Proceedings of the Federated Conference on Computer Science and Information Systems pp. 215–222
DOI: 10.15439/2020F79
ISSN 2300-5963 ACSIS, Vol. 21

An Exact Two-Phase Method For Optimal Camera Placement In Art Gallery Problem

Adis Alihodzic, Sead Delalic, Damir Hasic
University of Sarajevo, BiH
Department of Mathematics
ul. Zmaja od Bosne, 33-35, Sarajevo
Email: {adis.alihodzic, delalic.sead, damir.hasic}@pmf.unsa.ba

***Abstract*—It is well-known that determining the optimal number of guards which can cover the interior of a simple non-convex polygon presents an NP-hard problem. The optimal guard placement can be described as a problem which seeks for the smallest number of guards required to cover every point in a complex environment. In this paper, we propose an exact two-phase method as well as an approximate method for tackling the mentioned issue. The proposed exact approach in the first phase maps camera placement problem to the set covering problem, while in the second phase it uses famous state-of-the-art CPLEX solver to address set covering problem. The performance of our combined exact algorithm was compared to the performance of the approximate one. According to the results presented in the experimental analysis, it can be seen that the exact approach outperforms the approximate method for all instances.**

I. Introduction

THE ART gallery problem (AGP) dates back to the 1970s, and it was one of the earliest and most significant problems in sensor placement [1][2]. The calculation of optimal solutions for AGP is not only relevant from a theoretical aspect, but it also has practical importance in architecture, placement of radio antennas, urban planning, ultrasonography, sensors, mobile robotics, and other branches of science and industry [3]. In computational geometry, it presents a visibility problem of placing at least one security guard to cover every area of a museum or gallery [4]. Since the optimal camera placement (OCP) problem represents the process of finding the minimal number of cameras that are sufficient to cover every point in the environment, we can say the both AGP and OCP are very similar to each other. Art gallery problem in the original form was based on determining smallest number of security guards sufficient to see every point in an n-sided two-dimensional polygon P with or without holes. The scientists such as O'Rourke and Supowit, Lee and Lin, Katz and Rpoisman, Schuchardt and Hecker have shown that the process of looking for the smallest number of guards who can surveillance any polygons (ordinary or orthogonal) still presents an intractable NP-hard problem [5][6][7][8]. In 1975, Chvátal proved that only $\lfloor \frac{n}{3} \rfloor$ cameras are sometimes necessary and always sufficient to being covered the simple polygons composed of n vertices [9]. For n-sided polygon P with h holes, O'Rourke showed that it is necessary at most $\lfloor \frac{n+2h}{3} \rfloor$ vertex guards. On the other hand, Bjorling-Sachs, Souvaine, Hoffmann and others have shown that it is quite enough $\lfloor \frac{n+h}{3} \rfloor$ guards to being covered polygons with n vertices on the outer boundary which contain h holes inside them [10][11]. The researchers Györi, Hoffmann, Kriegel and Shermer have been shown that for the orthogonal polygons with h holes always is sufficient $\lfloor \frac{3n+4h+4}{16} \rfloor$ guards to being covered [12]. By using the colouring technique which has been used by Fisk [13] to prove the Chvátal result, the authors Avis and Toussaint have developed $O(n \log n)$ time complexity algorithm for camera placement in a simple polygon. However, the number of cameras is not minimal. Also, Bjorling-Sachs and Souvaine [10] proposed an $O(n^2)$ time algorithm for non-optimal guards positioning in a polygon P with h holes.

The placement of visual sensors in two-dimensional space can be modelled as AGP. Tasks such as surveillance require observing the interior of a polygon with a minimum number of sensors or cameras. This watching we call the interior covering (IC). For other tasks, such as inspection and image-based rendering, observing the boundaries of the environment is sufficient. In this case, keeping the boundaries, we call the edge covering (EC). Both interior covering and edge covering present NP-hard problem, and no deterministic (finite) algorithms are known for tackling this type of issue. In this paper, we propose two variants of algorithms such as an exact two-phase algorithm as well as the approximate method which participate in solving camera placement problem. For the experimental study, we developed two versions of the mentioned algorithms, to be able to process them in parallel for both edges covering and interior covering. The first phase of the exact two-phase approach serves for translating the art gallery problem into the famous set covering problem (SCP). In this phase, we will consider coverage of edges as well as surveillance of convex components, i.e. triangles, from which a polygon is composed. Also, in this phase, we exploit preprocessing algorithms, i.e. algorithms for creating a list of sets of vertices (components) that cover all vertices (parts) of a polygon P when take into consideration edge covering problem (interior covering problem). At the second phase of the algorithm, for a list of sets obtained in the previous step, we define a modified version of set covering problem and solve it as a linear optimization problem (LOP) by using a standard mathematical tool to find the exact optimal solution. The standard solving tool for such LOP is ILOG IBM CPLEX Solver [14]. The set covering problem is an NP-hard

problem in the strong sense, and many algorithms have been developed for its solving [15]. The SCP is vital in practice, as it has been used to model a broad range of problems arising from scheduling, manufacturing, delivery and routing, service planning, information retrieval, etc. [16][17]. The exact algorithms are almost all based on branch-and-bound and branch-and-cut [18][19]. Caprara et al. compared different exact and heuristic algorithms for solving the SCP [20]. They demonstrated that in practice, IBM ILOG CPLEX Solver is the best exact algorithm for tackling set covering issue [21]. ILOG CPLEX delivers high-performance, robust, flexible optimizers for solving linear, mixed-integer and quadratic programming problems (including mixed-integer quadratic constrained issues). ILOG CPLEX optimizer has a modelling layer that provides interfaces to C++, C#, Java, Python, Matlab, etc. In this paper, we have used the layer Concert Technology to integrate C# into the ILOG Optimization Studio, since all routines for the first part of the exact approach were written in C#. In order to show the power of proposed techniques, the two-phase algorithm has been tested on 268 various randomly generated simple nonconvex polygons. The results produced in the experimental analysis were compared with the one reached by our sub-optimal approximate algorithm, which we also have been developed for comparison purposes. For both observings, the experimental results show that tho-phase method is better technique and yields the optimal solutions in a reasonable amount of time.

The rest of the paper is organized as follows. For art gallery problem (AGP), an approximate method for both edges covering and interior covering of a simple nonconvex polygon is described in Sect. 2. The details of our exact two-phase algorithm are presented in Sect. 3. Experimental and comparative results of applying different versions of the algorithms for AGP are presented in Sect. 4. Finally, conclusions and suggestion for future work are discussed in the last section of the paper, Sect. 5.

II. An approximate method for AGP

In this section, before we describe the approximate algorithm for solving the AGP, we will briefly introduce some additional notation to facilitate the exposure. For any two distinct points v_1 and v_2 in the plane, we denote by $\overline{v_1v_2}$ the segment whose two endpoints are v_1 and v_2. A planar polygon P presents a closed plane figure whose boundary is composed of segments $\overline{v_iv_{i+1}}$ $(i = 0, 1, \cdots, n-1)$, where $v_n = v_0$. Also, a polygon P is simple if it is not self-crossing and has no holes. A planar polygon P is convex if it contains all the segments connecting any pair of its points. A nonconvex (concave) polygon P is a polygon that is not convex. In other words, a polygon P is nonconvex if there are two points u and w inside of P such that the segment $\overline{uw}$ is not entirely contained in the P. Also, a concave polygon must have at least four sides, and it always has at least one reflex interior angle, that is, an angle with a measure that is between 180 degrees and 360 degrees exclusive. Any point u in P is said to be visible from any other point w in P if and only if the segment $\overline{uw}$ does not intersect the exterior of P as well it is entirely contained in P. For any point $u \in P$, the set of all points in P which are visible from a vertex u is called the visibility region of u.

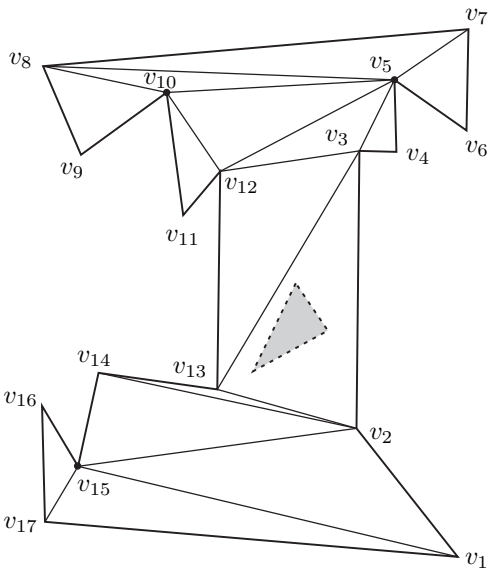

Fig. 1. The whole hull of the polygon P has covered by vertices v_5, v_{10} and v_{15}, but the inner coloured part is not visible by them.

In the following of this section, we will describe the approximate method. Let set V denotes the vertices of a simple non-convex polygon P which contains n vertices, i.e. $|V| = n$. Assume that vertices of a given polygon P are labeled by $v_1, v_2, \cdots, v_n$. Also, let $F(P, u)$ denotes the set of all points of P which can be observed from a point u. If the point u is a vertex of the polygon P, i.e. exists some index $k \in \{1, 2, \cdots, n\}$ such that $u = v_k$, then we call the subset $F(P, u)$ of P fan F_k, where the vertex v_k denotes the fan vertex of the set F_k. On the other hand, let u is not a vertex of the polygon P. Then, the set $F(P, u)$ is called a region under surveillance from the point u.

By taking into account these definitions, the main idea of our approximate method we will describe below is to being maximized fans. At the beginning of the method, we determine such fan F_{i_1} that covers the most vertices of the polygon P and set the number i_1 as the index of the first guard (camera). After that, we update the remaining sets F_j by removing from them all the elements which appear in the set F_{i_1}, i.e. we make difference $F_j \leftarrow F_j \setminus F_{i_1}$ for all fans. After this, the set F_{i_1} becomes empty, so it is no longer considered. For non-empty updated fans F_j, we repeat the same procedure as at the beginning of the algorithm, i.e. we select the fan F_{i_2} which has the most elements, and then take that index i_2 be the index of the second guard. It is clear now that the guard with index i_1 covers more vertices than the guard with the index i_2. By repeating the mentioned procedure, we can note that after a certain number of iterations, all fans F_i will be empty, which is an indicator for the end of the algorithm. Also, generated

TABLE I
THE FAN'S CALCULATION BY USING THE INDEXES OF COMPONENTS WERE COVERED BY THE VERTICES OF THE POLYGONS.

	$\mathbf{C}_1$	$\mathbf{C}_2$	$\mathbf{C}_3$	$\mathbf{C}_4$	$\mathbf{C}_5$	$\mathbf{C}_6$	$\mathbf{C}_7$	$\mathbf{C}_8$	$\mathbf{C}_9$	$\mathbf{C}_{10}$	$\mathbf{C}_{11}$	$\mathbf{C}_{12}$	$\mathbf{C}_{13}$	$\mathbf{C}_{14}$	$\mathbf{C}_{15}$
v_1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0
v_2	1	1	1	0	0	0	0	0	1	0	0	0	1	1	0
v_3	0	0	1	1	0	0	0	0	1	1	0	1	0	0	0
v_4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
$\mathbf{v}_5$	0	0	0	**1**	**1**	**1**	0	0	**1**	**1**	**1**	**1**	0	0	0
v_6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
v_7	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
$\mathbf{v}_8$	0	0	0	0	0	**1**	**1**	0	0	0	**1**	0	0	0	0
v_9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
v_{10}	0	0	0	1	0	1	1	1	0	1	1	1	0	0	0
v_{11}	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
$\mathbf{v}_{12}$	0	0	**1**	0	0	0	0	**1**	**1**	**1**	0	**1**	0	0	0
v_{13}	1	1	1	0	0	0	0	0	1	1	0	0	1	1	0
v_{14}	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
$\mathbf{v}_{15}$	**1**	**1**	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**
v_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
v_{17}	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1

numbers $i_1, i_2, \cdots, i_k$ were sorted in descending order, where k denotes the number of guards required to cover the boundary of the polygon P. Based above described procedure for a seeking a smallest number of guards to cover vertices of a simple nonconvex polygon P, the necessary steps of the vertex observing problem has summarized by Algorithm 1. From the pseudo-code presented in Algorithm 1, we can see that the method stops when all fans become empty. In other words, since the union of fans F_i $(\forall i = 0, 1, \cdots, n-1)$ denotes the vertices indexes of a polygon P, it is easy to conclude that the algorithm terminates as soon each vertex v_i has covered. Since a simple polygon P has been compounded of the segments $\overline{v_i v_{i+1}}$, and the mentioned algorithm can cover all vertices v_i of P, i.e. all endpoints of the segments $\overline{v_i v_{i+1}}$, immediately follows that proposed method can being exploited for edge covering (EC).

Algorithm 1 Approximate Method For Vertex Covering (VC)

1: Set $n_g \leftarrow 0$, $G \leftarrow \emptyset$, where n_g is a number of guards and G is their list. Initialize the list of all fan's indexes with $F \leftarrow \{0, 1, \cdots, n-1\}$.
2: For each vertex v_i $(i = 0, 1, \cdots, n-1)$ determine fan F_i by adding indexes of vertices $v_j \in P$ $(v_j \neq v_i)$ into F_i which are completely visible from the vertex v_i.
3: **while** $n_g \neq n$ **do**
4: From the list F, find the fan that has most elements and denotes its index by i.
5: Put to the list G the vertex (guard) $v_i \in P$ which was referred to the biggest founded fan F_i from the previous step.
6: From all fans F_j remove the elements which were appeared to the set F_i, i.e. $F_j \leftarrow F_j \setminus F_i$ $(\forall j \neq i)$.
7: Set $n_g \leftarrow n_g + |F_i|$ and remove the index i from the list F.
8: **end while**

From the pseudo-code presented in Algorithm 1, we can see that time complexity of our approximate method is proportional with $O(n^3)$ since the determination of fans F_i $(i = 0, 1, \cdots, n-1)$ is the most expensive step and it costs $O(n^3)$. More precisely, to examine which vertices are covered by an arbitrary vertex v_i of a polygon P, we first connect vertex v_i with the nonadjacent vertices v_k $(k \neq i-1, k \neq i, k \neq i+1)$, thus $n-3$ segments were obtained. Then in time $O(n^2)$ we check whether the generated segments intersect $n-2$ segments (segments $\overline{v_{i-1}v_i}$ and $\overline{v_i v_{i+1}}$ are not examined) which lie on the boundary of a polygon P. Since a polygon P has n vertices, then a total number of checks in the worst case is equal to $(n-3)(n-2)n$, which is proportional to $O(n^3)$.

Although the edge covering of a polygon is essential in image processing as well as in other applications, in this paper, we investigate the interior covering of a simple polygon. It is especially important to highlight here that there is a difference between the edge covering of a polygon and its interior covering. Namely, the number of guards necessary to cover the boundary of a polygon is not always sufficient to cover its overall interior, as can be seen in Figure 1. Conversely, it is always valid. For example, for the polygon has been shown in Figure 1, to perform its interior covering it was required exactly four guards such as v_5, v_8, v_{12}, and v_{15}, which represents the optimal number of guards. On the other hand, guards such as v_5, v_{10} and v_{15} can only cover the boundary of the polygon P, because it remains uncovered shaded triangle.

Before we perform interior covering of a nonconvex simple polygon P, we will address a polygon decomposition into a set of convex components C_k such that their union is the entire region of P. Now, interior covering (IC) can be modelled as a seeking the smallest number of guards required to cover the building components C_k such that their union is a whole polygon P. As earlier, in terms of fans, we define that fan F_i contains the indexes of components that can be covered by the vertex v_i. It is easy to note that each component belongs at least one of the fan so that the entire region of P is covered. The method shown in Algorithm 1 can also be exploited for

interior covering of a polygon by modifying its step 2 as follows. Instead of vertex covering, we will now visit the components, i.e. at the fan F_k, we will add those indexes of components that the vertex v_k can visit. In this way, we get the algorithm for interior covering (IC) of a simple nonconvex polygon.

Algorithm 2 The Exact Two-Phase Approach For Interior Covering (IC)

1: Determine a triangulation of of n sided nonconvex simple polygon P. Let us denote the obtained triangles as components $C_1, C_2, \cdots, C_{n-2}$.
2: For each vertex v_i $(i = 0, 1, \cdots, n-1)$, determine the fan F_i by adding indexes of components C_j into F_i which are completely visible from the vertex v_i.
3: Create the matrix A from the content of fans F_i.
4: Make preprocessing and reduce the number of rows for the matrix A.
5: Apply CPLEX solver to generate the smallest number of guards necessary for interior covering of a polygon.
6: Visualize the founded guards.

III. An Exact Two-Phase Approach For AGP

In this section, we will describe in detail our exact two-stage algorithm designed to solve the Art Gallery Problem (AGP). First of all, in the first stage, we will transform the art gallery problem to the well-known Set Covering Problem (SCP). After that, in the second stage, we will apply prominent CPLEX solver to address the adjusted set covering problem obtained in the first stage.

A. The mapping of the AGP on the SCP

In order to map the art gallery problem to the set covering one, we will first divide the interior of a polygon P into a set of nonoverlapping convex parts. There are several ways how to perform dividing a simple closed nonconvex polygon into nonoverlapping convex sub-polygons or pieces [22]. In this paper, partitioning a polygon into convex parts has obtained by exploiting triangulation. To efficiently perform triangulation, we have implemented a very efficient algorithm whose time complexity is proportional to the $O(n \log n)$ [23]. This algorithm consists of two steps. In the first step, we made a partition of a simple polygon with n vertices into monotone pieces in $O(n \log n)$ time, while in the second step, we triangulated monotone pieces (polygons) in linear time $O(n)$. The above steps together imply that any simple nonconvex polygon P can be triangulated in $O(n \log n)$ time.

The application of triangulation on any simple nonconvex polygon P composed of n vertices produces $n-2$ triangles, i.e. $n-2$ convex components $C_1, C_2, \cdots, C_{n-2}$. By introducing these components, optimal coverage of a polygon interior has reduced to seeking the smallest number of guards which can see all components. In order to determine those guards, we will first create fans F_j $(j \in \{1, 2, \cdots, n\})$ for each vertex v_j. In the context of components, arbitrary fan F_j contains the indexes of components which are visible from the vertex v_j. In the following, we consider the creating of 15 fans for a simple non-convex polygon shown in Figure 1. It is easy to see that for vertex v_1 fan F_1 has indexes 1, 2, 13, 14, since the vertex v_1 covers the components (triangles) C_1, C_2, C_{13}, C_{14}. In Table I, for each vertex v_j $(j = 1, 2, \cdots, n)$, we have presented calculated fans F_j in the form of rows. For example, with respect to the vertex v_1, the fan F_1 has indexes 1, 2, 13, and 14.

From the structure of data shown in Table I, it is easy to notice that an optimal covering of a polygon can be made by exploiting the adjusted version of the set-covering problem (SCP). The adjusted version of the SCP can be defined as follows. Let $A = (a_{i,j})$ be an zero-one matrix of $n \times n-2$ size. We say that a row i covers a column j if holds $a_{ij} = 1$. Let $I = \{1, 2, \cdots, n\}$ and $J = \{1, 2, \cdots, n-2\}$ be the row set and column set, respectively. The SCP needs determining the minimum subset $I' \subset I$ such that each column $j \in J$ is covered by at least one row $i \in I'$. A mathematical model for the adjusted SCP is defined as follows

$$Minimize\ f(x) = \sum_{i=1}^{n} x_i \tag{1}$$

subject to

$$\sum_{i=1}^{n} a_{ij} x_i \geq 1,\ \forall j \in J \tag{2}$$

$$x_i \in \{0, 1\},\ \forall i \in I \tag{3}$$

From Eq. 1 immediately implies that we have to minimize the number of guards (number of the selected rows), where $x_i = 1$ if the row i is in the solution and $x_i = 0$ otherwise. Each column j is covered at least by one row i. The SCP constraints guarantee this.

```
int M=...; //num. of cols
int N=...; //num. of rows
range rows=1..N;
range cols=1..M;
dvar boolean x[rows];

int A[rows][cols]=...;

minimize sum(i in rows) x[i];
    subject to{
        const1:
            forall(j in cols)
                sum(i in rows) A[i][j]*x[i]>=1;
    }
```

Fig. 2. The OPL Code For Solving Simplified SCP.

After we have described the mechanism for optimal interior covering of a simple nonconvex polygon, its realization is carried out by the method whose necessary steps were summarized in the pseudo-code of Algorithm 2. From the pseudo-code, we can see that the overall time complexity of this

approach is proportional to $O(n^3)$. Although this approach for both interior covering (IC) and edge covering (EC) has the same time complexity, there is a slight difference in the specified number of evaluations during their execution, which will be seen in the experimental analysis. Also, for both types of coverage, the preprocessing remained more costly in time compared to the finding of an optimal solution by using CPLEX solver.

B. The preporcessing of the adjusted SCP

It is well-known that preprocessing is a superior technique to accelerate the solving process by reducing the instance sizes additionally. There are a lot of preprocessing methods in the literature for the SCP [18]. In this paper, to reduce the instance size, i.e. the size of the matrix A, we apply methods such as *row domination* and *row inclusion*. The row domination can be defined as follows. We say that row i is dominated and can be removed from the matrix A if its columns J_i can be covered by other rows. In Table I, we can see that row 14 is dominated by rows 1 and 15, and it can be removed. Also, rows 1 and 17 are dominated by row 15, so they should be dropped. After the elimination of the rows from the matrix A, it was reduced on only six rows. Therefore, it consists of rows with indices 3, 5, 8, 12, 13 and 15. Row inclusion means that if a column j is covered by precisely one row after the above domination, then it row is included in an optimal solution. It can be noted that matrix A whose data has been shown in Table I before processing had 17 rows, and after processing it has only six rows.

C. The application of CPLEX solver on the adjusted SCP

The second phase of our proposed approach was based on solving of the adjusted set covering problem (ASCP). The adjusted set covering problem in IBM ILOG CPLEX Optimization Solver [14] is presented in the form of a binary integer programming problem, as we can see in Figure 2. From the code shown in Figure 2, we can note that the matrix A is being generated in the first phase of our algorithm. At the same time, the vector x is a decision binary vector whose components are being determined by CPLEX solver. On the beginning of the algorithm, CPLEX solver set the content of the vector x to zero. CPLEX solver by using techniques such as branch-and-bound as well as branch-and-cut, it is capable of determining an optimal solution just over several seconds, as we will see in the experimental analysis. For the polygon drawn in Figure 1, after the execution of CPLEX solver, the content of the vector x becomes this one x =[0 0 0 0 1 0 0 1 0 0 0 1 0 0 1 0 0], where ones (1) indicate that the vertices v_5, v_8, v_{12}, v_{15} are covered in case of interior covering. If we first make the elimination of rows, and after that, we apply CPLEX solver on the remaining rows of the matrix A, we will see that the size of vector x is reduced from 17 to 6, so the final solution x now has this form: x =[0 1 1 1 0 1].

IV. Experimental Results

In this experimental study, we compare two groups of deterministic algorithms for solving both edges covering (EC) and interior covering (IC) of a simple nonconvex polygon. The goal of our proposed methods was to minimize the number of cameras required for polygon coverage. The proposed techniques have been thoroughly tested to assess the quality of the results. In the first group of deterministic algorithms, there are exact methods, while the second group encompasses the approximated techniques. The algorithms have been applied to 268 various randomly generated simple nonconvex polygons. The simple nonconvex polygons have been produced with our random polygon generator developed for purposes of this paper, whose implementation details we omit. The interested reader can refer to similar random polygon generator (RPG) [24][25]. The examples of randomly generated simple nonconvex polygons with 50 edges are shown in Figure 3. Each instance is called RI-k-i, where k denotes the size of the ith instance. The coordinates of points (x, y) are chosen from the interval $[0, 500]$. Through the experimental evaluation, we assess the applicability of four algorithms for the AGP. The proposed approaches have been implemented in C# programming language, where two exact two-phase methods in their second phase use IBM ILOG CPLEX Optimizer 12.10 to address set covering problem expressed in the form of binary integer programming (BIP). This optimizer is being called from C# in conjunction with the Concert Technology. In order to evaluate time efficiency and the coverage rate of the proposed algorithms, a comparison test was performed using a PC with an Intel Core i7-3770K @3.5GHz with 64GB of RAM running under the Windows 10 x64 operating system.

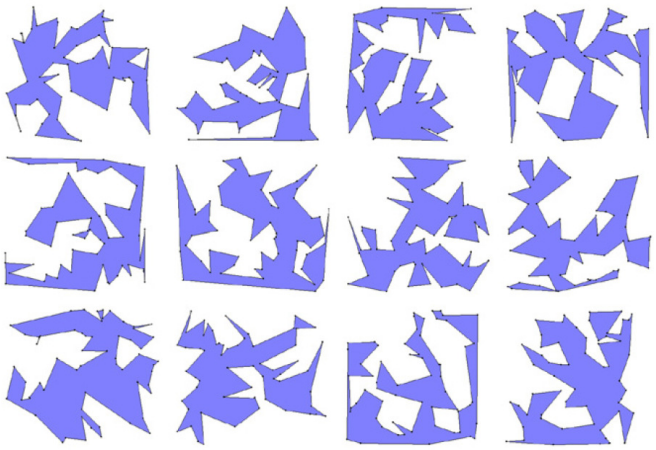

Fig. 3. Randomly generated simple nonconvex polygons with 50 vertices.

In Table II, for the purpose of checking the quality of obtained solutions as well as computational times, four algorithms were selected and tested through 8 groups of randomly generated polygons, where each group was composed of five randomly generated polygons containing 20, 40, 60, 100, 300, 500, 1000, and 2000 vertices, respectively. From obtained simulation results shown in Table II, we can see for edge covering (EC) problem that approximate method is slightly faster than the exact two-phase method for almost all groups except for the next to last group which contains 1000 vertices.

TABLE II
THE AVERAGE NUMBER OF GUARDS AND MEAN TIME PROCESSING PROVIDED BY THE EXACT AND APPROXIMATE ALGORITHMS FOR 40 RANDOMLY DISTRIBUTED INSTANCES.

	INTERIOR COVERING (IC)				**EDGE COVERING (EC)**			
	Two-phase approach		Approximate approach		Two-phase approach		Approximate approach	
Random instances	Number of guards	Time (sec.)	Number of guards	Time (sec.)	Number of guards	Time (sec.)	Number of guards	Time (sec.)
RI- 20-1	4	0.19	5	0.01	3	0.01	4	0.00
RI- 20-2	3	0.19	4	0.01	3	0.19	4	0.00
RI- 20-3	2	0.02	3	0.01	2	0.17	3	0.01
RI- 20-4	3	0.02	4	0.01	3	0.18	4	0.00
RI- 20-5	3	0.19	4	0.01	3	0.01	4	0.00
RI- 40-1	5	0.06	7	0.05	5	0.15	6	0.02
RI- 40-2	6	0.19	7	0.04	6	0.03	7	0.02
RI- 40-3	6	0.19	7	0.05	5	0.16	7	0.02
RI- 40-4	7	0.06	8	0.04	7	0.03	8	0.02
RI- 40-5	6	0.14	7	0.05	6	0.16	7	0.02
RI- 60-1	9	0.12	11	0.15	8	0.13	9	0.06
RI- 60-2	9	0.19	11	0.12	8	0.08	10	0.07
RI- 60-3	11	0.13	12	0.11	9	0.08	10	0.05
RI- 60-4	10	0.13	12	0.10	10	0.12	11	0.06
RI- 60-5	9	0.14	11	0.13	8	0.12	11	0.07
RI- 100-1	14	0.37	16	0.36	12	0.38	14	0.24
RI- 100-2	15	0.40	17	0.31	14	0.23	15	0.19
RI- 100-3	16	0.37	18	0.27	14	0.21	16	0.19
RI- 100-4	14	0.38	19	0.36	12	0.24	15	0.19
RI- 100-5	15	0.37	17	0.36	13	0.21	16	0.14
RI- 300-1	40	5.48	44	4.86	35	3.65	38	3.82
RI- 300-2	46	4.44	52	4.28	42	2.99	46	3.15
RI- 300-3	44	4.46	50	4.10	37	3.19	39	2.96
RI- 300-4	43	5.03	50	4.22	36	3.64	43	3.57
RI- 300-5	42	4.44	50	4.83	37	3.20	42	3.26
RI- 500-1	77	16.31	87	13.99	66	12.08	75	10.57
RI- 500-2	70	15.20	85	12.80	63	11.02	72	9.81
RI- 500-3	71	17.29	77	14.77	62	13.20	68	11.43
RI- 500-4	73	16.11	87	13.73	62	12.37	77	10.75
RI- 500-5	72	12.87	83	11.48	63	8.58	74	7.90
RI- 1000-1	148	74.09	162	73.92	129	60.92	147	66.20
RI- 1000-2	143	77.38	155	76.96	125	61.06	138	70.30
RI- 1000-3	141	82.21	154	83.04	120	69.34	137	80.68
RI- 1000-4	145	85.95	161	77.50	133	55.98	154	59.68
RI- 1000-5	140	93.73	155	92.02	124	75.53	141	81.98
RI- 2000-1	296	639.77	322	643.25	260	579.70	293	570.81
RI- 2000-2	303	618.68	335	621.49	265	561.08	305	548.73
RI- 2000-3	292	682.25	318	687.29	253	624.10	280	602.25
RI- 2000-4	285	455.22	318	445.97	254	398.18	291	385.15
RI- 2000-5	291	567.14	316	547.54	249	511.58	290	478.43

More precisely, the approximate method requires 3012.26 seconds to process all instances of polygons from all groups. On the other hand, the exact two-phase method costs 3072.93 seconds, which is 60.67 seconds more in comparison with the approximate method. In the sense of quality solutions, i.e. in the sense of the smallest number of guards, the exact two-phase method outperformed approximate method for any groups of instances. Namely, the exact method requires only 2566 guards (cameras) to cover all polygons from all eight groups, while the approximate method allocates 1558 cameras more, i.e. it needs 4124 cameras. Particularly superiority comes to the fore with an increase in the size of vertices. In practice for the case of observing a polygon consisting of 2000 vertices, e.g. in this paper if we take the randomly

TABLE III
THE AVERAGE NUMBER OF GUARDS AND MEAN TIME PROCESSING PROVIDED BY THE EXACT AND APPROXIMATE ALGORITHMS FOR 228 RANDOMLY GENERATED INSTANCES.

		INTERIOR COVERING (IC)				**EDGE COVERING (EC)**			
		Two-phase approach		Approximate approach		Two-phase approach		Approximate approach	
No. rand. instances	Size (n)	Mean no. of guards	Mean time (s)	Mean no. of guards	Mean time (s)	Mean no. of guards	Mean time (s)	Mean no. of guards	Mean time (s)
30	20	3.43	0.13	3.77	0.01	3.07	0.11	3.53	0.00
30	40	6.23	0.13	7.03	0.05	5.77	0.13	6.40	0.02
28	60	9.07	0.16	10.04	0.12	8.11	0.13	9.21	0.06
23	80	12.22	0.21	13.43	0.15	10.52	0.15	12.09	0.10
25	100	14.88	0.36	16.28	0.32	13.44	0.24	14.88	0.18
24	200	29.50	1.80	33.04	1.60	25.54	1.08	29.38	0.84
21	300	44.00	4.37	48.90	4.28	39.05	3.16	43.00	3.14
24	400	58.83	8.65	66.08	7.89	51.46	6.17	57.67	6.25
23	500	72.35	15.41	81.17	13.56	63.78	11.68	72.13	9.97

generated polygon such as RI-2000-5, we make earnings of 41 cameras. These earnings are not only referred to money for purchasing of additional cameras as well as on the savings of other resources. For example, each object covered by cameras requires electricity to power them, as well as specific hardware resources, such as external memory, which is used to store images (usually 30 frames per second) obtained via cameras daily. Based on the number of guards necessary to cover the boundary of a polygon, we can conclude that the approximate method is not able to find the global optimum. In contrast, the exact two-phase approach is capable of doing it in a short period. We have earlier shown on the example of the polygon shown in Figure 1 that edge covering (EC) is not the same as interior covering (IC). Namely, more guards are needed to perform interior covering compared to edge covering as we can see in Table Table II. For instance, in the case of polygon RI-2000-1, the exact two-phase method needs 296 cameras for interior covering, while it costs only 260 cameras for edge covering. Based on the results shown in Table II, we can note that also for interior surveillance of the polygon, the exact method yields a better solution (a smaller number of guards). On the other hand, the approximate approach is usually get trapped in some local optima, and as a consequence of it does not generate the optimal number of cameras.

In order to show the real robustness of the proposed methods, we tested them for a reasonably large dataset, i.e. for a dataset composed of 228 randomly generated simple nonconvex polygons, and the results obtained were saved in Table III. The simulation results show that the exact method gets in average better quality solutions compared with the approximate one for all sizes of the polygon. Other words, for both interior coverage and edge coverage, the mean number of cameras increases linearly concerning the size of vertices (n), so that a growth rate of the cameras being noticeably slower in the exact method compared to the price of growth generated by the approximate comparative approach. Also, by considering produced experimental results in Table III, we can conclude for all versions of polygon coverage, both exact and approximate methods are comparable in terms of CPU execution time, so that the approximate method being negligibly faster than the exact one.

Based on the experimental analysis, it can be concluded the exact method presents an appropriate practical tool that with a minimal number of cameras can cover the interior of the polygon as well as its hull, which has direct applications in security systems, computer graphics, computer vision, and other branches of industry.

V. CONCLUSION

In this paper, we studied the problem of guarding a simple nonconvex polygon and proposed four versions of algorithms for its solving. Quality of the proposed methods was tested throughout 268 randomly generated instances. Based on the obtained results, it can be concluded that our exact two-phase algorithm is convenient for this task, and it produces excellent overall performance. Also, our two-phase approach proved to be robust, in the sense that it was able to tackle different instances from a broad range of randomly generated. Since the first phase of our two-phase method is computationally expensive, in future work, we will investigate the efficient techniques in order to tackle these drawbacks. This further says that the improvements of the two-phase algorithm can be achieved. Also, we will consider other types of polygons with and without holes, such as orthogonal polygons, Von Koch polygons, and so for.

REFERENCES

[1] E. Tuba, I. Tuba, D. Dolicanin-Djekic, A. Alihodzic, and M. Tuba, "Efficient drone placement for wireless sensor networks coverage by bare bones fireworks algorithm," in *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, 2018. doi: https://doi.org/10.1109/ISDFS.2018.8355349 pp. 1–5.

[2] E. Tuba, R. Capor-Hrosik, A. Alihodzic, and M. Tuba, "Drone placement for optimal coverage by brain storm optimization algorithm," in *Hybrid Intelligent Systems*. Cham: Springer International Publishing, 2018. doi: https://doi.org/10.1007/978-3-319-76351-4%5F17 pp. 167–176.

[3] A. Elnagar and L. Lulu, "An art gallery-based approach to autonomous robot motion planning in global environments," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 2079–2084.

[4] J. O'Rourke, *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.
[5] J. O'Rourke and K. Supowit, "Some np-hard polygon decomposition problems," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 181–190, March 1983. doi: https://10.1109/TIT.1983.1056648
[6] D. Lee and A. Lin, "Computational complexity of art gallery problems," *IEEE Transactions on Information Theory*, vol. 32, no. 2, pp. 276–282, 1986. doi: https://10.1109/TIT.1986.1057165
[7] M. J. Katz and G. S. Roisman, "On guarding the vertices of rectilinear domains," *Computational Geometry*, vol. 39, no. 3, pp. 219 – 228, 2008. doi: https://doi.org/10.1016/j.comgeo.2007.02.002
[8] D. Schuchardt and H. Hecker, "Two np-hard art-gallery problems for ortho-polygons," *Mathematical Logic Quarterly*, vol. 41, no. 2, pp. 261 – 267, 1995. doi: https://
[9] V. Chvátal, "A combinatorial theorem in plane geometry," *Journal of Combinatorial Theory, Series B*, vol. 18, no. 1, pp. 39 – 41, 1975. doi: https://doi.org/10.1016/0095-8956(75)90061-1
[10] I. Bjorling-Sachs and D. L. Souvaine, "An efficient algorithm for guard placement in polygons with holes," *Discrete & Computational Geometry*, vol. 13, p. 77–109, January 1995. doi: https://doi.org/10.1007/BF02574029
[11] F. Hoffmann, M. Kaufmann, and K. Kriegel, "The art gallery theorem for polygons with holes," in *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*, October 1991. doi: https://10.1109/SFCS.1991.185346 pp. 39–48.
[12] E. Györi, F. Hoffmann, K. Kriegel, and T. Shermer, "Generalized guarding and partitioning for rectilinear polygons," *Computational Geometry*, vol. 6, no. 1, pp. 21 – 44, 1996. doi: https://doi.org/10.1016/0925-7721(96)00014-4
[13] S. Fisk, "A short proof of chvátal's watchman theorem," *Journal of Combinatorial Theory, Series B*, vol. 24, no. 3, p. 374, 1978. doi: https://doi.org/10.1016/0095-8956(78)90059-X
[14] CPLEX, *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual V 12.7*. International Business Machines Corporation, 2017. [Online]. Available: https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.1/ilog.odms.studio.help/pdf/usrcplex.pdf
[15] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co., 1990. ISBN 0716710455
[16] E. Balas and A. Ho, *Set covering algorithms using cutting planes, heuristics, and subgradient optimization: A computational study*. "Springer Berlin Heidelberg, 1980, pp. 37–60. ISBN 978-3-642-00802-3
[17] T.-P. Shuai and X.-D. Hu, "Connected set cover problem and its applications," in *Algorithmic Aspects in Information and Management*. Springer Berlin Heidelberg, 2006. doi: https://doi.org/10.1007/11775096%5F23. ISBN 978-3-540-35158-0 pp. 243–254.
[18] M. L. Fisher and P. Kedia, "Optimal solution of set covering/partitioning problems using dual heuristics," *Management Science*, vol. 36, no. 6, pp. 674–688, 1990. doi: https://doi.org/10.1287/mnsc.36.6.674
[19] E. Balas and M. C. Carrera, "A dynamic subgradient-based branch-and-bound procedure for set covering," *Operations Research*, vol. 44, pp. 875–890, December 1996. doi: https://doi.org/10.1287/opre.44.6.875
[20] A. Caprara, P. Toth, and M. Fischetti, "Algorithms for the set covering problem," *Annals of Operations Research*, vol. 98, no. 1-4, p. 353–371, December 2000. doi: https://doi.org/10.1023/A:1019225027893
[21] P. Laborie, J. Rogerie, P. Shaw, and P. Vilím, "Ibm ilog cp optimizer for scheduling," *Constraints*, vol. 23, pp. 210–250, April 2018. doi: https://doi.org/10.1007/s10601-018-9281-x
[22] J. O'Rourke, *Computational Geometry in C*. Cambridge University Press, September 1998.
[23] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational geometry: algorithms and applications*, 3rd ed. Springer, Berlin, Heidelberg, 2008. ISBN 978-3-540-77973-5
[24] T. Auer and M. Held, "Heuristics for the generation of random polygons," August 1996, pp. 38–43.
[25] S. Sadhu, S. Hazarika, K. K. Jain, S. Basu, and T. De, "Grp_ch heuristic for generating random simple polygon," in *Combinatorial Algorithms*. Springer Berlin Heidelberg, 2012. doi: https://doi.org/10.1007/978-3-642-35926-2978-3-642-35926-2 pp. 293–302.