



Coverage optimization of visual sensor networks for observing 3-D objects: survey and comparison

Xuebo Zhang¹ · Boyu Zhang¹ · Xiang Chen² · Yongchun Fang¹

Received: 2 April 2019 / Accepted: 9 September 2019 / Published online: 3 October 2019
© Springer Nature Singapore Pte Ltd. 2019

Abstract

Coverage is a basic and critical issue for design and deployment of visual sensor networks, however, the optimization problem is very challenging especially when considering coverage of three-dimensional (3-D) scenarios. This paper provides a brief survey of mainstream coverage optimization methods for visual sensor networks, including the greedy algorithm, genetic algorithm (GA), particle swarm optimization (PSO), binary integer programming (BIP) and differential evolution algorithm (DE). We provide an efficient open-source C++ implementation of these algorithms and compare their performance on a typical camera deployment problem for coverage of 3-D objects. In order to improve the computational efficiency, a parallel visual occlusion detection approach is proposed and implemented with graphic processing units (GPUs), which are then integrated into all of the aforementioned optimization approaches for a fair comparison. Evaluation results show that (1) the proposed parallel occlusion detection algorithm largely improves the computational efficiency; (2) among the five typical approaches, BIP has the best coverage performance yet with the highest time cost, and greedy algorithm is the fastest approach at the price of coverage performance; GA, PSO, and DE achieve a compromise between the performance and the time cost, while DE has better coverage performance and less time cost than PSO and GA. These results could serve as engineering guidelines and baselines for further improvement of coverage optimization algorithms.

Keywords Visual sensor networks · Coverage · Optimization · Parallel occlusion detection

1 Introduction

Visual sensors are widely used nowadays for their attractive advantages: low cost, light weight, non-contact sensing, rich information, etc. Applications with visual sensors are almost everywhere (Xu et al. 2011a; Shi et al. 2007; Kang and Medioni 2015; Miller et al. 2012). Because the sensing range of a single visual sensor is limited, multiple visual

sensors are usually combined to form a visual sensor network for many tasks, making visual sensor networks (VSNs) become a hot topic in recent years. Various applications of VSNs emerge such as surveillance (Wang 2013; Singh et al. 2008; Bodor et al. 2007), industrial inspection (Alarcon-Herrera et al. 2014a), object tracking Taj and Cavallaro (2011), etc. Interested readers are referred to monographs and surveys (Aghajan and Cavallaro 2009; Chen et al. 2008, 2011; Tarabanis et al. 1995; Mulligan and Ammari 2010).

As a prerequisite of many tasks in the field of VSNs, coverage has been one of the most fundamental and important research topics. Depending on whether the object model is known or not, the coverage approaches are classified as model-based approaches and non-model-based approaches (Chen et al. 2008). This paper focuses on survey and comparison of model-based approaches for coverage optimization, where the three-dimensional (3-D) object/environment model is supposed to be known, as encountered in tasks of industrial inspection, dimensional measurement, and so on. Model-based coverage optimization is usually solved by establishing a *coverage model* (performance function with

✉ Xuebo Zhang
zhangxuebo@nankai.edu.cn

Boyu Zhang
zby@mail.nankai.edu.cn

Xiang Chen
fangyc@nankai.edu.cn

Yongchun Fang
xchen@uwindsor.ca

¹ Institute of Robotics and Automatic Information System (IRAIS), Nankai University, Tianjin 300071, China

² Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B3P4, Canada

constraints) followed by *nonlinear coverage optimization* to obtain the sensor deployment configurations (Chen et al. 2008; Zhang et al. 2015b; Chen and Li 2004a).

A typical *coverage model* consists of a performance function with task and visual constraints, and an extensive survey of related works can be found in Mavrinac and Chen (2013). The performance function is used to evaluate the coverage which could be defined according to task requirements, for example, maximizing the covered area is one straightforward performance index. The task and visual constraints include camera field of view (FOV), resolution, blur, view angle, occlusion, etc. Most of these constraints could be formulated in an explicit and analytical form to be easily integrated into the subsequent optimization framework, except for occlusion. In practical implementation, the occlusion has to be checked one by one for each polygon face of the 3-D model, which is time-consuming and dramatically decreases the computational efficiency. Hence, fast visual occlusion detection is key to accelerate the computation and improve the overall efficiency of coverage approach, which will be studied in this paper.

Visual coverage optimization is generally modeled as a set cover problem based on the discrete representation of the 3-D surface model, which is NP-hard. In addition, the objective function and constraints are highly nonlinear with respect to the sensor configuration parameters including the position, orientation, etc, making the optimization even more difficult. Due to these two challenges, most coverage optimization approaches are based on greedy method (Alarcon-Herrera et al. 2014b) or metaheuristic approaches, such as genetic algorithm (GA) (Chen and Li 2004a) and particle swarm optimization (PSO) (Mavrinac et al. 2014; Xu et al. 2011b), to achieve suboptimal solutions.

In this paper, we will survey and compare five typical visual coverage optimization approaches including the greedy algorithm, GA, PSO, differential evolution (DE), and binary integer programming (BIP). We also provide an efficient open-source C++ implementation of these algorithms and compare their performance on a typical camera deployment problem for coverage of 3-D objects. In order to improve the computational efficiency, a parallel visual occlusion detection approach is proposed and implemented with graphic processing units (GPUs), which are then integrated into all of the aforementioned optimization approaches for a fair comparison. Evaluation results show that (1) the proposed parallel occlusion detection algorithm largely improves the computational efficiency; (2) Among the five typical approaches, BIP has the best coverage performance yet with the highest time cost, and greedy algorithm is the fastest approach at the price of coverage performance; GA, PSO, and DE achieve a compromise between the performance and the time cost, while DE has better coverage performance and less time cost than PSO and GA. These results could serve

as engineering guidelines and baselines for further improvement of coverage optimization algorithms.

In the remaining part, Sect. 2 gives an overview of related works about coverage model and optimization methods. Section 3 will give a statement of problem which is used to compare different approaches. Section 4 describes the serial and parallel approaches of generating discrete solution space with occlusion checking. Section 5 introduces five typical approaches for selecting cameras from the solution space. Section 6 provides the simulation results and Sect. 7 is an conclusion.

2 Survey of related works

VSNs could be viewed as a special kind of wireless sensor networks (WSNs). However, different from traditional WSNs whose sensors are usually point sensors to collect sensing information at specific points such as temperature and pressure, visual sensors capturing information of scenes within some area in its field of view rather than single points. Therefore, visual sensor networks have features of area sensing and (anisotropic) directional sensing, which makes its coverage problem more challenging than traditional WSNs. In Soro and Heinzelman (2005), Soro et al. point out and explain the important differences between WSNs and VSNs and show the differences by applying an algorithm designed for traditional WSNs to VSNs. The work in Charfi et al. (2009) also points out that the coverage optimization of VSNs is more complex than that of traditional WSNs.

Due to abundant applications of VSNs, it has become a rapidly developing research area in recent years. Many aspects of VSNs, such as data processing, communication, resource allocation, and so on Soro and Heinzelman (2009) have been investigated in the literature. Among all of these research topics, coverage is one of the most fundamental and active areas in VSNs. Several surveys have been conducted about coverage problems of visual sensor networks (Costa and Guedes 2010; Guvensan and Yavuz 2011; Soro and Heinzelman 2009). It is shown from these surveys that coverage problems can be classified into different categories according to different rules. Some works study about the Pan-Tilt-Zoom VSNs (Yen 2014; Loscrí et al. 2012) while others focus on VSNs with static cameras (Alarcon-Herrera et al. 2014a; Zhang et al. 2015b). Some researchers work on mobile targets such as (Azin et al. 2014; Chow et al. 2007) while others take into account static scenes (Zhang et al. 2014, 2015b, a). In addition, the area coverage (Fu et al. 2014) is also distinguished from the target coverage (Yen 2013). According to the spatial dimension, the coverage problems of VSNs can be divided into 3-D coverage problems and 2-D coverage problems, in which the complexity of three-dimensional coverage problems is much higher than

that of 2-D. For example, the sensing models for 3D visual coverage problems are more complex. Not only FOV should be considered, but also the image blur and resolution constraints should be taken into consideration. What's more, the occlusion detection of 3-D coverage problems is also much more time-consuming. Therefore, as one typical representative problem, the surface area coverage problem for a static 3-D scene or object will be the focus of this paper for subsequent comparison of different coverage optimization algorithms, which has vast applications in industrial inspection, dimensional measurement, surveillance and monitoring, and so on. To solve the coverage problem, a coverage model is established first and then nonlinear coverage optimization is applied. Hence, this brief survey is divided into two parts: coverage model (with special focus on occlusion detection) and coverage optimization.

2.1 Coverage model: occlusion detection

Different from general sensors such as lasers or infrared rays (Fanti et al. 2018), the sensing model of a visual camera is more complex. A precise model of VSNs will make a solution of coverage problem practically significant. The previous work (Mavrinac and Chen 2013) has provided a detailed survey about the modeling of visual sensor networks coverage, thus this paper will only give a brief introduction of four characteristics including FOV, image blur, resolution and occlusion in modeling the visual coverage, and in particular we will focus on the occlusion detection problem for which the existing literature still lacks in-depth survey, though it is very important to affect the computational efficiency.

2.1.1 Field of view

First of all, FOV is one of the most fundamental characteristics of a visual coverage model, which defines boundary of the sensing range. Different from traditional WSNs, the sensing range of a camera is an anisotropic area rather an isotropic circle or sphere. Thus, some works of two-dimensional (2-D) VSNs consider the directional nature of cameras and represent the FOV by a fan-shaped area (Ai and Abouzeid 2006; Yen 2013). In Hörster and Lienhart (2006b), (Costa et al. 2017; Jesus et al. 2018), the FOV is represented by a triangle. Altahir et al. (2017a, 2018) consider a coverage problem where the cameras are deployed in 3-D space to monitor a 2-D field so they represent the FOV by a changeable fan-shaped area. These are simplified representations of FOV, which make sense in 2-D scenarios. In 3-D space, a pyramid is usually used to describe FOV (Malik and Bajcsy 2008) whose boundary depends on the maximum angles that a camera is able to reach in four directions. Munishwar and Abu-Ghazaleh (2013) refer to the general FOV of cameras, which is represented by its angle-of-view (AOV) and

depth-of-field. In fact, it is more suitable to call the AOV as FOV. The depth-of-field characteristic represents the area of the visual scene that is acceptably sharp, which is actually depends on the image blur.

2.1.2 Image blur

The second characteristic is image blur. Points with different distances from the image plane result in image blur circles of different sizes. Only images that are sharp enough can be utilized in the practical applications. Therefore, the image blur constraint determines the acceptable maximum distance and minimum distance from the camera, thus coming out the near plane and far plane of a camera. FOV and the two planes determine the visible space which is called viewing frustum. 2-D coverage problem usually ignore this constraint, but works of 3-D coverage usually take it into consideration for the accuracy of modeling (Park et al. 2006).

2.1.3 Resolution

Another characteristic of VSNs is resolution. Resolution constraint guarantees the ratio between the pixel length and the corresponding physical length. In Mavrinac and Chen (2013), it is said that resolution is a function of depth along the optical axis. For a plane parallel to the image plane, it is true that the resolution depends only on the distance between them. While for a plane not parallel to the image plane, the angle between them should also be taken into consideration. Hörster and Lienhart (2006b) give an accurate definition for resolution when the object has a surface orthogonal to the optical axis and also mention that the resolution is smaller if the surface is not orthogonal. But they do not provide the specific formula. The work in Zhang et al. (2015b) combines the distance criterion and the view angle criterion together to form a new resolution criterion. They present the specific formula of resolution which account for both orthogonal and nonorthogonal situations, which is a more realistic definition of resolution. In this work, this new resolution criterion (Zhang et al. 2015b) will be adopted.

2.1.4 Occlusion

In addition to the mentioned characteristics above, occlusion is also an issue in VSNs and the handling of occlusion constraints is usually time-consuming especially for 3-D models represented by many small polygon faces. Some works do not consider the occlusion constraint in the modeling for their special tasks. The work in Malik and Bajcsy (2008) considers stereo camera placement for 3-D reconstruction and the objective is the coverage of a rectangular space, wherein the occlusion is not considered. Abdelkader et al. (2008) study coverage of moving target and the occlusion of

walls is taken into account but the occlusion between targets are not considered because the model of targets are simplified as points. Some works seek to avoid the computational complex occlusion detection by minimizing the probability of occlusion due to randomly moving objects by covering each priority area using multiple cameras (Indu and Chaudhury 2009). This approach has its value in surveillance but in inspection of static objects, the accurate computation of detection is required. In fact, for most area coverage problem in three dimensions, it is necessary to handle the occlusion for correct and more precise evaluation of coverage.

The detection of occlusion is usually conducted by checking whether the line of sight connecting the camera origin and a single point intersects with any obstacles (Mavrinac 2012). Since every element of the target could be an obstacle potentially, the occlusion should be checked one by one consuming a lot of time. The work in Zhang et al. (2014) uses polygon model for the object and detects occlusion by judging if the line segment connecting a vertex of triangle and the optical center is intersected with other triangles. Applications for industrial inspections usually represent an object by hundreds of thousands of triangles. To detect the occlusion in the view of one specific camera, $(n - 1)^2$ calculations need to be done (n denotes the total number of triangles in the object model), because every triangle needs to be detected if it is occluded by any other triangles. A kind of commonly used approach of visual coverage is the discrete solution space method. Alarcon-Herrera et al. (2014b) generate a candidate camera for each triangle to form an adjacent matrix. In this case, the computational complexity for occlusion detection will be $O(n^3)$. The huge computation will result in huge time cost especially when n is large. To avoid the high time cost in occlusion detection, The work in Chow et al. (2007) checks occlusion posteriorly to further update the coverage results. However, taking occlusion detection in the posterior checking process may lead to inaccurate coverage performance especially when the object model is complex and a lot of occlusion exists. To deal with the high time cost in occlusion detection, Zhang et al. (2015b) points out that parallel computation is able to be adopted to obtain efficient occlusion detection. Although the parallel implementation is not provided in the previous work (Zhang et al. (2015b)), it is indeed a feasible approach because the occlusion relationship can be judged independently.

In computer graphics, ray casting or rasterization is usually used to display 3-D models (Hughes et al. 2013) in a speed quicker than that of human eye's reaction. These 3-D models displayed are surely images after occlusion processing. The key of these approaches to handle the occlusion is the use of the depth buffer or z -buffer. For each pixel, only the distance of the closest point will be recorded in the depth buffer. As a result, points away from the pixel will not be

displayed. This is a successful implementation of occlusion detection. If the indices of triangles which are occluded can be obtained, it will be the ideal situation for our coverage problem. Although these depth buffer based approaches are now supported by special fixed-function units in graphics hardware and usually used for shading, there is no access to the index of the occluded triangles from the hardware. Thus, software methods for parallel occlusion that can return the index of occluded triangles still needs to be proposed.

In summary, as the existing hardware approaches cannot return the indices of occluded triangles in our problem, in this paper, a software parallel implementation based on GPU is presented to improve the efficiency. Furthermore, to reduce the computational complexity, the field of view constraint is judged first to filter out part of the triangles which lie outside the camera field of view.

2.2 Coverage optimization

The underlying optimization principle of visual coverage is similar to the NP-hard art gallery problem (AGP) (Urrutia 2000), which aims to monitor an art gallery with the least number of guards in order to make sure every point is seen by at last one guard. However, the visual coverage optimization is more complex due to the anisotropic directional sensing feature and other vision-related characteristics. The aim of camera network deployment is to minimize the number of cameras needed to fulfill the full-coverage task, or in contrast, maximize the covered area or targets with specified number of cameras. Hence, the deployment of VSNs is an NP-hard problem, which means that, although the best solution could be found theoretically, the most direct approach, simple enumeration or brute force search, is impractical to solve this problem. Consequently, how to find a near-optimal solution in a reasonable time is the focus of the study. To address this issue, different researchers propose a variety of algorithms and their variants to solve the coverage problem, such as harmony search, simulated annealing (Soltani et al. 2011; Morsly et al. 2012; Liu et al. 2014), and so on. Chen et al. (2014) utilize a mimetic algorithm based scheduling strategy and a heuristic recursive algorithm to maintain continuous coverage of WSNs. Altahir et al. (2017b) propose a dynamic programming algorithm for solving visual coverage problem and compare the outcomes with some existing optimization techniques such as local, global search and integer programming methods. All of these approaches can be divided into two categories, the centralized approaches and distributed approaches (Ai and Abouzeid 2006). The distributed approaches use more computing resources like memory and processors to improve the efficiency, but the overall performance may be degraded since usually only local information is used. In contrast, centralized approaches usually achieve better performance by sacrificing some efficiency.

Due to the space limitation, this paper will focus on centralized approaches and review five mainstream optimization approaches which are used frequently in visual coverage optimization, including greedy algorithm, genetic algorithm (GA), particle swarm optimization (PSO), differential evolution (DE), and binary integer programming (BIP).

Greedy algorithm: Greedy algorithms are the most basic approaches with high efficiency because of their simple operation. The general practice of greedy algorithms is to choose the best solution every single time and repeat the process. They can be good choices if the requirement of coverage performance is not critical. Ai and Abouzeid (2006) utilize different greedy approaches to solve a 2-D coverage problem. The aim is to maximize the number of targets covered by adjusting the FOV of randomly deployed directional sensors whereas the number of sensors to be activated is minimized. They first propose a centralized greedy algorithm as baseline for comparison and then provide a distributed greedy algorithm to improve the performance. Later, Munishwar et al. propose their centralized and distributed force-directed algorithm to further improve the performance. The force-directed methods assign weights to different FOVs of one camera depending on the *force* exerted by targets on them. The force in a pan is calculated as the ratio of the number of targets coverable by the pan to the total number of targets coverable by the camera. Pans with large forces will be selected with high priority. These are typical greedy approaches in 2-D coverage for pan-only cameras but also can be used for reference to 3-D camera deployment applications.

In 3-D coverage, the greedy algorithm is usually followed by generating a coverage matrix first. Alarcon-Herrera et al. (2014a) design a greedy algorithm for their vision system in industrial inspection. The aim is to maximize the visual coverage and to minimize the number of cameras. In their approach, the concept of ‘adjacency matrix’ is introduced and constructed to show the coverage strength between each scene-point and each camera viewpoint, based on which a weighted overlap graph is generated for each pair of viewpoints. The compounded degree which is the product between coverage strength and overlap strength is defined as the criterion for viewpoint selection. With this criterion, the greedy algorithm selects viewpoint one by one according to the sorted list of compounded degree.

To sum up, the biggest advantage of greedy algorithms lies in its high computation efficiency with low time cost. However, it can not guarantee high coverage performance every time. Algorithms with flexibility which can achieve better coverage performance are needed.

Genetic algorithm: To achieve a higher coverage rate than that of greedy algorithms, many heuristic approaches has been reported in the literature as the mainstream approaches. As a relatively mature approach, genetic algorithm (GA)

and its variants have been designed to solve the coverage optimization problem (Navin 2015; Indu and Chaudhury 2009; Heidali and Movaghar 2011; Wang et al. 2009; Yoon and Kim 2013). Indu and Chaudhury (2009) apply GA to visual surveillance of large spaces and provide maximum coverage of the priority areas by PTZ cameras. The work in Wang et al. (2009) applies standard GA to optimize coverage in VSNs. The work in Yoon and Kim (2013) proposes an efficient GA to deal with the sensor deployment problem with static sensors. Based on the property that the phenotype space of the problem is a quotient space of the genotype space in a mathematical view, a Monte Carlo method is adopted to design an efficient evaluation function. The efficient GA is compared with random deployment and other existing methods, and evaluation results demonstrate the superior performance of GA in terms of both time efficiency and coverage quality. Chen and Li (2004b) design a novel hierarchical GA to deal with the automatic sensor placement for model-based robot vision. The proposed hierarchical chromosome in Chen and Li (2004b) consists of parameter genes and control genes showing the topology of viewpoints. Topcuoglu et al. (2009) present a hybrid GA to solve the sensor placement problem. The hybrid GA includes specialized operators for hybridization, which are problem-specific heuristics for initial population generation, intelligent variation operators which comprise problem specific knowledge, and a local search phase. As GA has high computational complexity, some works begin to find faster approaches. Heidali and Movaghar (2011) propose an efficient genetic-based algorithm to deal with sensor network optimization problems that can achieve a good solution quickly. Navin (2015) proposes a distributed GA for WSNs to reduce the time consumption, and simulation results show that the distributed genetic algorithm achieves a near-optimal solution with better time performance compared with the centralized genetic algorithm.

Particle swarm optimization: Some works try other kinds of heuristic algorithms instead of GA, for instance, swarm intelligence optimization algorithms (Loscrí et al. 2012; Kulkarni and Venayagamoorthy 2011a; Konda and Conci 2013; Fu et al. 2014). PSO is the most common one of swarm intelligence optimization algorithms. It is a stochastic optimization technique inspired by flocking of birds, which is based on the multi-agent cooperation rather than competition in GA. Ready and Conci (2012) apply PSO to optimize video camera positioning in 2-D indoor environments, which considers both global coverage and local coverage. The work in Konda and Conci (2013) points out that there are three main advantages of PSO over the GA in terms of computational complexity, parameter tuning and prevention of stagnation, thus the authors apply the PSO to the coverage optimization and reconfiguration. Kulkarni and Venayagamoorthy (2011a) apply PSO to address WSN issues including static and mobile

WSN deployment and point out the advantages and disadvantages of PSO for optimization problems in WSNs. Variants of PSO are also designed for the camera placement problem. To improve the performance of the original PSO, Loscrí et al. (2012) propose a PSO algorithm for WSNs whose particle speed is updated by considering a consensus algorithm based on local optimum position. Simulation results show that their techniques obtain remarkable results for both coverage and energy consumption for movement. The work in Morsly et al. (2012) proposes a novel method based on binary particle swarm optimization (BPSO) inspired probability technique and compares it with the performances of some stochastic variants of optimization based particle swarm algorithms. Simulation results for 2-D and 3-D scenarios show the efficiency of their proposed technique. The work in Fu et al. (2014) also proposes a probability inspired binary PSO (PI-BPSO) to deal with the surveillance task. They assume that the probability of choosing each grid is the same and the coverage ratio can be determined by sampling. The simulation results show the effectiveness of the proposed PI-BPSO algorithm.

Differential evolution: The DE algorithm is one typical population-based heuristic evolutionary optimization method, which is developed for real-valued numerical optimization problems. Similar with GA, the DE algorithm contains the mutation, crossover, and selection module, yet it has stronger mutation strategies than GA, and the information of distance and direction from the current population is fully utilized to serve as some guidance in the search process (Iwan et al. 2012). Though the DE algorithm is applied in many applications including deployment of RFID sensors (Seok et al. 2010), design of antennas (Deb et al. 2014), object detection (Ugolotti et al. 2013), system modeling (Konar and Aytakin 2016), and so on, it is rare to be used for coverage optimization of visual sensor networks. Thus, we carry out the first trial in our recent work (Zhang et al. 2016a) to show that the performance of DE is significantly better than that of greedy algorithms for visual coverage optimization. However, the comparison with other algorithms are not conducted in Zhang et al. (2016a).

Binary integer programming: As coverage optimization of visual sensor networks is essentially a mathematical programming problem, integer linear programming is also applied as an effective solution in some works. Hörster and Lienhart (2006a) use an integer linear programming approach to solve the optimal visual sensor placement problem in 2-D space. The work in Hörster and Lienhart (2006b) considers four different visual sensors placement problems and propose the corresponding binary integer programming approaches for the problems to obtain the global optimal solutions. As the BIP problem is not able to be solved in a reasonable amount of time and memory with an arbitrarily large number of variables and constraints, they have to keep the number of variables and constraints as small as

possible and solve the problem using the LINGO package. Zhao and Cheung (2009) also adopt BIP to optimize deployment of visual sensors. The BIP approach seems like the best approach because it can provide the exact formulation of coverage problems. But in fact, even though the coverage problem can be transformed into a BIP problem, it is still an NP-hard problem. With the fast development of expensive commercial optimization softwares, the BIP can be solved efficiently when the scale of the problem is small. However, when the problem scale is large, the time cost increase fast and the final obtained solution may not be the optimal one.

Summary and comparison of different algorithms: To analyze the advantages and disadvantages of these optimization algorithms, some existing works conduct simulations and experiments among different algorithms. Deif and Gadallah (2014) discuss the strength and limitations of four mathematical approaches for WSN deployment: GA, computational geometry (CG), artificial potential fields (APF) and PSO. According to the work, it is shown that “GA and PSO are better suited for deploying WSNs with multiple design objectives than CG and APF approaches”. Morsly et al. (2012) apply several PSO based approaches, binary GA, Simulated Annealing and Tabu Search to the optimal camera network placement. Their results show the binary PSO inspired probability performs the best in both coverage rate and efficiency. Regarding comparison between DE and PSO, there have been no results reported for coverage optimization of visual sensor networks. However, some results have been published for comparison of PSO and DE for general benchmark problems (Iwan et al. 2012; Civicioglu and Erkan 2013; Kachitvichyanukul 2012; Ülker and Haydar 2012) and other specific applications (Deb et al. 2014; Ugolotti et al. 2013; Konar and Aytakin 2016; Basgumus et al. 2015). In those works for general benchmark problems, it is shown that DE outperforms PSO in terms of repeatability and convergence rate in Iwan et al. (2012), Civicioglu and Erkan (2013), Kachitvichyanukul (2012), and it is shown in Ülker and Haydar (2012) that DE has better performance for most benchmark problems than PSO and GA, while for two problems PSO has faster convergent speed. In the applications (Deb et al. 2014; Ugolotti et al. 2013; Konar and Aytakin 2016; Basgumus et al. 2015), DE achieves better performance for design of antennas (Deb et al. 2014), system modeling (Konar and Aytakin 2016), optical communication systems (Basgumus et al. 2015), while PSO has better performance for object detection (Ugolotti et al. 2013). Thus, it is necessary to investigate the performance of DE with respect to other algorithms for visual camera sensor networks.

To summarize, the greedy algorithm achieves the fastest computational speed while sacrificing coverage performance; the BIP algorithm is NP-hard, yet it presents the best coverage performance while the time cost increases dramatically as the problem scale increases including the number of

cameras and the size of the 3-D object model. GA, PSO and DE are metaheuristic optimization algorithms which achieve a compromise between greedy algorithm and BIP, yet comparative performance analysis among them still needs to be explored for the task of visual coverage optimization.

2.3 Contribution of this paper

According to the previous survey, it is seen that parallel visual occlusion detection needs to be studied as a common computational issue to improve the efficiency of the coverage optimization algorithm. In addition, fair comparative analysis of these five typical approaches needs to be conducted in the context of visual coverage optimization for 3-D scenarios. Compared with related works in the previous survey, the main contribution of this paper can be summarized as follows:

1. We conduct a survey of the five typical visual coverage optimization approaches including the greedy algorithm, GA, PSO, differential evolution (DE), and binary integer programming (BIP).
2. To improve the time efficiency of occlusion detection, this paper designs a parallel implementation to increase the computational efficiency and utilizes the field of view constraint to reduce the amount of computation.
3. An open-source GUP-based C++ implementation of the algorithms including the greedy approach, GA, PSO, DE, and BIP is provided, and these five approaches are compared in terms of the time efficiency, coverage performance, and so on.

It should be noted that the paper aims to provide more general guidelines and interpretation of existing algorithms, thus we use the standard version of PSO, GA and DE, while various variants of them are not involved in this paper.

3 Problem statement

In this section, the environment model and the camera model will be introduced and the criteria of coverage will be presented. Then, the general idea of camera network deployment for visual coverage optimization will be stated.

3.1 Environment (object) model and camera model

In this paper, the object model is expressed with a triangle mesh, wherein each triangle is described by three vertices. The coordinate of each vertex and the vertices of each triangle face are provided by a 3-D model file.

The camera model is described by its intrinsic parameters and external parameters. Its intrinsic parameters include

the focal length $f \in \mathbb{R}^+$ in mm, the horizontal and vertical pixel dimensions $s_u \in \mathbb{R}^+$ and $s_v \in \mathbb{R}^+$ in mm/pixel, the principle point $o = [o_u \ o_v] \in \mathbb{R}^2$ in pixel coordinates, the image width $w \in \mathbb{N}^+$ and the image height $h \in \mathbb{N}^+$ in pixel. $A \in \mathbb{R}^+$ denotes the effective aperture diameter of the optical lens in mm, $Z_s \in \mathbb{R}^+$ represents its subject distance. Its external parameters include its position and orientation relative to the world coordinate system. In this paper, we assume the intrinsic camera parameters are calibrated and the camera network deployment problem is to compute the extrinsic parameters to maximize the overall coverage area.

3.2 Criteria of visual coverage

To judge if a triangle is covered by a camera, four criteria which are discussed in the last section are considered including the field of view, the blur, the resolution and the occlusion.

- *Field of view*: let $x_{c_i}, y_{c_i}, z_{c_i}$ represent the coordinate of a triangle center under camera c_i . $\alpha_l, \alpha_r, \alpha_t, \alpha_b$ are respectively the left, right, top and bottom view angle of a camera. The field of view rule can be described as

$$\begin{aligned}
 -\tan\alpha_l &\leq \frac{x_{c_i}}{z_{c_i}} \leq \tan\alpha_r, \\
 -\tan\alpha_t &\leq \frac{y_{c_i}}{z_{c_i}} \leq \tan\alpha_b.
 \end{aligned} \tag{1}$$

- *Image blur*: let c_a represent the largest degree of blur that can be accepted, with z_n, z_f respectively denoting the nearest and farthest distance a camera can inspect with blur less than c_a , then the image blur rule is

$$\begin{aligned}
 z_n \leq z_{c_i} \leq z_f \\
 \text{where } z_n, z_f \text{ is computed as}
 \end{aligned} \tag{2}$$

$$\begin{aligned}
 z_n &= \frac{Z_s \cdot f \cdot f}{f \cdot f + A \cdot c_a \cdot Z_s}, \\
 z_f &= \frac{Z_s \cdot f \cdot f}{f \cdot f - A \cdot c_a \cdot Z_s}.
 \end{aligned} \tag{3}$$

The field of view and image blur criteria mean that a triangle has to be in the view frustum of a camera, or it is not covered. A more vivid explanation can be seen in Fig. 1 where the view frustum degenerates into a trapezoid on the XOZ plane.

- *Resolution*: resolution is represented by the number of pixels on the image plane occupied by the unit length on the object surface. Figure 2 shows the resolution of the object in the Y direction, where $r(P_{c_i})$ is the resolution and θ_i is the angle between the opposite direction of the normal vector

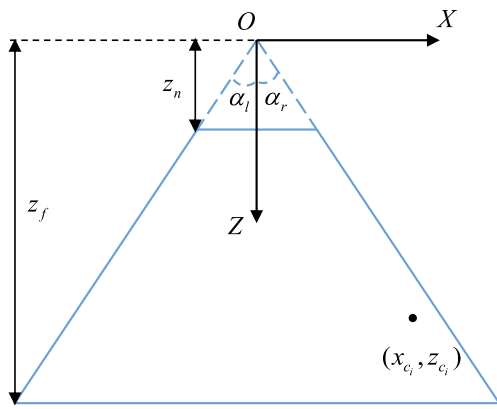


Fig. 1 Camera field of view and image blur

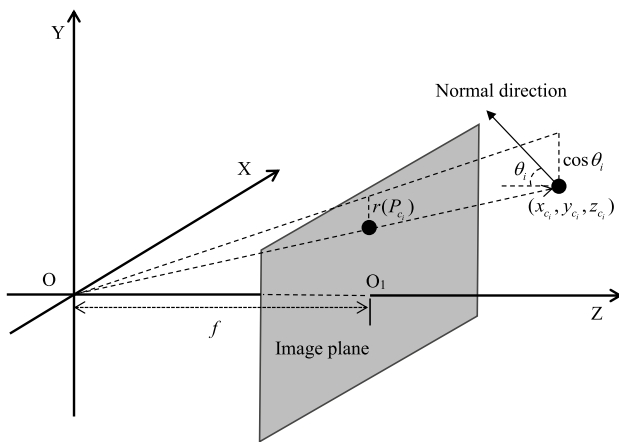


Fig. 2 Sensing resolution

of a triangle and the Z axis. According to the principle of triangle similarity, the following formula can be obtained

$$\frac{f}{z_{c_i}} = \frac{r(P_{c_i})}{\cos \theta_i} \tag{4}$$

The resolution constraint is satisfied only if the minimum required resolution in both X and Y directions is satisfied. Let r_a denote the minimum required resolution of the triangle, the resolution rule can be described as

$$r(P_{c_i}) = \frac{f \cdot \cos \theta_i}{z_{c_i} \cdot \max(s_u, s_v)} \geq r_a \tag{5}$$

- **Occlusion:** if the line segment which connects a vertex with a camera intersects with a triangle at an interior point of this triangle, then the vertex is occluded under the view of the camera. If one or more vertex of a triangle is occluded, then the triangle is occluded.

3.3 General idea to solve the problem

In this paper, we refer to the work in Alarcon-Herrera et al. (2014b) and solve the deployment problem with a set cover approach. It consists of two parts. In the first part, a candidate camera will be generated in the right above of each triangle with appropriate distance to form a coverage matrix. Let CF denote the matrix consisting of n rows and n columns. Its $F(j)$ covby $C(i)$ means the j th triangle is covered by the i th camera. $F(j)$ uncovby $C(i)$ means the j th triangle is not covered by the i th camera ($i = 1, 2 \dots n; j = 1, 2 \dots n$). Matrix CF describes the coverage relationship as follows:

$$CF(i, j) = \begin{cases} 0, & F(j) \text{ uncovby } C(i) \\ 1, & F(j) \text{ covby } C(i) \end{cases} \tag{6}$$

In the second part, a number of D cameras will be selected from the n candidate cameras so that the covered set of triangles is maximized (D is the given number of cameras to be placed).

To summarize, the first part is to generate discrete solution space (GDSS) and the second part is to select cameras (SC) by coverage optimization, which will be described in Sects. 4 and 5, respectively.

4 Different gdss approaches

This section will give the existing serial GDSS (S-GDSS) approach and propose a GPU-based parallel GDSS (P-GDSS) approach, wherein the visual occlusion detection is conducted in a parallel way with field of view detection being implemented first to suppress the computation burden.

4.1 Serial GDSS approach

In existing works, the GDSS procedure is conducted with a serial approach. It detects all the triangles respectively with the four coverage criteria presented in Sect. 3 to get the coverage matrix. The former three criteria are used to get a coverage matrix first. And then all the triangles will be detected by the occlusion criterion to renew the matrix and thus the final coverage relationship is obtained. Let $F(i)$ denote the i th triangle. $T(i) \in \mathbb{R}^{3 \times 3}$ and $R(i) \in \mathbb{R}^{3 \times 3}$ represent the translation (position) vector and orientation matrix of the i th camera in the world coordinate system, respectively. T and R are the discrete solution space of translation vector and orientation matrix. $F(j) < F(i)$ means $F(j)$ is occluded by $F(i)$. The S-GDSS approach can be described as Algorithm 1.

Algorithm 1 *S – GDSS*

Require: $Obj, \alpha_l, \alpha_r, \alpha_t, \alpha_b, A, f, w, h, s_u, s_v, Z_s, r_a, c_a$
Ensure: CF, R, T

- 1: Initialization: $CF = 0, R = 0, T = 0$
- 2: **for** $i = 1 \rightarrow n$ **do**
- 3: Generate $T(i)$ and $R(i)$ by $F(i)$
- 4: $R \leftarrow R(i), T \leftarrow T(i)$
- 5: **for** $j = 1 \rightarrow n$ **do**
- 6: **if** $-tan\alpha_l \leq \frac{x_{c_i}(j)}{z_{c_i}(j)} \leq tan\alpha_r$ **and** $-tan\alpha_t \leq \frac{y_{c_i}(j)}{z_{c_i}(j)} \leq tan\alpha_b$ **then**
- 7: **if** $z_n \leq z_{c_i}(j) \leq z_f$ **then**
- 8: **if** $r(P_{c_i})(j) \geq r_a$ **then**
- 9: $CF(i, j) = 1$
- 10: **else**
- 11: $CF(i, j) = 0$
- 12: **end if**
- 13: **else**
- 14: $CF(i, j) = 0$
- 15: **end if**
- 16: **else**
- 17: $CF(i, j) = 0$
- 18: **end if**
- 19: **end for**
- 20: **for** $j = 1 \rightarrow n$ **do**
- 21: **for** $k = 1 \rightarrow n$ **do**
- 22: **if** $F(j) < F(k)$ **then**
- 23: $CF(i, j) = 0$
- 24: **end if**
- 25: **end for**
- 26: **end for**
- 27: **end for**
- 28: **return** CF, R, T

4.2 Parallel GDSS approach

In Algorithm 1, the occlusion detection is a high time consuming step. For a model consisting of n triangles, n candidate cameras need to be generated. For camera $C(i)$, to detect whether a triangle $F(j)$ is occluded by other triangles, we need to calculate if the line segment links $C(i)$ with $F(j)$ intersects with the triangle $F(k)$ ($i = 1, 2 \dots n; j = 1, 2 \dots n; k = 1, 2 \dots n$). Therefore, the time complexity of the algorithm is $O(n^3)$, and the computational efficiency for the process of forming CF matrix is rather low. This paper improves the time efficiency from following two aspects:

Parallelization: A parallel approach through GPU is proposed in this paper to improve the computation efficiency. In this approach, n blocks are divided with each block containing n threads. The n^2 threads fulfill the whole task of occlusion detection in the field of view of one camera. As these threads can be seen as running at the same time, which means n^2 of calculation is done at one time, the whole time complexity will drop from $O(n^3)$ to $O(n)$. It should be noted that this paper only take small scaled models into account so the computer is able to allocate n^2 threads. Parallelization can also be applied to the occlusion detection of large scaled

Algorithm 2 *P – GDSS*

Require: $Obj, \alpha_l, \alpha_r, \alpha_t, \alpha_b, A, f, w, h, s_u, s_v, Z_s, r_a, c_a$
Ensure: CF, R, T

- 1: Initialization: $CF = 0, R = 0, T = 0, CF2 = 0, s = 1$
- 2: **for** $i = 1 \rightarrow n$ **do**
- 3: Generate $T(i)$ and $R(i)$ by $F(i)$
- 4: $R \leftarrow R(i), T \leftarrow T(i)$
- 5: **for** $j = 1 \rightarrow n$ **do**
- 6: **if** $-tan\alpha_l \leq \frac{x_{c_i}(j)}{z_{c_i}(j)} \leq tan\alpha_r$ **and** $-tan\alpha_t \leq \frac{y_{c_i}(j)}{z_{c_i}(j)} \leq tan\alpha_b$ **then**
- 7: $record(s) = j, s++$
- 8: **if** $z_n \leq z_{c_i}(j) \leq z_f$ **then**
- 9: **if** $r(P_{c_i})(j) \geq r_a$ **then**
- 10: $CF(i, j) = 1$
- 11: **else**
- 12: $CF(i, j) = 0$
- 13: **end if**
- 14: **else**
- 15: $CF(i, j) = 0$
- 16: **end if**
- 17: **else**
- 18: $CF(i, j) = 0$
- 19: **end if**
- 20: **end for**
- 21: **for** $j = 1 \rightarrow s$ **do**
- 22: Copy $F(j)$ from CPU to GPU
- 23: **end for**
- 24: Copy $T(i)$ and $R(i)$ from CPU to GPU
- 25: In the k^{th} thread of the j^{th} block
- 26: **if** $F(j) < F(k)$ **then**
- 27: $CF2(j, k) = 0$
- 28: **else**
- 29: $CF2(j, k) = 1$
- 30: **end if**
- 31: Copy $CF2$ from GPU to CPU
- 32: **for** $k = 1 \rightarrow s$ **do**
- 33: $CF(i, record(k)) = \min(CF2(j, :))$
- 34: **end for**
- 35: **end for**
- 36: **return** CF, R, T

models but the acceleration effect depends on the Parallel capability of the computer.

Reducing the computation cost: In the mentioned serial approaches above, we detect the occlusion relationship between each pair of triangles of the n triangles. However, if a triangle is not in the field of view of the camera, it is not necessary to detect whether it is occluded, meanwhile, it can not occlude any other triangles. Thus, we can utilize the field of view detection to screen out some triangles first and only detect these remaining triangles with other constraints including resolution, blur and view angle. In this way, the time efficiency of visual occlusion can be further improved.

Let $CF2$ denote the $n \times n$ matrix formed by n blocks with each block containing n threads in GPU, it describes the occlusion relationship in the view of one camera. $record$ is an

array that records the indices of triangles which are in the field of view of a camera. k and s are used to count. Algorithm 2 shows the pseudo code of the parallel GDSS approach.

It can be seen that the dual loop of occlusion detection in Algorithm 1 is replaced by a GPU-based implementation in line 26–30. Then an extra logical AND operation is used to renew the CF matrix. As the AND operation is very fast, the time consumption of the extra part is little and can be ignored. Thus the time consumption of occlusion detection is reduced from $O(n^3)$ to $O(n)$.

It can also be seen in Algorithm 2 that the field of view constraint is first detected and resultant triangles satisfying this constraint are recorded in an array. Then the occlusion detection is only conducted in a parallel way for triangles in the array. If only minority part of the object model satisfy the field of view constraint, it will largely reduce the computational complexity.

5 Different SC approaches

In this section, the greedy SC approach (G-SC) and two heuristic algorithms including genetic algorithm (GA) and particle swarm optimization (PSO), will be applied to the SC problem. After that, differential evolution (DE) will be adopted for visual coverage optimization, which has not been reported in the previous works. Finally, the SC problem will be reformulated into a binary integer programming (BIP) problem and solved by a commercial optimizer.

5.1 Greedy SC approach

To select D cameras from n candidate cameras, the work in Alarcon-Herrera et al. (2014b) gives a greedy approach as follows:

1. Search the row of CF whose sum is the largest, and choose the corresponding candidate camera with index k .
2. Renew the CF matrix: for $1 \leq s \leq n$, if $CF(k, s) = 1$, then $CF(:, s) = 0$, where s represents the index of triangles which the k th camera covers.
3. If the total number of chosen cameras is less than D , go to the first step. Otherwise, stop and return indices of the chosen cameras.

Let T_c and R_c denote the solution group of translation vector and orientation matrix. Each element of T_c denotes the translation vector of one camera. The corresponding element of R_c is the orientation matrix of the corresponding camera. $nCov(i)$ represents the number of triangles that $C(i)$ covers ($i = 1, 2 \dots n$). N is the total number of triangles covered by the D placed cameras. Then the greedy approach can be described as Algorithm 3.

Algorithm 3 $G - SC$

Require: D, CF, R, T

Ensure: N, R_c, T_c

```

1: Initialization:  $N = 0, R_c = 0, T_c = 0, nCov = 0$ 
2: for  $j = 1 \rightarrow D$  do
3:   for  $i = 1 \rightarrow n$  do
4:      $nCov(i) = \text{sum}(CF(i, :))$ 
5:   end for
6:   for  $k = 1 \rightarrow n$  do
7:     if  $nCov(k) = \max(nCov(i)), i = 1, 2, 3 \dots n$  then
8:        $R_c \leftarrow R(k), T_c \leftarrow T(k), N = N + nCov(k)$ 
9:       if  $CF(k, s) = 1$  then
10:         $CF(:, s) = 0$ 
11:       end if
12:     end if
13:   end for
14: end for
15: return  $N, R_c, T_c$ 

```

The time cost of this algorithm is very low because of its simple process, but its coverage performance is often not satisfactory. As a greedy algorithm, it chooses the camera that covers maximum triangles every time, which may be the best choice every single time. However, it ignores the fact that the combination of D single optimal solutions does not equal to the optimal overall solution. Therefore, new algorithms need to be explored to improve the coverage performance.

5.2 GA-SC and PSO-SC approaches

As relatively mature algorithms, GA and PSO are often applied to solve the coverage problem of WSNs or VSNs. In this paper, these two algorithms are also used for the SC procedure. GA is one of the most powerful meta heuristic methods to solve optimization problems. It is based on inherent selection, which drives biological evolution. It can reach near optimal solution with sufficient generations of evolution. As the steps of GA have been introduced by lots of previous works, in this paper we only give a brief introduction of our design in every step as follows:

Initialization: among numerous encoding strategies, this paper chooses the integer coding approach which is more suitable for the SC problem because the camera indices are integers. Genes are represented by integers whose range are from 1 to n . Each individual consists of D genes and one generation is made up of *pop* genes.

Selection: The number of triangles covered by each camera is computed as the fitness value and the roulette method is used for selection.

Crossover: Uniform crossover is used in this paper which means every gene in the individual has the same probability to cross.

Mutation: Uniform mutation is applied, which means every gene in the individual has the same probability to be replaced by a random integer ranged from 1 to n .

PSO is another meta heuristic method, which is based on flocks of birds. A set of particles are sprinkled in the solution space with their initial velocities. Then, the position and velocity of every particle will be updated again and again according to the position of the best particle at present and the best particle in the whole procedure. As PSO has also been introduced by many previous works, this paper will not introduce the detailed operation procedures of PSO and only focus on what is designed in our PSO approach for the specific visual coverage tasks. In this paper, each particle is an array made up of D integers with each integer represents the index of a camera. There are D initial velocities for the D elements in each particle which are all randomly generated from $-v_{max}$ to v_{max} . v_{max} is the speed limit for elements in the particle, which means the velocity of an element can not exceed v_{max} in the updating procedure.

Through lots of trials on the models used in this paper, crossover probability is chosen as 0.4 in GA because with this crossover probability the GA approach has the best convergence property. As the mutation probability of GA is usually low, mutation probability is chosen as 0.001 in this paper. In PSO, v_{max} is chosen as 20. As the population size influences the performance of GA and PSO, a lot of trials are made and the population size of GA and PSO are chosen as 30 and 20, respectively.

5.3 DE-SC approach

As a very effective global search algorithm with a quite simple mathematical structure, DE gains its popularity in various application in recent years. However, DE has not been reported to be adopted to solve the visual sensor deployment problem, and this paper makes an attempt. It contains four steps: initializing, mutation, crossover and selection. Though modules of DE and GA are similar, in fact these two algorithms are quite different. The essential difference is that the DE algorithm generates new individuals mainly by mutation, while GA mainly use crossover to generate new individuals with mutation playing a role of assistance. In addition, the strategies of DE and GA are also different in the modules of mutation and selection.

For the DE algorithm, its individual is an array made up of D integers. Each integer ranges from 1 to n , representing the index of a camera. pop is an integer which represents the population size. Let $x_i(0)$ be the i th individual of the 0th generation. $x_{ij}(0)$ denotes the j th gene of $x_i(0)$. v_i represents the i th individual after mutation. u_i denotes the i th individual after crossover. A brief introduction of the adopted DE algorithm is as follow.

Initializing the population:

$$\begin{aligned} \{x_i(0) | 1 \leq x_{ij}(0) \leq n, i = 1, 2 \dots pop; j = 1, 2 \dots D\} \\ x_{ij}(0) = randint(1, n) \end{aligned} \tag{7}$$

where $randint(1, n)$ is an uniformly distributed random integer from 1 to n .

Mutation: in this step, the individual v_i is generated by mutation

$$\begin{aligned} v_i(g) = x_{r_1}(g) + F \cdot (x_{r_2}(g) - x_{r_3}(g)) \\ i \neq r_1 \neq r_2 \neq r_3 \end{aligned} \tag{8}$$

where F is a zoom factor, and $x_i(g)$ is the i th individual of g th generation. r_1, r_2 , and r_3 are uniformly distributed random integers from 1 to n . In the process of evolution, if a gene of $v_i(g)$ is smaller than 1 or larger than n , it will be reproduced by the same random way as the population is initialized in (6).

Crossover: In this step, the individual u_i is generated by crossover between v_i and x_i .

$$u_{ij}(g) = \begin{cases} v_{ij}(g) & rand(0, 1) \leq p_c \\ x_{ij}(g) & otherwise \end{cases} \quad j = j_{rand} \tag{9}$$

where p_c is the crossover probability and $rand(0, 1)$ is an uniformly distributed random decimal between 0 and 1. j_{rand} is a randomly distributed integer between 1 and D . In this paper, $p_c = 0.3$. To ensure at least one gene of $v_{i,j}(g)$ delivers to the next generation, let the j_{rand} th gene pass to $u_{i,j}(g)$. The process is as Fig. 3. As is shown in the Fig. 1, the crossover procedure is started at the j_{rand} gene and the 4th gene of $u_i(g)$ is delivered to v_i as j_{rand} is 4. Then, the next gene of u_i comes from v_i as the random decimal r_1 is less than or equal to p_c . This procedure repeats until $u_i(g)$ is completely generated.

Selection: this step decides which one between $u_i(g)$ and $x_i(g)$ comes into the next generation as

$$x_i(g + 1) = \begin{cases} u_i(g) & f(u_i(g)) \leq f(x_i(g)) \\ x_i(g) & otherwise \end{cases} \tag{10}$$

where $f(x_i(g))$ is the number of triangles covered by individual $x_i(g)$. The condition to be left is that the individual covers more triangles.

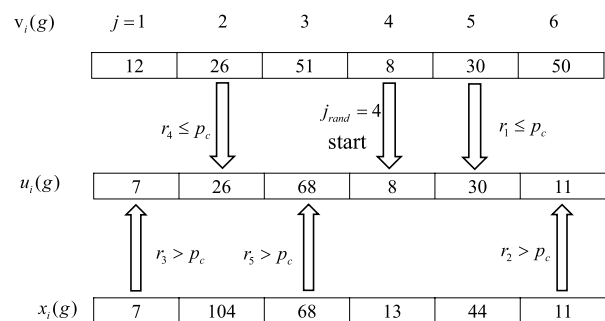


Fig. 3 Crossover

Table 1 Time cost of different GDSS methods

Triangles	100	200	300	400	500	600
S-GDSS (ms)	69	533	1739	4102	8055	13723
P-GDSS (ms)	131	223	382	476	610	724

Let $gbest$ represent the number of triangles covered by the best individual in the whole process, r_1, r_2 and r_3 represent random integers from 1 to n . $maxGeneration$ represents for which times the generation will evolve. The differential evolution approach for selecting cameras (DE-SC) can be described as Algorithm 4.

Algorithm 4 DE – SC

Require: D, CF, R, T

Ensure: N, R_c, T_c

```

1: Initialization:  $N = 0, R_c = 0, T_c = 0, gbest = 0$ 
2: for  $i = 1 \rightarrow pop$  do
3:   for  $j = 1 \rightarrow D$  do
4:      $x_{i,j} = randint(1, n)$ 
5:   end for
6: end for
7: while  $maxGeneration > 0$  do
8:   for  $i = 1 \rightarrow pop$  do
9:     for  $j = 1 \rightarrow D$  do
10:      if  $rand(0, 1) < p_c$  or  $j = j_{rand}$  then
11:         $u_{i,j} = x_{r_1,j} + F \times (x_{r_2,j} - x_{r_3,j})$ 
12:        if  $u_{i,j} > n$  or  $u_{i,j} < 1$  then
13:           $u_{i,j} = randint(1, n)$ 
14:        end if
15:      end if
16:    end for
17:  end for
18:  for  $i = 1 \rightarrow pop$  do
19:    if  $f(u_i) > f(x_i)$  then
20:       $x_i = u_i$ 
21:    end if
22:    if  $f(x_i) > gbest$  then
23:       $gbest = f(x_i)$ 
24:      for  $j = 1 \rightarrow D$  do
25:         $R_c(j) = R(x_{i,j}), T_c(j) = T(x_{i,j})$ 
26:      end for
27:    end if
28:  end for
29:   $maxGeneration = maxGeneration - 1$ 
30: end while
31:  $N = gbest$ 
32: return  $R_c, T_c, N$ 

```

5.4 BIP-SC approach

In this subsection, the deployment problem is formulated as a binary integer programming problem and solved by a commercial optimizer ‘Gurobi’.¹ The aim of optimization is to maximize the number of covered triangles, which is

equivalent to minimizing the number of uncovered triangles. Let a_i represent whether the i th camera is selected and b_i denotes whether the i th triangle is covered, where

$$a_i = \begin{cases} 0, & \text{the } i\text{th camera is not selected} \\ 1, & \text{the } i\text{th camera is selected} \end{cases} \quad (11)$$

$$b_i = \begin{cases} 0, & \text{the } i\text{th triangle is covered} \\ 1, & \text{the } i\text{th triangle is not covered} \end{cases} \quad (12)$$

e_{ij} is the i th row j th column element of matrix CF . Then the set cover problem can be formulated into the following binary integer programming problem:

$$\min \sum_{i=1}^n b_i \quad (13)$$

$$s.t. \begin{cases} \sum_{i=1}^n a_i = D \\ \sum_{i=1}^n a_i \cdot e_{i,1} + b_1 \geq 1 \\ \sum_{i=1}^n a_i \cdot e_{i,2} + b_2 \geq 1 \\ \dots \\ \sum_{i=1}^n a_i \cdot e_{i,n} + b_n \geq 1 \end{cases} \quad (14)$$

The optimization software Gurobi is used to solve this BIP problem due to its internal powerful numerical algorithms for mixed-integer problems with linear constraints.

6 Performance evaluation

This section will compare the two GDSS approaches presented in Sect. 4 and the five SC approaches in Sect. 5.²

6.1 Comparison for two GDSS approaches

To compare the time performance of the S-GDSS and P-GDSS, we employ 3-D models with increasing amount

¹ The optimizer Gurobi can be found at <http://www.gurobi.com>.

² An open source implementation of these approaches could be found at <https://github.com/MobFlyRobLab/COVSN> for interested readers to repeat the evaluation results.

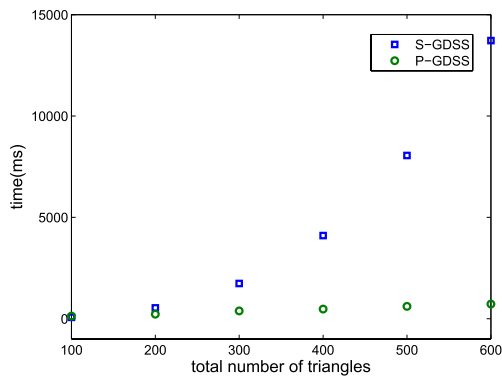


Fig. 4 Comparison of efficiency: the vertical axis indicates the time cost in ms, the horizontal axis denotes the amounts of triangles in object models

of triangles from 100 to 600. For each triangle in the model, a viewpoint is generated in the way stated in Sect. 3.3. The program is running on a computer of i5-6300HQ CPU and NVIDIA GeForce GTX950M GPU.

Table 1 shows the comparison of the S-GDSS approach and the P-GDSS approach. Figure 4 visualizes the data in the Table 1. It is seen from Fig. 4 that when the number of triangles is 100, the S-GDSS approach is faster than P-GDSS. This may result from the time cost of the data exchange between GPU and CPU in P-GDSS. But soon the time consuming of S-GDSS exceed that of P-GDSS. And the difference of efficiency between these two approaches becomes larger and larger as the amount of triangles grows. It is shown in Fig. 4 that the time cost of P-GDSS is roughly linear to the scale of the model while the time cost of S-GDSS

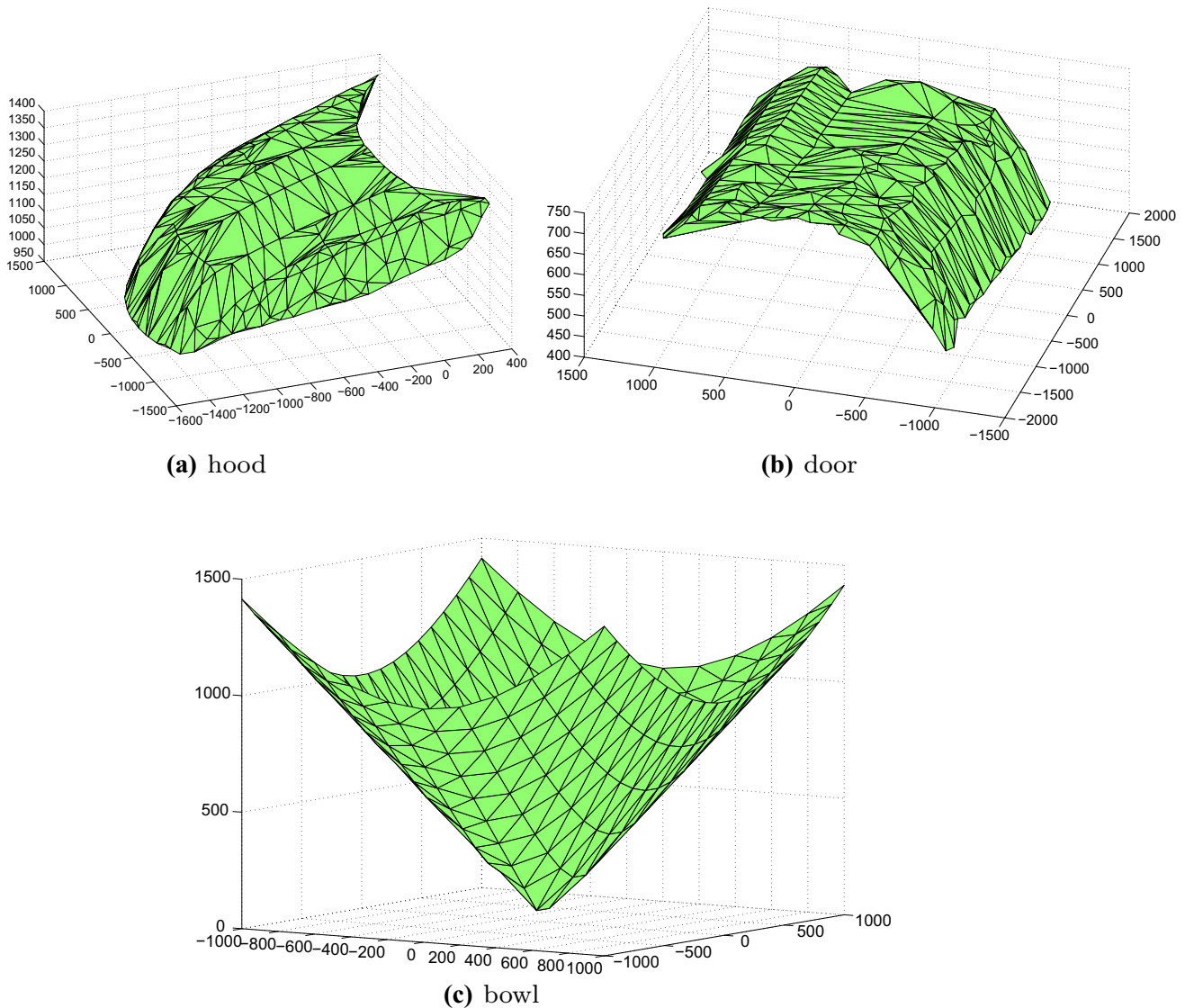


Fig. 5 Models of different shapes

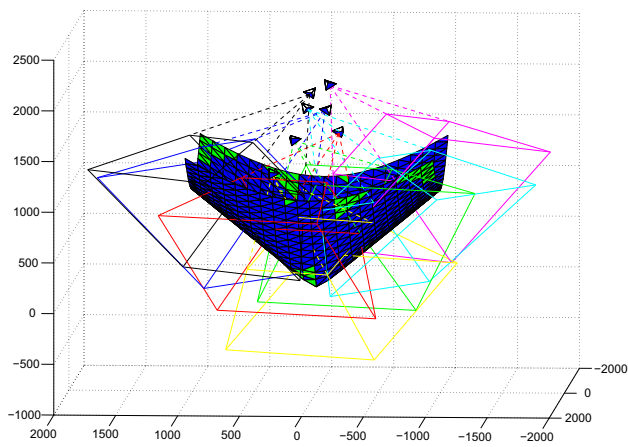


Fig. 6 Simulation result of G-SC

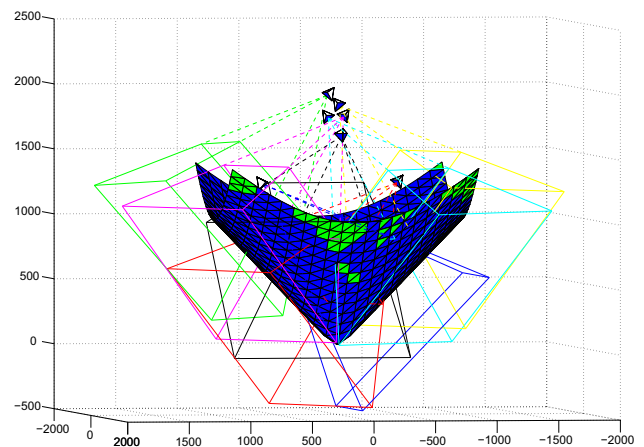


Fig. 8 Simulation result of PSO-SC

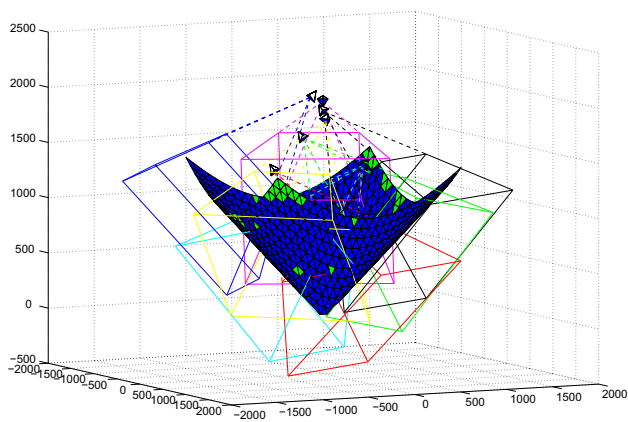


Fig. 7 Simulation result of GA-SC

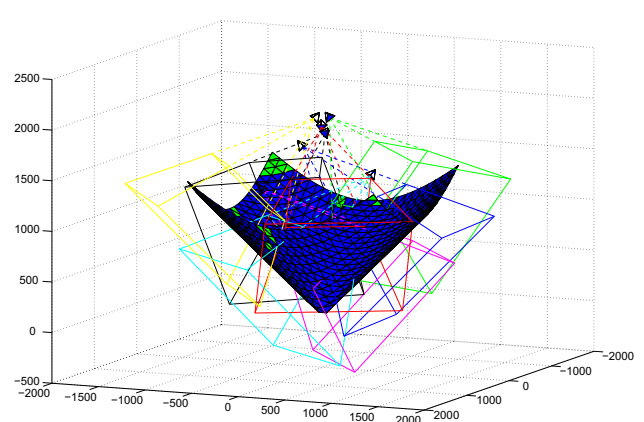


Fig. 9 Simulation result of DE-SC

increases sharply. It can be found from Table 1 that when the number of triangles in the used 3-D model grows to 600, the time cost of P-GDSS approach is still less than 1 s while the S-GDSS costs more than 13 s. Note that these approaches are running on the release mode of VS2013 which means the program has been optimized in time cost. Otherwise, the gap between these two approaches will be even larger.

6.2 Comparison for five SC approaches

To compare the performance of the the five SC approaches stated in Sect. 5, three models with different shapes but almost the same number of triangle faces are adopted. As shown in Fig. 5, the three models are door, hood and bowl. And then, bowl models with the same shape but higher resolutions are also used. A comparison of time efficiency and coverage results is depicted in Tables 2 and 3, where the number of cameras are set as $D = 7$. The coverage results

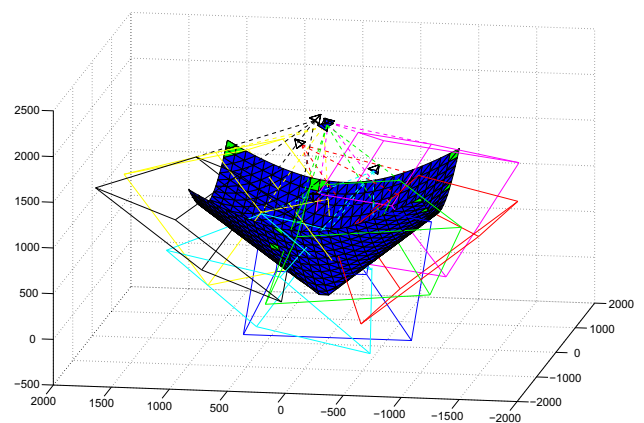


Fig. 10 Simulation result of BIP-SC

of five SC approaches on one model is shown in Figs. 6, 7, 8, 9 and 10, where the blue areas denote covered triangles and the green areas denote those uncovered.

Table 2 Number of triangles covered by different methods

Model	hood	door	bow11	bow12	bow13	bow14	bow15	bow16	bow17
Total triangles	378	380	380	600	800	1000	1200	1400	1600
Mean									
G-SC	370	350	331	535	722	937	1098	1314	1495
GA-SC	370.5	362.7	352.3	575	771.6	975.1	1167.7	1360.2	1558.7
PSO-SC	361.8	359.2	341.9	564	760.2	957.1	1141.6	1333.7	1539.4
DE-SC	372.2	364.6	357.2	574.7	776.5	977.2	1171.6	1371.1	1569.3
BIP-SC	373	366	360	579	781	985	1180	1380	1579
Best									
G-SC	370	350	331	535	722	937	1098	1314	1495
GA-SC	373	364	360	579	778	981	1176	1373	1572
PSO-SC	369	363	348	573	774	965	1164	1356	1561
DE-SC	373	366	359	576	779	982	1175	1378	1571
BIP-SC	373	366	360	579	781	985	1180	1380	1579
Worst									
G-SC	370	350	331	535	722	937	1098	1314	1495
GA-SC	368	361	344	570	761	970	1162	1341	1535
PSO-SC	356	352	335	559	743	947	1106	1304	1510
DE-SC	371	363	353	571	774	974	1167	1367	1567
BIP-SC	373	366	360	579	781	985	1180	1380	1579
Std dev.									
G-SC	0	0	0	0	0	0	0	0	0
GA-SC	1.74	1.00	5.36	2.52	6.08	4.50	4.71	10.71	9.48
PSO-SC	3.73	3.21	3.70	3.76	8.57	5.59	14.98	17.68	14.07
DE-SC	0.87	0.80	1.77	1.41	1.50	2.56	2.05	2.77	1.18
BIP-SC	0	0	0	0	0	0	0	0	0

6.2.1 Coverage performance

In Table 2, the coverage performance of the five SC approaches are presented. The first row of this table is the model name and the second row shows the numbers of triangles of different 3-D models, while other rows show the number of covered triangles by different approaches listed in the second column. Each approach runs ten times on every model and the best case, worst case, average and standard deviation are recorded.

It is seen that, among these five approaches, the G-SC approach possesses the worst coverage performance, which cannot satisfy the requirement of many practical applications. Thanks to the powerful Gurobi optimizer, BIP-SC holds the best coverage performance on all the models. Yet, since the problem is essentially NP-hard, the numerical algorithm cannot achieve the optimal solution in finite time when the size of the 3-D model is too large. The three heuristic approaches, GA-SC, PSO-SC and DE-SC achieve a compromise between the BIP-SC and G-SC, and they have slightly

worse performance than BIP-SC. To compare the performance among GA-SC, PSO-SC and DE-SC, we set termination conditions to make the three methods take roughly the same time and the average number of triangles covered by DE-SC is always the largest, which means DE-SC is the best heuristic approach for the visual coverage task.

6.2.2 Time cost

Table 3 shows time costs of different approaches with different models. It can be seen that the time cost of BIP-SC approach increases fast as the number of triangles grows. With a number of 1600 triangles, the time cost is nearly 3 min. It is also shown that BIP-SC approach can take much time even on a model with small triangles such as the hood model. Meanwhile, the three heuristic algorithms hold reasonable time costs which increase steadily as the amount of triangles grows. Therefore, regarding the time complexity, the three heuristic algorithms are better than BIP-SC.

Table 3 Time cost of five SC methods (unit: ms)

Model	hood	door	bowl1	bowl2	bowl3	bowl4	bowl5	bowl6	bowl7
Total triangles	378	380	380	600	800	1000	1200	1400	1600
Mean									
G-SC	370	350	331	535	722	937	1098	1314	1495
GA-SC	3080	2220	2068	3176	4423	5931	6594	7395	8335
PSO-SC	3282	2398	1993	3152	4236	5503	6123	7044	7872
DE-SC	3203	2275	1920	3197	4296	5424	6038	6956	7786
BIP-SC	28,493	289	746	9419	15,891	26,312	60,301	114950	175257
Best									
G-SC	2.8	2.9	2.9	6.6	11	19	27	39	53
GA-SC	2797	2065	2010	3060	4100	5571	6365	7158	8102
PSO-SC	3083	2266	1921	3035	4026	5156	5716	6835	7758
DE-SC	3103	2141	1865	3149	4099	5195	5941	6797	7709
BIP-SC	28,403	285	740	9388	15,756	25,549	60,118	114146	174869
Worst									
G-SC	3.0	3.2	3.3	7.2	12	22	33	46	58
GA-SC	3405	2389	2153	3391	4685	6216	7024	7697	8803
PSO-SC	3458	2602	2120	3368	4544	5737	6487	7362	7984
DE-SC	3336	2454	1987	3366	4794	5732	6198	7243	7891
BIP-SC	28,782	297	763	9490	16,150	27,118	60,648	117530	176146
Std dev.									
G-SC	0.05	0.07	0.12	0.19	0.34	1.00	2.05	2.29	1.53
GA-SC	157	88	35	102	208	197	177	176	213
PSO-SC	111	104	54	89	189	194	227	176	78
DE-SC	82	84	39	62	223	164	84	144	66
BIP-SC	120	3.7	6.4	34	138	448	172	970	360

6.2.3 Convergence speed of DE-SC, GA-SC and PSO-SC

Unlike the G-SC approach and BIP-SC approach, the three evolution algorithms (GA-SC, PSO-SC and DE-SC) produce different results at different time instants. Tables 2 and 3 only display those results at a specific time instant, which is not fully enough to demonstrate the comparative performance of them. Therefore, we record number of triangles covered by each of the three approaches as the time increases with the bowl models of different number of triangles, and the final results are provided in Fig. 11. It can be seen from Fig. 11 that the lines of DE-SC are on the top of GA-SC and PSO-SC in every subfigure, which means it can reach the best coverage performance with fixed given time or achieve the best time efficiency for a fixed coverage rate. Therefore, DE-SC is best among the three approaches.

To summarize, among these approaches, the G-SC approach holds the fastest speed at the cost of coverage rate; BIP-SC sacrifices the time efficiency and maintains the best coverage performance; GA-SC, PSO-SC, DE-SC reach compromises between time and coverage performance, wherein DE-SC outperforms GA-SC and PSO-SC.

7 Conclusions

In this paper, we give a survey and comparison of five typical visual coverage optimization approaches for observing 3-D objects, including the greedy algorithm, GA, PSO, DE, and BIP. In order to improve the computational efficiency, a parallel visual occlusion detection approach is proposed and implemented with graphic processing units (GPUs), which are then integrated into all of the aforementioned optimization approaches for a fair comparison. Evaluation results show that (1) the proposed parallel occlusion detection algorithm largely improves the computational efficiency; (2) among the five typical approaches, BIP has the best coverage performances yet with the highest time cost, and greedy algorithm is the fastest approach at the price of coverage performance; GA, PSO, and DE achieve a compromise between the performance and the time cost, while DE has better coverage performance and less time cost than PSO and GA. In the future, visual coverage of large-scale 3-D scenarios and distributed visual coverage algorithms still needs to be explored as a rather immature research field.

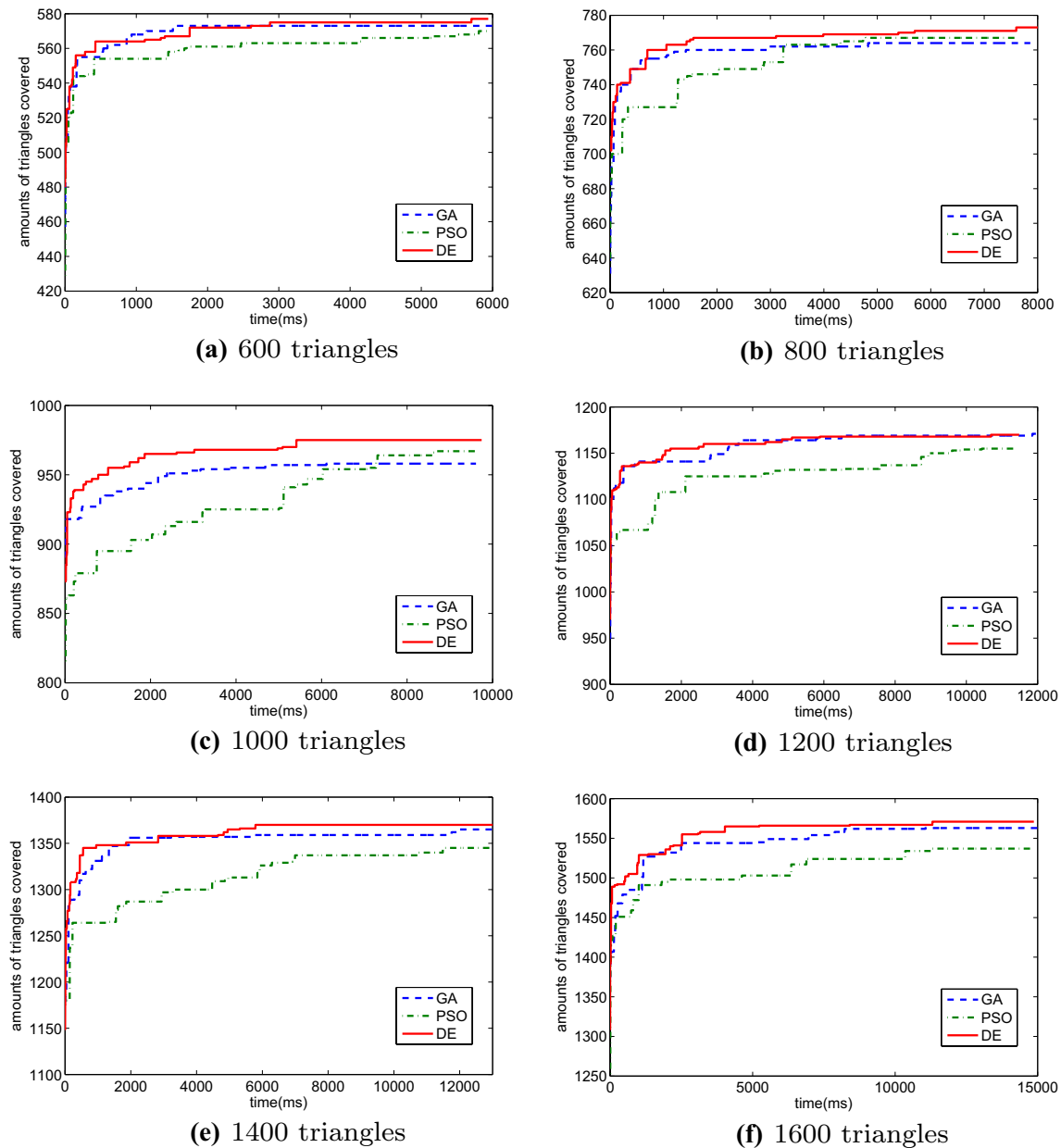


Fig. 11 Simulation results of three heuristic approaches with different models

Acknowledgements This work is supported in part by National Natural Science Foundation of China (NSFC) under Grant 61573195 and U1613210.

References

- Abdelkader, A., Mokhtar, M., El-Alfy, H.: Angular Heuristics for coverage maximization in multi-camera surveillance. In: Proceedings of IEEE International Conference on Advanced Video and Signal-Based Surveillance, pp. 373–378 (2008)
- Aghajan, H., Cavallaro, A.: Multi-Camera Networks: Principles and Applications. Academic Press, New York (2009)
- Ai, J., Abouzeid, A.A.: Coverage by directional sensors in randomly deployed wireless sensor networks. *J. Comb. Optim.* **11**(1), 21–41 (2006)
- Alarcon-Herrera, J.L., Chen, X., Zhang, X.: Viewpoint selection for vision systems in industrial inspection. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 4934–4939 (2014a)
- Alarcon-Herrera, J.L., Cheng, X., Zhang, X.: Viewpoint selection for vision systems in industrial inspection. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 4934–4939 (2014b)

- Altahir, A.A., Asirvadam, V.S., Hamid, N.H., Sebastian, P., Saad, N., Ibrahim, R., Dass, S.C.: Modeling multicamera coverage for placement optimization. *IEEE Sens. Lett.* **1**(6), 1–4 (2017a)
- Altahir, A.A., Asirvadam, V.S., Hamid, N.H., Sebastian, P., Saad, N., Ibrahim, R., Dass, S.C.: Optimizing visual surveillance sensor coverage using dynamic programming. *IEEE Sens. J.* **17**(11), 3398–3405 (2017b)
- Altahir, A.A., Asirvadam, V.S., Hamid, N.H., Sebastian, P., Saad, N., Ibrahim, R., Dass, S.C.: Optimizing visual sensor coverage overlaps for multiview surveillance systems. *IEEE Sens. J.* **18**(1), 4544–4552 (2018)
- Azin, N., Saeed, A., Harras, K., Mohamed, A.: On target coverage in mobile visual sensor networks. In: Proceedings of the 12th ACM International Symposium on Mobility Management and Wireless Access, pp. 39–46 (2014)
- Basgumus, A., Mustafa, N., Gunes, Y., Ahmet, A.: Performance comparison of the differential evolution and particle swarm optimization algorithms in free-space optical communications systems. *Adv. Electr. Comput. Eng.* **15**(2), 17–22 (2015)
- Bodor, R., Drenner, A., Schrater, P., Papanikolopoulos, N.: Optimal camera placement for automated surveillance tasks. *J. Intell. Rob. Syst.* **50**(3), 257–295 (2007)
- Charfi, Y., Wakamiya, N., Murata, M.: Challenging issues in visual sensor networks. *IEEE Wirel. Commun.* **16**(2), 44–49 (2009)
- Chen, S.Y., Li, Y.F.: Automatic sensor placement for model-based robot vision. *IEEE Trans. Syst. Man Cybern. B Cybern.* **34**(1), 393–408 (2004a)
- Chen, S.Y., Li, Y.F.: Automatic sensor placement for model-based robot vision. *IEEE Trans. Syst. Man Cybern.* **34**(1), 393–408 (2004b)
- Chen, S.Y., Li, Y.F., Zhang, J.W., Wang, W.L.: Active Sensor Planning for Multiview Vision Tasks. Springer, Berlin (2008)
- Chen, S.Y., Li, Y.F., Kwok, N.M.: Active vision in robotic systems: a survey of recent developments. *Int. J. Robot. Res.* **30**(11), 1343–1377 (2011)
- Chen, C., Mukhopadhyay, S.C., Chuang, C., Lin, T., Liao, M., Wang, Y., Jiang, J.: A hybrid memetic framework for coverage optimization in wireless sensor networks. *IEEE Trans. Cybern.* **45**(10), 2309–2322 (2014)
- Chow, K.Y., Lui, K. S., Lam, E. Y.: Maximizing angle coverage in visual sensor networks. In: Proceedings of IEEE International Conference on Communications, pp. 3516–3521 (2007)
- Civicioglu, P., Erkan, B.: A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artif. Intell. Rev.* **39**(4), 315–346 (2013)
- Costa, D.G., Guedes, L.A.: The coverage problem in video-based wireless sensor networks: a survey. *Sensors* **10**(9), 8215–8247 (2010)
- Costa, D.G., Duran-Faundez, C., Bittencourt, J.C.: Availability issues for relevant area coverage in wireless visual sensor networks. In: CHILEAN Conference on Electrical, pp. 1–6. Electronics Engineering, Information and Communication Technologies (CHILECON) (2017)
- Deb, A., Roy, J.S., Gupta, B.: Performance comparison of differential evolution, particle swarm optimization and genetic algorithm in the design of circularly polarized microstrip antennas. *IEEE Trans. Antennas Propag.* **62**(8), 3920–3928 (2014)
- Deif, D.S., Gadallah, Y.: Classification of wireless sensor networks deployment techniques. *IEEE Commun. Surv. Tutor.* **16**(2), 834–855 (2014)
- Fanti, M.P., Faraut, G., Lesage, J.J., Roccotelli, M.: An integrated framework for binary sensor placement and inhabitants location tracking. *IEEE Trans. Cybern.* **48**(1), 154–160 (2018)
- Fu, Y.G., Zhou, J., Deng, L.: Surveillance of a 2D plane area with 3D deployed cameras. *Sensors* **14**(2), 1988–2011 (2014)
- Guvensan, M.A., Yavuz, A.G.: On coverage issues in directional sensor networks: a survey. *Ad Hoc Netw.* **9**(7), 1238–1255 (2011)
- Heidali, E., Movaghar, A.: An efficient method based on genetic algorithms to solve sensor network optimization problem. *Int. J. Appl. Graph Theory Wirel. Ad Hoc Netw. Sens. Netw.* **3**(1), 18–33 (2011)
- Hörster, E., Lienhart, R.: Approximating optimal visual sensor placement. In: IEEE International Conference on Multimedia and Expo, pp. 1257–1260 (2006a)
- Hörster, E., Lienhart, R.: On the optimal placement of multiple visual sensors. In: Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks, pp. 111–120 (2006b)
- Hughes, J.F., Dam, A.V., McGuire, M., Sklar, D.F., Foley, J.D., Feiner, S.K., Akeley, K.: Computer Graphics: Principles and Practice, 3rd edn. Addison Wesley, Boston (2013)
- Indu, S., Chaudhury, S.: Optimal sensor placement for surveillance of large spaces. In: Proceedings of the Third International Conference on Distributed Smart Cameras, pp. 1–8 (2009)
- Iwan, M., Akmeliawati, R., Faisal, T., Al-Assadi, H.M.: Performance comparison of differential evolution and particle swarm optimization in constrained optimization. *Procedia Eng.* **41**, 1323–1328 (2012)
- Jesus, T.C., Costa, D.G., Portugal, P.: On the computing of area coverage by visual sensor networks: assessing performance of approximate and precise algorithms. In: IEEE 16th International Conference on Industrial Informatics (INDIN), pp. 193–198 (2018)
- Kachitvichyanukul, V.: Comparison of three evolutionary algorithms: GA, PSO, and DE. *Ind. Eng. Manag. Syst.* **11**(3), 215–223 (2012)
- Kang, Z., Medioni, G.: Progressive 3D model acquisition with a commodity hand-held camera. In: Proceedings of IEEE Winter Conference on Applications of Computer Vision, pp. 270–277 (2015)
- Konar, M., Aytakin, B.: Performance comparison of particle swarm optimization, differential evolution and artificial bee colony algorithms for fuzzy modelling of nonlinear systems. *Elektronika ir Elektrotechnika* **22**(5), 8–13 (2016)
- Konda, K.R., Conci, M.: Global and local coverage maximization in multi-camera networks by stochastic optimization. *Infocommun. J.* **5**(1), 1–8 (2013)
- Kulkarni, R.V., Venayagamoorthy, G.K.: Particle swarm optimization in wireless-sensor networks: a brief survey. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **41**(2), 262–267 (2011a)
- Kulkarni, R.V., Venayagamoorthy, G.K.: Particle swarm optimization in wireless-sensor networks: a brief survey. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **41**(2), 262–267 (2011b)
- Liu, J., Sridharan, S., Fookes, C., Wark, T.: Optimal camera planning under versatile user constraints in multi-camera image processing systems. *IEEE Trans. Image Process.* **23**(1), 171–184 (2014)
- Loscrí, V., Natalizio, E., Guerriero, F., Aloí, G.: Particle swarm optimization schemes based on consensus for wireless sensor networks. In: Proceedings of the 7th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks, pp. 77–84 (2012)
- Malik, R., Bajcsy, P.: Automated placement of multiple stereo cameras. In: Proceedings of 8th ECCV Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras, pp. 1–14 (2008)
- Mavrinac, A.: Modeling and optimizing the coverage of multi-camera system. Ph.D. Dissertation of University of Windsor (2012)
- Mavrinac, A., Chen, X.: Modeling coverage in camera networks: a survey. *Int. J. Comput. Vis.* **101**(1), 205–226 (2013)
- Mavrinac, A., Chen, X., Alarcon-Herrera, J.L.: Semiautomatic model-based view planning for active triangulation 3-D inspection systems. *IEEE/ASME Trans. Mechatron.* **20**(2), 799–811 (2014)
- Miller, A., White, B., Charbonneau, E., Kanzler, Z., Laviola, J.J.: Interactive 3D model acquisition and tracking of building block

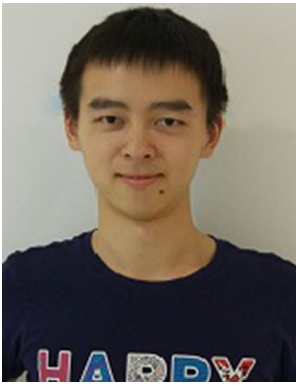
- structures. *IEEE Trans. Vis. Comput. Graph.* **18**(4), 651–659 (2012)
- Morsly, Y., Aouf, N., Djouadi, M.S., Richardson, M.: Particle swarm optimization inspired probability algorithm for optimal camera network placement. *IEEE Sens. J.* **12**(5), 1402–1412 (2012)
- Mulligan, R., Ammari, H.M.: Coverage in wireless sensor networks: a survey. *Netw. Protoc. Algorithms* **2**(2), 27–53 (2010)
- Munishwar, V.P., Abu-Ghazaleh, N.B.: Coverage algorithms for visual sensor networks. *ACM Trans. Sens. Netw.* **9**(4), 1–34 (2013)
- Navin, A.H.: Distributed genetic algorithm to solve coverage problem in wireless camera-based sensor networks. *Res. J. Resent Sci.* **4**(12), 106–109 (2015)
- Park, J., Bhat, P.C., Kak, A.C.: A look-up table based approach for solving the camera selection problem in large camera networks. In: *Proceedings of International Workshop on Distributed Smart Cameras*, pp. 72–76 (2006)
- Ready, K.K., conci, N.: Camera positioning for global and local coverage optimization. In: *ACM/IEEE International Conference on Distributed Smart Cameras*, pp. 1–6 (2012)
- Seok, J., Lee, J., Oh, C., Lee, J., Lee, H.J.: RFID sensor deployment using differential evolution for indoor mobile robot localization. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, pp. 3719–3724 (2010)
- Shi, Q., Xi, N., Sheng, W.: Recursive measurement process for improving accuracy of dimensional inspection of automotive body parts. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 4764–4769 (2007)
- Singh, V.K., Atrey, P.K., Kankanhalli, M.S.: Cooperative multi-camera surveillance using model predictive control. *Mach. Vis. Appl.* **19**(5), 375–393 (2008)
- Soltani, H.R., Haghghat, A.T., Rashidi, H., Chegini, T.G.: A couple of algorithms for k-coverage problem in visual sensor networks. In: *Proceedings of International Conference on Communication Engineering and Networks*, pp. 96–101 (2011)
- Soro, S., Heinzelman, W.B.: On the coverage problem in video-based wireless sensor networks. In: *Proceedings of 2nd International Conference on Broadband Networks*, pp. 9–16 (2005)
- Soro, S., Heinzelman, W.: A survey of visual sensor networks. *Adv. Multimedia* **2009**, 640386 (2009). <https://doi.org/10.1155/2009/640386>
- Taj, M., Cavallaro, A.: Distributed and decentralized multicamera tracking. *IEEE Signal Process Mag.* **28**(3), 46–58 (2011)
- Tarabanis, K., Allen, P., Tsai, R.: A survey of sensor planning in computer vision. *IEEE Trans. Robot. Autom.* **11**(1), 86–104 (1995)
- Topcuoglu, H. R., Ermis, M., Sifyan, M.: Hybrid evolutionary algorithms for sensor placement on a 3D terrain. In: *Proceedings of 9th International Conference on Intelligent Systems Design and Application*, pp. 511–516 (2009)
- Ugolotti, R., Nashed, Y., Mesejo, P., Špela Ivekovič, L., Mussi, S., Cagnoni, : Particle swarm optimization and differential evolution for model-based object detection. *Appl. Soft Comput.* **13**(6), 3092–3105 (2013)
- Ülker, E., Haydar, A.: Comparison of the performances of differential evolution, particle swarm optimization and harmony search algorithms on benchmark functions. *Acad. Res. Int.* **3**(2), 85–92 (2012)
- Urrutia, J.: Art gallery and illumination problems. In: *Handbook of Computational Geometry*, pp. 973–1027 (2000). <https://doi.org/10.1016/b978-044482537-7/50023-1>
- Wang, X.: Intelligent multi-camera video surveillance: a review. *Pattern Recogn. Lett.* **34**(1), 3–19 (2013)
- Wang, C., Qi, F., Shi, G.: Nodes placement for optimizing coverage of visual sensor networks. *Adv. Multimed. Inf. Process. PCM* **2009**, 1144–1149 (2009)
- Xu, J., Xi, N., Zhang, C., Shi, Q., Gregory, J.: Real-time 3D shape inspection system of automotive parts based on structured light pattern. *Opt. Laser Technol.* **43**(1), 1–8 (2011a)
- Xu, Y.C., Lei, B., Hendriks, E.A.: Camera network coverage improving by particle swarm optimization. *J. Image Video Process.* **3**, 1–10 (2011b)
- Yen, H.H.: Efficient visual sensor coverage algorithm in wireless visual sensor networks. In: *Proceedings of IEEE International Wireless Communications and Mobile Computing Conference*, pp. 1516–1521 (2013)
- Yen, H. H.: Novel visual sensor deployment algorithm in PTZ wireless visual sensor networks. In: *Proceedings of IEEE Asia Pacific Conference on Wireless and Mobile*, pp. 214–218 (2014)
- Yoon, Y., Kim, Y.H.: An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. *IEEE Trans. Cybern.* **43**(5), 1473–1483 (2013)
- Zhang, X., Alarcon-Herrera, J. L., Chen, X.: Optimization for 3-D model-based multi-camera deployment. In: *Proceedings of International Federation of Automatic Control*, pp. 10126–10131 (2014)
- Zhang, X., Alarcon-Herrera, J.L., Chen, X.: Coverage enhancement for deployment of multi-camera networks. In: *Proceedings of IEEE International Conference on Advanced Intelligent Mechatronics*, pp. 909–914 (2015a)
- Zhang, X., Chen, X., Alarcon-Herrera, J.L., Fang, Y.: 3-D model-based multi-camera deployment: a recursive convex optimization approach. *IEEE/ASME Trans. Mechatron.* **20**(6), 3157–3169 (2015b)
- Zhang, B., Zhang, X., Chen, X., Fang, Y.: A differential evolution approach for coverage optimization of visual sensor networks with parallel occlusion detection. In *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Banff, AB, pp. 1246–1251 (2016a)
- Zhang, X., Zhang, Y., Zhang, Q., Lee, V.C.S., Li, M.: Problem specific MOEA/D for barrier coverage with wireless sensors. *IEEE Trans. Cybern.* **47**, 1–12 (2016b). **(online published)**
- Zhao, J., Cheung, S.S.: Optimal visual sensor planning. In: *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 165–168 (2009)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Xuebo Zhang received the B.Eng. degree in automation from Tianjin University, Tianjin, China, in 2006, and the Ph.D. degree in control theory and control engineering from Nankai University, Tianjin, in 2011. From 2014 to 2015, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON, Canada. He is currently an Associate Professor with the Institute of Robotics and Automatic Information System and also Tianjin Key Laboratory of

Intelligent Robotics, Nankai University. His research interests include motion planning, visual servoing, and visual sensor network. Dr. Zhang is an Associate Editor of the *ASME Journal of Dynamic Systems, Measurement, and Control*.

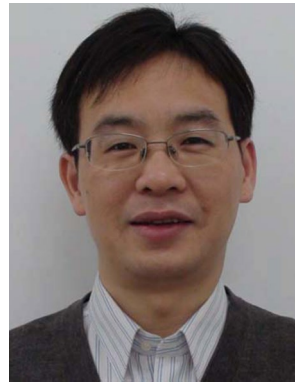


Boyu Zhang received the B.S. degree in automation from Nankai University, Tianjin, China, in 2016, and the M. Eng. degree in control science and engineering from Nankai University, Tianjin, in 2019. His research interests include visual sensor network and robot vision.



Xiang Chen received the M.Sc. and Ph.D. degrees from Louisiana State University, Baton Rouge, LA, USA, in 1996 and 1998, respectively, both in systems and control. Since 2000, he has been with the Department of Electrical and Computer Engineering, University of Windsor, ON, Canada, where he is currently a Professor. His research interests include robust control, vision sensor networks, vision-based control systems, networked control systems, and industrial applications of control

theory. He is currently an Senior Editor of IEEE/ASME Transactions on Mechatronics and an associate editor of SIAM Journal on Control and Optimization.



Yongchun Fang received the B.S. degree in electrical engineering and the M.S. degree in control theory and applications from Zhejiang University, Hangzhou, China, in 1996 and 1999, respectively, and the Ph.D. degree in electrical engineering from Clemson University, Clemson, SC, in 2002. From 2002 to 2003, he was a Postdoctoral Fellow with the Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY. He is currently a Professor with the Institute of

Robotics and Automatic Information System, Nankai University, Tianjin, China. His research interests include visual servoing, AFM-based Nano-systems, and control of underactuated systems including overhead cranes.