

Alternatively, the observer is sometimes able to capture the target —i.e., $q(t) = q^t(t)$ for some $t < t_{stop}$. The *capture time* (t_{cap}) is when the target is first captured.

In this paper, a path is optimal if it satisfies the following definition:

Definition 2.1: An observer path π^* is optimal if it is the shortest path among all possible paths that maximize t_{esc}. For the cases when t_{esc} does not exist (i.e., the observer never loses track of the target), then π^* is optimal if it minimizes t_{cap}. If capture before t_{stop} is not possible, then π^* minimizes the final separation between the target and the observer.

III. Overview of the Algorithm

Later in the paper, we will show that the optimal path π^* is composed of *straight-line segments* and *leaning* curves, as described below:

Straight-line segments: The observer moves towards a (carefully chosen) point M in a straight line without ever losing sight of the target.

Leaning curves: This is a curve defined by the connected part of the shortest path to a vertex C of P followed by the observer while moving at maximal speed, constrained to be on the sweeping line $\ell(t)$ rotating around C as the target moves along a straight line; see Figure 1.

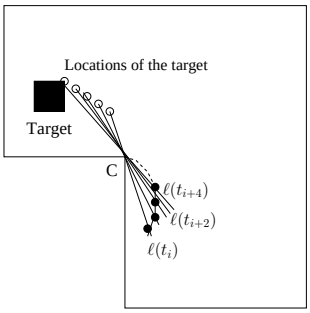


Fig. 1. The leaning curve: The observer is approaches the point C while remaining on $\ell(t)$ as it rotates about C.

Without any loss of generality, assume that the observer is at the origin at time $t = 0$. We now proceed to define the concept of a π-path.

Definition 3.1 (π-path): The path $\pi(q, t) \subset P$ is a *π-path* if it connects the origin to q and satisfies the following properties:

a. The target is always visible to the observer.
b. π is the shortest path that satisfies the above.
c. The observer traverses π at the top speed v_{obs}.
d. The observer arrives at q in time t.

Note that a path $\pi(q, t)$ does not necessarily exists for every combination of q and t. The following lemma establishes the uniqueness of a π-path:

Lemma 3.2: $\pi(q, t)$ is unique for any t and $q \in P$.

Proof: Assume that two distinct π-paths $\pi_1(q, t)$ and $\pi_2(q, t)$ exists, both reaching q at the arrival time t_f. Fix $0 < \alpha < 1$. Let $\pi_1(t)$ and $\pi_2(t)$ be the location of the observer at time t along the two paths, and define $\pi(t) = \alpha\pi_1(t) + (1 - \alpha)\pi_2(t)$. Note that the speed along π is always less or equal to v_{obs}, but there is an instant when the speed is strictly smaller since π_1 and π_2 both start at 0 (else the paths are identical). π is then shorter than π_1 or π_2, which are allegedly π-paths. Hence, π is infeasible in terms of target visibility and an obstacle exists between π_1 and π_2; see Figure 2.

Now, assume that at t_0 there is a value α for which $\alpha\pi_1(t) + (1 - \alpha)\pi_2(t)$ does not see the target, and t_0 is the earliest time when said α exists. Consider the triangle $T(t) = \Delta\pi_T(t)\pi_1(t)\pi_2(t)$ (where $\pi_T(t)$ is the target's position at t). Select δ such that $T(t)$ does not intersect any part of ∂P for any $0 < t \leq t_0 - \delta$ (see Figure 2). That is, $T(t_0 - \delta)$ is entirely below the obstacle. However, for a time arbitrarily close to the arrival time t_f, the triangle $T(t_f - \epsilon)$ is entirely above the obstacle. Therefore, the triangle $T(t)$ must cross the obstacle, and either the edge $(\pi_T(t), \pi_1(t))$ or the edge $(\pi_T(t), \pi_2(t))$ intersect the obstacle at some t. But this contradicts our assumption that both $\pi_1(t)$ and $\pi_2(t)$ see the target for all t. ■

From here on we omit t in the notation of $\pi(q, t)$, and denote this path as $\pi(q)$. Let $\Gamma(t)$ be the subset of all points q such that the length of $\pi(q)$ is $v_{obs}t$. $\Gamma(t)$ contains the arrival point for the optimal path π^* at time t.

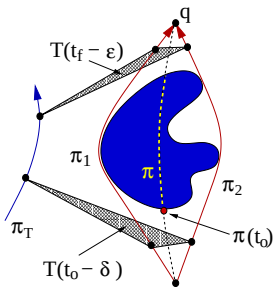


Fig. 2. π-paths are unique: $T(t)$ must cross the obstacle to go from configuration $T(t_0 - \delta)$ to $T(t_f - \epsilon)$, contradicting the assumption that both π_1 and π_2 see the target at all times.

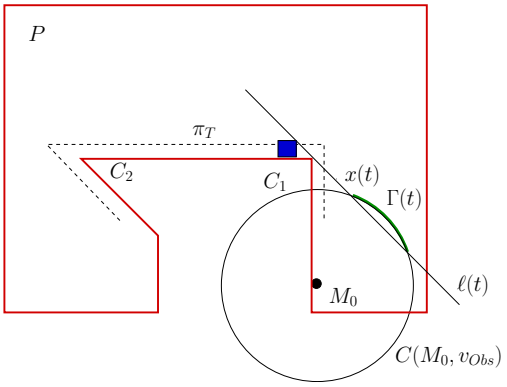


Fig. 3. A simple example: target follows path π_T. $\Gamma(t)$ is also shown.

A. Simple Example

Before we present the general algorithm for finding the optimal path π^*, we first describe a simple example, illustrated in Figure 3. This scenario was simulated and screen shots are shown in Figure 6(a)-(d). The target starts a distance d_0 from the observer, and moves up along the right wall of the polygon, until it reaches the vertex C_1. After rounding up this vertex the target proceeds to go left until it reaches vertex C_2. Afterwards, the target continues around C_2, going down until it reaches its destination. Let the initial position of the observer be M_0.

First Critical Point: M_1

Initially, for small values of t, the possible locations for the observer are all the points of P bounded by the circle $C(M_0, v_{\text{obs}}t)$ of radius $v_{\text{obs}}t$ and center M_0. In this simple case, $\Gamma(t)$ is a portion of this circle. Let $\ell(t)$ denote the visibility sweeping line to the target, once the target moves beyond the vertex C_1. That is, the target at time t is visible from all points above $\ell(t)$ and but not from those below $\ell(t)$; see Figure 3.

Let $\mathbf{x}(t)$ denote the higher intersection point of $C(M_0, v_{\text{obs}}t)$ with $\ell(t)$. As the target moves to the left the point $\mathbf{x}(t)$ moves to the right; see Figure 4.

Let $\theta(t)$ be the angle defined by the segment $\overline{M_0\mathbf{x}(t)}$ and a vertical line. Initially, θ increases with t. After $\theta(t)$ reaches a maximum either the observer loses the target or $\theta(t)$ decreases again. Assume the latter. Let t_1 be the time at which $\theta(t)$ reaches its maximum, and let M_1 be

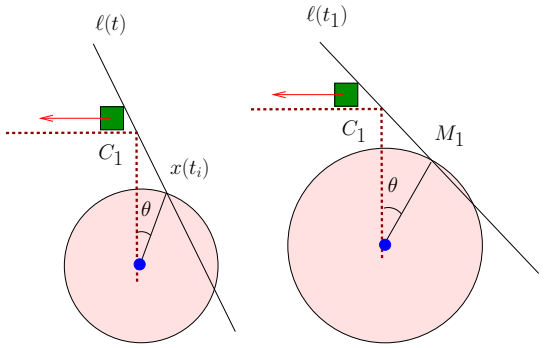


Fig. 4. The angle θ achieves its maximum at time t_1. We set M_1 to be the upper intersection point of $C(M_0, v_{\text{obs}}t_1) \cap \ell(t_1)$.

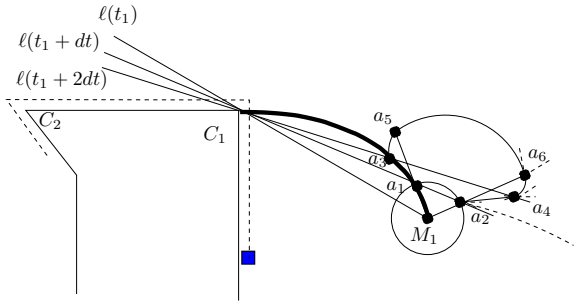


Fig. 5. Computing the leaning curve between M_1 and C_1.

the location of $\mathbf{x}(t_1)$. This point is significant because the observer's optimal path passes through M_1.

It's possible that $\ell(t)$ becomes tangent to $C(M_0, v_{obs}t)$ at time $t = t_1$. For $t'' > t_1$ we have $C(M_0, v_{\text{obs}}t'') \cap \ell(t'') = \emptyset$. That is, the circle $C(M_0, v_{\text{obs}}t'')$ and $\ell(t'')$ do not intersect after time t_1, and target visibility is broken. The observer is doomed to lose the target.

Thus, M_1 is either a tangent point or a local maximum. In either case is optimal to go directly to M_1 because doing so ensures that the target is seen for as long as possible. The difference is that π^* terminates at M_1 in the former case, but continues after M_1 in the latter.

After M_1: Leaning Curves

The optimal path after point M_1 becomes more complicated. In order to make the explanation simpler, we will assume that $\Gamma(t_1)$ contains only the point M_1. In general, M_1 may only be a local maximum, and $\Gamma(t_1)$ must include the circular arc above $\ell(t = t_1)$ (the complete algorithm acknowledges this). But for the simple example shown in Figure 3 we are confident that M_1 also happens to be the global maximum and that π^* passes through this point.

We proceed as follows: Let dt be an infinitesimal time interval. Draw a circle of radius $v_{\text{obs}}dt$ centered at M_1. This circle intersects $\ell(t_1 + dt)$ at points a_1 and a_2; see Figure 5. $\Gamma(t_1 + dt)$ becomes the arc a_1a_2.

Next, we draw circles of radius $v_{\text{obs}}dt$ at every point on the edge a_1a_2. Again, we consider the intersection of these circles with $\ell(t_1+2dt)$ and take all the points above $\ell(t_1 + 2dt)$. $\Gamma(t_1 + 2dt)$ is then composed of 3 edges:

The first one is an arc defined by the circle of radius $v_{obs}2dt$ centered at M_1, bounded by the segments $\overline{M_1 a_1}$ and $\overline{M_1 a_2}$. The second is the arc defined by the circle of radius $v_{obs}dt$ centered at point a_1. The third is the arc defined by the circle of $v_{obs}dt$ centered at point a_2. The points on these 3 circular arcs above $\ell(t_1+2dt)$ define the curve $\Gamma(t_1+2dt)$. Let a_3, a_5, a_6, a_4 be the intersections of these arcs (ordered as in Figure 5).

The growth process continues from the new points and $\Gamma(t_1+3dt)$ can also be determined. Note that the path formed by the curve $M_1, a_1, a_3...$ in Figure 5 forms a lower envelope, below which the target is not visible. This is exactly the *leaning curve*.

By definition, if the observer moves along the leaning curve the target will remain in sight. However, the observer may deviate from the leaning curve before reaching C_1. This will happen if the target makes a turn around a second vertex C_2.

Critical Point M_2

Following an analysis similar to that for M_1, we compute a second maximal point M_2 determined by C_2.

Figure 6(a)-(d) shows the sequence of tangents to the leaning curve as time increases. The observer may decide to depart the leaning curve along a tangent anywhere between M_1 and the vertex C_1. In fact, these tangential paths are π-paths. Therefore, the locus of points reachable by the observer at $t > t_1$ following such strategy is part of $\Gamma(t)$. We call this locus a *ρ-edge* of $\Gamma(t)$. Every point in ρ-edge is induced by a tangent to the leaning curve.

Next, we consider the intersections of the sweeping lines $\ell_2(t)$ defined by C_2 with the ρ-edges. At time t, the ρ-edge intersects $\ell_2(t)$ at two points. Consider the one closest to C_2. As t increases, the intersection achieves a maximum in the sense that it corresponds to the earliest tangent to the leaning curve; see Figure 6(d). This point is analogous to the maximal θ when computing M_1. We mark this point as M_2.

Similar to the computation of M_1, the point M_2 may be a tangent to the ρ-edge, in which case the observer is doomed to lose the target. Going to M_2 nevertheless keeps the target in view for as long as possible, thus π^* terminates at M_2. If M_2 is not tangent to the ρ-edge then this point becomes the new vertex of π^*.

So far the computed path contains the following edges: A segment connecting the observer's initial position M_0 to M_1; a leaning curve connecting M_1 to the departure point; and a segment connecting this point to M_2.

Once the observer reaches M_2 we grow a new leaning curve; Figure 6(d). The process continues until a future sweeping line fails to intersect $\Gamma(t)$, the target is caught, or the target stops. In the first case we lose the target, but π^* maximizes the escape time. In the latter cases, the observer either captures the target in the minimum time or minimizes the final distance to the target.

B. Approximating Leaning Curves

A leaning curve is defined by the sequence of intersection points between the visibility sweeping line $\ell(t)$ and $\Gamma(t)$ as $\ell(t)$ rotates around a vertex of P. To our knowledge, this curve has no close-form solution.

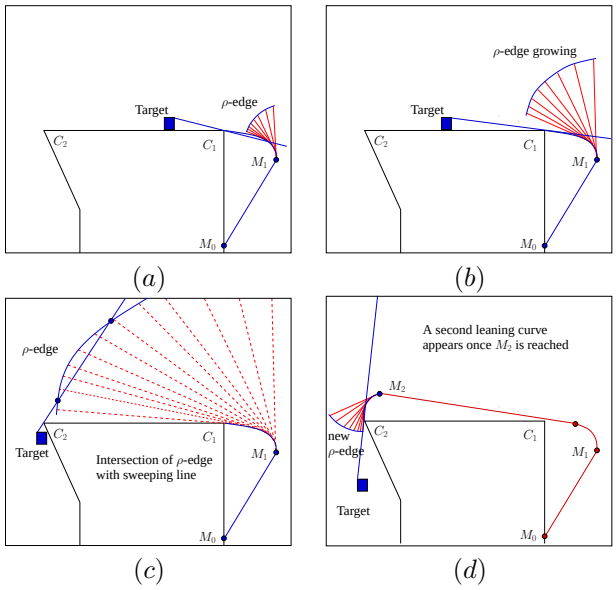
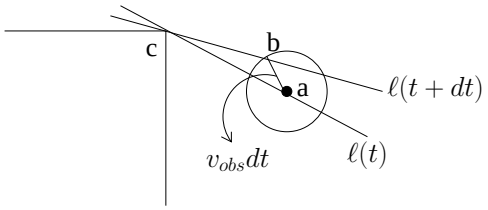


Fig. 6. Evolution of a leaning curve through time: (a) initial growth of a ρ-edge; (b) a fully-formed ρ-edge; (c) the intersection between $\ell(t)$ and the ρ-edge; (d) critical point M_2 and the resultant optimal path π^*.

The leaning curve and the ρ-edge can be approximated as follows. Select a small dt (ideally within 2% of the duration of the leaning curve). Calculate $\ell(t+dt)$ for the chosen value of dt (the line $\ell(t)$ is known because the target is predictable). Consider the following scenario:



Now compute a circle of radius $v_{obs}dt$. Find the intersection of this circle with $\ell(t+dt)$. Take the point b which is closer to the vertex C and use it as the next starting point. The slope between the points a and b approximates the slope of the tangent at b.[1]

IV. Optimal Target Tracking Algorithm

Let M_0 and $\pi_T(0)$ be the initial positions of the observer and the target, respectively. The algorithm computes the evolution of the set $\Gamma(t)$ as the target moves at maximum speed v_T along π_T. At time t, $\Gamma(t)$ contains the candidate points through which π^* might pass. As shown below, $\Gamma(t)$ is a connected path in the plane composed of pieces we refer to as *edges*. Each edge has a constant description complexity.

[1]Alternatively, we could write the differential equation of the curve and use an ODE solver. This will be more efficient, but the geometric method illustrates better the mechanics of the leaning curve.

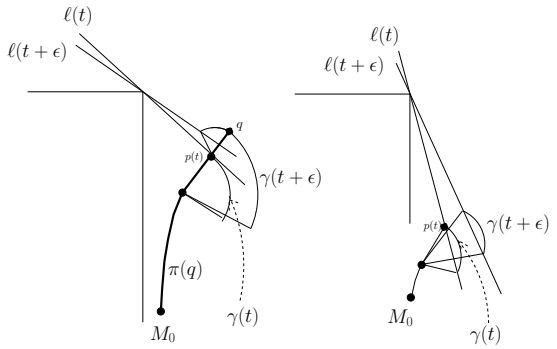


Fig. 7. Characterization of the path $\Gamma(t)$: (a) the curve $\gamma(t)$ is stretched by $p(t)$; (b) the curve $\gamma(t)$ shrinks and the point $p(t)$ does not contribute to $\gamma(t+\epsilon)$.

Informally, $\Gamma(t+\Delta)$ is computed from $\Gamma(t)$ by growing disks of radius $v_{obs}\Delta$ at every point of $\Gamma(t)$, taking the union of these disks, and defining $\Gamma(t+\Delta)$ as the set of points in the boundary of the union that see the target at $t+\Delta$. Matters are complicated by the edges and vertices of P, and the sweeping lines induced by the target's motion. If $\Gamma(t)$ becomes empty at any $t < t_{stop}$ the observer will lose the target. Otherwise, we have an optimal path that keeps the target in view until its final destination.

Calculating $\Gamma(t+\Delta)$ by discretizing t and growing circles $\forall\ q \in \Gamma(t)$ is indeed possible. But this procedure is highly inefficient. Instead, our algorithm maintains a set $\mathcal{O}$ of obstacles which implicitly define $\Gamma(t)$ in the following sense: Every π-path $\pi(q)$ can be described as the shortest path $\pi'(q)$ connecting M_0 to q such that $\pi'(q) \cap \mathcal{O} = \emptyset$. The algorithm finds all the critical events that affect the shape of $\Gamma(t)$ as the target moves, and stores these events in a priority queue Q. After each combinatorial change, we fetch a new event from Q, update $\Gamma(t)$ and $\mathcal{O}$, compute future events, and insert these into Q.

$\Gamma(t)$ is made up of two kinds of *edges*: circular arcs and ρ-edges. At any instant, one or two visibility sweeping lines define the endpoints of one or two edges. Let $\gamma(t)$ be one of these edges, and let $p(t)$ be its endpoint on the sweeping line $\ell(t)$. We say that $p(t)$ *stretches* $\gamma(t)$ if there is a point $q \in \Gamma(t+\varepsilon)$ such that $\pi(q)$ fails to go through $p(t)$ for some small $\varepsilon > 0$ (i.e., the last segment of $\pi(p(t+\varepsilon))$ does not intersect $\gamma(t)$). Otherwise, we say that $p(t)$ *shrinks* $\gamma(t)$. See Figure $7(a)$-(b).

We say that a point $p(t_0) \in \gamma(t_0)$ is an M*-point* if $p(t)$ shrinks $\gamma(t)$ slightly before t_0, and $p(t)$ stretches $\gamma(t)$ slightly after t_0. It can be shown that every ρ-edge contains at most one M-point. We assume that given a description of a ρ-edge, the M-point for this edge can be computed in time $O(1)$. Note that some numerical instability may occur because ρ-edges are approximated, and the computation of a ρ-edge is affected by numerical errors in a previous ρ-edge. However, this is not a problem, since a good approximation scheme produces enough accuracy.

A. Types of Obstacles

(1) Edges of ∂P: A simple type of obstacle is an edge of the polygon that breaks the line of sight between the observer and the target.

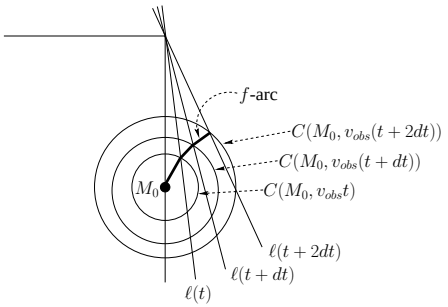


Fig. 8. An illustration of an f-arc.

(2) f-arcs: These are formed by the intersection point $p(t)$ of some edge $\gamma(t)$ of $\Gamma(t)$ with a visibility sweeping line $\ell(t)$ as time progresses while $p(t)$ is shrinking $\gamma(t)$; see Figure 8. The shortest path π^* never intersects an f-arc, except possibly at the endpoints of the f-arc. Thus we are concerned only with the endpoints of f-arcs. The f-arcs can be computed in $O(1)$ time given the description of $\gamma(t)$ and $\ell(t)$.

(3) δ-arc: These are formed by the trace of the shortest path that an observer can take on its way to a reflex vertex of P such that it remains on the sweeping line $\ell(t)$ as the target moves along a straight line. The leaning curves introduced earlier compose the δ-arcs. A δ-arc usually connects an M-point (defined above) to the reflex vertex of P that define $\ell(t)$; see Figure 5. A leaning curve is always a connected part of a δ-arc.

B. Types of Edges of $\Gamma(t)$

(1) Circular edges. A circular edge is denoted by its center, its angular span, and its radius at time t; see Figure $9(a)$. The centers for circular edges of $\Gamma(t)$ can be one of three different types:

a) the starting point of the observer,
b) an M-point,
c) a reflex vertex of the polygon P.

(2) ρ-edges. Describing these edges is more involved. Each ρ-edge is induced by a δ-arc d. A point q is on the ρ-edge $\gamma(t)$ induced by d if the last segment e of $\pi(q)$ is a straight segment of length $v_{\text{obs}}(t-t')$, connecting q to a point q' of d, where $\Gamma(t)$ reaches q' at time t'; see Figure 10 . Since $\pi(q)$ is a shortest path, the tangent to r at q' (if it exists) must be in the direction of e.

Now that we have defined the obstacles and the edges of $\Gamma(t)$ we can describe the data structure needed to maintain and update the event queue. At time t, we maintain the curve $\Gamma(t)$ as a list of the endpoints of the different edges of $\Gamma(t)$, together with the parameters for each edge. For circular edges these parameters are clear, while ρ-edges are stored as pointers to the parameters of the δ-arcs they emerged from.

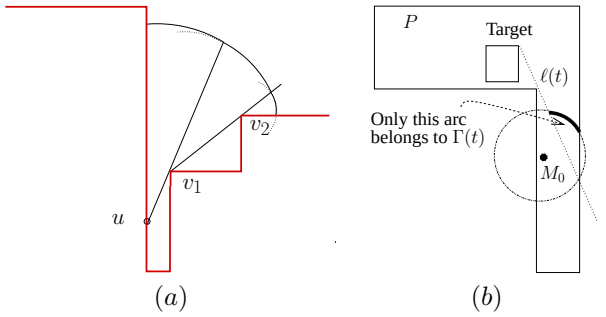


Fig. 9. (a): The circular edges which are parts of $\Gamma(t)$ are emerging from the vertices of P. (b): $\Gamma(t)$ slightly after an event at which a circular edge of $\Gamma(t)$ hits an edge of P. Only the thick edge closer to the target is stored in $\Gamma(t)$.

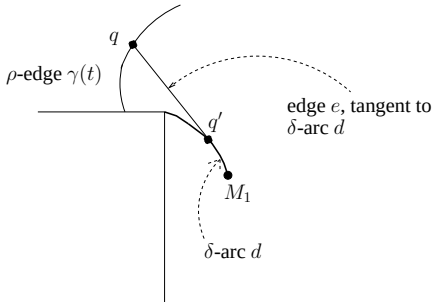


Fig. 10. Defining a ρ-edge. An observer following $\pi(q)$ leaves the δ-arc at time t', and the length of $\overline{qq'}$ is $(t-t')v_{\text{obs}}$.

C. *Types of Events as t Increases*

(1) $\Gamma(t)$ runs into an edge e of ∂P or into a sweeping line $\ell(t)$; see Figure 9(b). In this case, an edge of $\Gamma(t)$ is split into two. Since the observer tries to stay as close as possible to the target, only the edge closest to the target has to be maintained. Let $\gamma(t)$ denote this piece. (Note: the points on ∂P or on $\ell(t)$ are not on $\Gamma(t)$, since they are not endpoints of optimal paths.) We also compute and insert into Q the future time at which a combinatorial event defined by $\gamma(t)$ and e or $\ell(t)$ occurs, (e.g., $\ell(t)$ reaches the endpoint of $\gamma(t)$). In the case of an intersection with $\ell(t)$, we also insert into Q the M-point that occurs on $\gamma(t)$, if it exists.

(2) An edge of $\Gamma(t)$ runs into a reflex vertex C; see Figure 9(b). In this case we begin growing a circular edge at C, and insert it into $\Gamma(t)$.

(3) The first or last edge $\gamma(t)$ of $\Gamma(t)$ has degenerated to a single point. Let $\gamma'(t)$ be the adjacent edge of $\gamma(t)$ along $\Gamma(t)$. Upon reaching this event, $\ell(t)$ begins intersecting $\gamma'(t)$. We compute the future events for $\gamma'(t)$ as in the first case.

(4) The intersection point of a visibility sweeping line and an edge $\gamma(t)$ of $\Gamma(t)$ reaches a M-point q. We insert a new ρ-edge into $\Gamma(t)$ from this point, and if needed, a new circular edge. Note that $\gamma(t)$ must be the first or the last edge of $\Gamma(t)$. We also insert a new a δ-arc into the list of objects $\mathcal{O}$. The other endpoints of these ρ-edges and δ-arcs are not known at this point, and will be revealed as time increases; see Figure 11.

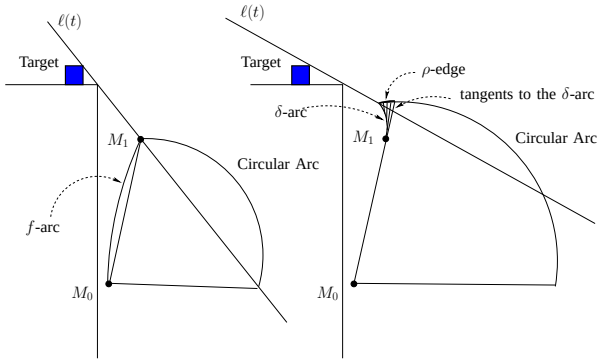


Fig. 11. The M point is reached and a ρ-edge is grown.

(5) The intersection point of a visibility sweeping line $\ell(t)$ and the first or last edge $\gamma(t) \in \Gamma(t)$ intersects an edge of ∂P at a point q. In this case, the observer loses the target at point q, and $\pi(q)$ is the optimal path; see Lemma 4.1. A similar event occurs if $\ell(t)$ becomes tangent to $\gamma(t)$.

(6) A δ-arc d defined by the visibility sweeping line $\ell(t)$ reaches the vertex C of P around which $\ell(t)$ rotates. We finish inserting this arc into $\mathcal{O}$.

(7) The target has reached a bend-point on its path π_T (i.e., the target has changed its direction), or one of the visibility sweeping lines $\ell(t)$ hits a (new) vertex v of P. We re-compute the times of all future events in the queue in which $\ell(t)$ is involved. For each intersection point $p(t)$ of $\ell(t)$ and $\Gamma(t)$ we check whether it stretches or shrinks the edge $\gamma(t)$ of $\Gamma(t)$ that it intersects. In the former case, we insert a new ρ-edge into $\Gamma(t)$ and a new δ-arc into $\mathcal{O}$. In the latter case, we gradually shrink $\gamma(t)$ and insert a new f-arc into $\mathcal{O}$; see Figure 12.

Lemma 4.1: Let g be the f-arc determined by a visibility sweeping line $\ell(t)$, that is closer to the target. Assume that g intersects an edge of ∂P in the interior of g, and let q be the intersection point. Then $\pi(q)$ is the optimal path π^*, and the observer would lose the target at this point.

Proof: Assume that this intersection occurs at time t'. Clearly, if that observer follows $\pi(q)$ then it sees the target for all $t \leq t'$. On the other hand, it is not hard to verify that if t'' is slightly larger than t', then any point that the observer can reach in time t'' (not necessarily along a shortest path) is hidden from the target either by $\ell(t'')$ or by ∂P. ■

Termination of the algorithm: The events of type 5 above takes care of the case where the observer loses the target. The other possibility is that the target reaches its destination at time t_M, and $\Gamma(t_M)$ is not empty. In this case, we take the point $q \in \Gamma(t_M)$ which is the closest to the final position of the target. Finally, the output of the algorithm is reconstructed by tracing back from q, through the objects of $\mathcal{O}$ around which $\pi(q)$ bends. For this, every edge of $\Gamma(t)$ and object of $\mathcal{O}$ "knows" from which obstacle of $\mathcal{O}$ it emerges.

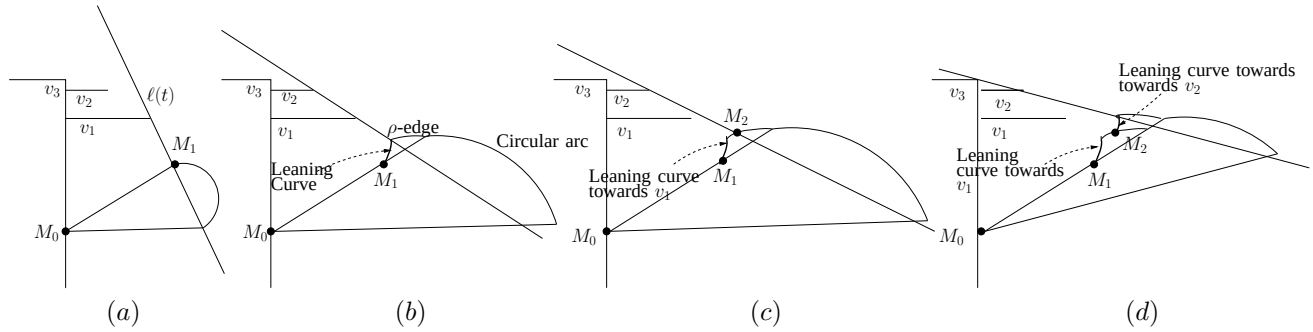


Fig. 12. (a) The first M-point, M_1; (b) a leaning curve is started towards v_1. Now the visibility sweeping line changes from v_1 to v_2 and; (c) a new M-point, M_2, is reached; (d) A leaning curve toward v_2 is started.and the visibility line changes from v_2 to v_3.

D. Structure of $\Gamma(t)$

Lemma 4.2: $\Gamma(t)$ is a connected curve.

Proof: $\Gamma(t)$ *starts as a circular edge and it goes through combinatorial changes only at certain discrete events. The lemma holds for* $t = 0$*, as* $\Gamma(t)$ *is a single point. It is easy to show that if the lemma holds before each of the events listed above, then it holds afterwards as well.* ■

We define a *flight plan* Π to be a path of the observer in the plane, together with the speed of the target at each point along this path. A *legal flight plan* is a flight plan at which the observer moves at velocity $\leq v_{\text{obs}}$ starting from M_0, seeing the target at any point and time along its way.

Let $\mathcal{F}(t)$ denote the set of all points that the observer can reach at time t, following a legal flight plan. We say that $\Gamma(t)$ *hits* a segment e at time t' and point q if $\Gamma(t')$ intersects e at q, while there is no other point of $\mathcal{F}(t') \cap e$ in a small enough vicinity of q.

Lemma 4.3: For every segment $e \subseteq P$, the intersection $e \cap \mathcal{F}(t)$ consists of a single connected interval.

Proof: Consider two points $q_1, q_2 \in e \cap \mathcal{F}(t)$*. Let* Π_1 *and* Π_2 *be two legal flight plans leading to* q_1 *and* q_2*, resp. Let point* q *be on* e *between* q_1 *and* q_2*, whose distance from* q_1 *is* $\alpha||q_1 - q_2||$ *for some* $0 \leq \alpha \leq 1$*.*

Define $\Pi(t') = \alpha\Pi_1(t') + (1-\alpha)\Pi_2(t')$*. Then* Π *is a flight plan from* M_0 *to* q*, and the velocity of the observer along* Π *is* $\leq v_{obs}$*. Moreover, for every* $0 \leq t' < t$*, both* $\Pi_1(t')$ *and* $\Pi_2(t')$ *are on the sides of the sweeping lines from where the target is visible at time* t'*. Thus,* $\Pi(t')$ *also sees the target which implies that* Π *is a legal flight plan, and hence* $q \in \mathcal{F}(t)$*.* ■

We say that a connected curve γ is convex if the line segment connecting every two points on γ does not intersect γ in its interior.

Corollary 4.4: $\Gamma(t)$ is convex for every t.

Lemma 4.5: There are no two times $t_1 < t_2$ at which $\Gamma(t)$ hits e.

Proof: We show this by contradiction. Assume that $\Gamma(t_1)$ *hits* e *at* q_1 *and* $\Gamma(t_2)$ *hits* e *at* q_2*. If at time* t_2 *there are other points of* $\mathcal{F}(t_2)$ *on* e *(except* q_2*) then by Lemma 4.3 there are also points of* $\mathcal{F}(t_2)$ *arbitrary close to* q_2*, which is a contradiction.*

Let $E = \{q \in e | q \in \mathcal{F}(t), t < t_2\}$*. If* $q_2 \in E$ *then the observer reaches* q_2 *at an earlier time, hence* q_2 *cannot be in* $\Gamma(t_2)$*. If* $q_2 \notin E$ *then let* q' *be the closest point of* E *to* q_2*, and let* t' *be the last time that* q' *sees the target. At this time a visibility sweeping line* $\ell(t')$ *passes through* q'*, and* $\ell(t')$ *separates* q' *from* q_2*. Otherwise, an observer located at* q' *at time* t' *could have reached closer to* q_2*. But in this case, it is not hard to verify that* q_2 *could not see the target for any time* $t > t'$*. That is, it cannot be on* $\Gamma(t_2)$*, which is a contradiction.* ■

E. Implementation and Running Time

We construct a triangulation $\mathcal{T}$ of P. Once $\Gamma(t)$ intersects a triangle $\Delta \in \mathcal{T}$ we compute the future time at which $\Gamma(t)$ hits each of the remaining edges of Δ, or hits an edge or vertex of P adjacent to Δ. This takes $O(\log n)$ time per a triangle edge using Corollary 4.6 below. Here n is the number of edges in the polygon P.

Corollary 4.6: Assume that in the time interval $[t_1, t_2]$ neither $\Gamma(t)$ nor the visibility sweeping line $\ell(t)$ go through any combinatorial changes. Given a visibility sweeping line $\ell(t)$, we can find in time $O(\log n)$ the earliest time $t' \in [t_1, t_2]$ and point at which $\ell(t')$ intersects $\Gamma(t')$, if t' exists.

Proof: We perform a binary search on the order of the vertices of $\Gamma(t)$*, as follows: we pick one such vertex* $u(t)$ *and we observe that* $u(t)$ *moves along a curved path in the plane as* t *changes. We can compute in time* $O(1)$ *the time* t_u *of the intersection point of* $u(t)$ *and* $\ell(t)$*, and by checking the directions in which* $\ell(t_u)$ *stabs* $\Gamma(t_u)$*, deduce on which side of* $u(t)$ *the next vertex should be picked.* ■

Lemma 4.7: Each vertex defines at most one circular edge of $\Gamma(t)$.

Proof: Assume that v *is the center of two circular edges. These edges cannot exists at the same time and have the same radius, because of the way we maintain* $\Gamma(t)$*. On the other hand, if they are defined for two different times* t_1, t_2*, then* v *must be in both* $\Gamma(t_1), \Gamma(t_2)$*, contradicting Lemma 3.2.* ■

Lemma 4.8: The total number of events of type 7 that occur in the course of the algorithm is $O(n)$.

Proof: Let v *be a vertex of* P*. Observe that in the course of the algorithm, there is at most one connected time interval* $[t_1, t_2]$ *in which* v *sees the target. In other words, once* v *has stopped seeing the target, it will never see it again. To verify this, consider* $\mathcal{V}(v)$*, the visibility polygon of* v *and note that since the target moves along a shortest path, the union of its locations inside* $Vis(v)$ *must be connected.*

Now consider an event of type 7 as defined above. We need to bound the number of times that a visibility sweeping line moves from rotating about vertex v *to another vertex,* v'*. Once this happens,* v *would not see the target anymore, hence this event can happen only* $O(n)$ *times.* ■

Lemma 4.9: The total number of edges of $\Gamma(t)$, that appear in the course of the algorithm is $O(n)$.

Proof: The bound on the number of circular edges, centered at vertices of P, is $O(n)$ (from Lemma 4.7). To bound the number of other ρ edges of $\Gamma(t)$, recall that each of their endpoints occurs when one of the sweeping lines goes through an event of type 7. Since the number of these events is $O(n)$ (from Lemma 4.8), we obtain the desired bound. Note that this bound also bounds the number of objects in $\mathcal{O}$. ∎

Lemma 4.10: The number of events of type 1 is $O(n)$.

Proof: The bound on the number of times that $\Gamma(t)$ hits a edge of P follows from Lemma 4.5. The bound on the number of times a visibility sweeping line can hit $\Gamma(t)$ follows from the fact that this event can happen only after some other type of combinatorial event (either the target changed its direction, or the sweeping line began rotating around another vertex of P, or the sweeping line began rotating around another vertex of the target). There are only $O(n)$ such events. ∎

We also need to efficiently handle when a sweeping line $\ell(t)$ rotating around v hits and proceeds to rotate around a new vertex v'. For this we maintain a list containing all vertices of P above $\ell(t)$, and a complementary list for those below $\ell(t)$. Each list is maintained using the data structure for dynamic convex hull of Chan [13]. As the target moves along a straight line, the next vertex intersected by $\ell(t)$ can be computed in time $O(\log n)$. Afterwards, the structure is updated in time $O(\log^{1+\varepsilon} n)$. Again, we use the fact that once a vertex of P fails to see the target it would never see it again. Thus, the total time required here is $O(n \log^{1+\varepsilon} n)$. Combining all of the above results yields the following theorem:

Theorem 4.11: We can find the optimal path π^* in time $O(n \log^{1+\varepsilon} n)$.

V. Conclusion and Future Work

This paper introduced an $O(n \log^{1+\varepsilon} n)$-time algorithm for computing the optimal robot motion strategy that maintains line-of-sight visibility of a target moving inside a polygon with n vertices. The computed path maximizes t_{esc} (and minimizes t_{cap} when the target cannot escape). Our algorithm does not consider system dynamics. Nevertheless, the algorithm could prove useful in practice since t_{esc} for the kinematic case is sometimes an adequate approximation of t_{esc} for the dynamic case.

We ignored non-holonomic constraints in our analysis. However, this is not a serious shortcoming for non-holonomic robots capable of steering quickly because the paths computed by our algorithm tend to be smooth (i.e., few sharp turns). However, incorporating non-holonomic constraints is a topic for future research.

The inclusion of *range constraints* to the visibility model will be an important extension to our work to address a typical limitation in most sensors. A more difficult problem is the inclusion of *field-of-view constraints* and remove the assumption of omnidirectional vision. In this problem, the tangent of π^* must always be directed (roughly) towards the target in order to keep it in view.

Although the assumption that the target's path is known a priori is quite strong, we believe that our algorithm could be the basis of an on-line algorithm that maximizes the *true minimum time-to-escape*, defined as the shortest escape time among all possible target strategies in response to a sequence of observer actions. Worst-case tracking becomes a minimax problem for unpredictable targets. We believe that our algorithm may be a first step in designing a strategy for this general problem.

Acknowledgements This research was partially supported by NSF grant, ACR-0222920.

VI. References

[1] S. Hutchinson, G. D. Gager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

[2] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Transactions on Robotics and Automation*, 1993.

[3] S. M. Lavallée, J. Troccaz, L. Gaborit, A. L. Benabid P. Cinquin, and D. Hoffmann, "Image guided operating robot: A clinical application in stereotactic neurosurgery," in *Computer Integrated Surgery: Technology and Clinical Applications*, R.H.Taylor, S. Lavallée, G. Burdea, and R. Mösges, Eds. 1995, pp. 342–351, MIT Press.

[4] A. J. Briggs and B. R. Donald, "Visibility-based planning of sensor control strategies," *Algorithmica*, vol. 26, no. 3-4, pp. 364–388, 2000.

[5] S. M. LaValle, H. H. González-Baños, C. Becker, and J.-C. Latombe, "Motion strategies for maintaining visibility of a moving target," in *IEEE Conference on Robotics and Automation*, 1997.

[6] H. Gonzalez-Banos, C.-Y. Lee, and J.-C. Latombe, "Real-time combinatorial tracking of a target moving unpredictably among obstacles," in *IEEE Conference on Robotics and Automation*, 2002.

[7] P. Fabiani and J. C. Latombe, "Dealing with geometric constraints in game-theoretic planning," in *Proc. Int. Joint Conf. on Artif. Intell.*, 1999, pp. 942–947.

[8] T.-Y. Li and T.-H. Yu, "Planning object tracking motions," in *IEEE Conference on Robotics and Automation*, 1999.

[9] B. Nelson and P. K. Khosla, "Integrating sensor placement and visual tracking strategies," in *IEEE Conference on Robotics and Automation*, 1994.

[10] K. N. Kutulakos, C. R. Dyer, and V. J. Lumelsky, "Provable strategies for vision-guided exploration in three dimensions," in *IEEE Conference Robotics and Automation*, 1994.

[11] R. Sharma and S. Hutchinson, "On the observability of robot motion under active camera control," in *IEEE Conference on Robotics and Automation*, 1994.

[12] J.-C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, MA, 1991.

[13] Chan, "Dynamic planar convex hull operations in near-logarithmaic amortized time," *JACM: Journal of the ACM*, vol. 48, 2001.