


## Research Article

# A More Accurate Metaheuristic Approach for the Art Gallery Problem

Bahram Sadeghi Bigham<sup>1\*</sup>, Sahar Badri<sup>2</sup>, Maziar Zahedi-Seresht<sup>3</sup>, Shahrzad Khosravi<sup>3</sup>, Nazanin Padkan<sup>4</sup>

<sup>1</sup>Department of Computer Science, Faculty of Mathematical Sciences, Alzahra University, Tehran, Iran

<sup>2</sup>Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila, Italy

<sup>3</sup>Department of Quantitative Studies, University Canada West, Vancouver, Canada

<sup>4</sup>Department of Information Engineering and Computer Science, University of Udine, Udine, Italy

E-mail: [b\\_sadeghi\\_b@alzahra.ac.ir](mailto:b_sadeghi_b@alzahra.ac.ir)

**Received:** 22 November 2024; **Revised:** 19 January 2025; **Accepted:** 28 February 2025

**Abstract:** The Art Gallery problem is one of the most important non-deterministic polynomial (NP)-hard optimization problems in computational geometry, with many applications in localization, robotics, telecommunications, etc. The goal of the Art Gallery problem is to find the minimum number of guards needed within a simple polygon to observe and protect its entirety. There are several approaches to solving the Art Gallery problem, and this paper presents an efficient method based on the Particle Filter algorithm, which solves the most fundamental case of the problem in a nearly optimal manner. Experimental results on random polygons generated show that the new method is more accurate, providing solutions that are, on average, 9.94% better than Bottino's results for the same sample set. The approach was also applied to four groups of random orthogonal polygons and compared with the optimal solution. Results show that the new method finds the optimal solution with a 0.16% error. Furthermore, this paper discusses the impact of resampling and particle numbers in minimizing runtime.

**Keywords:** art gallery problem, optimization, heuristic algorithm, localization, particle filter

**MSC:** 65D18, 68T20, 11Y16

## 1. Introduction

The Art Gallery Problem has been studied for several decades as one of the most important optimization problems in computational geometry. The classical art gallery problem was posed by Victor Klee in 1973 [1]. It has many applications in minimizing landmarks, trilateration [2], pose estimation [3], and robot localization [4]. Imagine an art gallery or museum with many priceless objects like plantings and sculptures. The gallery needs to be equipped with some number of guards (Guards are positioned to ensure visibility of all areas within the polygon) to supervise all areas within the polygon. to supervise the whole gallery. If the number of guards is small, some parts of the gallery may not be monitored. Having many guards, would be expensive and difficult to maintain. The main goal is to find the minimum number of guards to protect all the parts of the gallery.

This problem can be written in a geometric version: assume the art gallery as a simple polygon, designated  $P$  that may or may not have holes. The goal is to find the minimum number of guards to cover  $P$  [1]. These guards can be positioned

at vertices (vertex-guard), on edges (edge-guard), or anywhere within the polygon (point-guard). There are several types of the art gallery problems. The 2-guard [5] problem in a simple polygon  $P$  requires two guards on the boundary of  $P$  from the starting vertex  $u$  to the ending vertex  $v$ . One of the guards goes around the boundary clockwise and the other counterclockwise, so they are able to see each other. The line segment connecting the two guards lies completely in the  $P$  all the time [5].

The 3-guard problem in a simple polygon  $P$  asks if three guards can move from vertex  $u$  to vertex  $v$ , with the first and third guards placed on two boundary chains of  $P$  from  $u$  to  $v$ , and the second guard always visible to the other two inside  $P$  [6]. The art gallery problem is an  $NP$ -hard problems. Lee and Lin [7] showed a reduction from 3-SAT and proved that this problem is  $NP$ -hard. These problems cannot be solved in polynomial time, so finding the exact solution to them is time-consuming. Therefore, approximation and metaheuristic algorithms are used to find approximate solutions to these hard problems. These algorithms can find near-optimal solutions in less times.

In 2018, Abrahamsen et al. [8] proved that the problem is  $\exists\mathbb{R}$ -complete. In [9], they explore the complexity of the Point-Boundary Art Gallery Problem, showing that it is  $\exists\mathbb{R}$ -hard. This problem involves finding configurations of guards to protect the walls of an art gallery shaped like a polygon. Their results shed light on the algebraic constraints involved in solving this problem and provide insights into its complexity. The Point-Boundary Art Gallery Problem, a variant of the classic Art Gallery Problem, involves guarding a polygonal art gallery's boundary with a minimum number of guards. Guards must be positioned only along the boundary, ensuring every boundary point is visible to at least one guard. In [10], the authors discuss the complexity of the Art Gallery Problem and how it can be encoded as a system of polynomial equations over real numbers. They prove that the problem is  $\exists\mathbb{R}$ -completeness, showing its computational challenge.

The class  $\exists\mathbb{R}$ -complete consists of problems that can be reduced in polynomial time to the problem of deciding whether a system of polynomial equations with integer coefficients and any number of real variables has a solution [8]. The art gallery problem has been widely studied over the years. In 1975, Chvatal established a theorem known as Chvatal's Art Gallery Problem [11]. It states that  $M = \left\lfloor \frac{n}{3} \right\rfloor$  guards are always sufficient and sometimes necessary to protect an  $n$ -gon art gallery. In 1978, another simple proof based on the polygon triangulation and coloring of vertices was proposed by Fisk [12]. Based on Fisk's proof, Avis and Toussaint [13] proposed an  $O(n \log n)$  time divide-and-conquer algorithm for placing guards in a simple polygon.

The art gallery problem also has been studied for orthogonal polygons [14]. Orthogonal polygons, have internal angles of  $90^\circ$  or  $270^\circ$  [15]. In 1983, Kahn et al. [16] proved that  $M = \left\lfloor \frac{n}{4} \right\rfloor$  guards are always sufficient and sometimes necessary to protect an orthogonal polygon (gallery). Later in 1983, O'Rourke [14] proved again that the maximum number of guards in orthogonal polygons are  $M = \left\lfloor \frac{n}{4} \right\rfloor$ . Katz and Roisman [17] proved that the art gallery problem for orthogonal polygons is  $NP$ -hard.

Gosh [18] proposed an approximation algorithm for the vertex-guard problem. Bajuelos et al. [19] presented a metaheuristic genetic algorithm for solving the vertex-guard problem. Amit et al. [20] introduced a heuristic algorithm for solving the point-guard problem. In 2011, Bottino et al. [21] introduced optimal solutions for the point-guard problem and two years later, Tozoni et al. [22] introduced an algorithm that successfully found the exact solution when tested on a large collection of instances from publicly available benchmarks, but whose convergence could not be in general guaranteed. In 2021, Bigham et al. [23] proposed a new metaheuristic approach for the art gallery problem, but this article here mentioned a more accurate and extended state.

Furthermore, Tozoni et al. [24] proposed an algorithm based on integer linear programming (ILP) to solve the art gallery problem. According to the paper, the algorithm optimally solves the problem with point guards and shows an efficacy rate of more than 98%. The optimal solutions are obtained in less than an hour of computational time.

In 2012, Kröller et al. [25] introduced a procedure based on linear programming (LP) to obtain the bounds of the art gallery problem, and this procedure, in the case of convergence, provides optimal solutions. In 2015, Fekete et al. [26] introduced another method for solving the art gallery problem that has a better speed and more quality solutions in comparison with [25].

This article proposed a new method for solving the art gallery problem based on the particle filter algorithm (Each particle represents a potential configuration of guard positions within the polygon) with the highest accuracy. The

experimental results on random polygons created by Bottino et al. [21] show that the proposed method is more accurate with the same or fewer guards. The simulation results are also shown for four groups of random orthogonal polygons. Finally, we discuss resampling (Resampling involves generating new particles around high-weight particles to improve solution accuracy.) and particle numbers to minimize the run time. The achievements of this article can be used in issues such as 2-guard, trilateration, simultaneous localization and mapping (SLAM), and pose estimation.

This method can also have practical applications in engineering companies and IoT-focused organizations. By simulating complex environments and optimizing the placement of sensors or cameras, the proposed approach helps achieve full coverage with minimal resources. This leads to cost savings and improved efficiency, particularly in monitoring and managing industrial or internet of things (IoT) environments.

**Our Contribution:** This work presents a new method to solve the Art Gallery Problem using a particle filter algorithm. It improves previous methods by adding a better resampling process that helps particles spread more effectively, leading to more accurate guard placement. The new algorithm is faster and provides better results when tested on random and orthogonal polygons. Compared to other methods, it shows a 9.94% improvement for arbitrary polygons and finds nearly perfect solutions for orthogonal cases. This approach also creates a foundation for applying the algorithm to other versions of the Art Gallery Problem, like 2-Guard and 3-Guard setups.

The remaining sections of this paper are organized as follows: Some important definitions related to the article are studied in Section 2. Then, in Section 3 a new metaheuristic method for the art gallery problem will be discussed and the simulation results are studied in Section 4. The conclusion and future work are discussed in Section 5. All symbols and variables used throughout the paper are introduced in Table 1.

**Table 1.** Symbols and variables used throughout the article

Symbols	Description
$P$	A simple polygon
$n$	The number of polygon vertices
$N$	The number of particles
$M$	The number of guards in the worst case
$ P $	Polygon's area
$k$	The counter of guard numbers
$X_i$	A particle
$[x_i, y_i]$	A random point
$X$	A set of $N$ particles
$VP(X_i)$	Visibility polygon for particle $X_i$
$VP([x_i, y_i])$	Visibility polygon for point $[x_i, y_i]$
$w_i$	Weight or importance of a particle
$vp(X_i)$	The ratio of visibility area for particle $X_i$
$RN$	The resampling number
$vp_{max}$	The best ratio for visibility area
$Best(X)$	The best particle (with most coverage) in $X$
$\tau$	Threshold or the acceptable percentage for entering the resampling stage

## 2. Basic definitions

In this section, some theoretical basics related to the proposed algorithm are going to be introduced.

## 2.1 Visibility polygon

The visibility polygon is one of the foremost critical issues in computational geometry. Given a simple polygon  $P$ , two points are said to be visible to each other if the line segment that joins them does not cross any obstacles [27]. Polygon  $P$  will be covered by point  $q$ , if it is entirely visible from  $q$ . In 1980, the first algorithm for computing the visibility polygon was introduced by Avis and El Gindy [28]. After that, different versions of this problem were introduced, each having their own applications [29].

Consider  $P$  as an  $n$ -gon and  $q$  as a point in it. The goal is to compute the visibility polygon or the covered area. In order to compute the visibility polygon in concave polygons, let  $P = (v_0, v_1, \dots, v_n)$  and  $E = (v_0v_1, \dots, v_{n-1}v_n)$  be, respectively, the polygon's vertices and edges. Asano et al. [30] introduced an algorithm to compute the visibility polygon in concave polygons using the Sweep Line algorithm which the complexity of it is  $O(n \log n)$ .

## 2.2 Particle filter

The particle filter is a recursive algorithm that was introduced in 1996 by Del Moral [31] and has been widely used in diverse fields such as robotics [32]. The particle filter, which approximates the posterior distribution  $bel(x_t)$  by a set of random state samples (particles) drawn from this posterior, serves as a nonparametric approach to the Bayes filter [33]. Clearly, the set  $X_t$  of  $M$  particles are denoted:

$$X_t = ([x_t^{[1]}, \dots, x_t^{[M]}])$$

Each particle  $x_t^{[m]}$  ( $1 \leq m \leq M$ ) is an instantiate of the state at time  $t$ , such as the position and orientation of an agent. These particles represent the Bayes filter posterior distribution

$$x_t^{[m]} \sim bel(x_t) = p(x_t | z_{1:t}, u_{1:t})$$

which  $z_{1:t}$  is the set of all previous observations and  $u_{1:t}$  the set of all previous actions. Conceptually, the particle filter consists of the following steps [33]:

**Action update:** Sample  $M$  particles  $\bar{X}_t = ([x_t^{[1]}, \dots, x_t^{[M]}])$  at random from the state transition distribution  $p(x_t | z_{1:t}, u_{1:t})$ . These particles form a nonparametric approximation of  $\bar{bel}(x_t)$ , where  $\bar{bel}(x_t) = p(x_t | z_{1:t}, u_{1:t})$  is the prediction posterior before considering observation  $z_t$ .

**Calculate weights:** Calculate an importance factor (weight)  $w_t^{[M]} = p(z_t | x_t^{[M]})$  for each particle  $x_t^{[M]}$  given observation  $z_t$ . Each weighted particle  $\langle x_t^{[M]}, w_t^{[M]} \rangle$  to the temporary particle set  $\bar{x}_t$ .

**Resample:** Draw  $M$  particles (with replacement) from  $\bar{x}_t$ , and add to the particle set  $X_t$ . The probability of drawing the particle  $x_t^{[M]}$ , is proportional to the corresponding weight  $w_t^{[M]}$  and often results in the inclusion of duplicate particles.

The particle filter uses a set of particles to find feasible solutions to a problem, with each particle representing a solution. Initially, particles are distributed randomly within a polygon, all having equal weight. Over time, particle weights are updated based on information gathered, and low-weighted particles are replaced by new ones around high-weighted particles, a process known as resampling. The following section introduces the proposed algorithm for the art gallery problem based on these theoretical foundations.

## 3. New approach for the art gallery problem using the particle filter

Since the proposed algorithm is based on the particle filter algorithm, the particles firstly are uniformly distributed in the polygon space. For example, if 50 particles are dispersed, each particle has an equal probability of  $\frac{1}{50}$  of being a

solution, which represents the particle's initial weight or importance. As the algorithm progresses, the visibility polygon of each particle is computed, and weights are updated based on the ratio of the area covered by the particle to the total polygon area. Particles with lower weights are eliminated, while high-weight particles are used to generate new particles around them. This resampling process ensures that the algorithm focuses on promising regions within the polygon, improving efficiency and accuracy. The resampling continues until the optimal solution is found or the maximum number of iterations is reached. Although each particle is regarded as a feasible solution in the particle filter algorithm, each particle is considered as a set of points (guards) in the proposed algorithm. Consider  $n$ -gon  $P$  as the input of the algorithm.

As mentioned before,  $M = \left\lfloor \frac{n}{3} \right\rfloor$  ( $M = \left\lfloor \frac{n}{4} \right\rfloor$  for orthogonal polygons) guards are needed to protect an  $n$ -gon in the worst case. The number of guards in each stage is represented by  $k$ . The maximum value of  $k$  is  $M$ . The binary search is done in each stage until reaches an optimal solution.

### 3.1 Binary search

The binary search algorithm is used to find the optimal solution (optimal  $k$  and positions) in an ordered array (this set is started from 1 to  $M$ ). In the binary search, the search space is split in half  $\left(\frac{M}{2}\right)$ . First,  $\frac{M}{2}$  guards are checked whether they can cover the polygon or not. If  $\frac{M}{2}$  guards are enough to cover the polygon, the first split  $\left(\left[1, 2, \dots, \frac{M}{2}\right]\right)$  is divided into two minor sets again and the binary search will be continued. If  $\frac{M}{2}$  guards are not adequate, the second split  $\left(\left[\frac{M}{2}, \dots, M\right]\right)$  is divided into two minor sets again and the binary search will be continued. This process is continued until the optimal number of guards is obtained or all the elements are checked.

Let  $X$  be a set of  $N$  particles:

$$X = (X_1, X_2, \dots, X_N). \quad (1)$$

Then, the particles are distributed monotonously in the polygon and the particle weights are defined as follows:

$$w_i = \frac{1}{N} \quad st \quad 1 \leq i \leq N. \quad (2)$$

Each particle is regarded as a set of  $M$  random points (guards) in the polygon such as an  $M$ -footed spider as follows:

$$\begin{aligned} X_1 &= ([x_1^{[1]}, y_1^{[1]}], [x_2^{[1]}, y_2^{[1]}], \dots, [x_{[M]}^{[1]}, y_{[M]}^{[1]}]) \\ X_2 &= ([x_1^{[2]}, y_1^{[2]}], [x_2^{[2]}, y_2^{[2]}], \dots, [x_{[M]}^{[2]}, y_{[M]}^{[2]}]) \\ &\vdots \\ X_N &= ([x_1^{[N]}, y_1^{[N]}], [x_2^{[N]}, y_2^{[N]}], \dots, [x_{[M]}^{[N]}, y_{[M]}^{[N]}]). \end{aligned} \quad (3)$$

Using Asano Algorithm [30], the visibility polygon of each particle is computed by the union of the visibility polygons of the set of its points:

$$\begin{aligned}
VP(X_1) &= VP([x_1^{[1]}, y_1^{[1]}]) \cup VP([x_2^{[1]}, y_2^{[1]}]) \cup VP... \cup VP([x_{[M]}^{[1]}, y_{[M]}^{[1]}]) \\
VP(X_2) &= VP([x_1^{[2]}, y_1^{[2]}]) \cup VP([x_2^{[2]}, y_2^{[2]}]) \cup VP... \cup VP([x_{[M]}^{[2]}, y_{[M]}^{[2]}]) \\
&\vdots \\
VP(X_N) &= VP([x_1^{[N]}, y_1^{[N]}]) \cup VP([x_2^{[N]}, y_2^{[N]}]) \cup VP... \cup VP([x_{[M]}^{[N]}, y_{[M]}^{[N]}]).
\end{aligned} \tag{4}$$

Then, the particle weights are being updated:

$$\begin{aligned}
vp(X_1) &= \frac{|VP(X_1)|}{|P|} \\
vp(X_2) &= \frac{|VP(X_2)|}{|P|} \\
&\vdots \\
vp(X_N) &= \frac{|VP(X_N)|}{|P|}.
\end{aligned} \tag{5}$$

For each particle  $X_i$ , the weight  $vp(X_i)$  is obtained from the ratio of the area of the visibility polygon for each particle ( $|VP(X_i)|$ ) to the area of the polygon  $|P|$ . Then, the resampling stage is started and repeated until  $vp(max) = 1 \left( \frac{|VP(X_i)|}{|P|} = 1 \right)$ . The counter of the resampling stages is  $j$ . It is started from 1 and is updated incrementing by 1 at each stage until it reaches the maximum defined number  $RN$  (the defined resampling number). In this stage, low-weighted particles will be eliminated and reproduced again around the high-weighted particles. For example, high-weighted particles whose their  $vp(X_i)$  are more than a specific threshold such as  $\tau = 0.9$  (the particles that cover more than 90% of the polygon) are sampled and the particles that cover the polygon lower than 90% will be omitted. Then, the omitted particles are reproduced around the high-weighted particles. Therefore, the number of particles  $N$  is constant and their locations are changed.

In the following, the visibility polygon for each particle is computed using Asano Algorithm [30] (the weights are updated). Another resampling stage is done again (The maximum number of the resampling stages is limited, for example 20). A particle may be found that  $vp = 1$  (the particle can cover the polygon). Then, this particle will be saved as an optimal solution. Regard that this particle is not unique. This procedure will be performed for all particles using the binary search. It is possible to obtain no particle after completing all the resampling stages. In this situation, the particle that covered the polygon in the last stage is regarded as the optimal solution. The proposed approach is presented in the algorithm. This algorithm finds the minimum number of required guards to protect a polygon using the particle filter algorithm.

Reproducing the omitted particles around other particles is based on their weights. The omitted particles are reproduced more around a particle which has a higher  $vp$ . The best particle  $X$  is obtained when  $vp(X_i) = 1$  which is

indicative a feasible solution to the problem. No other particles are needed to be checked. It may not be necessary to complete all the resampling stages.

If  $vp(max) \neq 1$ , all the resampling stages  $RN$  are completed, but the particle that covers the polygon is not obtained yet. Therefore, the particle that covered the polygon in the last stage is selected as the optimal solution (Best(X)).

**Algorithm 1** Minimum number of guards for the Art Gallery Problem.

**Require:** Simple polygon  $P$  with  $n$  vertices

**Ensure:** Location of the minimum number of guards to guard  $P$

```

1:  $kmin = 1$ ,  $kmax = \lfloor n/3 \rfloor$  and  $k = \frac{kmax + kmin}{2}$ 
2: while 1 do
3:   Generate  $X = \{X_1, X_2, \dots, X_N\}$ 
4:   for  $j = 1$ ;  $j = RN$ ;  $j++$  do
5:      $vp_{max}(X) = 0$ 
6:     for  $i = 1$ ;  $i = N$ ;  $i++$  do
7:        $vp(X_i) = \frac{|VP(X_i)|}{|P|}$ 
8:       if  $vp(X_i) > vp_{max}(X)$  then
9:          $vp_{max}(X) = vp(X_i)$ 
10:      end if
11:      if  $vp_{max}(X) == 1$  then
12:         $BestX = X_i$ 
13:         $BestX = X_i$ 
14:      end if
15:    end for
16:    if  $vp_{max}(X) == 1$  then
17:       $kmax = k$ 
18:       $kopt = k$ 
19:      break(); /* No need to other resampling */
20:    else
21:       $kmin = k$ 
22:    end if
23:  end for
24: end while
25: Return( $kopt$ ); /* When any particle cannot cover  $P$  with  $k$  guards*/
26: Return(BestX) /* The best result for previous  $k$  */

```

The runtime complexity of our proposed algorithm has been explained in detail in the paper. The algorithm has a complexity of  $O(N \log N)$  for  $N$  particles, and the visibility polygon for an  $n$ -gon is computed in  $O(n \log n)$  time. In the worst case, the binary search method requires  $O(\log M)$  steps, where  $M = 3n$  (for orthogonal polygons,  $M = 4n$ ). The algorithm also runs for  $RN$  resampling steps, which is at most  $O(n/3)$  (or  $O(n/4)$ ). Therefore, the overall complexity of the algorithm is  $O(n^3 \log^2 n)$ . This means the algorithm may run slower as the number of vertices ( $n$ ) increases, which could affect its scalability for very large polygons or many particles.

## 4. The experimental results

The simulation results are presented here to evaluate the proposed algorithm.

Example 1: According to [20], a simple polygon with 18 vertices is considered. The polygon's area ( $P$ ) is  $92,243.5m^2$  and its coordinates are as follows:

$$P = ([558, 497], [513, 148], [477, 413], [439, 413], [403, 150], [384, 410],$$

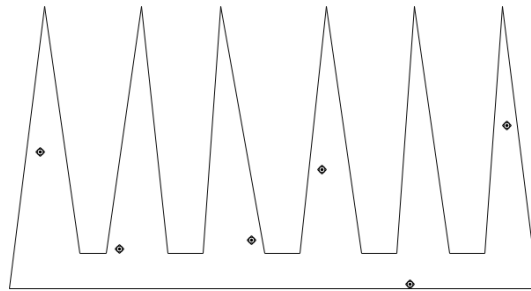
$$[339, 409], [298, 152], [267, 409], [228, 409], [192, 151], [161, 412],$$

$$[124, 412], [80, 151], [74, 413], [52, 413], [25, 147], [11, 497]).$$

According to  $M = \left\lfloor \frac{18}{3} \right\rfloor = 6$  (6 the maximum dimension for each particle) and the binary search algorithm, the proposed procedure for  $[1, 2, 3, \dots, 6]$  is as follows:

Firstly, the algorithm is performed on  $3D$  particles. If a particle that covers the polygon can be found, the algorithm will be performed on the first part of the array  $[1, 2, 3]$  using the binary search. Here, the particle is not found, so the second part of the array  $[3, 4, 5, 6]$  is checked using the binary search. In fact, the algorithm is examined for  $5D$  particles. If a  $5D$  particle that covers the polygon is found, then the algorithm will be performed for  $4D$  particles. Since the covering  $5D$  particle is not found here, the algorithm will be checked for  $6D$  particles. After implementing the algorithm on particle dimension using the binary search, it is concluded that the algorithm for  $6D$  particles (6 guards) reaches the optimal solution. The guards' locations for the  $6D$  particle are shown in Figure 1. Their coordinates are shown in

$$Best(x) = ([83, 402], [22, 276], [227, 414], [514, 231], [320, 360], [404, 487]).$$



**Figure 1.** The simulation results of the proposed algorithm for the given polygon in [20] shows that 6 guards (small black diamonds) are needed to be selected in this polygon

Example 2: According to Amit et al. [20], another simple polygon with 21 vertices is considered. The polygon's area ( $P$ ) is  $91,201.5m^2$  and its coordinates are as follows:

$$P = ([475, 512], [146, 512], [284, 486], [146, 480], [147, 366], [118, 415]$$

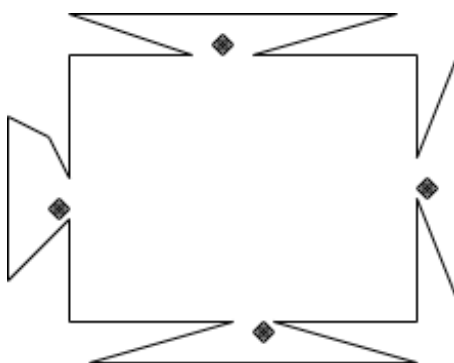
$$[99, 288], [128, 295], [146, 336], [151, 226], [256, 226], [151, 191], [405, 190],$$

$$[294, 226], [438, 223], [437, 316], [475, 230], [472, 418], [437, 343], [428, 474], [314, 480]).$$



According to  $M = \left\lfloor \frac{21}{3} \right\rfloor = 7$  and the binary search algorithm, for  $[1, 2, 3, \dots, 7]$  the proposed procedure is as follows. Firstly, the algorithm is performed on the  $4D$  particles. The  $4D$  particle that covers the polygon is found, so the first part of the array  $[1, 2, 3, 4]$  is examined using the binary search. The algorithm tries to find a  $2D$  particle to cover the polygon. Here, the  $2D$  particle to cover the polygon that covers the polygon is not found. Then, the algorithm is checked to detect a  $3D$  particle to cover the polygon and this particle is not found either. After implementing the algorithm on particle dimension using the binary search, it is concluded that the algorithm for  $4D$  particles (4 guards) reaches the optimal solution. The guards' locations for the  $4D$  particle are shown in Figure 2. Their coordinates are

$$Best(X) = ([116, 348], [208, 209], [435, 324], [292, 495]).$$



**Figure 2.** The simulation results of the proposed algorithm for the given polygon in [20] shows that 4 guards (small black diamonds) are needed to be located in this polygon

Using results obtained for the above examples and comparing them with Amit et al. [20], it can be inferred that the results are approximately optimal.

A comparison between the average accuracy of the proposed and Bottino et al.'s [21] algorithms for random polygons is shown in Table 2. In this comparison, 20 random 30-gons, 40-gons, 50-gons, and 60-gons are studied. These polygons are the same as the ones in [21] and some of them are shown in Figure 3.

**Table 2.** A comparison of the results between the proposed method and Bottino et al. [21]

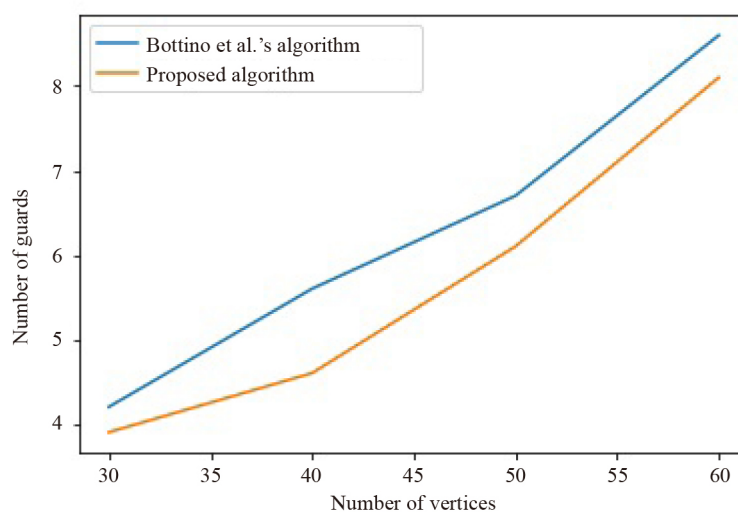
$n$	Proposed method	Bottino et al.'s algorithm [21]	Improvement percentage
30	3.9	4.2	7.14
40	4.6	5.6	17.85
50	6.1	6.7	8.95
60	8.1	8.6	5.81



**Figure 3.** Some of the studied polygons and the locations of the guards using the proposed method

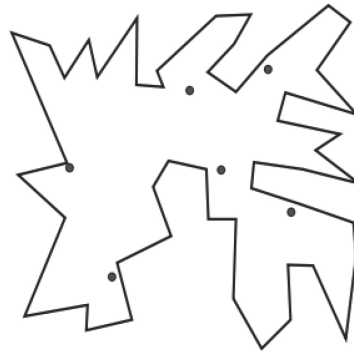
Table 2 shows that the average optimal number of guards for random polygons in the proposed method is 9.94% (**average improvement percentage**) better than Bottino's answers for the same sample set that is shown in Figure 4.

Table 2 shows that in the proposed method the average optimal number of guards for random polygons average 9.94% (average improvement percentage) better than Bottino's answers for the same sample set that is shown graphically in Figure 4.

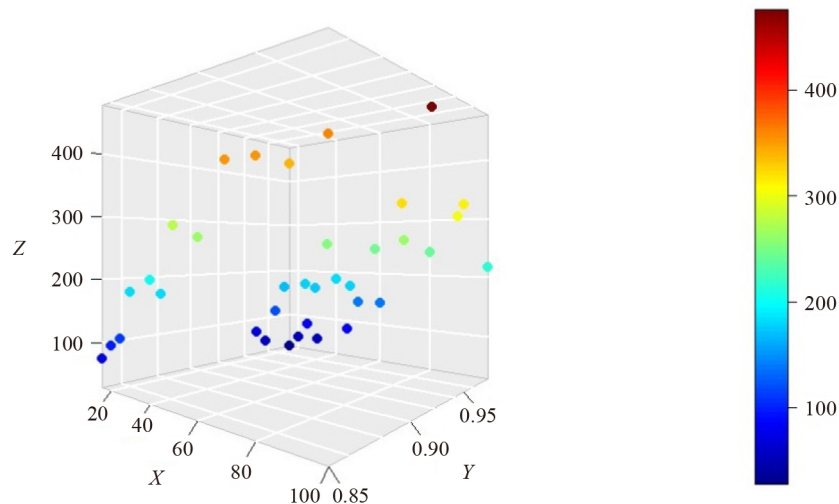


**Figure 4.** Comparison of the average guard number of the proposed algorithm and the method in [21]

Using different numbers of particles and thresholds ( $\tau$ ), the proposed algorithm is examined on an arbitrary 50-gon (Figure 5). The optimal solutions (6 Guards) are obtained at different times and illustrated in Figure 6. As shown there, the X,Y, and Z axes indicate the number of particles, threshold ( $\tau$ ), and time, respectively. The color spectrum on the right shows the time in seconds and its color changes from blue to red with the increasing time. The points in the range of 1 to 100 seconds are blue. As the time increases, the color changes from dark blue to light blue.



**Figure 5.** According to the simulation results of the proposed algorithm, 6 guards are needed to be selected in this 50-gon

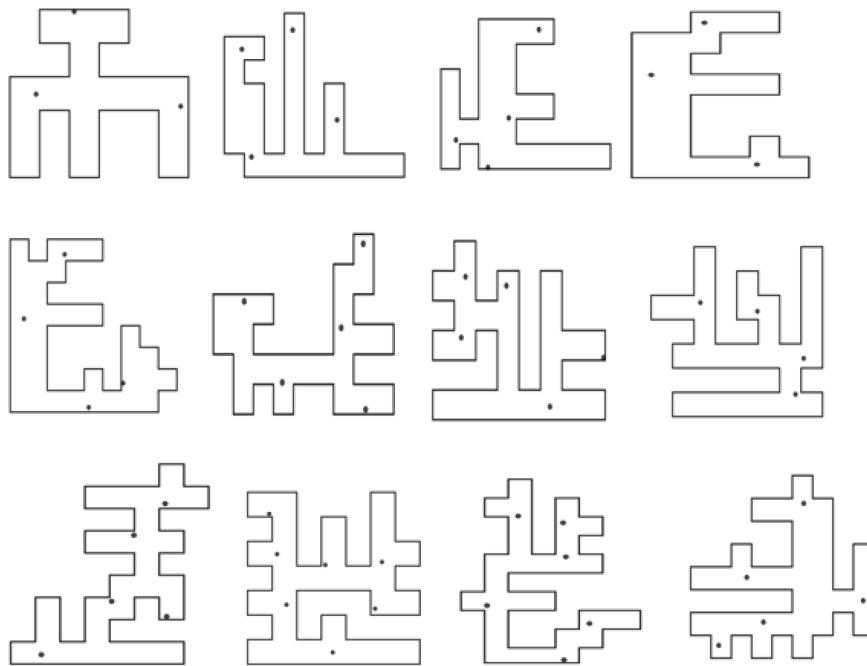


**Figure 6.** The optimal solutions at different time intervals

The darkest red point shows the optimal solution at the latest time (476 s) that is obtained when the number of particles is 90 and  $\tau = 0.95$ . The darkest blue point indicates the optimal solution with the minimum running time (28 s) and is obtained when the number of particles is 15 and  $\tau = 0.98$ . Both the darkest red and blue points have found 6 guards for covering the 50-gon. The choice of parameters, specifically the threshold ( $\tau$ ) and number of particles ( $N$ ), is justified through a balance between solution quality and computational efficiency, as higher  $\tau$  values lead to faster results with acceptable accuracy. These parameter choices were validated through sensitivity analysis, showing that the results (Figure 6) with  $[N = n/3]$  and  $\tau = 0.98\%$  are acceptable settings for the proposed algorithm. Table 3 shows that the average number of guards for orthogonal polygons in the proposed method is 0.16% (**average error percentage**) worse than the average optimal solutions and some of them are shown in Figure 7.

**Table 3.** The simulation of the proposed algorithm for orthogonal polygons

$n$	Samples	Proposed method	Optimal (average)	Error percentage
10	30	2	2	0
20	30	3.4	3.4	0
30	30	4.76	4.76	0
40	30	6	5.96	-0.67

**Figure 7.** Some of the studied polygons and the locations of guards using the proposed algorithm

Note: If we have a polygon with many vertices, our algorithm will require more complex and extensive computations. This is because, in this case, the maximum dimension for each particle increases in proportion to the number of polygon vertices. As the maximum dimension for each particle increases, the computational load grows significantly, which leads to longer processing times and reduced algorithm performance. In fact, this limitation could harm the algorithm and decrease its effectiveness in complex scenarios.

## 5. Conclusion and future work

The art gallery problem is one of the most vital NP-hard problems in computational geometry applicable in robot localization and SLAM. In this paper, a new heuristic method has been proposed to find the near optimal solution to the art gallery problem using the particle filter. In this method, each particle provides an optimal solution to find the number and locations of the adequate guards that would guard the given polygon. The experimental results show that the method finds a fewer (or equal) number of guards with respect to the previous methods. The new method improves the results presented by Bottino et al. [21] in arbitrary polygons and presents new results in orthogonal ones. Quantum computing shows potential for solving NP-hard problems like the Art Gallery Problem, but it remains impractical for widespread use. In the meantime, heuristic methods, such as our proposed approach, are essential for efficient, near-optimal solutions. In

future work, guarded guard, 2-Guards, and 3-Guards problems may be considered. Also, one can extend our results to solve the problem of achieving maximum coverage for a given polygon using a fixed number of guards. In addition, future advancements may explore hybrid methods combining AI and cloud computing for dynamic environments, enhancing scalability and adaptability for real-time industrial or IoT applications. We recognize that sensitivity analysis could improve our understanding of the algorithm's performance, and we plan to include it in future work to strengthen its robustness and evaluation.

## Acknowledgement

The authors would like to express their sincerely gratitude to Dr. A. Bottino for providing the data that helped us make a better comparison.

This research has been facilitated by the Data Science Lab of the Faculty of Mathematical Sciences at Alzahra University.

## Authors' contributions

Dr. Sadeghi Bigham has contributed to the main idea of the algorithm and the technique of comparing it with the Bottino algorithm and the topics mentioned in the article. He has played an important role in supervising all stages of the article, including algorithm, the way of writing, literature, to ensure that the literature review presented is up to date. The first draft manuscript was written under the supervision and participation of Dr. Sadeghi Bigham. All authors commented on previous versions of the manuscript and read and approved the final manuscript.

Sahar Badri has been contributed to implementing the algorithm and programming and the results obtained and comparing it with the Bottino et al.'s algorithm, related work done in this area, the shapes and literature examined in the article. The first manuscript of the article was written by Sahar Badri under the supervision of Dr. Sadeghi Bigham. All authors commented on previous versions of the manuscript and read and approved the final manuscript.

After the initial version of this paper was uploaded to the arXiv by three other authors, Mazyar Zahedi-Seresht identified several issues and informed the authors. Subsequently, in a close collaboration, he and Shahrzad Daneshvar meticulously reviewed the paper and evaluated its accuracy. They also assessed the column related to optimal solutions and prepared a better version of the paper.

Nazanin Padkan has been contributed to writing and reviewing the article.

All authors commented on previous versions of the manuscript and read and approved the final manuscript.

## Conflict of interest

Authors declare that they have no conflict of interest.

## References

- [1] Klee V. *Unsolved Problems in Intuitive Geometry*. University of Washington; 1960.
- [2] Sadeghi Bigham B, Dolatikalan S, Khastan A. Minimum landmarks for robot localization in orthogonal environments. *Evolutionary Intelligence*. 2022; 15(3): 2235-2238.
- [3] Shamsfakhr F, Bigham BS. GSR: geometrical scan registration algorithm for robust and fast robot pose estimation. *Assembly Automation*. 2020; 40(6): 801-817.
- [4] Shamsfakhr F, Sadeghi Bigham B, Mohammadi A. Indoor mobile robot localization in dynamic and cluttered environments using artificial landmarks. *Engineering Computations*. 2019; 36(2): 400-419.

- [5] Icking C, Klein R. The two guards problem. *International Journal of Computational Geometry & Applications*. 1992; 2(03): 257-285.
- [6] Tan X. An efficient algorithm for the three-guard problem. *Discrete Applied Mathematics*. 2008; 156(17): 3312-3324.
- [7] Lee D, Lin A. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*. 1986; 32(2): 276-282.
- [8] Abrahamsen M, Adamaszek A, Miltzow T. The art gallery problem is  $\exists\mathbb{R}$ -complete. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. New York, NY, USA; 2018. p.65-73.
- [9] Stade J. The point-boundary art gallery problem is  $\exists\mathbb{R}$ -hard. *arXiv:221012817*. 2023. Available from: <https://doi.org/10.48550/arXiv.2210.12817>.
- [10] Abrahamsen M, Adamaszek A, Miltzow T. The art gallery problem is  $\exists\mathbb{R}$ -complete. *Journal of the ACM*. 2021; 69(1): 1-70.
- [11] Chvátal V. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*. 1975; 18(1): 39-41.
- [12] Fisk S. A short proof of Chvátal's watchman theorem. *Journal of Combinatorial Theory, Series B*. 1978; 24(3): 374.
- [13] Avis D, Toussaint GT. An efficient algorithm for decomposing a polygon into star-shaped polygons. *Pattern Recognition*. 1981; 13(6): 395-398.
- [14] O'Rourke J. An alternate proof of the rectilinear art gallery theorem. *Journal of Geometry*. 1983; 21: 118-130.
- [15] Michael T, Pinciu V. The orthogonal art gallery theorem with constrained guards. *Electronic Notes in Discrete Mathematics*. 2016; 54: 27-32.
- [16] Kahn J, Klawe M, Kleitman D. Traditional galleries require fewer watchmen. *SIAM Journal on Algebraic Discrete Methods*. 1983; 4(2): 194-206.
- [17] Katz MJ, Roisman GS. On guarding the vertices of rectilinear domains. *Computational Geometry*. 2008; 39(3): 219-228.
- [18] Ghosh SK. Approximation algorithms for art gallery problems in polygons and terrains. In: Rahman MS, Fujita S. (eds.) *WALCOM: Algorithms and Computation*. Berlin, Heidelberg: Springer; 2010. p.21-34.
- [19] Bajuelos Domínguez AL, Canales Cano S, Hernández Peñalver G, Martins AM. Optimizing the minimum vertex guard set on simple polygons via a genetic algorithm. *WSEAS Transactions on Information Science and Applications*. 2008; 5(11): 1584-1596.
- [20] Amit Y, Mitchell JS, Packer E. Locating guards for visibility coverage of polygons. *International Journal of Computational Geometry & Applications*. 2010; 20(05): 601-630.
- [21] Bottino A, Laurentini A. A nearly optimal algorithm for covering the interior of an art gallery. *Pattern Recognition*. 2011; 44(5): 1048-1056.
- [22] Tozoni DC, de Rezende PJ, de Souza CC. The quest for optimal solutions for the art gallery problem: a practical iterative algorithm. In: Bonifaci V, Demetrescu C, Marchetti-Spaccamela A. (eds.) *Experimental Algorithms*. Berlin, Heidelberg: Springer; 2013. p.320-336.
- [23] Bigham BS, Badri S, Padkan N. A new metaheuristic approach for the art gallery problem. *arXiv:210705540*. 2021. Available from: <https://doi.org/10.48550/arXiv.2107.05540>.
- [24] Tozoni DC, de Rezende PJ, de Souza CC. A practical iterative algorithm for the art gallery problem using integer linear programming. *Optimization Online*. 2013; 2013: 1-21.
- [25] Kröller A, Baumgartner T, Fekete SP, Schmidt C. Exact solutions and bounds for general art gallery problems. *Journal of Experimental Algorithmics*. 2012; 17: 2.1-2.23.
- [26] Fekete SP, Friedrichs S, Kröller A, Schmidt C. Facets for art gallery problems. *Algorithmica*. 2015; 73(2): 411-440.
- [27] Toth CD, O'Rourke J, Goodman JE. *Handbook of Discrete and Computational Geometry*. CRC press; 2017.
- [28] El Gindy H, Avis D. A linear algorithm for computing the visibility polygon from a point. *Journal of Algorithms*. 1981; 2(2): 186-197.
- [29] Eskandari M, Yeganeh M. S-visibility problem in VLSI chip design. *Turkish Journal of Electrical Engineering and Computer Sciences*. 2017; 25(5): 3960-3969.
- [30] Asano T. An efficient algorithm for finding the visibility polygon for a polygonal region with holes. *IEICE Transactions*. 1985; 68(9): 557-559.
- [31] Del Moral P. Nonlinear filtering: interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*. 1997; 325(6): 653-658.

- [32] Pozna C, Precup RE, Földesi P. A novel pose estimation algorithm for robotic navigation. *Robotics and Autonomous Systems*. 2015; 63: 10-21.
- [33] Thrun S, Burgard W, Fox D. Probabilistic robotics. *Kybernetes*. 2006; 35(7/8): 1299-1300.