Algorithms for Computing Grazing Area

L. Gewali, R. Venkatasubramanian, and D. Glasser
HRH College of Engineering, University of Nevada, Las Vegas

Abstract: We consider the problem of sweeping (or grazing) the interior/exterior of a polygon by a flexible rope whose one end point (anchor) is attached on the boundary of the polygon. We present a linear time algorithm to compute the grazing area inside a simple polygon. We show how to extend the algorithm for computing the internal grazing area, without increasing its time complexity, to compute the grazing area in the exterior of a simple polygon. For grazing in the exterior of a convex polygon , we present an $O(n)$ time algorithm to locate the anchor point that maximizes the simple grazing area. All three algorithms are optimal within a constant factor. Grazing area problems can be viewed as guard placement problems under *L-visibility*.

Keywords: Computational geometry, visibility, guard placement.

I. Introduction

Problems dealing with visibility properties of polygons have attracted the interest of many researchers in recent years [1]. One such problem is the computation of "visibility region" from a given point inside/outside a polygon. Under the standard definition of visibility, two points inside a polygon are visible if the line segment joining them does not intersect with the exterior of the polygon. Linear time algorithms for computing the visibility polygon under standard visibility have been reported [2]. Many visibility problems have been investigated by phrasing them as the placement of point guards inside a polygonal gallery and are known as "art gallery problems" [1]. The single guard placement problem asks for locating a point inside/outside a polygon from which the area of the visibility region is maximized. Approximation algorithms for solving single guard placement problems have been reported [3]; and the existence of exact algorithm for this problem is open.

Guard placement problems have also been pursued by allowing interesting variations on the standard notion of visibility. Under stair-case visibility (s-visibility, in short) two points inside a polygon are visible if they can be connected by a stair-case path (a path consisting of only horizontal and/or vertical line segments and monotone along x- and y-axis) without intersecting the exterior of the polygon. The notion of s-visibility has been useful for decomposing orthogonal polygon onto simpler components [4]. Also, concepts of rectangular visibility (r-visibility) and distance limited visibility (d-visibility) have been introduced [6]. (It may be noted that under *d-visibility*, two points inside a polygon are visible if the length of the line segment l connecting them is no more than d and l does not intersect with the polygon [6].) Recently, algorithms for computing visibility region from a point under diffuse reflection/refraction have been considered [7].

In this paper we consider problems dealing with the sweeping (or **grazing**) of the interior/exterior of a polygon by a rope whose one end is attached to a fixed point which we call the **anchor point**. The set of points that can be reached by the free-end of the rope is the grazing area. Figure 1 shows examples of grazing area in the exterior of a convex polygon. An example of grazing area in the interior of a simple polygon is shown in Figure 6. The specific problems addressed are as follows.

(IGAP) Internal Grazing Area Problem: Given a polygon Q, an internal point o on its boundary, and a rope of length L, find the grazing area inside the polygon when one end of the rope is anchored at o. External Grazing Area Problem (**EGAP**) is similarly defined when the grazing area is outside the polygon.

MaxGAP (Maximum Grazing Area Problem): Given a convex polygonal obstacle Q of n vertices and a rope of length L, find an anchor point on the boundary of the polygon that maximizes the grazing area. **MinGAP** (Minimum Grazing Area Problem) is defined similarly.

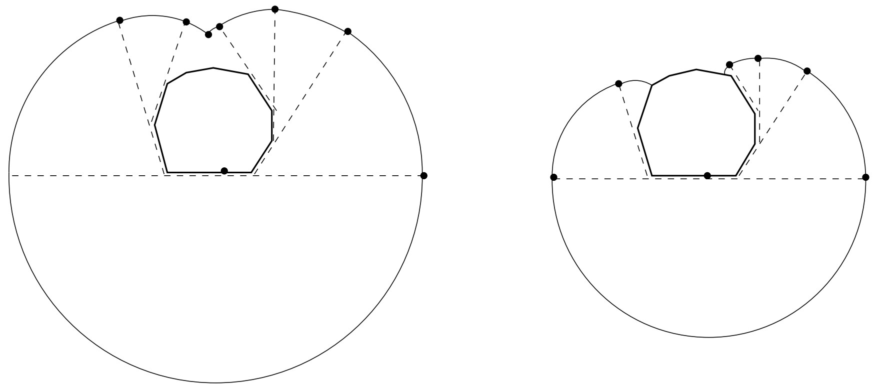
Grazing area problems can be viewed as variations of standard visibility problems. Two points a and b in the presence of a polygonal obstacle are **L-visible** if they can be connected by a rope of length L. (L-visibility is like diffuse-visibility limited by length L). Then we can view IGAP/EGAP as the problem of computing the visibility polygon under L-visibility. Similarly, MaxGAP can be viewed as the problem of placing a single guard to maximize the visibility area, under L-visibility. In [9], the notion of grazing area has been applied to compute the area of intersection between two closed 2-D objects, which also contains an $O(n^2)$ time algorithm to locate an anchor point that minimizes the grazing area inside a convex polygon.

In section II, we develop preliminaries for grazing in the exterior of a convex polygon and characterize two types of grazing areas: (a) simple grazing area, and (b) non-simple grazing area. We prove that the anchor point yielding the maximum simple grazing area is given by one of the $3n$ points on its boundary. In section III, we present a non-trivial algorithm to compute the maximum simple grazing area in $O(n)$ time, which is optimal within a constant factor. In section IV, we present an optimum $O(n)$ time algorithm to solve IGAP for simple polygons by constructing the geodesic-tree (shortest path tree) induced by the problem instance. We, then, show how to extend the algorithm to compute grazing area in the exterior of simple polygons without increasing its time complexity. We conclude by discussing the extension of grazing area problems.

II. Preliminaries

The vertices of the convex polygonal obstacle Q in clockwise order are $v_0, v_1, v_2, ..., v_{n-1}$ and the edge (v_i, v_{i+1}) is denoted by e_i. The length of e_i is denoted by l_i and the angle between e_{i-1} and e_i, as shown in Figure 3, is referred to by θ_i. All additions and subtractions on indices are done modulo n. When the length of the rope is greater than one half the perimeter p of the obstacle, the grazing area completely encloses the obstacle. This type of grazing area is similar to a polygon with a hole and we call it a **non-simple grazing area** (Figure 1a). For the other case ($L < p/2$), the grazing area is like a simple polygon and we call it a **simple grazing area**, (Figure 1b).

It is straightforward to solve IGAP/EGAP for convex polygons by expressing the grazing area in term of l_i's, θ_i's, and the length of the rope L.



(a): General Grazing Area (b): Simple Grazing Area

Figure 1: Two Types of Grazing Areas

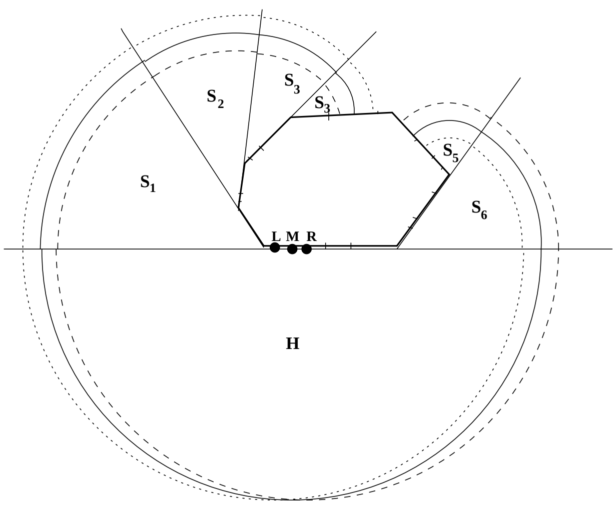


Figure 2: Illustrating the Proof of Lemma 2

Lemma 1: EGAP/IGAP for a convex polygon can be solved in $O(n)$ time.

Proof: Here, we sketch the algorithm for solving EGAP. (Algorithm for IGAP is similar.) The edges bounding the grazing area consist of the edges of the polygon and the circular arcs corresponding to the external angles of the polygon. By knowing the lengths of the edges of the boundary of the grazing region, its area can be easily determined in $O(n)$ time. For the case of simple grazing area, the lengths and radii of all bounding arcs can be determined in $O(n)$ time, in a straightforward way, by scanning the

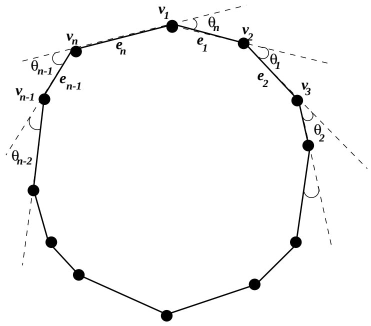


Figure 3

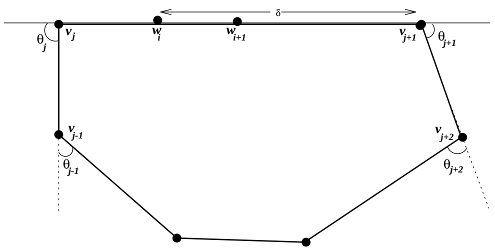


Figure 4

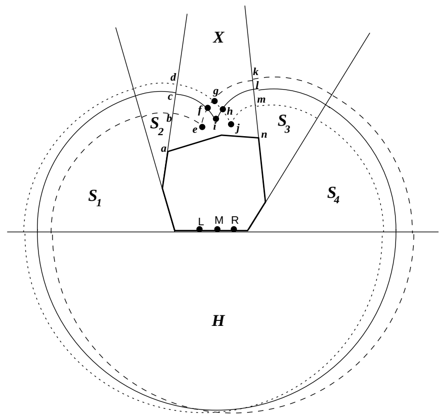


Figure 5

boundary of the polygon in the left (counter clockwise) and the right (clockwise) sides, starting from the anchor point. The case of the non-simple grazing area is similar, except for the determination the point where the arcs from the left and the right side meet (point I in Figure 1a). Meeting point I can be determined by performing "advance and check" process as follows. Starting from the anchor point, edges of the polygon are examined by advancing in the left and the right sides of the anchor point. Each advance can be done by selecting an edge either from the left or from the right side; however, to make a balanced advance, the next edge is selected from that side which makes the total length of scanned edges in one side to be as close as possible to the total length of scanned edges in the other side. During each advance, circular arcs corresponding to newly scanned edges in each side are checked for intersection. The "advance and check" process stops whenever the arcs from left and right sides intersect. All these checks take $O(n)$ time. □

A point q is said to be **covered** by the rope if that point lies on the grazing area. If we slide the anchor point from v_i towards v_{i+1}, the set of vertices covered by the rope may change. We increase the total number of vertices to $3n$ by adding $2n$ **steiner** vertices, so that, when the anchor point slides between adjacent vertices (in the new set), the set of non-steiner vertices covered by the rope do not change. Let v_i^c (respectively v_i^a) be the farthest point on the boundary that can be covered by the rope from the clockwise (respectively, counter clockwise) direction when the anchor point is at v_i. Points v_i^c and v_i^a are the steiner vertices induced by v_i. Including such $2n$ steiner vertices, there are now a total of $3n$ vertices and $3n$ edges. We assign new labels to these vertices and edges in their order of occurrence in the clockwise traversal of the boundary as $w_0, w_1, w_2, \cdots, w_{3n-1}$ and $g_0, g_1, \cdots, g_{3n-1}$, respectively. Note that non-steiner vertices have two labels, w_i and v_j. The new set of edges g_i 's are called the **sub-edges**.

In the rest of this section and in Section III, unless otherwise stated, we use the term grazing area to indicate simple grazing area, i.e., we assume that L is less than one half the perimeter of the obstacle.

Lemma 2: Solution for MaxGAP in the exterior of a convex polygon is given by one of the vertices (including steiner vertices) of the polygon.

Proof: We prove the lemma by showing that any anchor point lying on the interior of a sub-edge can not yield the maximum grazing area. Consider two points p_l and p_r on a sub-edge e separated by a distance 2ϵ. Let p_m be the mid-point of the segment (p_l, p_r) (Figure 2). We denote by $A(p_m)$, $A(p_l)$, and $A(p_r)$ the grazing area when the anchor point is on p_m, p_l, and p_r, respectively. If $A(p_m)$ is maximum then we must have:

$$2A(p_m) > A(p_l) + A(p_r) \quad (1)$$

The free space , outside of the polygon, can be partitioned into half plane H and infinite sectors $S_1, S_2, S_3, ..., S_k$ as shown in Figure 2. In Figure 2, the boundary of the grazing area, corresponding to the anchor points at p_m, p_l, and p_r, is shown by dotted, solid, and dashed arcs, respectively. Let $A(S_i, q)$, and $A(H, q)$ denote the portions of the grazing area lying on S_i

and H, respectively, when the anchor point is on a point q. Since $A(H, p_l)$, $A(H, p_r)$, and $A(H, p_m)$ are equal, we have:

$$A(H, p_l) + A(H, p_r) = 2A(H, p_m) \tag{2}$$

The grazing areas $A(S_i, p_m)$, $A(S_i, p_l)$, and $A(S_i, p_r)$ can be written in term of sector angle γ_i and some radius r_i as:

$$\begin{aligned} A(S_i, p_m) &= \frac{\gamma_i}{2} r_i^2 \\ A(S_i, p_l) &= \frac{\gamma_i}{2}(r_i + \epsilon)^2 \\ A(S_i, p_l) &= \frac{\gamma_i}{2}(r_i - \epsilon)^2 \end{aligned}$$

By inspecting the above three equations we have:

$$A(S_i, p_l) + A(S_i, p_r) > 2A(S_i, p_m) \tag{3}$$

From equations 2 and 3, we have:

$$A(p_l) + A(p_r) > 2A(p_m) \tag{4}$$

This implies that equation 1 can not be true and hence the anchor point yielding the maximum grazing area can not be in the interior point of an edge. □

Here onwards, for the purpose of convenience, we use a slightly different notation for grazing areas (in comparison to that used in Lemma 2). We use $\mathcal{A}_i$ to denote the grazing area when the anchor point is at vertex v_i. When the anchor point is at v_i, let the furthest vertices covered by the rope in the clockwise and the counter clockwise directions be v_{i+m} and v_{i-k}, respectively. Then $\mathcal{A}_i$ can be written as follows.

$$\begin{aligned} \mathcal{A}_i \;=\; & \frac{\pi L^2}{2} + \theta_i L^2 \\ & + \theta_{i+1}(L - l_i)^2 + \theta_{i+2}(L - l_i - l_{i+1})^2 + \theta_{i+3}(L - l_i - l_{i+1} - l_{i+2})^2 + \cdots \\ & + \theta_{i+m}(L - l_i - l_{i+1} - l_{i+2} - \cdots - l_{i+m-1})^2 \\ & + \theta_{i-1}(L - l_{i-1})^2 + \theta_{i-2}(L - l_{i-1} - l_{i-2})^2 + \cdots \\ & + \theta_{i-k}(L - l_{i-1} - l_{i-2} - \cdots - l_{i-k})^2 \end{aligned} \tag{5}$$

Let us analyze the variation of simple grazing area when the anchor point varies along a sub-edge $g_i = (w_i, w_{i+1})$ lying on edge e_j. Let the anchor point be at a distance x from w_i towards w_{i+1}. Let δ be the length of the line segment (w_i, v_{j+1}), i.e., δ is the distance from w_i to its nearest non-steiner vertex in the clockwise direction (Figure 4). The grazing area as a function of distance x from w_i, denoted by $\mathcal{A}_i^+(x)$, can be expressed as:

$$\begin{aligned}\mathcal{A}_i^+(x) \quad = \quad & \frac{\pi L^2}{2} \\ & + \theta_{j+1}(L-\delta+x)^2 + \theta_{j+2}(L-\delta-l_{j+1}+x)^2 \\ & + \cdots + \theta_{j+m}(L-\delta-l_{j+1}-l_{j+2}-\cdots-l_{j+m-1}+x)^2 \\ & + \theta_j(L+\delta-l_j-x)^2 + \theta_{j-1}(L+\delta-l_j-l_{j-1}-x)^2 \\ & + \cdots + \theta_{j-k}(L+\delta-l_j-l_{j-1}-l_{j-2}-\cdots-l_{j-k}-x)^2 \\ & + \phi_1 x^2 \end{aligned} \tag{6}$$

where $\phi_1 = \theta_{j+m+1}$ if w_i is a steiner vertex such that the non-steiner vertex corresponding to w_i lies in the clockwise direction; otherwise, $\phi_1 = 0$. Notice that the superscript $'+'$ in $\mathcal{A}_i^+(x)$ emphasizes that the distance x increases in the clockwise direction. $\mathcal{A}_i^+(x)$ can be rewritten as:

$$\mathcal{A}_i^+(x) \quad = \quad a_i^+ x^2 + b_i^+ x + c_i^+ \tag{7}$$

where

$$a_i^+ \quad = \quad \theta_{j+1} + \theta_{j+2} + \cdots + \theta_{j+m} + \theta_j + \theta_{j-1} + \cdots + \theta_{j-k} + \phi_1 \tag{8}$$

$$\begin{aligned} b_i^+ \quad = \quad & 2\{\theta_{j+1}(L-\delta) + \theta_{j+2}(L-\delta-l_j-l_{j+1}) + \cdots \\ & + \theta_{j+m}(L-\delta-l_j-l_{j+1}-\cdots-l_{j+m-1})\} \\ & - 2\{\theta_j(L+\delta-l_j) + \theta_{j-1}(L+\delta-l_j-l_{j-1}) + \cdots \\ & + \theta_{j-k}(L+\delta-l_j-l_{j-1}-\cdots-l_{j-k})\} \end{aligned} \tag{9}$$

$$\begin{aligned} c_i^+ \quad = \quad & \frac{\pi L^2}{2} \\ & + \{\theta_{j+1}(L-\delta)^2 + \theta_{j+2}(L-\delta-l_{j+1})^2 - \theta_{j+3}(L-\delta-l_{j+1}-l_{j+2})^2 + \cdots \\ & + \theta_{j+m}(L-\delta-l_{j+1}-l_{j+2}-\cdots-l_{j+m-1})^2\} \\ & + \{\theta_j(L+\delta-l_j)^2 + \theta_{j-1}(L+\delta-l_j-l_{j-1})^2 + \cdots \\ & + \theta_{j-k}(L+\delta-l_j-l_{j-1}-l_{j-2}-\cdots-l_{j-k})^2\} \end{aligned} \tag{10}$$

Observation 1: From equation 7 we find that the grazing area as a function of x has the form of a parabola. The second derivative of the area function with respect to x is a_i^+ i.e., $\frac{d^2\mathcal{A}_i^+(x)}{dx^2} = a_i^+$. The coefficient a_i^+ is the sum of positive angles (equation 8), and hence the parabola is in the first quadrant with its arms going towards the positive y-axis.

Observation 1 is useful to compute the minimum grazing area efficiently (discussed in the Conclusion).

III. Algorithm for Maximum Simple Grazing Area

Lemma 1 and Lemma 2 can be directly used to compute the maximal simple grazing area in $O(n^2)$ time. To speed up the algorithm we have to find ways of avoiding $O(n)$ computations per vertex. The idea is to relate grazing areas corresponding to adjacent vertices and use this fact to compute $\mathcal{A}_{i+1}$ from $\mathcal{A}_i$. An inspection of the expressions for $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$ (equation 5) shows that the number of terms present in $\mathcal{A}_{i+1}$ but not in $\mathcal{A}_i$ is $O(n)$. Hence, it does not seem possible to compute $\mathcal{A}_{i+1}$ from $\mathcal{A}_i$ in constant time by a straightforward comparison of the corresponding terms.

Consider the variation of grazing area when the anchor point varies along a sub-edge $g_i = (w_i, w_{i+1})$. This variation can be expressed in two ways. The grazing area as a function of distance x from w_i, which was denoted by $\mathcal{A}_i^+(x)$ earlier, is given by equation 7. The same variation can be viewed as a function of distance x from w_{i+1}, which we denote by $\mathcal{A}_{i+1}^-(x)$, and can be written as:

$$\mathcal{A}_{i+1}^-(x) \quad = \quad a_{i+1}^- x^2 + b_{i+1}^- x + c_{i+1}^- \qquad (11)$$

The superscript $'-'$ in $\mathcal{A}_{i+1}^-(x)$ emphasizes that the distance x increases in the counter clockwise direction.

Our goal is to compute all coefficients $(a_i^+, b_i^+, c_i^+), 1 \leq i \leq 3n$, and $(a_i^-, b_i^-, c_i^-), 1 \leq i \leq 3n$, efficiently. Functions $\mathcal{A}_i^+(x)$ and $\mathcal{A}_i^-(x)$ can be used to relate coefficients (a_i^+, b_i^+, c_i^+) and (a_i^-, b_i^-, c_i^-). Similarly, functions $\mathcal{A}_i^+(x)$ and $\mathcal{A}_{i+1}^-(x)$ can be used to relate coefficients (a_i^+, b_i^+, c_i^+) and $(a_{i+1}^-, b_{i+1}^-, c_{i+1}^-)$.

Lemma 3: Coefficients a_i^+, b_i^+, and c_i^+ can be computed from coefficients a_i^-, b_i^-, and c_i^- in constant time.

Proof: The expression for $\mathcal{A}_i^-(x)$ is:

$$\mathcal{A}_i^-(x) \quad = \quad a_i^- x^2 + b_i^- x + c_i^- \tag{12}$$

where,

$$\begin{aligned}
a_i^- &= \theta_{j+1} + \theta_{j+2} + \cdots + \theta_{j+m} + \theta_j + \theta_{j-1} + \cdots + \theta_{j-k} + \phi_2 && (13)\\
b_i^- &= -2\,\{\theta_{j+1}(L-\delta) + \theta_{j+2}(L-\delta-l_j-l_{j+1}) + \cdots \\
&\quad + \theta_{j+m}(L-\delta-l_j-l_{j+1}-\cdots-l_{j+m-1})\} \\
&\quad + 2\{\theta_j(L+\delta-l_j) + \theta_{j-1}(L+\delta-l_j-l_{j-1}) + \cdots \\
&\quad + \theta_{j-k}(L+\delta-l_j-l_{j-1}-\cdots-l_{j-k})\} && (14)\\
c_i^- &= \frac{\pi L^2}{2} \\
&\quad + \{\theta_{j+1}(L-\delta)^2 + \theta_{j+2}(L-\delta-l_{j+1})^2 - \theta_{j+3}(L-\delta-l_{j+1}-l_{j+2})^2 + \cdots \\
&\quad + \theta_{j+m}(L-\delta-l_{j+1}-l_{j+2}-\cdots-l_{j+m-1})^2\} \\
&\quad + \{\theta_j(L+\delta-l_j)^2 + \theta_{j-1}(L+\delta-l_j-l_{j-1})^2 + \cdots \\
&\quad + \theta_{j-k}(L+\delta-l_j-l_{j-1}-l_{j-2}-\cdots-l_{j-k})^2\} && (15)
\end{aligned}$$

In equation 13, $\phi_2 = \theta_{j-k+1}$ if w_i is a steiner vertex such that the non-steiner vertex corresponding to w_i lies in the counter clockwise direction; otherwise, $\phi_2 = 0$. Comparing coefficients a_i^+, b_i^+, and c_i^+, given by equations 8-10, with a_i^-, b_i^- and c_i^-, given by equations 13-15, we observe that:

$$\begin{aligned}
a_i^+ &= a_i^- + \phi_1 - \phi_2 && (16)\\
b_i^+ &= -b_i^- && (17)\\
c_i^+ &= c_i^- && (18)
\end{aligned}$$

□

Lemma 4: Coefficients a_{i+1}^-, b_{i+1}^-, and c_{i+1}^- can be computed from coefficients a_i^+, b_i^+, and c_i^+ in constant time.

Proof: Functions $\mathcal{A}_i^+(x)$ and $\mathcal{A}_{i+1}^-(|g_i| - x)$ are equal in the interval $|g_i|$. By Taylor's theorem of calculus, the coefficients of equal powers of x (in $\mathcal{A}_i^+(x)$ and $\mathcal{A}_{i+1}^-(|g_i| - x)$) should be equal, which results in:

$$a_{i+1}^- \quad = \quad a_i^+ \tag{19}$$

$$b_{i+1}^{-} = -2a_i^{+}|g_i| - b_i^{+} \quad (20)$$
$$c_{i+1}^{-} = c_i^{+} + b_i^{+}|g_i| + a_i^{+}|g_i|^2 \quad (21)$$

□

Theorem 1: The anchor point yielding the maximum simple grazing area in the presence of a convex polygonal obstacle can be computed in $O(n)$ time.

Proof: We first maintain the vertices w_i's and the data associated with them (e.g., coordinates of w_i's, angle θ_i's, etc) in a doubly linked circular list C. Each node of C also contain pointers to the nearest non-steiner vertices on both sides (clockwise and counter clockwise). Such a list C can be made from the original list of non-steiner vertices v_i's in $O(n)$ time. In $O(n)$ time, we can compute a_i^-, b_i^-, c_i^- and the corresponding area $\mathcal{A}_i^-(x = 0)$ by traversing the doubly linked list and using equations 13-15. We relate the coefficients of area functions corresponding to adjacent sub-edges and propagate the computation from the current vertex to the next by walking in the list C and using Lemma 3 and Lemma 4. During each advance, we update the value of the maximum grazing area. Since there are $3n$ vertices in total, the total time is bounded by $O(n)$. The correctness of the algorithm follows from Lemma 2. □

IV. Grazing in the Presence of Simple Polygons

In this section we consider the computation of the grazing area from an interior boundary point o when the polygon Q is simple (not necessarily convex). As mentioned earlier, the length of the rope is L and the anchor points is fixed at o. We use $I(Q, L, o)$ to denote an instance of IGAP. An example of the grazing area inside a simple polygon is shown in Figure 6, where the region that cannot be reached by the rope is shown shaded. Our approach to solve IGAP is to partition the polygon into carefully chosen triangles and obtain the solution by computing grazing areas in each triangle by introducing appropriate rope lengths and anchor points. Unlike the triangles in the standard triangulation, where triangle vertices must be the vertices of the polygon, the triangulation appropriate for computing grazing area may have the vertices of their triangles at any point on the boundary of the polygon.

Let $SP(o, v_i)$ denote the geodesic path (i.e., shortest path) connecting o to v_i. The union of the shortest paths, $SP(o, v_i), 1 \leq i \leq n$, forms the shortest path tree T. Figure 7 shows the shortest path tree rooted at the anchor point o. The shortest path tree partitions the polygon into **funnel** polygons [8]. Note that a funnel polygon is a simple polygon consisting of a pair of **walls** and a **lid**: the pair of wall consist of two outward convex chains sharing a common vertex (**apex** of the funnel); and the **lid** is the line segment that connects the end of the walls. In Figure 7, vertices of one of the funnels are o, a, b, c, d, and e with lid bc, apex o, and walls oab and $oedc$.

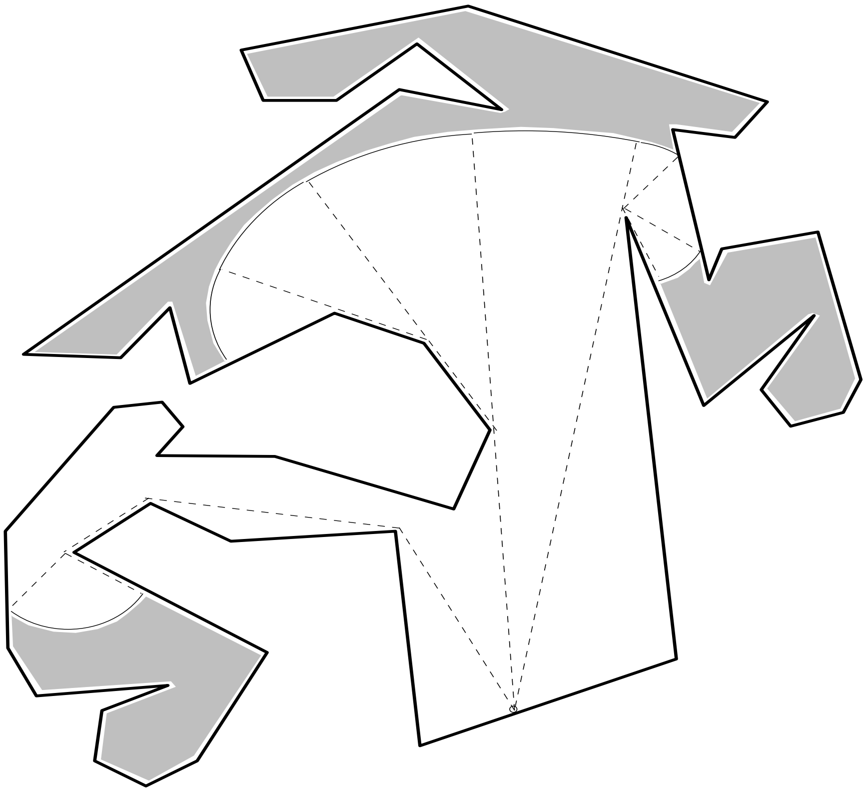


Figure 6: Illustrating grazing area in the interior of a simple polygon

In addition to the diagonals of the polygon given by the edges of the shortest path tree T, we introduce a few chords (called **extension chords**) as follows. (Note that a chord of a polygon is a line segment in its interior with end points on the boundary.) Each edge of the tree T forming the wall of the funnel is extended in the forward direction (parent to children) to meet the boundary of the polygon. The structure of the funnel guarantees that the extension chords formed as above meet the boundary of the polygon without intersecting any other tree edges or chords. Thus we have the following observation.

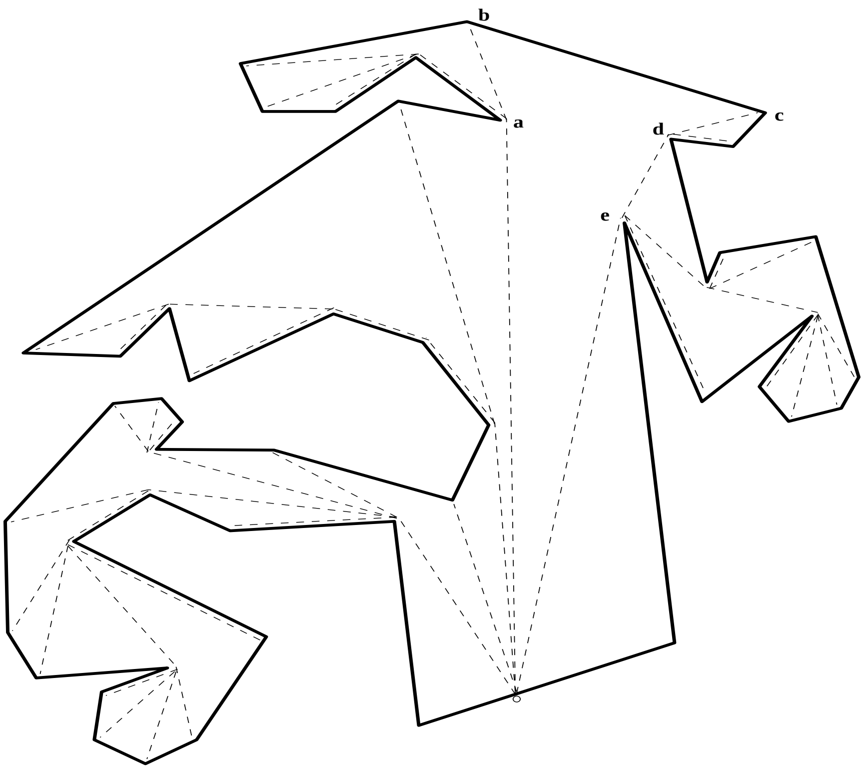


Figure 7: Illustrating the shortest path tree induced by an anchor point

Observation 2: The set of edges of the shortest path tree T and the set of extension chords partition the polygon into triangles. The partitioning thus achieved, which we denote by $R(Q)$, is referred to as the **triangulation induced by instance** $I(Q, L, o)$. Figure 8 illustrates an example, where extension chords are shown as solid lines.

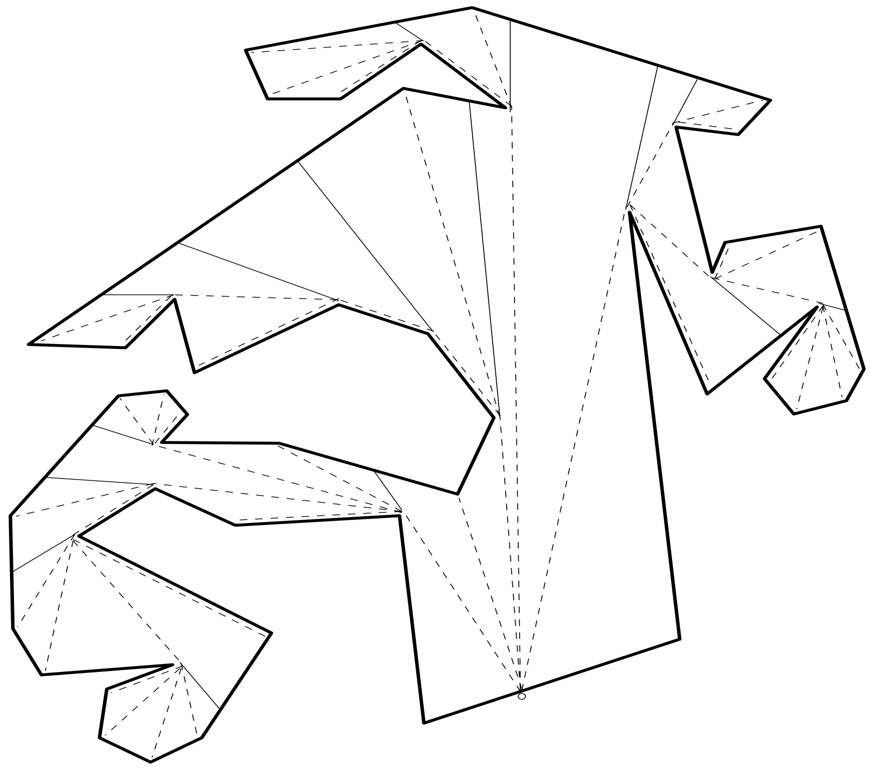


Figure 8: Triangulation obtained by extending the shortest path tree

Lemma 5: Triangulation $R(Q)$ contains $O(n)$ vertices.

Proof: Tree T contains $n + 1$ vertices (n vertices of Q and the anchor point o). Each chord formed by extending a funnel edge introduces only one steiner vertex. The total number of the edges of the polygon used for creating extension chords is no more than the total number of reflex vertices

r in the polygon. Hence the total number of vertices in $R(Q)$ is at most $n + r + 1$. $\square$

Let $t_1, t_2, ...t_k$ be the triangles of $R(Q)$. We refer to the vertex of the triangle t_i from which the diagonal/chord originates as its **tip vertex** f_i. Let d_i be the length of the shortest path from anchor point o to f_i.

Lemma 6: The set of points in triangle t_i that can be reached by a rope of length L anchored at o is the same as the set of points that can be reached by a rope of length $L - d_i$ anchored at f_i.

Proof: Let p be any point in t_i that can be reached by a rope of length L anchored at o. Let $SP(o, p)$ be the shortest path connecting o to p. Clearly the length of $SP(o, p)$ is no more than L; otherwise, the rope anchored at o would not reach p. Since f_i is a vertex of the shortest path tree T rooted at o, $SP(o, p)$ must pass through vertex f_i. Thus the distance between f_i and p is no more than $L - d_i$, which implies that p can be reached by a rope of length $L - d_i$ anchored at f_i. $\square$

Lemma 6 implies that the solution for the original instance $I(Q, L, o)$ of IGAP can be obtained by solving the straightforward instances, $I(t_i, L - d_i, f_i), 1 \leq i \leq k$. (certainly, we need to consider only those instances for which $L - d_i$ is positive.) A formal sketch of the algorithm follows.

Algorithm Internal Grazing

Input: A simple polygon Q, boundary point o in its interior, and rope length l.
Output: Grazing Area.
Step 1: Generate the shortest path tree T for Q rooted at o.
Step 2: Introduce chords by extending the edges of the walls of all funnels to obtain the triangulation $R(Q)$.
Step 3: Report the output by aggregating the solutions for instances, $I(t_i, L - d_i, f_i), 1 \leq i \leq k$, such that $L - d_i$ is positive.

Theorem 2: IGAP for simple polygons can be solved in $O(n)$ time.

Proof: Computation of the shortest path tree in Step 1 can be done in $O(n)$ time by using the balanced decomposition algorithm given in [8]. Once we have the shortest path tree T rooted at o , Step 2 can be done in $O(n)$ time by checking the intersection of the lines passing through the tree edges with the lids of the corresponding funnels. Within the same time complexity, shortest paths d_i's can be obtained from the shortest path tree. There are only $O(n)$ grazing area computations in Step 3, each of which can be solved in constant time; hence Step 3 also takes $O(n)$ time. □

The approach described above can be modified to solve the external grazing area problem (EGAP) for simple polygons in linear time by transforming an instance of $EGAP$ to at most three instances of $IGAP$. This is stated in the following theorem.

Theorem 3: EGAP for simple polygons can be solved in $O(n)$ time.

Proof: Let $I(Q, o, L)$ be an instance of $EGAP$. We enclose polygon Q by a large square U such that no boundary point of U is within distance L from any boundary point of Q. This can be easily done by making the diagonal of the square U sufficiently larger than the diameter of the convex hull of Q. The above condition implies that the end of the rope never touches the boundary of U.

We first consider the case when the anchor point o is on the convex hull of Q (Figure 9a). We construct an extremely narrow tunnel $xyx'y'$ near the anchor point o. By the introduction of the tunnel, we can view the region bounded by the tunnel and the boundaries of Q and U as the interior of the simple polygon W with ordered vertices $x, v_n, v_{n-1}, ..., v_2, v_1, x', y', d, a, b, c, y$, and x. Now, the solution for the $EGAP$ instance $I(Q, o, L)$ is given by the union of the solutions for $IGAP$ instances $I(W, x, L)$ and $I(W, x', L)$.

The other case is when the anchor point is inside the convex hull of Q. In this case the anchor point is in a cave polygon (denote it by K) formed by a convex hull edge and part of the boundary of Q (Figure 9b). Consider the shortest path tree rooted at o inside the cave polygon K. We extend the two edges of the tree incident at the convex hull vertices (labeled as g and h in Figure 9b) to meet the boundary of U at g' and h'. Let Z denote the polygon formed by the union of K and the polygon with ordered vertices g, h, h', d, and g'. Let Z' denote the polygonal region in the interior of U

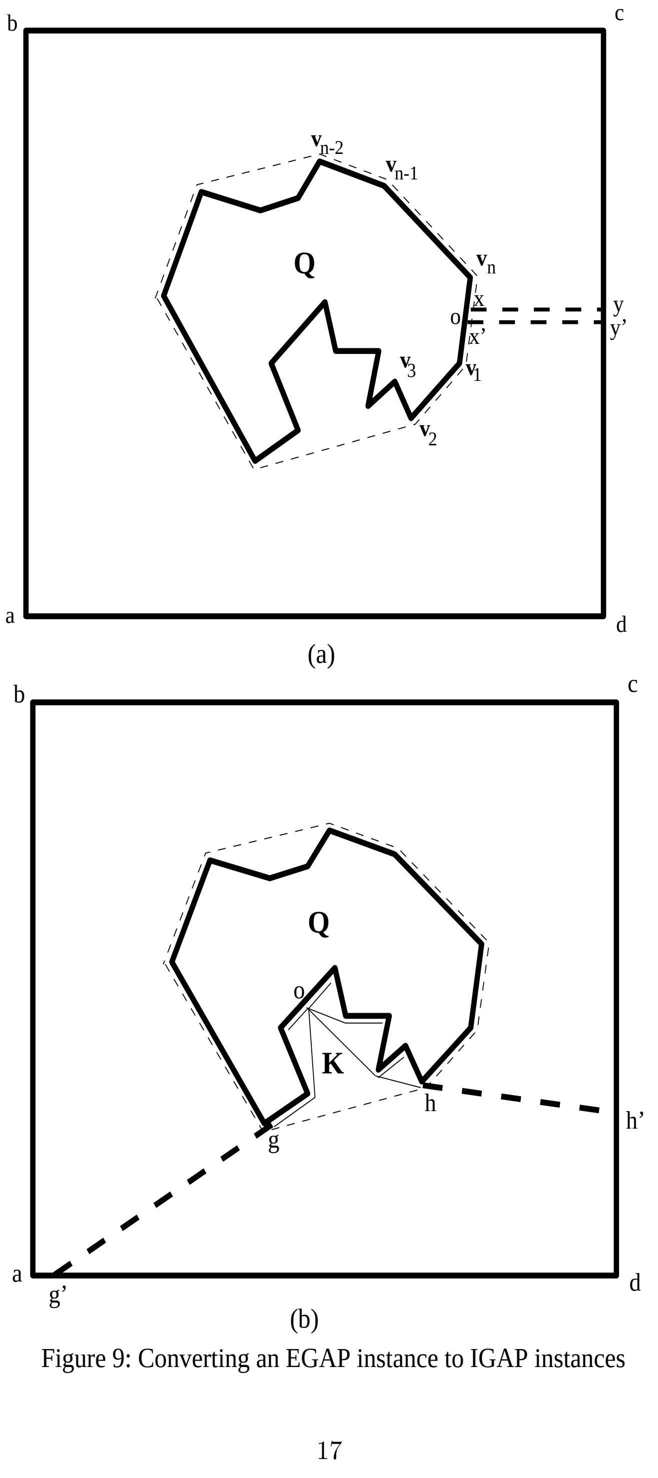


Figure 9: Converting an EGAP instance to IGAP instances

but to the exterior of the union of W and Q. Observe that the solution for the $EGAP$ instance $I(Q, o, L)$ is given by the union of the solutions for $IGAP$ instances $I(Z, o, L)$, $I(Z', g, L - s(g))$, and $I(Z', h, L - s(h))$, where $s(h)$ and $s(g)$ denote the lengths of shortest paths from o to h and g, respectively. □

V. Conclusions

We have presented an $O(n)$ time algorithm for computing the maximum simple grazing area in the presence of a convex polygonal obstacle; the algorithm is optimal within a constant factor. The algorithm can be easily modified to compute the minimum simple grazing area within the same complexity: the equations of the parabola corresponding to each $3n$ sub-edges of the polygon are examined to identify the candidate anchor points (one per sub-edge), and the minimum grazing area corresponds to one of these anchor points. We conjecture that Lemma 2 is valid even for non-simple grazing area. It is interesting to note the following observation for non-simple grazing area.

Observation 3: For the case of non-simple grazing area, when the anchor point moves along an edge, the meeting point moves along an ellipse. In Figure 5, points e, i, and j are on the ellipse whose focii are points a and n.

Observation 3, together with some other insight may be useful for developing a sub-quadratic time complexity algorithm to compute the maximum non-simple grazing area for convex polygons.

We have presented an $O(n)$ time algorithm to solve $IGAP$ for simple polygons. We have explained how to modify the algorithm to solve EGAP for simple polygons, without increasing its time complexity. It would be interesting to develop efficient algorithms to compute the grazing area inside a polygon with holes.

Acknowledgments: The authors gratefully acknowledge the stimulating technical discussions with Professor Simeon Ntafos. We are grateful to an anonymous referee for her/his comments and corrections which helped us to improve the paper. For this research, Laxmi P Gewali was partially supported by grants from DOE and CRAY Research Inc.

References

[1] T. C. Shermer, “Recent results in art galleries”, Proceedings of the IEEE, Vol. 80, No. 9, Sept. 1992.

[2] H. ElGindy and D. Avis, “A linear time algorithm for computing the visibility polygon from a point,” *Journal of Algorithms*, Vol. 2, 1981, pp. 186-197.

[3] S. Ntafos and M. Tsoukalas, “Optimum placement of guards”, *Third Canadian Conference on Computational Geometry*, Vancouver, British Columbia, (1991), 122-125. Also in “Information Sciences”, 1993.

[4] R. Motwani, A. Raghunathan and H. Saran, “Covering orthogonal polygons with star polygons: the perfect graph approach,” *Journal of Computer and Systems Sciences*, Vol. 40, 1990, pp. 19-48.

[5] L. Gewali, M. Keil, and S. Ntafos, “On covering orthogonal polygons with star-shaped polygons”, *Information Sciences*, Vol 65, 1992, pp. 45-63.

[6] S. Ntafos, “Watchman routes under limited visibility”, *Computational Geometry: Theory and Applications*, Vol. 1, No. 3, (1992) pp 149-170.

[7] B. Aronov, A. R. Davis, T. K. Dey, S. P. Pal, and D. C. Prasad, “Visibility with reflection”, *Proceedings of the eleventh acm symposium on computational geometry*, pp. 316-325, 1995.

[8] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan, “Linear time algorithms for visibility and shortest path problems inside triangulated simple polygon”, *Algorithmica*, Vol. 2, No. 2, 1987, pp. 209-233.

[9] R. Venkatasubramanian, “On the Area of Intersection Between Two Closed 2-D Objects”, *Information Sciences*, Vol. 82, (1995), pp. 25-44.