World Applied Sciences Journal 10(10): 1255-1263, 2010
ISSN 1818-4952

On the Number of Guards in Sculpture Garden Problem

[1]M. Eskandari, [2]A. Mohades and [3]B. Sadeghi Bigham

[1]Department of Mathematics, Alzahra University, Tehran, Iran
[2]Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran
[3]Institute for Advanced Studies in Basic Sciences, Zanjan, Iran

Abstract: In this paper, we consider the problem of placing a small number of angle guards inside a simple polygon P such that each point in the plane can determine if it is in or out of P from a monotone Boolean formula (and (.) and or (+) operations only) composed from the angle guards. Each angle guard views an infinite wedge of the plane. This class of new art gallery type problems is called sculpture garden problems. First we present an efficient algorithm for placing $n-[c/2]$ angle guards inside a simple n-gon P and show that this bound is tight, where c is the number of vertices of convex hull of P. Then we show that, for any polygon P, there is a set of $n-[c/2]-h$ angle guards that solve the sculpture garden problem for P, where h is the number of holes in P and show that this bound is tight, too.

Key words: Computational geometry · Art gallery · Sculpture garden problem · Wireless localization

INTRODUCTION

The main goal in classic art gallery problems is to place a small number of point guards inside a polygon P so that this set of guards can see all of P. Sometimes such problems involve angle guards and study the number of angel guards that are sufficient to see P. An angle guard g views an infinite wedge of the plane and can be defined as a Boolean predicate, $B_g(.)$ such that for a point p in the plane, $B_g(p)$ is true if p is inside the angle associated with g and is false otherwise. Given a polygon P, sculpture garden problem is interested in a placement of angle guards in, around and outside P in such a way that a monotone Boolean formula, $F(p)$, built from the angle-guard predicates, $B_g(p)$ can be obtained, so that $F(p)$ is true if and only if p is inside P. Moreover, we desire that the number of angle guards needed to define such a formula be small. This problem can be viewed as a kind of art gallery problem, where it is not sufficient that the guards merely see all of the art gallery, but instead they must collectively define the geometry of the art gallery.

In [1], Eppstein *et al.* showed there is a polygon P such that a "natural" angle-guard vertex placement cannot fully distinguish between points on the inside and outside of P, which implies that Steiner-point guards are sometimes necessary. And showed that, for any polygon P, there is a set of $n+2(h-1)$ angle guards that solve the sculpture garden problem for P, where h is the number of holes in P (so a simple polygon can be defined by $n-2$ guards). In addition, they proved that, for any orthogonal or convex polygon P, the sculpture garden problem can be solved using $[n/2]$ angle guards.

Motivation for sculpture garden problems comes, for example, from localization problems in wireless mobile computing (e.g., see [2]), where we wish to determine with certainty the position of a wireless device in a geometric environment. Sculpture garden problems could be used, for example, in a localization problem where we are asked to deploy a collection of locators in what can be viewed as a two-dimensional space so that a wireless device can prove that it belongs to a given polygonal environment. In this case, the locators would be simple, fixed base stations that can each broadcast information inside a certain angle. Such guards could be realized physically using angular RF antennas, by IR transmitters with angular shields, or even visible/LASER light transmitters with angular shields. Localization is becoming an important topic in wireless mobile computing. For example, Bulusu *et al.* [3] study how RF strength and angle can be used for sensor localization and Savvides *et al.* [4] show how to improve the consistency of such an approach by iterative algorithms.

On the other hand, "prison yard" problems [5-8] seek a set of guards that can simultaneously see both the interior and exterior of a simple polygon, in which case $[n/2]$ guards are sufficient and sometimes necessary [5].

Corresponding Author: M. Eskandari, Department of Mathematics , Alzahra University, Tehran, Iran.

Relating to angle guards, Estivill-Castro *et al*. [9] show that vertex angle guards with angles of 180 are sufficient to see any simple polygon and there are polygons such that any fixed angle less than this will not. Likewise, Steiger and Streinu [10] and Bose *et al*. [11] study the complexity of illuminating wedges with angle-restricted floodlights placed at a fixed set of points. Another motivating application comes from constructive solid geometry (CSG), where we wish to construct a geometric shape from simple combinations of simple primitive shapes. Dobkin *et al*. [12] describe a method for constructing a formula F that defines a simple polygon using primitives that are halfplanes defined by lines through polygonal edges, so that each halfplane is used exactly once. More recently, Walker and Snoeyink [13] study the problem of using polygonal CSG representations, a la Dobkin *et al*. [12], for performing point-in-polygon tests. They experimentally consider several interesting heuristics for improving the efficiency of such tests, by "flattening" the CSG tree defined by the formula.

In this paper, we improve the large upper bound $n+2(h-1)$ for an arbitrary n-gon with h holes by presenting five algorithms for placing guards and obtain a tight bound n–,[c/2]–h where c is the number of vertices of convex hull of P.

Preliminaries: An angle guard G with angle $\alpha \in (0,360)$ is a pair (a, w_α) of a point a and an infinitive wedge w_α of aperture α at apex a and views w_α. It can be viewed as a Boolean predicate, $B_G(.)$ such that for a point p in the plane, $B_G(p)$ is true if p is inside the angle associated with G and is false otherwise:

$$B_G(p) = \begin{cases} True & p \in w_\alpha \\ False & p \notin w_\alpha \end{cases}$$

For simplification, we use G instead of $B_G(p)$. Given a polygon P with n vertices, we want to allocate a small number of angle guards with arbitrary angles at points of P so that a monotone[1] Boolean formula, $F(p)$, built from the angle-guard predicates, $B_G(p)$ yields and we have:

$$F(p) = \begin{cases} True & p \in Int(P) \\ False & p \notin Int(P) \end{cases}$$

Where *int(P)* is interior of polygon P. If $F(.)$ is a solution of sculpture garden problem for P, we said P is defined by F. For example the pentagon in figure 1.a is determined by the formula $A.B.C$ and the 12-vertex polygon in figure 1.b is determined by $A.B.F + D.C.E + D.C.F$.

As defined in [1], a natural angle-guard vertex placement is one where we place each angle guard at a vertex of the polygon with the angle of that vertex as the angle of the guard. They showed that for general polygons, natural angle guards do not suffice. Even one at every vertex of the pentagon shown in figure 2.a fails to distinguish between points x and y. Figure 2.b shows that three guards suffice for this pentagon, but one, D, broadcasts both inside and outside P. This implies that the sculpture garden problem cannot be solved using natural guards for some polygons and must use Steiner points or Steiner angles.

Now we want to show that the sculpture garden problem can be solved for any simple polygon by at most n–[n/2] guards which is tight. To prove this bound we need to establish some definitions and preliminary results.

Definition 1: The complement of an angle guard $G = (a,w_a)$ is an angle guard G' at point a with angle $2\pi - \alpha$ such that the wedge associated with G' is the complement of w_α in plane.

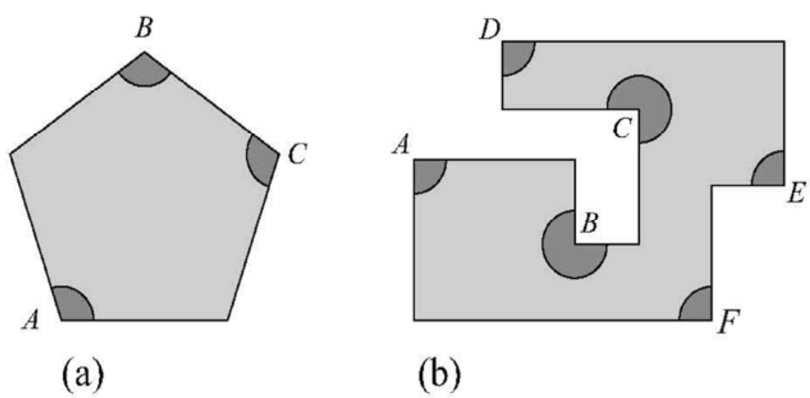

Fig. 1: (a) Convex 5-gon; (b) Orthogonal polygon, n = 12

[1]A Boolean formula is monotone if it contains only AND (.) and OR (+) operators; hence, has no NOT operations.

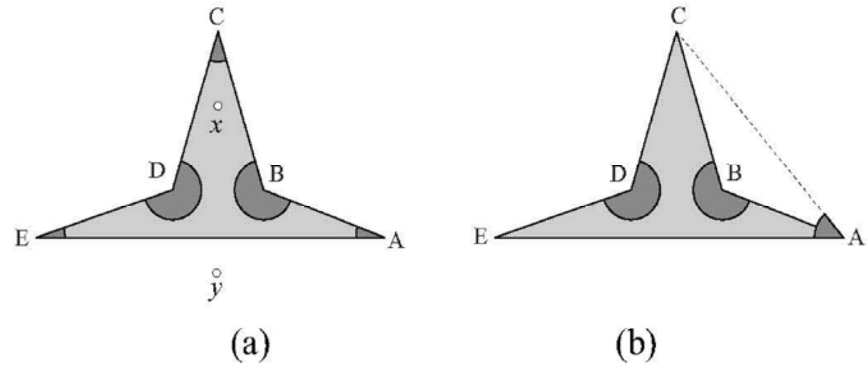

Fig. 2: (a) natural angle guards do not suffice; (b) Coverage by three guards; formula A.B.D.

Definition 2: If formula F is a solution of sculpture garden problem for polygon P, the complement of solution F which is denoted by F' is defined as follows: first replace any angle guard G by its complement G', then change (.) operators to (+) and (+) operators to (.).
I
Proposition 3: f P is defined by F, then F' defines the exterior of P.

Proof: By Jordan curve theorem, P as a simple closed plane curve divides the plane into two components, the interior and exterior of P. Note that $F' = NOT(F)$. If x is an exterior point of P, $F(x)=False$ and then $F'(x)=True$ and x is an interior point of the area outside P. If x is an interior point of P, $F(x)=True$ and then $F'(x)=False$ and x is an exterior point of the area outside P. This means F' is a solution of sculpture garden problem for the area outside of P.

Definition 4: The pockets of a simple polygon are the areas outside the polygon, but inside its convex hull.

Lemma 5: Suppose that P is a simple polygon and Q is a simple polygon inside $CH(P)$ whose vertices are vertices of P and it contains P entirely. The area outside P but inside Q are subpolygons of Q which are denoted by $Q_1, Q_2, ..., Q_k$. Let t_i be the number of vertices of Q_i and q be the number of vertices of Q. We have:

$$\sum_{i=1}^{k} t_i - 2k + q = n$$

Proof: The edges of P which do not belong to Q are counted in $\sum_{i=1}^{k} t_i - 1$ once. On the other hand the number of edges of P which belong to Q is q-k. See figure 3.

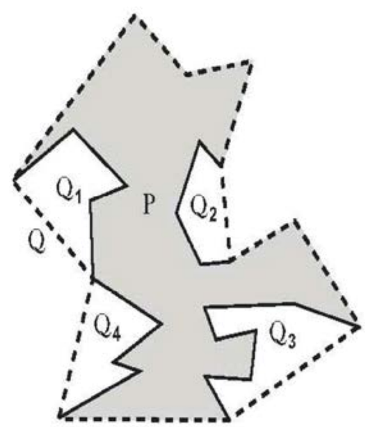

Fig. 3: Illustration of lemma 5.

So the number of edges of P which do not belong to Q is n-(q-k). So we have:

$$\sum_{i=1}^{k} t_i - k = n - q + k$$

Corollary 6: Suppose that P is a simple polygon with n vertices and $P_1, P_2, ..., P_m$, are its pockets. Let n_i be the number of vertices of P_i and c be the number of vertices of $CH(P)$. We have:

$$\sum_{i=1}^{m} n_i - 2m + c = n$$

A Tight Upper Bound for Simple Polygons: In this section we show that the sculpture garden problem can be solved for any simple polygon by at most n–[n/2] guards. To prove this bound we present four algorithms to place a small number of guards and obtain a formula F to define four classes of simple polygons. The first class is the polygons whose convex hulls have at least six vertices,

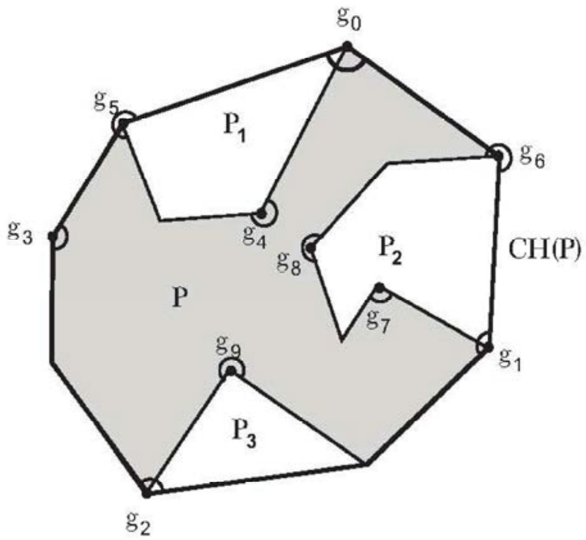

Fig. 4: Illustration of algorithm 1: F = (g_0, g_1, g_2, g_3) . (g_4+g_5) . (g_6 + g_7 + g_8). g_9

the second is the polygons whose convex hulls have exactly five vertices, the third is the polygons whose convex hulls have exactly four vertices and the fourth is the polygons whose convex hulls have exactly three vertices.

Algorithm 1: First class, *c>5*:

Step 1: Find convex hull of *P*, *CH(P)*. Let $v_0,v_1,...,v_{c-1}$, be the vertices of *CH(P),* clockwise. Place [*n*/2] natural angle guards on vertices $v_0,v_2,...,v_{2[c/2]}$ and define *CH(P)* by formula F_c $v_0,v_2,...,v_{2[c/2]}$ [5].

Step 2: Denote the pockets of *P* by $P_1,P_2,...,P_m$ clockwise. If *m=0*, go to step 3. If P_i is a triangle, denote the vertex of P_i which is not on *CH(P)* by *u.* Place a natural angle guard G_i at *u* as a vertex of *P* and let $F_i = G_i$. Otherwise, find a solution to define P_i by using one of algorithms 1 (recursively), 2, 3 or 4 and obtain a formula f_i for P_i. Then find the complement of above solution and place these complement angle guards at vertices of *P* and let $F_i = f'_i$.

Step 3: Let $F = F_c.F_1.F_2.....F_m$. See figure 4 for illustration of algorithm 1.

Algorithm 2: Second class, *c=5:*
Step 1: Find convex hull of *P, CH(P).* Let $v_0,v_1,...,v_4$ be the vertices of *CH(P),* clockwise.

Step 2: Denote the pockets of P by $P_1,P_2,...,P_m$, clockwise.

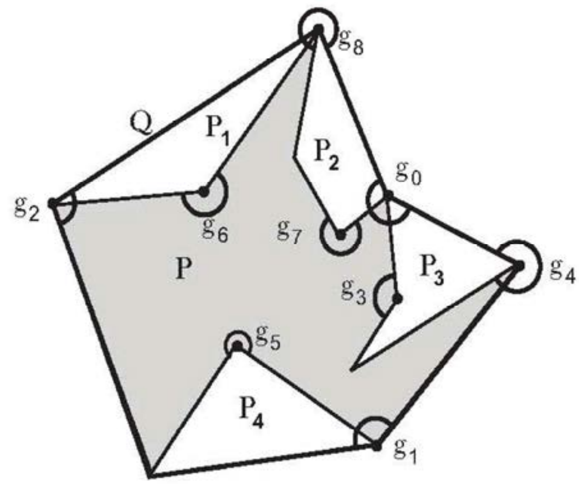

Fig. 5: Illustration of algorithm 2: F = (g_0, g_1, g_2) . (g_3+g_4) . (g_7+ g_8). g_5,g_6

Step 3: If *m=0*, then *P* is a convex pentagon *ABCDE*. Place three natural angle guards at *A, C* and *E* and let *F=A.C.E* and exit.

Step 4: Otherwise, in an arbitrary pocket P_1 of *P*, denote the common edge between *CH(P)* and P_1 by $e = v_i\ v_{i+1}$ and denote the closest vertex of P_1 to *e* by *s*. Add the line segments v_is and sv_{i+1} to *CH(P)* and remove e from it and denote this new polygon by *Q*. *Q* is a hexagon with only one reflex vertex. See figure 5. *P*lace a natural angle guard A at this reflex vertex as a vertex of *Q* and two natural guards *B* and *C* at v_{i-1} and v_{i+2} as vertices of *Q* and let F_c = *A.B.C*.

Step 5: Let P_0 be the areas outside *P*, but inside *Q* which contains edge v_is and P_1 be the areas outside *P*, but inside *Q* which contains edge sv_{i+1}. If P_0 or P_1 is a line segment F_i = *True*. If P_i is a triangle, denote the vertex of P_i which is not on *Q* by *u*. *P*lace a natural angle guard G_i at *u* as a vertex of *P* and let $F_i = G_i$. Otherwise, find a solution to define P_i by using one of algorithms 1, 2 (recursively), 3 or 4 and obtain a formula f_i for P_i. Then find the complement of above solution and place these complement angle guards at vertices of *P* and let $F_i = f'_i$. Let $F = F_c.F_0.F_1.....F_m$.

See figure 5 for illustration of algorithm 2.

Algorithm 3: Third class, c=4:

Step 1: Find convex hull of *P*, *CH(P)*. Let $v_0,v_1,...,v_3$ be the vertices of *CH(P)*, clockwise.

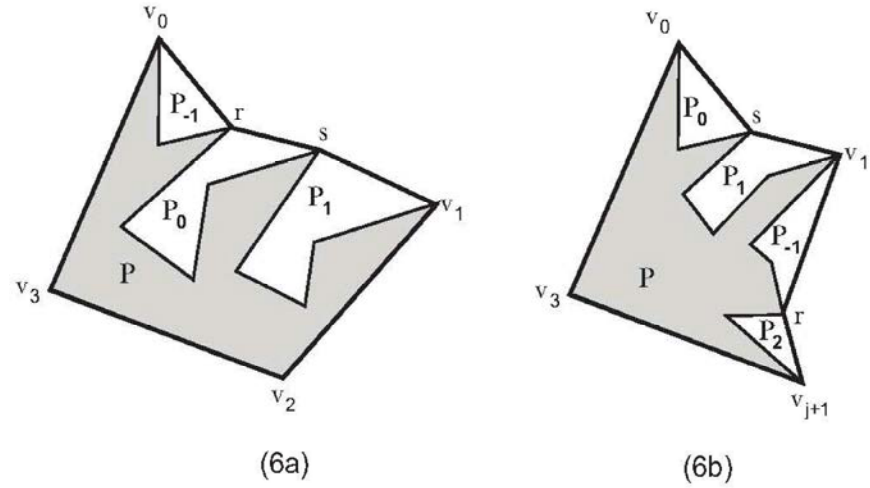

Fig. 6: Illustration of algorithm 3,

Step 2: Denote the pockets of P by $P_1,P_2,...,P_m$, clockwise.

Step 3: If $m=0$, then P is a convex tetragon *ABCD,* place two natural angle guards at two opposite corners, A and C, let $F=A.C$ and exit.

Step 4: If $m=1$ and P_1 is a triangular pocket, P is a pentagon with only one reflex vertex. *P*lace a natural angle guard A at this reflex vertex as a vertex of P and two natural guards B and C at two opposite corners of *CH(P)*; let $F=A.B.C$ and exit.

Step 5: Otherwise ($m=1$ and P_1 is not triangular), denote the common edge between *CH(P)* and P_1 by $v = v_0v_1$ and denote the closest vertex of P_1 to e by s. Add the line segments v_0s and sv_1 to *CH(P)* and remove e from it and denote this new polygon by Q. Let P_0 be the areas outside P, but inside Q which contains edge v_0s and P_1 be the areas outside P, but inside Q which contains edge sv_1.One of P_0 or P_1 is not a line segment, e.g., P_1. Denote the closest vertex of P_0 to edge v_0s by r. Add the line segments v_0r and rs to Q and remove edge v_0s from it. Let P_{-1} be the areas outside P, but inside Q which contains edge v_0r and P_0 be the areas outside P, but inside Q which contains edge rs. *P*lace three natural angle guards *A, B* and C at v_0, s and v_2 as vertices of Q and let $F_c = A.B.C$. See figure 6a.

Step 6: Otherwise, we have $m\neq 1$. Denote the common edge between *CH(P)* and P_1 by $e = v_0v_1$ and denote the closest vertex of P_1 to e by s. Add the line segments v_0s and sv_1 to *CH(P)* and remove e from it and denote this new polygon by Q. Let P_0 be the areas outside P, but inside Q which contains edge v_0s and P_1 be the areas outside P, but inside Q which contains edge sv_1. Denote the common edge between Q and P_2 by $d = v_jv_{j+1}$ and denote the closest vertex of P_2 to d by r. Add the line segments v_jr and rv_{j+1} to Q and remove d from it. Let P_{-1} be the areas outside P, but inside Q which contains edge v_jr and P_2 be the areas outside P, but inside Q which contains edge rv_{j+1}. See figure 6b. If $j=1$, place three natural angle guards *A, B* and C at *r, s* and v_3 as vertices of Q and let $F_c = A.B.C$. If $j\neq 1$, place three natural angle guards *A, B* and C at r, v_0 and v_1 as vertices of Q and let $F_c = A.B.C$.

Step 7: If P_i is a line segment, let F_i = *True*. If P_i is a triangle, denote the vertex of P_i which is not on Q by u. *P*lace a natural angle guard G_i at u as a vertex of P and let $F_i = G_i$. Otherwise, find a solution to define P_i by using algorithm 1, 2, 3 (recursively) or 4 and obtain a formula f_i for P_i. Then find the complement of above solution and place these complement angle guards at vertices of P and let $F_i = f'_i$. Let $F = F_c.F_{-1}.F_0.F_1.....F_m$.

Algorithm 4: Third class, *c=3:*

Step 1: Find convex hull of P, *CH(P)*. Let v_0,v_1,v_2, be the vertices of *CH(P)*, clockwise.

Step 2: Denote the pockets of P by $P_1.P_2,...,P_m$, clockwise.

Step 3: If $m=0$, then P is a triangle, place two natural angle guards A and B at its two vertices and let $F=A.B$ and exit.

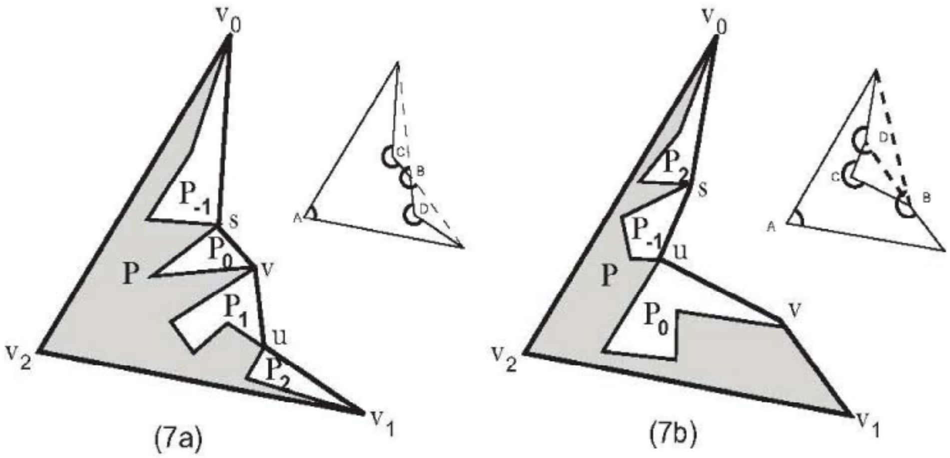

Fig. 7: Illustration of step 4-2 of algorithm 4,

Step 4: If $m=1$ and P_1 is a triangular pocket, P is a tetragon with one reflex vertex. *P*lace a natural angle guard A at this reflex vertex as a vertex of P and a natural guard B at the opposite corner; let $F=A.B$ and exit.

If P_1 is a tetragonal pocket, P is a pentagon with at least one reflex vertex. Denote the common edge between $CH(P)$ and P_1 by $e = v_0v_1$ and the closest vertex of P to e by v. *P*lace two natural angle guards A and B at vertices v_2 and v as vertices of tetragon $v_0v_1v_2$ and place a natural angle guard C at the fifth vertex of P; let $F=A.B.C$ and exit.

If P_1 has more than four vertices, denote the common edge between $CH(P)$ and P_1 by $e = v_0v_1$ and the closest vertex of P to e by v. Add the line segments v_0v and vv_1 to $CH(P)$ and remove e from it and denote the new polygon by Q. Let P_0 be the areas outside P, but inside Q which contains edge v_0v and P_1 be the areas outside P, but inside Q which contains edge vv_1. By above assumption one of P_0 or P_1 has at least three vertices, e.g., P_1. Denote the closest vertex of P_1 to edge vv_1 by u. Add the line segments vu and uv_1 to Q and remove edge vv_1 from it. Let P_1 be the areas outside P, but inside Q which contains edge vu and P_2 be the areas outside P, but inside Q which contains edge uv_1. By above assumption one of P_0 or P_1 or P_2 has at least three vertices, e.g., P_0. Denote the closest vertex of P_0 to edge v_0v by s. Add the line segments v_0s and sv to Q and remove edge v_0v from it. Let P_{-1} be the areas outside P, but inside Q which contains edge v_0s and P_0 be the areas outside P, but inside Q which contains edge sv. See figure 7a. *P*lace four angle guards A, B, C and D as shown in figure 7 and let $F_c = A.B.D$.

Step 5: For $m=2$, if both of P_1 and P_2 are triangular pockets, P is a pentagon with two reflex angles. *P*lace two natural angle guards A and B at these reflex angles as vertices of P and a natural angle guard C at v_2 as a vertex of $CH(P)$; Let $F=A.B.C$ and exit.

Otherwise, denote the common edge between $CH(P)$ and P_i by $e_i = v_{i-1}s_i$, where $i=1,2$. And denote the closest vertex of P_i to e_i by s_i. Clearly one of P_1 or P_2 is not triangular, e.g., P_1. For $i=1,2$, add the line segments $v_{i-1}s_i$ and v_is_i to $CH(P)$ and remove e_i from it to obtain pentagon Q. Let P_2 be the areas outside P, but inside Q which contains edge v_1s_2 and P_{-2} be the areas outside P, but inside Q which contains edge v_2s_2. Let P_0 be the areas outside P, but inside Q which contains edge v_0s_1 and P_1 be the areas outside P, but inside Q which contains edge v_1s_1. One of P_0 or P_1 has more than three edge, e.g., P_0. Denote the closest vertex of P_0 to v_0s_1 by u. Add the line segments v_0u and us_1 to Q and remove v_0s_1 from it to obtain hexagon Q. Let P_{-1} be the areas outside P, but inside Q which contains edge v_0u and P_0 be the areas outside P, but inside Q which contains edge us_1. See figure 8. *P*lace four angle guards A, B, C and D as shown in figure 8 and let $F_c = A.B.C$.

Step 6: If $m=3$, denote the common edge between $CH(P)$ and P_i by $e_i = v_{i-1}v_i$, where $i=1,2,3$. And denote the closest vertex of P_i to e_i by s_i. Add the line segments $v_{i-1}s_i$ and v_is_i to $CH(P)$ and remove e_i from it to obtain hexagon Q. *P*lace three natural angle guards A, B and C at s_1, s_2 and s_3 as vertices of Q and let $F_c = A.B.C$. Let P_i be the areas outside P, but inside Q which contains edge $v_{i-1}s_i$ and P_{-i} be the areas outside P, but inside Q which contains edge v_is_i.

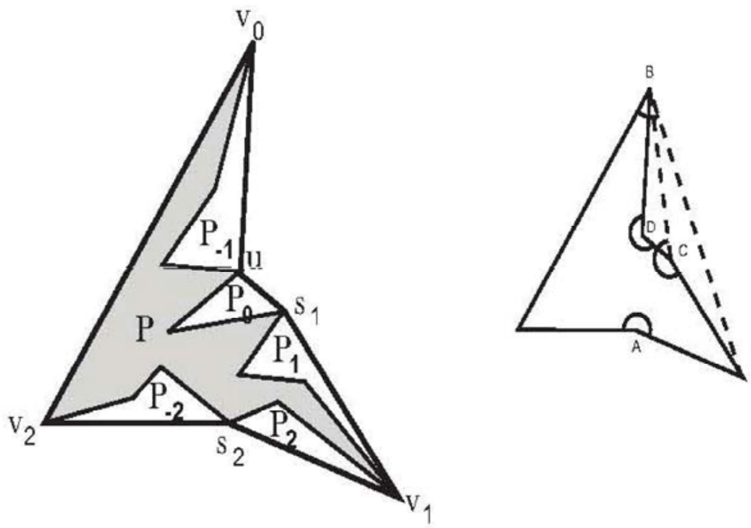

Fig. 8: Illustration of step 5-1 of algorithm 4,

Step 7: If P_i is a line segment or does not exist F_i = *True*. If P_i is a triangle, denote the vertex of P_i which is not on Q by u. *P*lace a natural angle guard G_i at u as a vertex of P and let $F_i = G_i$. Otherwise, find a solution to define P_i by using algorithm 1, 2, 3 or 4 (recursively) and obtain a formula f_i for P_i. Then find the complement of above solution and place these complement angle guards at vertices of P and let $F_i = f'_i$. Let $P_i = F_c F_{-3}.F_{-2}.F_{-1}.F_0.F_1.....F_m$

Now we will show that the above algorithms solve the sculpture garden problem for any simple polygon by at most n–[$n/2$] guards. To prove this bound first we need to show that the obtained formula F define the inside of P correctly and then we show that the number of the angle guards is at most n–[$n/2$] for the first class and at most *n-2* for the other classes.

Theorem 7: The obtained formula from above algorithms, F, is a solution of sculpture garden problem for a given simple polygon P.

Proof: We prove this theorem only for algorithm 1; it is similar for the other algorithms. First note that in step 1 of algorithm 1, we place [$c/2$] guards to define *CH(P)* which is proved in [5]. Now for a given point $x \in P$, we have $F_c(x)$ = *True* and $F_i(x)$ = *True* then *F(x)=True*. For a given point $x \in P$, if x is not inside of *CH(P)*, $F_c(x)$ = *False* then *F(x)=False*. If $x \in CH(P)$ so it must be in a pocket like P_i, so $F_i(x)$ then *F(x)=False*. Therefore F defines P.

Theorem 8: For a given simple n-gon P, the number of the angle guards used for defining P, in algorithm 1, is at most n–[$c/2$] and the number of the angle guards used for defining P, in the other algorithms, is at most *n-2*.

Proof: We prove this theorem by induction on n, the number of vertices of P. Theorem is true for $n=3$, because for defining a triangle we place two guards in step 3 of algorithm 4. For an arbitrary simple polygon P, we suppose that the theorem is hold for smaller polygons. We establish four cases:

c>5: In step 1 of algorithm 1, we place [$n/2$] guards to define *CH(P)*. In step 2, we place at most $n_i - 2$ guards to define P_i. Because if $CH(P_i)$ has more than five vertices, by induction we use n–[$c_i/2$] guards which is less than n_i–2, where c_i is the number of vertices of $CH(P_i)$; if $CH(P_i)$ has less than six vertices, by induction we use n_i–2; and for the triangular pockets we use only one guard. Therefore, by corollary 6, the number of guards is at most:

$$\sum_{i=1}^{m}(n_i-2)+\lceil c/2 \rceil = n - \lfloor c/2 \rfloor$$

c=5: In step 3 of algorithm 2, we place three guards to define a convex pentagon. In step 4 we use three guards to define hexagon Q and in step 5, we place at most n_i–2 guards to define P_i (similar the above case). Therefore, by lemma 5, the number of guards is at most:

$$\sum_{i=1}^{m}(n_i-2)+3 = n-6+3 \le n-2$$

c=4: In step 3 of algorithm 3, we place two guards to define a convex tetragon. In step 4, we place three guards to define a pentagon. In steps 5 and 6 we use three guards to define hexagon Q and in step 7, we place at most n_i–2 guards to define P_i. Therefore, by lemma 5, the number of guards is at most:

$$\sum_{i=1}^{m}(n_i-2)+3=n-6+3\le n-2$$

c=3: In step 4 of algorithm 4, we place two guards to define a nonconvex tetragon. In steps 4-1 and 5, we place three guards to define a pentagon. In steps 4-2 and 5, we use four guards to define hexagon Q and in step 7, we place at most n_i–2 guards to define P_i. Therefore, by lemma 5, the number of guards is at most:

$$\sum_{i=1}^{m}(n_i-2)+4=n-6+4=n-2$$

In step 6, we use three guards to define hexagon Q and in step 7, we place at most n_i–2 guards to define P_i. Therefore, by lemma 5, the number of guards is at most:

$$\sum_{i=1}^{m}(n_i-2)+3=n-6+3\le n-2$$

Theorem 9: The upper bound n–[c/2] for number of angle guards in sculpture garden problem is tight.

Proof: As shown in [5] any convex n-gon needs [n/2] guards, so the bound n–[c/2] is achieved by convex polygons and is tight.

A Tight Upper Bound For Polygons With Holes: In this section we show that the sculpture garden problem can be solved for an arbitrary polygon with h holes by at most n–[c/2] –2h+t guards, where t is the number of triangular holes and find a tight upper bound.

Algorithm 5: Polygons with holes:

Step 1: Denote the holes of P by $H_1,H_2,...H_h$. Remove holes from P temporary to obtain new polygon R, then by using algorithms 1,2,3 and 4, find a solution f for R.

Step 2: Every H_i is supposed to be a simple polygon whose interior lies outside of P. By using algorithms 1,2,3 and 4, find a solution f_i for this simple polygon H_i, $1\le i\le h$. Let $F_i = f'_i$.

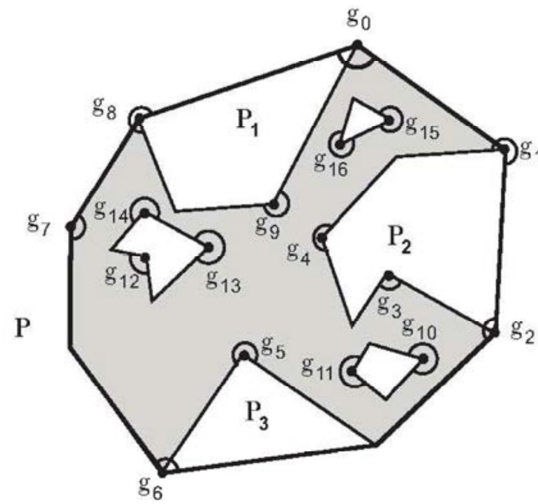

Fig. 9: Illustration of algorithm 5,

Step 3: Let $F = f.F_1.F_2.....F_h$.

See figure 9 for illustration, the obtained formula is

$$F = (g_0.g_2.g_6.g_8).(g_8 + g_9).(g_1 + g_3 + g_4).g_5.(g_{10} + g_{11}).(g_{12} + g_{13} + g_{14}).(g_{15} + g16).$$

Theorem 10. The obtained formula from algorithm 5, F, is a solution of sculpture garden problem for a given polygon P with h holes.

Proof: For a given point $x \in P$, we have $x \in R$ and $x \notin H_i$, so $f(x)=True$ and $F_i(x)=True$ then $F(x)=True$. For a given point $x \notin P$, if x is not inside of R, $f(x)=False$ then $F(x)=False$. If $x \in R$ so it must be in a hole like H_i, so $F_i(x)$ then F(x)=False. Therefore F defines P.

Theorem 11: For a given n-gon P with h holes, the number of the angle guards used for defining P, in algorithm 5, is at most n–[c/2]–2h+t, where t is the number of triangular holes.

Proof: Denote the vertices of H_i by m_i. In step 1 of algorithm 5, we use $n-(\sum_{i=1}^{h}m_i)-\lceil c/2\rceil$ guards, by theorem 8. If the number of triangular holes is t, in step 2, we use $n-\sum_{i=1}^{h}(m_i-2)+t$ guards, because one guard is added to $n-\sum_{i=1}^{h}(m_i-2)$ for every triangular hole. Therefore the number of guards is at most n–[c/2]–2h+t.

Theorem 12: The upper bound n–[c/2]–h angle guards in sculpture garden problem for polygons with h holes is tight.

Proof: A convex polygon whose holes are all triangular achieves this bound.

CONCLUSION

In this paper, we study the sculpture garden problem for placing angle guards to define inside of a polygon *P*. We presented five algorithms concerning the kinds and number of guards needed to define various polygons. We obtained a tight bound $n-[c/2]$ for simple polygons and tight bound $n-[c/2]-h$ for polygons with *h* holes, where *c* is the number of vertices of *CH(P).*

REFERENCES

1. Bose, P., L.J. Guibas, A. Lubiw, M.H. Overmars, D.L. Souvaine and J. Urrutia, 1997. The floodlight problem. International J. Computational Geometry and Applications, 7(1/2): 153-163.
2. Bulusu, N., J. Heidemann and D. Estrin, 2000. Gps-less low cost outdoor localization for very small devices.
3. Cho, Y., L. Bao and M. Goodrich, 2005. Protocols for Region Locality Among Mobile Devices. Technical Report.
4. Dobkin, D.P., L. Guibas, J. Hershberger and J. Snoeyink, 1993. An efficient algorithm for finding the CSG representation of a simple polygon. Algorithmica, 10: 1-23.
5. Eppstein, D., M.T. Goodrich and N. Sitchinava, 2007. Guard Placement For Efficient Point-in-Polygon Proofs. SOCG, 457-467.
6. Estivill-Castro, V., J. O'Rourke, J. Urrutia and D. Xu, 1995. Illumination of polygons with vertex lights. Information Processing Letters, 56: 9-13.
7. Furedi, Z. and D. Kleitman, 1994. The prison yard problem. Combinatorica, 14: 287-300.
8. O'Rourke, J., 1987. *Art Gallery Theorems and Algorithms*. Number 3 in Int. Ser. Monographs on Computer Science. Oxford Univ. Press.
9. O'Rourke. J., 1997. Visibility. In J. E. Goodman and J. O'Rourke, editors, Handbook of Discrete and Computational Geometry, chapter, 25: 467-480. CRC Press LLC, Boca Raton, FL.
10. Savvides, A., C.C. Han and M.B. Strivastava, 2001. Dynamic fine-grained localization in ad-hoc networks of sensors. In Mobile Computing and Networkin], pp: 166-179.
11. Steiger, W. and I. Streinu, 1998. Illumination by floodlights. Computational Geometry. Theory and Applications, 10(1): 57-70.
12. Urrutia. J., 2000. Art gallery and illumination problems. In J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry, pp: 973-1027. North-Holland.
13. Walker, R.J. and J. Snoeyink., 1999. Practical point-in-polygon tests using CSG representations of polygons. In M. T. Goodrich and C. C. McGeoch, editors, *Algorithm Engineering and Experimentation (Proc. ALENEX '99), volume 1619 of Lecture Notes Comput. Sci*, pages 114-123. Springer-Verlag.