Proposing a New Heuristic Algorithm to Solve Minimum-Vertex Guard in Art-Gallery Problem

Mohammad Reza Nami
Department of Electrical, Computer and IT Engineering, Islamic Azad University-Qazvin Branch, Qazvin, Iran.
Email: nami1352@gmail.com
Received: August 2012 Revised: October 2012 Accepted: January 2013

ABSTRACT
Finding minimum vertex guard to cover an art gallery is one of outstanding open problems in computational geometry. In this problem, a given polygonal art gallery is given. The aim is to find minimum vertex guard to cover it. This is an NP-hard problem. The purpose of this paper is to propose a heuristic algorithm that finds minimum number of vertex guard, who is put on the vertex of polygon. This algorithm has been implemented with C#. An arbitrary polygon with $n$ vertices is randomly developed. Computational result of the proposed algorithm shows that the average number of vertex guard needed to cover a polygon with $n$ vertices is $n/6.48$. This result is better than other algorithms developed for this problem. For this, we finally compare the results of our heuristic algorithm with the result of genetic algorithm and well-known art-gallery theorem.

KEYWORDS: Art-Gallery Problem, Minimum Guard, Polygon Covering.

1. INTRODUCTION
The art gallery theorem was stated and proved by Vasek Chvátal [1] in 1975 in response to a query from VictorKlee. Chvatal proved that at most $\lfloor n/3 \rfloor$ guards is needed to cover an art-gallery imagined as a polygon. V. Kelli in 1973 posed that how many stationary guards are needed to guard an art gallery room with $n$ walls [2].

Let $x$ be any point in an art gallery. The point $y$ is visible to $x$ provided the line segment joining $x$ and $y$ does not exit the gallery. Meanwhile, we also assume that every point is visible to itself. The segment represents the sight line of a guard. A guard is considered to be a fixed vertex in polygon $P$ with visibility range $2\pi$. A set of guards covers $P$ if each vertex of $P$ is visible by at least one guard. Thus, the art gallery problem deals with setting a minimum number of guards in a polygonal art gallery so that a set of guards see all vertex of this polygon al art gallery so that a set of guards see all vertex of this polygon. Different kinds of this problem are introduced in computational geometry. Some of papers are consider cameras as guards. Some of other papers are discussed on four kinds of guards: point guards who place anywhere in the polygon, vertex guards who place on vertices of the polygon, edge guards who are allowed to patrol along an edge of the polygon, and mobile guards who are allowed to patrol along a segment lying inside of the polygon. This paper focuses on vertex guards.

This paper is organized as follows. In section 2, related work is presented. Section 3 explains the proposed algorithm. Section 4 discusses on the results of the proposed algorithm and compares them with results of genetic algorithm and well-known art-gallery theorem. Finally, conclusion is expressed.

2. RELATED WORK
As mentioned above, V. Kelli in 1973 posed that how many stationary guards are needed to guard an art gallery room with $n$ walls [3]. Two years after posing Klee's question, Chvatal established the well-known Art Gallery Theorem: $\lfloor n/3 \rfloor$ guards are occasionally necessary and always sufficient to cover a simple polygon of $n$ vertices. Over the years, numerous variations of the original problem have been considered and studied. Discussing on kinds of guards as mentioned before and kind of polygon e.g., orthogonal simple polygons, i.e., simple polygons whose edges meet at right angles and simple polygons including holes were some of assumptions and variations of the original problem. Table 1 shows minimum number of guards for guarding polygons without holes.
Designing approximation algorithms [4] is one of approaches to solve computational geometry problems. In general, these algorithms can be based on heuristic and general meta-heuristics (e.g., simulated annealing and genetic algorithm) [5] [6] [7]. Authors in [4][5], were developed approximation algorithms to solve the minimum guards problem. In [3], a genetic algorithm as an approximation algorithm was proposed to solve the minimum vertex guard problem. Based on this algorithm, authors concluded that, on average, the minimum number of vertex-guards needed to cover a simple polygon with n vertices was \( n/6.38 \) and the approximation ratio was observed to be less than or equal to 2. This result for orthogonal polygons was \( n/6.40 \) with approximation ratio 1.9. Table 2 shows minimum number of guards for guarding polygons with \( h \) holes.

From point of view of the problem class, minimum guard problem has been discussed and proved by the following researchers:
- O'Rourke [1983]: NP-complete for any polygons with \( h \) holes
- Aggarwal [1984]: NP-complete for any polygons with holes, focus on point guards
- Lee and Lin[1986]: NP-hard for any polygons without holes
- O'Rourke [1987]: NP-hard for any polygons with \( h \) holes (Vertex, edge, and point guards)
- Liaw [1993]: NP-hard for k-spiral polygons
- Hecker [1995]: NP-hard for orthogonal polygons

### 3. PROPOSED ALGORITHM

At first, some definitions and notations are defined. In a polygon with \( n \) vertices, \( r \) represents the number of reflex vertices. For a \( P \in P \), the visible polygon from \( P \) is called \( \text{Visible}(P) \) which includes the set of vertices \( \text{Visible}(P) \) that are visible from \( P \) and is defined as

\[
\text{Visible}(P) = \{ P_i \in P \mid P_i \text{ sees } P \}.
\]

The cardinality of a vertex guard set is denoted by \( |G| \). It is also assumed that all vertices of polygon are in list \( P_i \) (0≤\( i \)≤\( n-1 \)) and list of guards is \( G(g) \) which \( g \) will be number of guards.

Now, we present the steps of proposed algorithm. At the first, \( n \) random vertices are generated with \( x \) and \( y \) coordinates as vertices of polygon \( P \). \textit{Random-Generate-Points (n)} performs it. Then for generation a simple polygon with these given vertices, \textit{Random-Polygon-Generator (n)} is called. This function simulates the Two_Opt_Moves algorithm to generate arbitrary random polygons. For any \( P \in P \), its \( \text{Visible}(P) \) is specified with finding the internal diagonals \( \text{Visible}(P) \) which indicate all visible vertex \( P_i \) from \( P \). Finally, the guards are determined by calling \textit{Find-Guards()} function. Figure 1 indicates the procedure of determining minimum vertex guard in

### Table 1 Minimum number of guards for guarding polygons without holes

<table>
<thead>
<tr>
<th>Author [Year]</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chavatal [1]</td>
<td>Needing at most ( [n/3] ) guards</td>
</tr>
<tr>
<td>Fisk [4]</td>
<td>Proof of above formula with 3-coloring technique</td>
</tr>
<tr>
<td>Toussaint [1981]</td>
<td>( [n/4] ) mobile guards are always sufficient and occasionally necessary</td>
</tr>
<tr>
<td>Avis [1981]</td>
<td>Presenting algorithm with running time: ( O(n \log n) )</td>
</tr>
<tr>
<td>Kahn [1983]</td>
<td>Needing at most ( [n/4] ) stationary guards for an orthogonal polygon</td>
</tr>
<tr>
<td>O'Rourke [1983]</td>
<td>Proof of above formula (Decomposing into quadrangle convex polygons)</td>
</tr>
<tr>
<td>Aggarwal [1984]</td>
<td>Needing at most ( [3n+4/16] ) mobile guards</td>
</tr>
<tr>
<td>O'Rourke [1987]</td>
<td>Needing at most ( [n/4] ) mobile guards</td>
</tr>
<tr>
<td>Bjorling [1998]</td>
<td>Proof of above formula for edge guards</td>
</tr>
<tr>
<td>Urrutia [2000]</td>
<td>Needing at most ( [n/4] ) edge guards are sufficient, except for a few polygons</td>
</tr>
</tbody>
</table>

Table 2 describes researches done on polygons with \( h \) holes.

### Table 2 Minimum number of guards for guarding polygons without hole

<table>
<thead>
<tr>
<th>Author [Year]</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shermer [1982]</td>
<td>Needing ( [(n+h)/3] ) vertex guards (For ( h&gt;1 ) is Open Problem!!)</td>
</tr>
<tr>
<td>O'Rourke [1987]</td>
<td>Needing ( [(n+2h)/3] ) vertex guards</td>
</tr>
<tr>
<td>Bjorling [1992]</td>
<td>Needing ( [n-2]/5 ) edge guards for monotone polygon</td>
</tr>
<tr>
<td>Elnager [2004]</td>
<td>The no. of guards ( &lt;= [(n+h)/3]+1 ) is sufficient</td>
</tr>
</tbody>
</table>
which g is the returned number of guards by this algorithm.

Function Heur_Minimum_Vertex_Guard(n)
1. Random_Generate_Points(n);
2. P=Random_polygon_Generator();
3. For i=0 to n-1 do
   3.1 Determine Visible(p_i)
4. g=Find_Guards();
5. Return g;

Figure 1: procedure of determining minimum vertex guard

3.1. Random Polygon Generation
After generating random vertices, random polygon generator function simulates arbitrary random polygons. In this procedure, iteratively a pair of intersected edges (u,v) and (s,t) is searched and is replaced by the pair of (s,v) and (u,t).

3.2. Visible Polygons
To determine guards, at the first, for any \( p_i \), its visible polygon, \( \text{Visible}(p_i) \) is specified by finding the internal diagonals \( p_i \rightarrow p_j \) which indicate all visible vertex \( p_j \) from \( p_i \). Two cases happen. At the first case, Vertex \( p_j \) is a convex vertex and \( p_{i-1} \) does not lie left of \( p_j p_i \) and \( p_{i+1} \) does not lie right of \( p_j p_i \). In second case, Vertex \( p_j \) is a reflex vertex and \( p_{i-1} \) does not lie left or on of \( p_j p_i \) and \( p_{i+1} \) does not lie right of \( p_j p_i \). In the next step, the diagonals which have intersected the polygon in a point except the end of edges are removed. Thus, the remaining internal diagonals represent that any vertex sees which vertices and its visible polygon. Finally, the number of guards is computed for any vertices with respect to the \( \text{Visible}(p_i) \).

3.3. Finding Vertex Guards
At the first, the number of visible vertices of any vertex \( p_i \) is counted in \( \text{Visible}(p_i) \) and is placed in counting list \( C(\text{Visible}(p_i)) \). Also, for any \( v_j \), for any element of its \( \text{Visible}(p_j) \), number of any element repetition in all \( p_j \) in \( \text{Visible}(p_j) \) is obtained and is placed in counting list \( C(\text{Visible}(p_j)) \). Then the vertices are sorted according to vertex which sees maximum of vertices from its maximum amount \( C(\text{Visible}(p_j)) \) to its minimum amount. If for both vertices \( p_j \) and \( p_{j+1} \), \( C(\text{Visible}(p_j)) \) is equal to \( C(\text{Visible}(p_{j+1})) \), the preference for sorting will be with the vertex which its visible vertices have less number of repetition in other visible lists, because it gives a better cover. Now, based on this sorting, \( p_j \) will be the vertex which sees maximum of vertices. This vertex is added to guards list, namely G(g) and is incremented to the number of guards g. All vertices which see vertex guard \( p_j \) in these visible lists, the lists of other vertices visible polygon and the list of all of vertices, i.e., \( P \), are removed. This procedure continues until no vertex remains without guard and finally each vertex is guarded by a guard and each vertex will be without a guard. This procedure is represented as follows.

Find_Guards algorithm()
1. K=n-1;
2. While (K>=0) do Begin
   1. For each \( p_j \in F \) do
      Compute \( C(\text{Visible}(p_j)) \) and \( C(\text{Visible}(p_j)) \), END FOR;
   2. Sort vertices \( p_j \) from Max \( C(\text{Visible}(p_j)) \) to Min \( C(\text{Visible}(p_j)) \);
   3. For each \( j=1 \) do If \( C(\text{Visible}(p_j))=C(\text{Visible}(p_j)) \) then Select \( \text{Min}(C(\text{Visible}(p_j)), C(\text{Visible}(p_j))) \) and sort vertices P;
   4. Add \( p_j \) into G(g);
   5. For each \( p_j \in \text{Visible}(p_j) \) do
      5.1. For i=0 to n-1 do delete \( p_j \in F \); delete \( p_j \in \text{Visible}(p_j) \);
      5.2. For j=1 to n-1 do delete \( p_j \in \text{Visible}(p_j) \);
   End for; End for;
6. K=K-1; END WHILE;

Figure 2: Finding guards algorithm

4. COMPUTATIONAL RESULT
We have implemented our heuristic algorithm with C#.
In this section, output of our program is analyzed especially for large number of nodes. The purpose of these experiments is to find the average number of minimum vertex guards on 4 sets of polygons of 30, 50, 70, and 100 vertices randomly generated. The results put for genetic algorithm in figure 3 has been extracted from [8].

Figure 3 shows average number of minimum vertex guards for covering arbitrary polygons randomly developed. It also represents a comparison between our algorithm and the results of using genetic algorithm and well-known art-gallery theorem.
Using the least squares method, linear adjustment \((n/6.48)+0.51\) is obtained. As a result, minimum number of vertex guard needed to cover an arbitrary polygon is \((n/6.48)\). This computational result is more effective than old related algorithms.
5. CONCLUSIONS AND FUTURE WORK
This paper presents a new heuristic algorithm to find the minimum vertex guard in order to cover a given random simple polygon. Computational experiments indicated that the average number of minimum vertex guards needed to cover an arbitrary simple polygon is \( \frac{n}{6.48} \). As future work, we attempt to improve this algorithm and run it on specific polygons such as monotone polygons. We will also enhance our algorithm on multi processors.

REFERENCES