Recent Advances in Camera Planning for Large Area Surveillance: A Comprehensive Review

JUNBIN LIU, SRIDHA SRIDHARAN, and CLINTON FOOKES,
Queensland University of Technology

With recent advances in consumer electronics and the increasingly urgent need for public security, camera networks have evolved from their early role of providing simple and static monitoring to current complex systems capable of obtaining extensive video information for intelligent processing, such as target localization, identification, and tracking. In all cases, it is of vital importance that the optimal camera configuration (i.e., optimal location, orientation, etc.) is determined before cameras are deployed as a suboptimal placement solution will adversely affect intelligent video surveillance and video analytic algorithms. The optimal configuration may also provide substantial savings on the total number of cameras required to achieve the same level of utility.

In this article, we examine most, if not all, of the recent approaches (post 2000) addressing camera placement in a structured manner. We believe that our work can serve as a first point of entry for readers wishing to start researching into this area or engineers who need to design a camera system in practice. To this end, we attempt to provide a complete study of relevant formulation strategies and brief introductions to most commonly used optimization techniques by researchers in this field. We hope our work to be inspirational to spark new ideas in the field.

Categories and Subject Descriptors: G.1.6 [**Mathematics of Computing**]: Optimization

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Camera planning, camera placement, optimization, binary integer programming, swarm intelligence, simulated annealing, video surveillance

ACM Reference Format:
Junbin Liu, Sridha Sridharan, and Clinton Fookes. 2016. Recent advances in camera planning for large area surveillance: A comprehensive review. ACM Comput. Surv. 49, 1, Article 6 (May 2016), 37 pages.
DOI: http://dx.doi.org/10.1145/2906148

1. INTRODUCTION

In recent years, large scale camera networks have become ubiquitous due to their ability to provide abundantly rich video information and superior coverage of physical spaces of interest. Networks of cameras are now being deployed widely for Intelligent Video Surveillance (IVS). Based on user defined requirements, IVS systems can automatically identify potential risks by detecting, localizing, tracking, and recognizing targets or events of interest from video footages [McCool et al. 2008; Xu et al. 2012; Denman et al. 2011; Ryan et al. 2011]. For these video analytic and content analysis systems, it is of vital importance that the optimal camera configuration (i.e., optimal location,

This research was supported partly by an ARC Discovery grant DP110100827.
Authors' addresses: J. Liu, S. Sridharan, and C. Fookes, School of Electrical Engineering and Computer Science, Queensland University of Technology, 2 George Street, Brisbane, Australia 4000; emails: {junbin.liu, s.sridharan, c.fookes}@qut.edu.au.

DOI: http://dx.doi.org/10.1145/2906148

orientation, etc.) is determined before cameras are deployed, as the cost of modification is expensive. Optimal camera configuration will provide savings on the total number of cameras used to achieve the same or greater level of utility. Moreover, a properly placed camera network will bring significant benefits to subsequent computer vision tasks. For example, if the placement of the cameras in an airport environment can be configured such that the faces of passengers are always in view across the cameras at important locations of interest, the subsequent face recognition and tracking algorithm will be simplified. As another example, consider an optimal placement of cameras in a space such that 100% video coverage is obtained. If such a placement could be achieved with a minimal number of cameras, the subsequent development of an algorithm for multicamera based person tracking in a crowded environment will be considerably simplified.

With the increasing deployment of these large scale multicamera surveillance networks, the development of clever camera placement algorithms has become a research area of significant interest to the computer vision community. The camera placement problem has recently become a complex offline optimization problem due to constrained camera capabilities of various types, complex physical space that has to be covered, and stringent requirements specified by the end user, such as the requirement for automatic tracking of subjects or recognition of faces.

The simplest scenario of achieving a predefined video coverage of a given physical space while minimizing the number of cameras is still a difficult optimization problem in itself. Due to the NP hard nature of this problem [Erdem and Sclaroff 2006], the true optimum may not be feasible if the search space is large and thus preventing the use of simple enumeration and search techniques.

The current research in automated camera placement has focused on two main directions: (i) addressing specific user requirements; and/or (ii) developing an effective optimization strategy. In this article, we provide a comprehensive introduction to the problem and a concise review of recent advances in this field.

1.1. Scope and Contribution

Camera placement is an interdisciplinary study that can be linked to a large variety of fields. The earliest work can be traced back to the 1970s in the field of computational geometry where the famous art gallery problem was tackled [Chvatal 1975]. In sensor networks, the placement of sensors has been examined, especially within the emerging wireless sensor networks. Camera planning is also a topic in robotics where cameras are used as a means of acquiring information for robot navigation and control. Camera placement is also related to optimization since the problem is to find the optimal strategy to achieve certain functional requirements. Despite the large variety of related fields of study, in this article, our scope is limited to providing an introduction to the camera placement problem suitable for use in the surveillance industry. A special focus is given to how the various requirements and constraints are formulated as an offline optimization problem as well as how the optimization problems are solved by researchers in this field. To aid the understanding of the problem, we provide an overview to the camera placement problem, a review of the relevant background literature, and a coarse categorization of the relevant approaches.

The review provided in this article is capturing the most important work in the field and is focusing on recent advances (post 2000) where most of the development in this area has occurred. This article is targeting not only researchers already active in the field and those about to commence, but also engineers who have a specific problem at hand and would like to acquire knowledge on how the problem may be tackled. To this end, we give an in-depth review of relevant formulation strategies and brief introductions to most commonly used optimization techniques by researchers in this field. We hope our work to be inspirational to spark new ideas in the field.

1.2. Article Organization

The article is organized as follows. In Section 2, we provide the background and an overview of the camera placement problem in general. Section 3 and Section 4 describe the commonly used camera models and environment models. A number of techniques used to reduce the problem to a manageable level are also discussed in Section 4. Section 5 addresses the main focus of the article: the various formulations and design variables for camera placement that researchers have investigated. Then, the optimization methods used to solve the formulations are presented in Section 6. Finally, conclusions are drawn in Section 7.

2. BACKGROUND AND PROBLEM OVERVIEW

2.1. Computational Geometry

The earliest work was published by Chvatal [1975] and O'Rourke [1987] in the field of computational geometry. Chvatal formulated the Art Gallery Problem (AGP), which is concerned with the assignment of guards to different positions in an art gallery in order to achieve the maximum visual coverage of the valuable assets. In an AGP, the layout of the gallery is represented as a simple polygon, and a number of guards, represented as points, are placed inside the polygon such that all the points inside the polygon can be observed by at least one guard. In essence, this is equivalent to finding the best locations for a set of infinite-depth omnidirectional cameras on a 2D floorplan to achieve the optimal coverage of the observation area. Although the upper bound of the number of guards required to be placed at the vertices (vertex guards) has been shown to be $\lfloor n/3 \rfloor$ for a polygonal environment of n vertices [Chvatal 1975; Fisk 1978], the optimal solution has been proven to be NP-hard [Lee and Lin 1986]. Only recently, Couto et al. developed exact algorithms for the orthogonal AGP problem [Couto et al. 2008] and AGP with vertex guards [Couto et al. 2011].

Since its proposal, the AGP has received much attention from the computational geometry community mainly on the theoretical aspect of the work. However, the problems of adapting the classical AGP as well as many of its variants (such as the orthogonal AGP [Couto et al. 2008] or AGP with restricted guard visibility [Kazazakis and Argyros 2002]) to camera placement problems are evident: the environment is oversimplified and the guards are assumed to have unrealistic capabilities such as unlimited visibility and depth, which cannot be used as a replacement for visual cameras. In addition, the task of AGP is floor coverage, whereas most camera placement approaches consider many more objectives, such as those discussed in Section 5.

2.2. An Overview of Camera Placement

The widespread deployment of video surveillance camera networks in areas such as public spaces, transportation hubs, and critical infrastructure attracts greater interest from the research community. This has resulted in a shift from a problem of theoretical interest in issues such as the AGP to a problem where practical constraints are emphasized in order to maximize the utility and efficiency or to minimize the cost of camera networks with much more realistic assumptions.

Conventional camera placement approaches usually involve four steps, as shown in Figure 1. This offline process starts with users specifying a set of inputs, including the cameras to be used and the environment for which this placement strategy is designed. If the objective of the camera network includes tasks such as object tracking, then additional object space data must be supplied. Most importantly, users need to specify

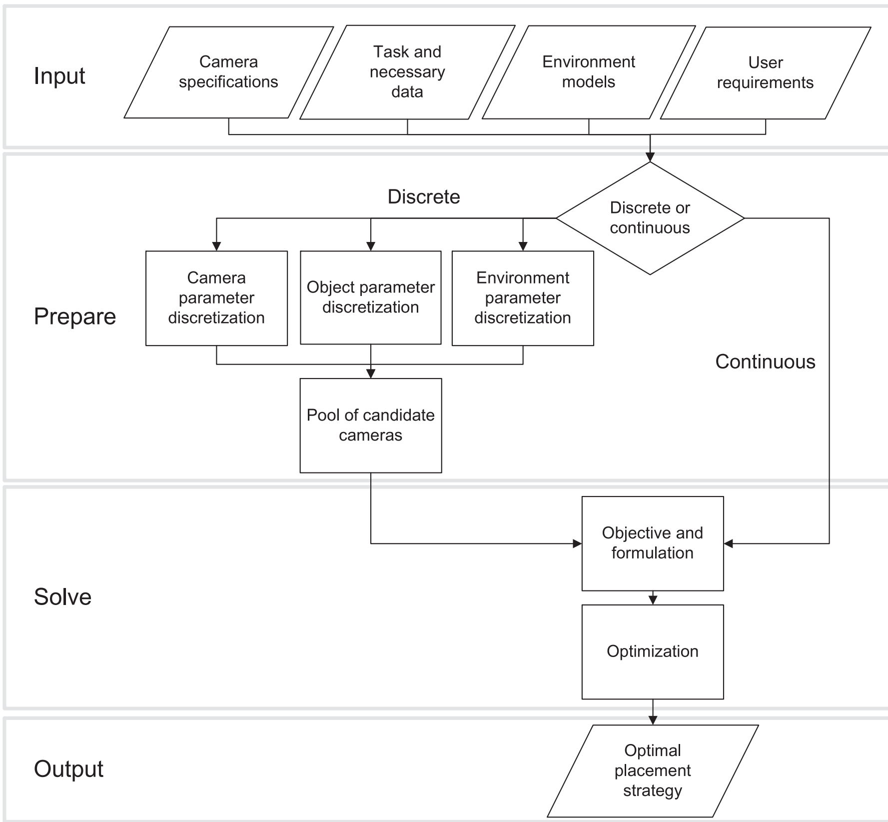

Fig. 1. Overview of camera placement approaches.

the requirements and constraints (Section 5), such as the percentage of the floor that needs to be covered or the minimal pixel resolution the system must achieve. If the problem is decided to be tackled in a discrete domain, an extra preparation step needs to be taken. The preparation step involves discretization of the input parameters and formation of a pool of candidate cameras (Section 4.3), from which the optimal subset is to be selected. The next step is the problem formulation step (Section 5) where all user objectives and requirements are expressed mathematically as an optimization problem, which is then solved by an appropriate optimization technique (Section 6) in the next stage.

Recent research on camera placement generally has progressed into two main directions. The first direction focuses on better modeling real-world camera properties, scene dynamics, user objectives, and system constraints, to form optimization problems that can be solved using standard optimization techniques, such as Branch and Bound [Land and Doig 1960]. The second direction of current camera placement research is the design of tailored optimization algorithms or adaptation methods that allows a formulated problem to be solved more effectively or efficiently. It is to be noted that almost all surveyed camera placement approaches use the formulation-optimization methodology as presented in Figure 1.

2.2.1. Versatile Formulations. Most camera placement research focuses on the formulation of the problem. This includes general problem formulations [Erdem and Sclaroff

2006; Horster and Lienhart 2009; Liu et al. 2014] and quality metrics of a camera network in performing various tasks such as persistent object tracking [Yao et al. 2008, 2010], task observability [Bodor et al. 2005, 2007], tag visibility [Zhao and Cheung 2007; Zhao et al. 2009], object localization [Ercan et al. 2006], occlusion metric [Chen and Davis 2008; Mittal and Davis 2004, 2008], and object recognition [Takahashi et al. 2006].

Erdem and Sclaroff [2004, 2006] proposed a general formulation of the problem. Given the set of all constraints required to achieve a specific task, the problem is to find the optimal placement for a set of cameras in the area satisfying the constraints and minimizing a given cost function such as the monetary cost of the cameras. Horster and Lienhart [2006, 2009] proposed a set of similar formulations. The difference is that instead of restricting the areas to be polygon shaped, Horster and Lienhart used points to represent space. These points may have different weights representing their importance, for example, points at entrances may have a higher importance than other areas. Both papers, as well as many others [Gonzalez-Barbosa et al. 2009; Mostafavi and Dehghan 2011; Zhao and Cheung 2009] treat camera placement as a combinatorial optimization over a binary solution space. Instead of finding the optimal parameters of each camera, a decision version of the problem is considered: given a set of candidate cameras computed by discretization of the parameter space of the camera and the environment, an optimal subset that satisfies the constraints is selected. In these formulations, the objective functions and constraints are all linear expressions of the binary decision variable, which indicates whether a candidate is chosen. This approach to formulating the problem complies with the format required for Binary Integer Programming (BIP), which can be solved using well studied algorithms such as Branch and Bound [Land and Doig 1960].

In contrast, a few authors formulate the problem in a continuous domain. Liu et al. [2012, 2014] proposed a general statistical formulation. All constraints and objectives of the camera network are summarized in a single objective function, which is then solved using a customized simulated annealing routine. Camera placement schemes designed for certain tasks do not admit linear objective or linear constraints. For example, Mittal and Davis [2004, 2008] probabilistically estimated dynamic occlusion, Bodor et al. [2005, 2007] considered aggregated motion observability, and Ercan et al. [2006] optimized multiview object localization accuracy. All of these objectives are formulated in a nonlinear form and solved using simulated annealing in Mittal and Davis [2004, 2008], hill climbing in Bodor et al. [2005, 2007], and semidefinite programming in Ercan et al. [2006].

Although maximizing coverage (area in 2D, volume in 3D) remains the most studied topic [Fleishman et al. 1999; Erdem and Sclaroff 2006; Horster and Lienhart 2009; Becker et al. 2009; Liu et al. 2012; Zhou and Long 2011; Amriki and Atrey 2011; Xu et al. 2010, 2011; Mohd Yusoff et al. 2011; Konda and Conci 2012, 2013a, 2014; Wang et al. 2013; Rudolph et al. 2014], there is a clear trend that other computer vision related tasks are becoming the center of attention. Yao et al. [2010] argued that only maximizing visibility is insufficient for persistent and automated tracking. The authors proposed to incorporate a hand-off success rate (the percentage of successful hand offs) analysis when determining camera placement. This approach attempts to preserve necessary uniform overlapped Field of Views (FoVs) between adjacent cameras for an optimal balance between coverage and hand-off success rate. Bodor et al. [2005, 2007] tried to optimize the camera network's ability to observe a set of predefined tasks, such as human motions. The authors defined their observation metric as a combination of the motion statistics and the resolution of the observed actions.

The observability of frontal faces has been incorporated in the process of finding the optimal camera placement by Ram et al. [2006]. The authors used the probability of observing an object of random orientation from multiple sensors as the quality metric for optimization. Similarly, Zhao et al. [2009] described a visibility model that can be used to evaluate the objective of maximizing the probability of tracking planar tags, such as a name tag or a face.

Takahashi et al. [2006] considered the task of planning a multicamera network for object recognition, using a quality metric that depends on the distance between manifolds. Similarly, [Chen 2009] and [Chen and Davis 2008] proposed a metric that incorporates camera resolution and occlusions. Ercan et al. [2006] considered object localization as their camera placement objective. Olague and Mohr [2002] minimized errors of accurate measurements of approximately known 3D targets. Denzler et al. [2002] developed a planning strategy to select the optimal intrinsic parameters for Kalman filter based object tracking.

2.2.2. Tailored Optimizations. Another stream of current camera placement research is the design of camera placement specific optimization algorithms or adaptation methods that allow formulated problems to be solved more effectively or efficiently. The camera placement problem has been largely considered as a discrete optimization through various quantization processes. This results in a NP-hard problem and is often tackled by using BIP methods as the first attempt [Mostafavi and Dehghan 2011; Gonzalez-Barbosa et al. 2009; Zhao et al. 2009; Horster and Lienhart 2009]. Despite the existence of freely or commercially available integer programming packages, obtaining an exact solution using BIP methods is generally impractical for any reasonable-size camera placement problems [Zhao 2011; Liu et al. 2012; Horster and Lienhart 2009]. For this reason, many have developed approximation techniques. Various greedy heuristics have been developed, such as the dual sampling algorithm [Horster and Lienhart 2009; Zhao et al. 2009]. Despite the variations in implementations, a greedy heuristic is a constructive heuristic that selects the optimal set of cameras in a sequential manner. At each iteration, the camera that provides the optimal gain is chosen. A novel adaptation to Semidefinite Programming (SDP) has been proposed by Zhao [2011] who used a "life and project" process to relax the original BIP problem to a SDP problem and showed promising results in terms of accuracy of the approximation and speed advantage.

Although powerful, these methods are not able to handle nonlinear objectives or constraints. In camera placement, nonlinearity typically arises when the task deviates from coverage optimization since the aggregated effect of a set of candidate cameras cannot be computed as a linear combination of the effect of individual candidates. For example, if the task is to find the optimal set of cameras to localize an object, a localization accuracy metric for a candidate set can only be determined through a nonlinear collaboration process involving all the candidates. In these circumstances, metaheuristic search algorithms are shown to be effective. In particular, Genetic Algorithms (GAs) have found their use in many camera placement approaches [David et al. 2007; Feng et al. 2011; van den Hengel et al. 2009; Indu et al. 2009; Kim et al. 2008; Murray et al. 2007; Olague and Mohr 2002; Yao et al. 2010]. A GA is a metaheuristic optimization algorithm that mimics the process of natural selection through using techniques inspired by natural evolution, such as crossover and mutation [Goldberg 1989]. Genetic algorithms have been proven to have advantages in dealing with nonlinear complex objectives coupled with large or poorly understood search spaces, where traditional optimization techniques such as hill climbing may fail. Many variants of GA have been used in this domain. For example, Indu et al. [2009] coded the parameters of a camera as a gene in a chromosome, which represents an array of cameras. A population of chromosomes are randomly chosen and evaluated against a fitness function. Well-performing chromosomes are chosen to reproduce the next population through genetic operations of crossover and mutations. The process iterates until stopping criteria are met. Olague and Mohr [2002] designed a multicellular GA represented by a tree structure. Kim et al. [2008] introduced a tailored GA capable of finding equally good alternative solutions to a camera placement where multiple objectives are desirable simultaneously.

A relatively new class of algorithms, namely, Swarm Intelligence (SI) has attracted a greater amount of attention in camera placement. As a subclass of SI, the Particle Swarm Optimization (PSO) has been widely used in this domain [Burelli et al. 2008; Conci and Lizzi 2009; Hsieh et al. 2011; Morsly et al. 2010, 2012; Konda and Conci 2012, 2013b; Xu et al. 2010, 2011, 2013; Zhou and Long 2011]. In a PSO algorithm, a feasible solution is termed a particle and a population of particles are updated in each iteration of the algorithm according to their velocity measures. The movement of a particle is subject to its local best-known locations and the best-known location of the entire population. This process will move the swarm of particles toward the optimal location. Variants to standard PSO exist. For example, Morsly et al. [2010, 2012] proposed a PSO-Inspired Probability algorithm to extend the standard binary PSO by updating the value of the velocity of a particle probabilistically according to a measure computed using the fitness of the best-known locations of the particle as well as the best-known location of the population. Instead of incorporating additional constraints into the fitness function, Xu et al. [2010, 2013] proposed three different PSOs, giving constraint specific treatments in the optimization stage.

Other optimization techniques have been adopted or modified for this problem. Liu et al. [2012, 2014] proposed a Trans-Dimensional Simulated Annealing (TDSA) which can deal with selecting the optimal number of cameras in a continuous domain. Chrysostomou et al. [2012] and Chrysostomou and Gasteratos [2012] designed variants of the Artificial Bee Colony (ABC) algorithms that simulates the foraging behavior of bees. Wang et al. [2013] proposed a Distributed Mean-Shift algorithm (DMSA).

2.3. Other Related Areas

Camera placement is a subfield of the more general field of sensor planning if a camera is considered as a sophisticated sensor. The sophistication comes from two areas: First, image formation is a complex process. A wider range of factors such as focal length and resolution must be considered, usually leading to a much more extended sensor model. Secondly, a camera provides an abundant amount of information which is to be subsequently processed to fully exploit the potential of the sensor. The quality of this processing, often the core design consideration of most optimal camera placement approaches, is sensitive to the way the sensors are originally placed. In the broader sensor placement context, the coverage of the network and the detection of events are the two main topics. When coverage of the network is considered, the problem is similar to camera placement for optimal coverage with the exception that occlusions, static or dynamic, which dramatically changes the sensing capability of a camera, is either treated in a completely different manner or not incorporated. When a sensor network is built for event detection, sensors are modeled probabilistically and metrics such as false alarm rates are often used. A review of sensor models can be found in Wang [2011]. As a related field, sensor placement in Wireless Sensor Networks (WSNs) has attracted a significant amount of research attention since WSNs are often severely constrained by not only the sensing range of the sensors but also the wireless channel fading, packet routing, energy conservation, and possible sensory failures. Younis and Akkaya [2008] provide a comprehensive review of sensor placement in WSNs.

Online vision sensor planning has been considered in the field of robotics and visual sensor networks. For example, Abrams et al. [1999] proposed an approach to continuously plan the orientations, locations, and optical settings of cameras in an environment containing objects moving in known ways (e.g., in a robotic work cell) such

Table I. Summary of Related Surveys

Paper	Focus
Roy et al. [2004]	Camera planning for 3D object recognition and scene analysis
Scott et al. [2003]	View planning for object reconstruction
Abidi et al. [2009]	Multimodel sensor integration
Mavrinac and Chen [2013]	Geometrical and topological coverage models
Tarabanis et al. [1995]	Camera planning in robotic inspection prior to 1995
Natarajan et al. [2015]	Optimal control of multiple active cameras
Our work	Camera placement as an offline optimization

that the cameras offer the best feature detectability. Chen and Li [2004] developed a strategy to determine the shortest sequence of known viewpoints for a single vision sensor to inspect an object with known geometric information. However, camera planning in robotics is mostly used for visual servoing, robotic navigation, or human-computer interaction. Compared to the camera placement for surveillance considered in this article, the main difference is that most vision sensor planning approaches often involve dynamic planning of sensor locations [Abrams et al. 1999], orientations [Munishwar and Abu-Ghazaleh 2010, 2011, 2013; Konda and Conci 2013a, 2014; Rudolph et al. 2014; Natarajan et al. 2014], and/or make use of sensory feedbacks [Nelson and Khosla 1994], and the quantity of cameras to be planned is relatively small. In contrast, most cameras used in surveillance carry out passive monitoring where sensory feedback is not used nor are the locations of the cameras frequently altered.

Camera selection has been considered as an important topic in large surveillance camera networks. The idea of camera selection is to select a subset of cameras from all cameras such that this subset is capable of accomplishing some required tasks, such as target tracking [Monari and Kroschel 2010], target localization [Liu et al. 2010], optimal observation of an object [Snidaro et al. 2003], or occlusion avoidance for improved multitarget tracking and pose estimation [Gupta et al. 2007b]. However, camera selection has a few distinct differences to camera placement problems considered in this article. First of all, camera selection is an online process where the selection process changes as the environment changes, for example, as the object of interest moves through the network. Camera placement, in contrast, is an offline process that incorporates long term scene dynamics in a probabilistic manner. Second, both camera selection and camera placement require certain metrics to be defined but camera selection rarely uses an extended optimization procedure as the result of the speed requirement. For example, heuristics are used in Gupta et al. [2007b], Monari and Kroschel [2009, 2010], Liu et al. [2010], and Soro and Heinzelman [2007]. Lastly, the vast majority of camera selection approaches select cameras from a limited pool of real cameras, whereas camera placement selects cameras from a theoretically infinite pool of candidates.

In terms of surveys in related areas, Roy et al. [2004] discussed camera planning approaches for 3D object recognition and scene analysis. Scott et al. [2003] surveyed view planning methods for object reconstruction and inspection. Abidi et al. [2009] discussed multimodal sensor integration, in which cameras are included as a modality. Recently, Mavrinac and Chen [2013] surveyed geometrical and topological coverage models for camera networks. Tarabanis et al. [1995] provided a summary of camera planning work in the context of robotic inspection prior to 1995. The work that is closest to ours is Natarajan et al. [2015], in which the authors surveyed recent approaches for the optimal control of multiple active cameras. The list of related surveys is presented in Table I. These works have a few distinctive differences to ours: (1) Our work mostly addresses the problem of determining the optimal placement of cameras, which often occurs prior to the deployment stage and it is usually an offline process that

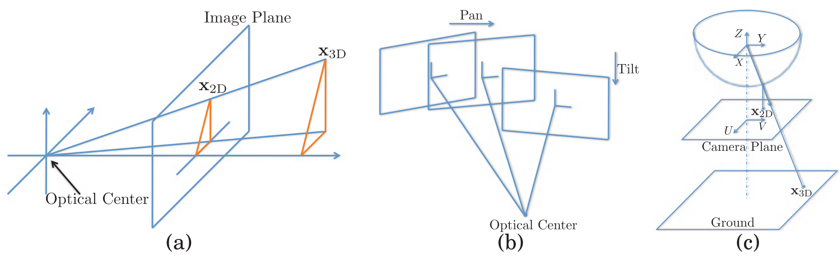

Fig. 2. Commonly used camera models in camera placement: (a) 3D static perspective camera; (b) pan and tilt motion of a perspective camera; (c) omnidirectional camera.

normally incurs significant computing resources. The problem discussed in Natarajan et al. [2015] is an online problem. (2) Our work concerns objectives in the surveillance area in general but not restrictive to 3D recognition or reconstruction. To our best knowledge, there is no recent (post-2000) review work concerning advances in optimal camera placement approaches for the purpose of wide area surveillance which we are addressing in this work.

3. CAMERA MODELS AND VISIBILITY

For the sake of completeness of this work and ease of understanding of subsequent sections, we briefly describe the camera models commonly used in camera placement approaches and discuss constraints applicable to a single camera. For comprehensive studies, readers are referred to Tarabanis et al. [1995] for discussions on camera parameters and effects, Mavrinac and Chen [2013] for camera network coverage models, and Sturm et al. [2011] for a survey of theoretical models used in computer vision research. The camera models considered in this work are similar to those used in Zhao et al. [2009], Horster and Lienhart [2009], Erdem and Sclaroff [2006], and Mittal and Davis [2008], which are illustrated in Figure 2. These models are applicable for cameras mounted on surfaces such as ceilings and walls in an area of interest to provide monitoring capability. Although mobile cameras have been considered previously such as in Abrams et al. [1999], in this work only stationary cameras are considered.

3.1. Perspective Camera

A pinhole camera model (Figure 2(a)) is a simplified camera model often used to model the perspective effect of camera projections. The model can be expressed as $\mathbf{x} = \mathbf{PX}$, where $\mathbf{X} = [x, y, z, 1]^\top$ is a point in the world coordinate. $\mathbf{x} = [x, y, w]^\top$ is the corresponding point in the image coordinate, and $\mathbf{P}$ is a projection matrix that maps the world coordinates to image coordinates. The matrix $\mathbf{P}$ may be decomposed as $\mathbf{P} = \mathbf{K}[\mathbf{R} \mid -\mathbf{Rt}]$, where $\mathbf{R}$ is a rotation matrix representing the camera orientation $(\theta_x, \theta_y, \theta_z)$ and $\mathbf{t}$ is a translation vector representing the camera center (camera location) in the world coordinate frame. The matrix $\mathbf{K}$ is referred to as the camera intrinsic parameter matrix and consists of the camera focal length, aspect ratio, principal point, and skew.

3.1.1. Camera FoV. The FoV of a camera is a volume within which objects can be projected onto the image plane of the camera. For a static perspective camera with no lens distortion, the FoV region is essentially a rectangular pyramid whose apex is located at the camera center. For these cameras, FoV is usually characterized by a horizontal FoV angle and a vertical FoV angle. Furthermore, as noted in Abrams et al. [1999], camera planning systems generally consider cameras whose image planes are symmetrical about the optical axis, and thus removing the need to optimize the third

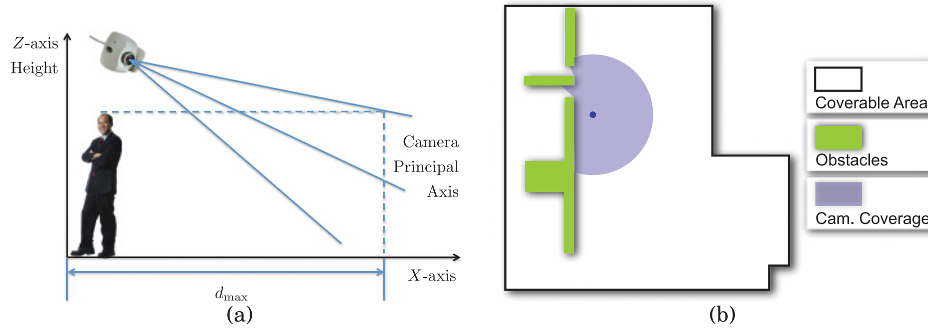

Fig. 3. (a) Exemplar definition of maximum viewing distance, which is constrained by the tilting angle of the camera and the average height of the object of interest. (b) Static occlusion. It can be seen that the disk-shaped coverage region of the camera (blue dot) is partially blocked by the obstacles in the environment.

orientation (roll about the optical axis) of the camera. The horizontal FoV angle can be computed as $\alpha_{\text{FOV}} = 2\tan^{-1}(\frac{s_x}{2f})$, where s_x is the width of the image and f is the focal length. Similarly, replacing s_x with s_y, the image height, one can compute the vertical FoV.

3.1.2. Depth of Field, Pixel Resolution, and View Limits. For a perspective camera, there are a number of factors limiting the range within which objects are considered visible. The distance at which an object is visible is referred to as viewing distance in this article. The factors influencing the maximum viewing distance are depth of field, pixel resolution, and most importantly, the application scenario. A real camera containing a thick-lens imaging system with a finite aperture can only focus sharply at a particular distance. In practice, if a point images to a blurred circle smaller than a specified size, it is considered in focus. This leads to a definition of minimum and maximum distances considered visible by the camera. The depth of field can be tuned by changing the focal length in a way that it extends from a minimum value to infinity. In this case, the pixel resolution may be the limiting factor of visibility of the camera. Pixel resolution refers to the ability of the lens system to distinguish features at a distance. As an object moves away from a static camera, the object will eventually appear to be a single pixel on the captured image (based on the assumption that a CMOS type imaging sensor is used). Whether an object is resolvable depends predominantly on four factors: the object's shape, size, orientation, and distance to the cameras. For instance, one possible way to define pixel resolution is provided in Tarabanis et al. [1994]: For a line segment located on the plane at the depth of field distance and parallel to the image plane of the camera, it will be considered resolvable if the length of its image is greater than or equal to the largest distance between pixels (the diagonal for square pixel cameras). With given or assumed knowledge of the smallest object in the scene to be monitored, pixel resolution effectively places an upper bound on the viewing distance of a camera.

Despite this knowledge of resolution and its relation to viewing distance, the application requirements of the camera system are often much more restrictive in determining maximum viewing distances. For example, in order to guarantee a certain success rate of face recognition [Fookes et al. 2012], the minimum size required is w_i pixels in width for a normal human face of w_r meter in width; then the maximum viewing distance of a perspective camera with no distortion and a focal length of f pixels can be approximated by $d_{\max} = (f w_i)/w_r$, when a direct frontal view is assumed. As another example, Figure 3(a) shows one possible definition of the maximum viewing distance adopted from Erdem and Sclaroff [2006]. In this case, it is related to the tilting angle of the camera and the camera's vertical FoV.

3.1.3. Perspective Distortion. Sometimes the perspective effect of projection must be considered, as in many applications a large perspective distortion may be undesirable. For example, a ceiling-mounted camera pointing toward the ground may be good for floor coverage but performs poorly for subsequent face detection algorithms.

3.1.4. Occlusion. Three types of occlusion may occur in a scene for any camera-related applications: static/environmental occlusion, self-occlusion, and dynamic occlusion. Static occlusion is the occlusion caused by fixtures in the environment such as walls and furniture. With a precise model of the environment, the visible area/volume of a camera with respect to the static-occlusion constraint can be computed using algorithms such as Radial Sweep [Erdem and Sclaroff 2006] in 2D or Z-buffering in 3D [Greene et al. 1993]. An example of 2D static occlusion is shown in Figure 3(b).

Self-occlusion is applicable when a specific side of an object of interest is required. For example, in some surveillance systems, it is required that the frontal view must be available (e.g., frontal gait recognition [Sivapalan et al. 2011]). Another example is in casinos where dealers hands must be clearly visible. In these cases, the frontal faces or the hands may be occluded by the persons of interest themselves. Therefore, it is necessary, with the preacquired knowledge of the orientation of the objects, that the cameras are planned in a way that the object is the least likely to cause self-occlusion.

Dynamic occlusion, also called interobject or mutual occlusion, can only occur when there is more than one object of interest in an environment and that one object is occluded by another in the view of a camera. Dynamic occlusions are much harder to be accounted for simply because of the free dynamics of the objects, which are not deterministic at all and therefore can only be approximated by probability distributions in a real environment. One way to tackle this is to compute the worst-case mutual occlusion angle. This will be described in more detail in Section 5.

3.1.5. Other Factors. There are many other sensor parameters that affect the quality of captured images. Typically these include aperture size, sensor contrast, aperture speed, and shutter mechanism. These parameters may be explicitly considered in formulating solutions to the camera placement problem.

3.2. Pan Tilt Zoom (PTZ) and Omnidirectional Cameras

PTZ cameras [Liles 2005] are commonly used to extend the FoV of static perspective cameras. As such, they undergo panning and tilting motions only when required or as programmed. An image of panning and tilting motion is depicted in Figure 2(b). The zooming option is mostly used when the operator observes events of interest and would like to zoom in for greater details. Therefore, each PTZ camera is often treated as an aggregation of all static perspective cameras at different pan and tilt combinations.

Omnidirectional cameras [Scaramuzza 2014] are often used for wide area surveillance due to their extremely wide FoV. There are mainly two types of omnidirectional cameras: dioptric and catadioptric. Dioptric cameras are created with a fisheye lens and catadioptric cameras are built from a pinhole camera with a parabolic mirror. The images captured by omnidirectional cameras usually lack detail due to the large FoV. As a result, in practice, they are mostly used for tasks such as floor coverage, and their viewing regions are usually approximated as circular disks on the ground defined by a maximum and minimum radius.

4. ENVIRONMENT MODEL AND PROBLEM SIMPLIFICATIONS

4.1. Environment Model

For most camera placement approaches, a comprehensive environment model is necessary to provide the greatest amount of useful information, especially for computing static occlusions from fixtures in the environment. Therefore, a good 3D scene model for a region where supervision is to be carried out must contain elements describing the physical layout of the region coupled with the objectives such as a description of the exact area that needs to be covered. The shape of the region as well as the shapes and positions of fixtures such as separators, pillars, and furniture must be described. In addition, if the camera network is to be used for tasks such as tracking, the disturbances of physical flow must be included (i.e., the object space data). The computed camera placement strategy is only optimal to the current setup of the environment including all the furniture layouts.

4.2. Problem Simplifications

Solving the camera placement problem may prove to be challenging if the most general case is considered. In order to deal with a more manageable problem size, a number of recent approaches [Liu et al. 2012; Horster and Lienhart 2009] consider simplification and discretization techniques, which often involve one or more of the following processes: (1) use of 2D models combined with polygonal representations to approximate the environment and coverage regions of cameras, as efficient algorithms exist in the field of computational geometry, and (2) discretization of camera (e.g., position and orientation), environment (e.g., floor coverage), and object space (e.g., object tracking data) parameters to restrict the optimal selection to a confined subset.

4.2.1. 2D Models. A number of authors consider a simplification of the general 3D environment model: the 2D floorplan of the environment of interest. Despite being vastly simpler than the full 3D environment, a major drawback that cannot be avoided with 2D floorplans is the lack of ability to model obstacles not at full height which might occlude cameras' vertical FoV angles partially. For example, office partitions cannot be included in the floorplan as they do not block camera views completely. However, utilizing a full 3D model will incur significantly more computational cost due to a much larger search space. Interesting work that categorizes the entire region into different types of zones according to various properties that can influence camera placement design can be found in David et al. [2007].

The coverage area of a perspective camera in 2D is often approximated by an isosceles triangle [Horster and Lienhart 2009; Gonzalez-Barbosa et al. 2009]. The angle subtended by the two equal sides is the horizontal FoV and the height corresponding to the third edge is the maximum viewing distance of the camera. The maximum viewing distance, as discussed in Section 3, is the maximum distance that some sort of resolution can be guaranteed.

Omnidirectional cameras possess significant distortion but are capable of capturing full 360° around them. In 2D, they are often approximated by disks with a circular opening in the center. The model is constrained by two parameters, the maximum and minimum viewing distance. The maximum viewing distance is the radius of the disk and it can be determined in a similar fashion to that of a perspective camera. The minimum distance is the radius of the central opening that represents the region unable to be covered by the camera. The radius of this opening is usually a physical limitation. The coverage model used for PTZ cameras is often a sector of a disk determined as the union of the coverage of a set of 2D perspective cameras at a continuous range of orientations. Depending on the model of the servos used, certain types of PTZ cameras can also achieve full 360° coverage.

4.3. Quantization of Camera, Environment, and Object Spaces

Regardless of the choice of a 3D or 2D camera model, when the placement problem is treated as a discrete optimization, the continuous camera parameters including camera positions and poses are often quantized to further reduce the complexity of the problem. The camera positions are restricted to a set of specific points in the environment normally with constant intervals. The poses of the cameras are also discretized such that a very small subset of angles are allowed. Even with very coarse discretization, the problem may still not be feasible if a true optimum is to be sought.

Similarly, the environment can also be discretized. While a number of authors use simple representation of equally sized blocks or uniform points [Murray et al. 2007; David et al. 2007], other discretization methods have also been considered. Horster and Lienhart [2009] also used weighted sampling to assign different weights to different areas according to their importance. Yabuta and Kitazawa [2008] used rectangles of varying sizes to mimic different resolutions according to the importance. A smaller rectangle represents an area of greater importance.

When the task involves the use of object space (e.g., object tracking data needed in optimizing for tag/face detection and tracking hand off) the object space *may* be discretized as well [Zhao et al. 2009]. The object space may be discretized in the same way as environmental space. Alternative methods including random sampling, stratified sampling, and systematic sampling can also be employed [Zhao et al. 2009].

Discretization of the environment space and object space allows different cameras to be brought to a common ground for comparison. Furthermore, although the optical characteristics of the cameras may be vastly different, each of them generates a "visibility" vector indicating their performance in a particular task. For example, omnidirectional cameras and perspective cameras can both be assigned vectors of the same length that indicate the block coverage of a given floorplan.

5. DESIGN VARIABLES AND FORMULATIONS

5.1. Overview

Camera placement by definition is an offline optimization problem often considered in one of the two classical approaches: maximization of the utility (Max-Utility) while keeping a constant cost of the system or minimization of the system cost (Min-Cost) while maintaining a satisfactory utility. The definition of utility varies according to the objective of the camera network to be planned. Its definition ranges from simple coverage percentage to the aggregated observability of a target profile [Bodor et al. 2007; Mostafavi and Dehghan 2011]. It can also be a combination of coverage, foreshortening effect, and resolution [Fehr et al. 2009]. The cost function is usually the number of cameras [Liu et al. 2012] but it can represent monetary value associated with the particular configuration or the cabling cost as well.

Max-Utility and Min-Cost are the most basic formulations upon which almost all of the variety of formulations are built. The variants usually extend the basic formulations by either designing utility functions to better reflect the fitness of a camera network for a specific task or adding additional constraints to better model the physical restrictions of the environment or user requests. In this article, Any variable that has to be determined before commencing the design of a camera placement approach is referred to as a design variable. Design variables include user objectives, constraints, types of cameras available, etc.

5.1.1. Notations. Before introducing the formulations, the notations that will be used are outlined. Unless otherwise stated, a lowercase bold letter represents a vector; an uppercase bold letter represents a matrix. Functions are usually represented using

Table II. Common Notations used Throughout this Article

Notation	Meaning	Notation	Meaning
C, c_i	A network of cameras, and the ith camera in the set.	e_i	Cost associated with camera i.
$\mathbf{x}$	The binary decision variable indicating the optimal subset of cameras in a pool of candidates.	N_{cand}	Number of cameras in a pool of candidates.
N_{region}	Number of discretized regions in an environment.	E	The environment of interest.
Q	Quality metric. Multiple definitions exist.	η	Visibility measure of a camera with respect to a target. Multiple definitions exist.
$\mathbf{p}$	Location of a target.	$\mathbf{q}$	Location of a camera.
(s_x, s_y)	Width and height of the image captured by a camera.		

a single uppercase letter but there are circumstances where they are represented using their names, for example, UniRandom($\bullet$) intuitively represents a function that generates a uniformly distributed random number. A number of common notations used throughout this article are listed in Table II.

5.1.2. Number of Cameras, Continuous and Discrete Formulations. Regardless of the problem at hand or the camera model to be used, there is always a discrete variable unavoidable in all camera placement formulations: the number of cameras. The treatment of the problem can be dramatically different for the case when the number of cameras is known or when it is a variable that has to be determined during optimization.

In the case when the number of cameras is not known, camera placement problems usually adopt a discrete formulation: given the set of discretized parameters (as described in Section 4.2), the pool of candidate cameras is first computed and then the subset that optimizes the objective and meets the constraints is selected. The search space in a discrete formulation is $\mathbb{B}^{N_{\text{cand}}}$, where $\mathbb{B} = \{0, 1\}$ and N_{cand} is the total number of candidate cameras.

In contrast, when the number of cameras is known a priori, the problem can be formulated in either the discrete domain in the same fashion as the case of unknown number of cameras, or in a continuous domain. In a continuous formulation, a feasible solution is a vector of known dimension that contains the camera parameters. The optimization is to find the parameter vector that optimizes the objective while satisfying the constraints. The search space of the continuous domain is $\mathbb{C}^{N_{\text{camera}}}$ where $\mathbb{C}$ is the joint parameter space for each camera and N_{camera} is the number of cameras. For example, if each camera is modeled by the vector representing its operation $c = [x, y, z, f]^\top$, where (x, y, z) is the location of the camera and f is the focal length, then the joint parameter space is $\mathbb{C} = \mathbb{R}^4$.

5.2. Coverage

Maximal coverage of the environment of interest has been perceived by both system developers and researchers as the most important task in the area of surveillance. In a Min-Cost formulation, it is the problem that seeks the camera placement configuration with the smallest cost on the condition that all user required regions are covered. A large body of work focuses on this problem [Angella et al. 2007; David et al. 2007; Murray et al. 2007; Yabuta and Kitazawa 2008; Conci and Lizzi 2009; Fehr et al. 2009].

The problem is represented as a Set Coverage Problem [Slavik 1997],

$$\begin{aligned} &\min \quad \sum_{i=1}^{N_{\text{cand}}} e_i x_i \\ &\text{s.t.} \quad \mathbf{Ax} \geq \mathbf{b} \quad \text{and} \quad x_i \in \{0, 1\}, \end{aligned} \tag{1}$$

where e_i is the cost associated with the ith candidate camera and x_i is a binary variable that indicates whether the ith camera is selected from the N_{cand} candidates. $\mathbf{A}$ is a $N_{\text{region}} \times N_{\text{cand}}$ matrix where N_{region} is the total number of discretized regions. The ith column of $\mathbf{A}$ represents the coverage of all the regions by the ith camera. $\mathbf{b}$ is an N_{region} dimensional binary vector indicating the regions that need to be covered and this is to be specified by the end user.

5.3. Critical Region Coverage (Backup Coverage)

In addition to maximal coverage, some important regions such as doorways, entrances, and exits may require extra monitoring. For the case when strict coverage is required, this can be easily incorporated in the formulation in Equation (1) by allowing the vector b to take values greater than 1, indicating the number of cameras required for covering each region.

However, it is often the case where the user requirement does not have to be strictly achieved [Yabuta and Kitazawa 2008]. For a tight budget surveillance system, the cameras should at least cover all of the the critical regions and cover as many noncritical regions as possible. It is only worthwhile if the addition of a new camera can increase the number of observed regions by at least K. In this case, the formulation can be written as [Yabuta and Kitazawa 2008]

$$\begin{aligned} &\min \quad \sum_{i=1}^{N_{\text{cand}}} e_i x_i - w \sum_{j=1}^{N_{\text{region}}} (1 - s_j) t_j \\ &\text{s.t.} \quad \mathbf{Ax} \geq \mathbf{b} \quad \text{and} \quad x_i \in \{0, 1\}, \end{aligned} \tag{2}$$

where

$$s_j = \begin{cases} 1 & \text{if region } j \text{ is critical,} \\ 0 & \text{otherwise,} \end{cases} \quad \text{and} \quad t_j = \begin{cases} 1 & \begin{array}{c}\text{if region } j \text{ is not critical} \\ \text{and region } j \text{ is covered,}\end{array} \\ 0 & \text{otherwise.} \end{cases}$$

Here the objective is to achieve a balance between finding the minimum cost of the cameras and the maximum number of observed noncritical regions. The inclusion of a camera takes place when the number of observed regions increases by K and this behavior is controlled by the constant w. Therefore, w should satisfy $1 - wK < 0$ and $1 - w(K - 1) > 0$.

Contrary to Yabuta's work, Fehr et al. [2009] argue that the coverage of the critical region is optimal only when the projections of the camera frustums exactly cover the critical regions (represented as convex hulls derived from motion data in Fehr et al. [2009]) to provide sufficient coverage as well as maximum amount of target details. A quality metric is therefore defined as a Gaussian-like function,

$$Q = \exp\left\{-\beta \left(\frac{\text{Max}\,(F - A, A - F)}{F}\right)^{\alpha}\right\}, \tag{3}$$

where α and β are control variables. As can be seen, the function behaves like a normal function with the maximum only occurring when the coverage regions F of the cameras match the critical regions A. The quality metric can be incorporated as a constraint in the formulation presented in Equation (1).

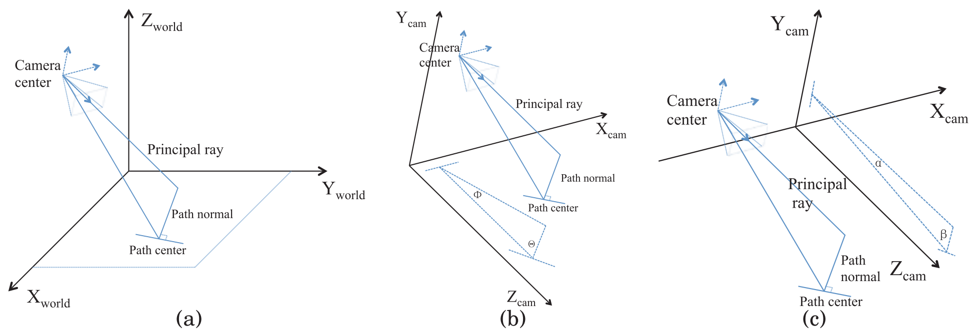

Fig. 4. Relative orientation between a camera and a path: (a) shows the relative orientation of a path and a camera in the world coordinate frame; (b) shows the projection of the camera-path relationship on the camera's x-z plane; and (c) shows the projection of the camera-path relationship on the camera's y-z plane.

5.4. Task and Path Observability

Instead of considering the optimal layout for observing the critical regions, one may design a system to directly compute the optimal task observability [Bodor et al. 2005, 2007]. The goal is to optimize the coverage of the cameras relative to the expected path distributions of the object of interest. The object of interest can be of various types including vehicles, pedestrians, body parts, and objects. Given a set of object trajectories $\mathbf{s}_j \in \mathbf{S}$ as input, the quality of a set of cameras can be defined as

$$Q(c_1, ..., c_n) = \sum_{j=1}^{M} G(\mathbf{s}_j, c_1, \ldots, c_n), \tag{4}$$

where $G(\mathbf{s}, c_1, \ldots, c_n)$ is a gain function of the trajectory states $\mathbf{s}_j \in \mathbf{S}$ and c_i is the ith candidate camera. $\mathbf{S}$ represents the set of states of the system which is the set of possible trajectories in this case. M is total number of random sample trajectories drawn from the prior probability distribution on the trajectory states, which is assumed to be known. Each trajectory $\mathbf{s}_j$ consists of a set of linear paths which are defined by an orientation (θ_j), path center, and path length. Trajectories with large curvatures can be treated as a number of individual approximately linear paths with low residual errors.

The G function of a single camera c_i and a single path $\mathbf{s}_j$ can be expressed as a combination of three metrics: the camera-path distance, approximated pixel resolution, and strength of foreshortening effect. The camera must maintain a minimum distance d_0 from each path to ensure the successful capture of motions. While satisfying the minimum distance requirement, it should be as close to the path as possible to maximize the resolution of the path on the captured image. Lastly, the camera should be oriented in a way that there is minimal foreshortening effect. The three metrics are combined as

$$G_{ij} = \begin{cases} 0 & \text{if } d_{ij}^2 < d_0^2, \\ \frac{d_0^2}{d_{ij}^2} \cos(\theta_{ij}) \cos(\phi_{ij}) \cos(\alpha_{ij}) \cos(\beta_{ij}) & \text{otherwise}, \end{cases} \tag{5}$$

where d_{ij} is the camera-path distance and $\alpha, \beta, \theta, \phi$ are the angles that define the relative orientation between a camera-path pair as shown in Figure 4. Based on this definition of G_{ij} of the gain of a camera with respect to a path, the path observability of a camera network in Equation (4) can be expanded as

$$Q = \tau\left(\sum_{j=1}^{M} G_{1j}, \ldots, \sum_{j=1}^{M} G_{nj}\right), \tag{6}$$

where the function τ is defined as

$$\begin{aligned}
&\tau\left(x_1, x_2, \ldots, x_n\right) = && (7)\\
&x_1 + x_2 +, \ldots, x_n \\
&-x_1x_2 - x_1x_3 -, \ldots, -x_{n-1}x_n \\
&+x_1x_2x_3 + x_1x_2x_n +, \ldots, +x_{n-2}x_{n-1}x_n \\
&\ldots \\
&(-1)^{n+1}x_1x_2 \ldots x_n. && (8)
\end{aligned}$$

Compared to the more intuitive approach of simply setting regions where paths exist as critical regions and computing the placement strategy as described in Equation (2), Equation (6) is much more flexible as it allows the dynamic prioritization of trajectories with a higher likelihood of being observed.

5.5. Target Visibility

More direct approaches of defining target observability can be found in Zhao et al. [2009], where Zhao et al. proposed a metric of measuring the observability of frontal tags which can be faces or name tags. In the approach, the target is assumed to be a small flat surface perpendicular to the ground and all the targets are of the same square shape with known edge length l. The targets are further assumed to be at the same height with centers lying on a single plane z parallel to the ground. The line of intersection between each square tag and the plane z is referred to as center line, which also has the same length l as a tag edge. The visibility η_j of a tag in the view of a camera c_i is defined using the projected size of the target on the image plane of the camera. Due to different camera parameters, the image of the target will appear to be differently shaped quadrilaterals in different cameras. Instead of using the quadrilateral, Zhao et al. used the projection of the center line as a simplified representation. Given this setup, the visibility of a target j can be parameterized as $V(\mathbf{p}_j, \mathbf{v}_j, \beta_j | l, c_i, E)$, where $\mathbf{p}_j$ is the location of the target. $\mathbf{v}_j$ is the pose vector of the target which lies on plane z. β_j is a fixed worst-case occlusion angle, which is measured at the center of the target on plane z. If the projection of the line of sight between camera c_i and another target center falls within β_j, occlusion is said to have occurred. E encapsulates the environment. c_i, l, and E are known parameters for a particular problem. If the projected length of the center line is greater than a predefined threshold, then the target is visible to the camera,

$$\eta_{i,j} = \begin{cases} 1 & \text{if } V(\mathbf{p}_j, \mathbf{v}_j, \beta_j | l, c_i, E) > T, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

The optimization can be defined as

$$\begin{aligned}
\min \quad & \sum_{i=1}^{N_{\text{cand}}} e_i x_i \\
\text{s.t.} \quad & \sum_{i=1}^{N_{\text{cand}}} x_i \eta_{i,j} \geq 1 \quad \forall j \quad \text{and} \quad x_i \in \{0, 1\}.
\end{aligned} \tag{10}$$

In the optimization defined in Equation (10), the constraint ensures that for all targets (indexed by j), there has to be at least one camera that can observe it. Similar computation for the target visibility has been described by Liu et al. [2012, 2014]. The difference between Liu et al.'s approach and Equation (9) is that Liu et al. did not consider mutual occlusion since the authors assumed that cameras are mounted on the ceiling so that mutual occlusions do not occur frequently. However, Liu et al. considered thresholding the area of the projected quadrilateral (resulting from assuming a rectangle-shaped object) to determine whether an object is marked as visible. Compared to thresholding the center line in Zhao et al. [2009], quadrilaterals can correctly model the effect of increasing titling angles of the cameras. In these cases, the center lines are constant but the areas of the quadrilaterals decrease to correctly indicate the growing difficulty in identifying the target due to the perspective effect.

5.6. Localization Error

Ercan et al. [2006] studies the problem of optimal camera placement for single target localization in a wireless vision sensor network context. The objective is to compute the optimal camera placement strategy that minimizes the localization error of a target. The problem is considered in a 2D environment in which the cameras are assumed to be mounted at the same height as the target and point to horizontal directions. In order to reduce communication and on-board computation, the cameras are treated as weak perspective cameras, outputting 2D locations of the target as a series of one-dimensional scan lines. The relationship between a 2D object location $\mathbf{p}$ and the measurement reported by N cameras $\mathbf{y} = [y_1, y_2, \dots, y_N]^\top$ can be described as

$$\mathbf{y} = \mathbf{K}\mathbf{p} + \boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon}$ is a measurement error and $\mathbf{K}$ is given by

$$\mathbf{K} = \left\{ \begin{array}{cc} -\sin\theta_1 & \cos\theta_1 \\ -\sin\theta_2 & \cos\theta_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ -\sin\theta_N & \cos\theta_N \end{array} \right\}.$$

Since both environment and cameras are essentially 2D models, only one variable θ_i is used to describe the orientation of each camera. The error of the estimated target location can be expressed in terms of the mean square error as

$$Q = \frac{4\left(\frac{\alpha+1}{\sigma_o^2} + \sum_{i=1}^{N}\frac{1}{\sigma_{v_i}^2}\right)}{\left(\frac{\alpha+1}{\sigma_o^2} + \sum_{i=1}^{N}\frac{1}{\sigma_{v_i}^2}\right)^2 - \left(\frac{\alpha-1}{\sigma_o^2} + \sum_{i=1}^{N}\frac{\cos 2\theta_i}{\sigma_{v_i}^2}\right)^2 - \left(\sum_{i=1}^{N}\frac{\sin 2\theta_i}{\sigma_{v_i}^2}\right)^2}, \quad (11)$$

where σ_{v_i} are elements along the diagonal of the measurement noise, which is assumed to be independent for different cameras, $\Sigma_v = \mathrm{diag}(\sigma_{v_1}^2, \dots, \sigma_{v_n}^2)$. σ_o and α are elements in the covariance matrix for the object prior, $\Sigma_o = \sigma_o^2\mathrm{diag}(1, 1/\alpha)$. Under weak perspective projection, the distance from a camera to the object in a 2D scenario does not affect the measured position on the scan line, but instead it affects the measurement variance according to $\sigma_v^2 = \theta_e d^2 + \sigma_f^2$, where d is the camera object distance, σ_f is the error in 2D camera location, and θ_e is the camera orientation error.

The camera placement problem can be formally defined as

$$\max \quad \left(\frac{\alpha+1}{\sigma_o^2}+\sum_{i=1}^{N}\frac{x_i}{\sigma_{v_i}^2}\right)^2-\left(\frac{\alpha-1}{\sigma_o^2}+\sum_{i=1}^{N}\frac{x_i\cos 2\theta_i}{\sigma_{v_i}^2}\right)^2-\left(\sum_{i=1}^{N}\frac{x_i\sin 2\theta_i}{\sigma_{v_i}^2}\right)^2$$

$$\text{s.t.} \quad \sum_{i=1}^{N} x_i = N_{\text{camera}} \quad \text{and} \quad x_i \in \{0,1\},$$

where N_{camera} is the number of cameras the network is required to have, which is a known parameter. The objective function, as can be observed, ignores the numerator part of the Q function. This approximation is made due to mainly two reasons. First, the denominator is essentially the mutual information between the measurements and the object location, which is naturally a good metric for optimization [Ertin et al. 2003; Wang et al. 2004]. Secondly, the denominator involves only square terms of the unknown, which facilitates an optimization routine of Semidefined Programming (discussed in Section 6.2). Alternatively, Zhao [2011] linearized the quality function, allowing the problem to be solved by Binary Integer Programming.

5.7. Tracking Hand Off

Yao et al.'s camera placement approach [Yao et al. 2008, 2010] is concerned with optimizing the camera parameters such that a sufficient amount of overlap between neighboring cameras is preserved to ensure successful target hand over among cameras. This is an essential requirement for persistent and automated tracking in real-time surveillance systems. The necessary overlapping FoV is termed hand-off safety margin and it is subject to the target-camera distance. When the target is away from the camera, the same motion will result in a smaller displacement on the image plane than if the target was close to the camera.

Given the objective, the concept of visibility has been revisited, taking into account the metric for measuring resolution M_r as well as margin distances M_d. Traditionally, with the employment of a static perspective camera as described in Section 3.1, the camera-target distance of a target on the ground $(p_x, p_y, 0)$ can be easily computed as

$$d_t = \frac{-q_z}{p_x/f\cos\theta_y + \sin\theta_y},$$

where q_z is the z component of the camera location $\mathbf{q} = (q_x, q_y, q_z)$, θ_y is the tilting angle of the camera, and f is the focal length. Given d_t, a resolution metric can be defined as

$$M_r = \begin{cases} \alpha_r/d_t & d_t > |q_z/\tan\theta_y|, \\ \frac{\alpha_r}{(d_t+q_z/\tan\theta_y)^2 - q_z/\tan\theta_y} & d_t \le |q_z/\tan\theta_y|, \end{cases} \tag{12}$$

where α_r is a controlling coefficient. The metric for measuring the margin distances has been proposed to be

$$M_d = \left\{\alpha_d\left[(1-2|p'_x|/s_x)^2 + (1-2|p'_y|/s_y)^2\right]\right\}^{\beta_1 d_t+\beta_0}, \tag{13}$$

where s_x and s_y are the image width and height, (p'_x, p'_y) is the projected point of a 3D point $\mathbf{p}$ on the image, and α_d is a normalization coefficient. β_1 and β_0 are coefficients controlling the power of the polynomial in order to mimic the effect that the movement of an object will result in greater displacement on the image captured when it is closer to the camera. The observability of a static perspective camera is a linear combination

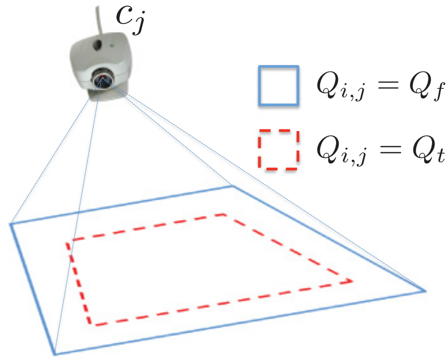

Fig. 5. Hand-off safety margin for jth camera and i is the index of discretized grids. Areas outside of the solid lines are invisible areas in which the Q metric of the camera does not pass the failure threshold ($Q_{i,j} < Q_f$). On the other hand, the area enclosed by the red dashed line is the visible region where target takes place on a single camera ($Q_{i,j} > Q_t$). The area in between the two lines is the hand-off safety margin ($Q_t > Q_{i,j} > Q_f$) where the target is still visible but hand off to a different camera must be initiated to enable persistent multicamera tracking.

of M_r and M_d,

$$Q = \begin{cases} w_r M_r + w_d M_d & \text{within the FoV,} \\ -\infty & \text{otherwise,} \end{cases} \tag{14}$$

where w_r and w_d are weights that balance the trade-off between M_r and M_d. Having a higher Q metric indicates better performance in terms of both resolution and distance to the edges. However, in practice, once the resolution and distance requirement are met, any further increase does not provide extra benefits and the extra resources are better reallocated to other requirements such as coverage. Along with this line of reasoning, a failure threshold Q_f and a trigger threshold Q_t are defined, forming a hand-off safety margin as shown in Figure 5. In order to make use of the hand-off safety margin, the traditional Min-Cost formulation is modified to take in the extra complication introduced by having three types of regions. Let $\mathbf{A}_c$ denote the conventional binary coverage matrix where each element indicates whether the corresponding grid can be covered by the camera (i.e., all grids where $Q_{i,j} > Q_f$). Two extra matrices are constructed: Let $\mathbf{A}_h$ denote the binary matrix indicating the coverage of the hand-off safety margin where $Q_t > Q_{i,j} > Q_f$ and let $\mathbf{A}_v$ be the binary matrix for the coverage of the true visible regions, that is, all grids where $Q_{i,j} > Q_t$. Given a solution vector $\mathbf{x}$, the vectors $\mathbf{A}_c\mathbf{x}$, $\mathbf{A}_h\mathbf{x}$, and $\mathbf{A}_v\mathbf{x}$ represent the grids that are covered, in the hand-off safety margin, and truly visible, respectively. The Min-Cost camera placement problem can be defined as

$$\begin{aligned} &\min \quad \sum_{j=1}^{N_{\text{cand}}} e_j x_j \qquad \text{then} \quad \max \quad \sum_{i=1}^{N_{\text{region}}} o_i \\ &\text{s.t.} \quad \mathbf{A}_c\mathbf{x} \geq \mathbf{b}_c \\ &\quad x_i \in \{0, 1\}, \end{aligned}$$

where o_i are elements of the vector $\mathbf{o} = w_c(\mathbf{A}_c\mathbf{x} > 0) + w_h(\mathbf{A}_h\mathbf{x} = 2) - w_v(\mathbf{A}_v\mathbf{x} > 1)$. The problem is constructed as a consecutive optimization. First, identical to the standard formulation, the setup where the minimal cost that can achieve a desired coverage is found. Then in the second step, with the knowledge of the number of cameras, the set of cameras that optimize a combined objective is sought. The objective is a weighted sum of three independent ones: cover as many regions as possible; each hand-off safety region should be covered by two cameras; and covering the true visible regions by more than one camera is a waste of resources and should be penalized.

5.8. Stereo Cameras

In certain image processing applications, each target may require a stereo camera to cover a scene for the purpose of subsequent processing tasks such as location estimation or pose reconstruction. The stereo camera can be formed by intelligently placing single static cameras with additional constraints to form stereo pairs. The additional constraints, in particular, are (1) the distance between a target k and the cameras in a pair c_i, c_j must be within a minimum and maximum distance $d_{\min}$ and $d_{\max}$ and (2) the angle subtended by the cameras at the target is less than a predefined threshold $\phi_{\max}$. Mathematically this constraint can be presented as [Al Hasan et al. 2008]

$$v^k_{i,j} x_i x_j = 1 \quad \forall k \in \{1, 2, \ldots, K\}, \tag{15}$$

where $v^k_{i,j}$ is a stereo visibility metric for target k, which is defined as

$$v^k_{i,j} = \begin{cases} 1 & d_{\min} \leq \mathrm{Dist}(c_i, k) \leq d_{\min} \text{ and} \\ & d_{\min} \leq \mathrm{Dist}(c_j, k) \leq d_{\min} \text{ and} \\ & \phi(c_i, k, c_j) \leq \phi_{\max}, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

Here $\mathrm{Dist}(c_i, k)$ is the Euclidean distance between camera c_i and target k and $\phi(c_i, k, c_j)$ stands for the smaller angle subtended by camera c_i, target k, and camera c_j. This objective ensures an upper bound of $d_{\max}\sqrt{2 - 2\cos\phi_{\max}}$ on the length of the baseline between the stereo pair.

5.9. Active Cameras

Active cameras with panning and tilting capabilities are often seen as a natural extension to perspective static cameras since they can cover a much greater area with simple panning and tilting motions. However, at any particular time, the coverage is rather limited, which is especially the case for those cameras sitting on slow motors. When an event of interest occurs in a particular area, these cameras may not be able to tune to the correct orientation in time. In order to address this requirement, Erdem and Sclaroff [2006] considered a worst-case scenario where a maximum allowable time $T_{\max}$ is specified for any active camera trying to turn to the event of interest. With the knowledge of the angular speed of a panning head ω_p, this requirement is translated to a reduced coverage region. Suppose that a panning camera can only rotate between $[-\theta_p, \theta_p]$ with 0 being the natural or central orientation. In a worst-case scenario, the camera is currently pointing to $-\theta_p$ and with the specified time $T_{\max}$, the camera can only be rotated to $\theta_{\mathrm{real}} = -\theta_p + \omega_p T_{\max}$. Similarly, when the camera is at its maximum angle θ_p it can only be turned back to $\theta_p - \omega_p T_{\max} = -\theta_{\mathrm{real}}$ within $T_{\max}$. Therefore, to be able to satisfy both worst-case scenarios, the camera's coverage region must lie within $[-\theta_{\mathrm{real}}, \theta_{\mathrm{real}}]$. However, if the angular speed of the panning head is too slow and the θ_{real} is negative, the region of coverage of the two worst-case scenarios will not overlap each other, meaning the specified time cannot be guaranteed. This idea for panning cameras is illustrated in Figure 6. Although tilting can be treated in the same manner, it is often not necessary as tilting of a surveillance camera is usually within a much smaller range than that of panning motions.

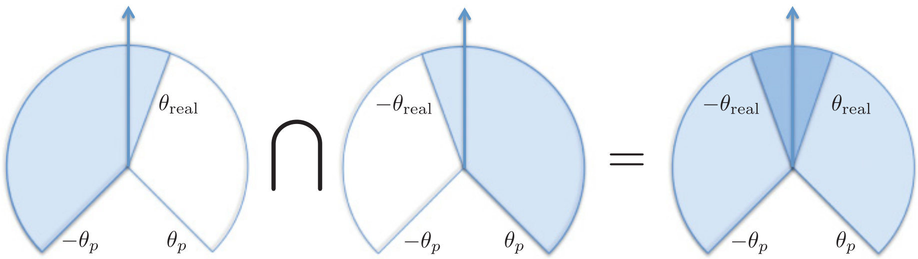

Fig. 6. Reachable region of a panning head. The first two sectors show the coverage of the camera in worst-case scenarios: the camera is at $-\theta_p$ and θ_p, respectively. The last sector shows the valid coverage region in which the camera is guaranteed to be able to turn to any direction within this region under the time constraint T_{max}.

Table III. Camera Placement Main Objectives

Main objective	Paper
Space coverage (Static occlusion)	Erdem and Sclaroff [2006], Angella et al. [2007], David et al. [2007], Murray et al. [2007], Yabuta and Kitazawa [2008], Conci and Lizzi [2009], Fehr et al. [2009], Gonzalez-Barbosa et al. [2009], van den Hengel et al. [2009], Morsly et al. [2010, 2012], Chrysostomou et al. [2012], Liu et al. [2012, 2014], and Konda and Conci [2012, 2013a, 2013b, 2014]
Backup coverage	Murray et al. [2007], Kim et al. [2008], van den Hengel et al. [2009], and Liu et al. [2012, 2014]
Dynamic occlusion	Mittal and Davis [2004, 2008], Gupta et al. [2007a], Mostafavi and Dehghan [2011], and Zhao et al. [2009]
Target observability	Bodor et al. [2005, 2007], Mostafavi and Dehghan [2011], Konda and Conci [2012, 2013b], and Munishwar and Abu-Ghazaleh [2010, 2011, 2013]
Target localization	Ercan et al. [2006]
PTZ cameras	Erdem and Sclaroff [2006], Indu et al. [2009], Rudolph et al. [2014], Konda and Conci [2013a, 2014], and Natarajan et al. [2012, 2014]
Tag visibility	Zhao and Cheung [2007], Zhao et al. [2009], and Zhao [2011]
Stereo cameras	Al Hasan et al. [2008]
Target hand off	Yao et al. [2008, 2010]
Resolution	Fehr et al. [2009], and Chrysostomou et al. [2012]

5.10. Summary of Design Variables and Formulations

Although this section has described the two basic formulations and how researchers extend them to incorporate various design variables such as tag visibility, tracking hand off, and path observability, the techniques surveyed in this section do not provide an exhaustive list. Table III provides a summary of a more complete set of papers arranged according to the main design variable of each paper. Note that many papers incorporate multiple objectives and constraints.

6. OPTIMIZATION

Despite the various formulations and objectives studied in Section 5, many camera placement problems are treated in the discrete domain: Given a precomputed pool of candidate cameras, select a subset that fully satisfies a set of user defined constraints while minimizing some cost function which can be a mixture of multiple types of cost (Min-Cost). This discrete placement problem has been proven to be NP-hard, meaning the true optimal can be hard to find in polynomial time. Although integer linear programming based methods are capable of finding the exact solution for small problems, the combination of the volume of the environment, discritization levels, and available types of cameras often result in a large search space for any reasonable-size problems. Furthermore, it is usually prohibitive for integer programming based methods to find the exact solution (if it exists) in a feasible time frame. There exist many approximation algorithms but only a select few are often used in camera placement algorithms. In this section, we provide an introductory discussion on these approximation algorithms, providing insight to readers who face the selection of the most appropriate optimization technique for their specific camera placement problems. Furthermore, many approximation algorithms require a unique adaptation (linear programming is an adaptation that requires the approximation of nonlinear constraints by their linear equivalences) and therefore we attempt to summarize and highlight the appropriateness of each different adaptation available in the current literature. We build on the recent work of Zhao [2011] who compared integer linear programming based algorithms, greedy algorithms, semidefinite programming, and simulated annealing. But we also include the studies of algorithms with increasing popularity in this field: Artificial Bee Colony, Particle Swarm Optimization, Genetic Algorithms, and Trans-dimensional Simulated Annealing.

6.1. Linear Programming (LP)

Linear programming is a mathematical formulation for determining the optimal value of an objective function. As the name suggests, the objective function as well as all the constraints must be expressed as linear functions of the vector of decision variables. The set of inequalities define a feasible region to be a convex polytope (a polyhedral set) over which the objective is to be optimized. Linear programs ensure that the optimum of the objective function (if it exists) occurs at one of its vertices (extreme points) and thus the problem is significantly reduced to identifying extreme points. This property inspired practically efficient algorithms to be developed for general linear programming problems such as the simplex method [Dantzig 1963].

Integer Linear Program (ILP) is a special type of linear program. It includes additional constraints that some or all of the variables must take integer values. This allows a much greater range of problems that can be modeled. In the case of camera placement, the majority of authors formulate the problem in a way that a candidate pool is first created and then a decision variable $\mathbf{x}$ is associated with each candidate indicating their selection in a particular camera configuration. Therefore, by definition, the decision variable must take the value of 0 or 1 and the resultant formulation is a special type of ILP that is named BIP). While the additional binary constraint is intuitive, it causes the problem to be unsolvable in polynomial time (and usually takes an excessively long time).

A widely used method of solving BIPs is to relax the integer constraint by replacing it with the noninteger alternatives, that is, $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{x} \leq \mathbf{1}$. This relaxation transforms a NP-hard problem into a form solvable in polynomial time. However, in general the solution to the relaxation cannot be guaranteed optimal unless the coefficient matrix $\mathbf{A}$ in Equation (1) is unimodular, which results in the optimal solution being binary automatically. However, as pointed out by Erdem and Sclaroff [2006] the $\mathbf{A}$ unfortunately is not always unimodular. Nonetheless, the LP relaxation has inspired many exact algorithms to solve the BIP problem, for example, the commonly used branch and bound routine [Land and Doig 1960].

6.2. Semidefinite Programming (SDP)

As another alternative to LP relaxation of BIP, SDP is able to provide a tighter relaxation [Laurent 2001] of the binary constraint and offers the capability to optimize nonlinear convex objective functions. Most importantly, SDPs can be solved very efficiently. A SDP can be defined as [Vandenberghe and Boyd 1996]

$$\min \mathbf{c}^{\top}\mathbf{x} \quad\quad (17)$$
$$\text{s.t.} \quad \mathbf{Y} \succeq 0,$$

where $\mathbf{Y} = \mathbf{Y}_0 + \sum_{i=1}^{N} x_i \mathbf{Y}_i$, and $\mathbf{c} \in \mathbb{R}^N$. $\mathbf{Y}_0, \ldots, \mathbf{Y}_N \in \mathbb{R}^{M\times M}$ are symmetric matrices. The $\succeq$ sign means $\mathbf{Y}$ is positive semidefinite, that is, $\mathbf{z}^{\top}\mathbf{Y}\mathbf{z} \geq \mathbf{0}$ for all $\mathbf{z} \in \mathbb{R}^n$.

A BIP formulation, such as Equation (1), can be converted to a SDP using the "Lift and Project" process as follows [Zhao 2011]. A standard BIP formulation is first rewritten as an equivalent quadratic program,

$$\min \mathbf{c}^{\top}\mathbf{x} \quad\quad (18)$$
$$\text{s.t.} \quad \mathbf{A}\mathbf{x} \geq \mathbf{b} \quad \text{and} \quad (1 - x_i)x_i = 0,$$

where $\mathbf{A} \in \mathbb{R}^{M\times N}$, $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{b} \in \mathbb{R}^M$. M is the number of inequality constraints and N is the total number of decision variables. In the simplest problem posted in Equation (1), M is the number of discrete regions that need to be covered and N is equivalent to the number of cameras in the pool of candidates. One wishes to introduce a new matrix $\mathbf{Y}$ such that

$$\mathbf{Y} = \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix}\begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix}^{\top} = \begin{pmatrix} 1 & \mathbf{x}^{\top} \\ \mathbf{x} & \mathbf{x}\mathbf{x}^{\top} \end{pmatrix},$$

and $\mathbf{Y} \in \mathbb{R}^{(N+1)\times(N+1)}$. Then, the constraint that $\mathbf{Y}$ is semidefinite is imposed, $\mathbf{Y} \succeq 0$. For convenience $x_0 = 1$ has been prefixed to $\mathbf{x}$ and therefore $y_{0,0} = 1$. The $y_{i,0} = y_{i,i}$ constraint is derived from the binary constraint $x_i x_i = x_i$. The SDP adoption also includes additional linear inequalities derived from the original constraints $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and the relaxation of the binary constraint $0 \leq x_i \leq 1$,

$$\begin{gathered}(1 - x_i)\mathbf{A}_r\mathbf{x} \geq (1 - x_i)b_r, \\ x_i\mathbf{A}_r\mathbf{x} \geq x_i b_r, \\ \forall i \in \{1, \ldots, N\}, \\ \forall r \in \{1, \ldots, M\},\end{gathered}$$

where $\mathbf{A}_r$ is the rth row of matrix $\mathbf{A}$ and b_r is the rth element of vector $\mathbf{b}$. These inequalities can easily be rewritten in terms of $y_{i,j}$ using properties specified previously. Lastly, $\mathbf{Y}$ is solved using standard SDP algorithms, such as the interior point method [Karmarkar 1984], and x_i can be recovered from $x_i = y_{0,i}$.

6.3. Greedy Heuristics

As the most intuitive method among all the approximation methods, greedy algorithms are the simplest algorithms for constructing feasible solutions to many combinatorial optimization problems. In camera placement, a greedy algorithm selects one camera to be included in the final optimal set at each iteration according to some predefined rules. It ensures that at each stage the selected camera offers a locally optimal contribution to the result. For example, if the objective is to maximize the coverage area, during each iteration, the camera that provides the most coverage of the uncovered region will be selected. The iteration may stop when the maximum number of allowable cameras is reached or all constraints are satisfied. Presented in Appendix A, Algorithm 1 is a representative example of a greedy algorithm similar to the ones proposed in Horster and Lienhart [2009] and Zhao et al. [2009].

Greedy algorithms have been used widely in many fields of study due to their superior computational efficiency. However, their suitableness has to be considered carefully. Greedy algorithms are deterministic iterative algorithms that never reconsider their choices made previously. Therefore, they only make locally optimal steps but may not be globally optimal overall. If the camera placement problem exhibits optimal substructure (as opposed to overlapping substructure), then greedy algorithms are one of the best approximation algorithms. This may be the case for the simplest set covering problem [Zhao 2011] but for a more sophisticated objective, such as dynamic occlusion, greedy algorithms are often inadequate. Many attempts have been made to improve greedy algorithms and a particular one that has been seen in camera placement is the GRASP routine (Greedy Randomized Adaptive Search Procedure) [Al Hasan et al. 2008], which borrows the idea of random sampling to avoid being trapped into local optima. Instead of always making the greedy choice, it randomly selects one from a number of near-optimal solutions.

6.4. Swarm Intelligence (SI)

Recently, a family of bioinspired algorithms, namely, Swarm Intelligence, has found its use in many areas including optimization. SI algorithms model the "collective behavior of decentralized, self-organized systems" [Bonabeau et al. 1999]. These systems are usually simple agents that interact with each other and with the environment. Although they lack the knowledge of the global state of the system and act according to a set of simple rules, interactions among them often lead to the emergence of intelligent global behaviors that are optimal in certain circumstances. Any swarm intelligent behaviors possess two fundamental characteristics: self-organization and division of labor [Karaboga et al. 2012]. Self-organization is a set of dynamical mechanisms that establish basic rules for the interactions of agents purely at a local level and result in structures observable at a global level. Division of labor indicates that different tasks are simultaneously performed by specialized individuals who cooperate with each other to accomplish larger objectives. SI algorithms are usually named after the natural events they mimic: Ant Colony Optimization [Dorigo et al. 1996], Artificial Bee Colony (ABC) algorithm [Karaboga et al. 2012], Artificial Immune Systems [Dasgupta 1999], Glowworm Swarm Optimization [Krishnanand and Ghose 2009], and PSO [Kennedy and Eberhart 1995]. Among these, of particular interest in the context of camera placement are the ABC and PSO algorithms.

6.4.1. Artificial Bee Colony (ABC). The ABC algorithm is an emerging SI algorithm capable of optimizing complex objective functions through mimicking the foraging behavior of a colony of bees. The algorithm is comprised of three essential components and two modes of behavior: food sources, employed foragers (employed bee), and unemployed foragers (onlooker and scout bee), and the recruitment to a rich nectar source and the abandonment of a poor source. In an optimization context, a food source is a feasible solution to the problem associated with a single measure of "profitability" or a fitness value of the solution. Each employed bee tries to fully explore a particular food source. Onlooker bees are responsible for processing the information obtained by the employed bee and scout bees randomly search the whole problem space to identify potential food sources for further exploration. More specifically, the standard ABC algorithm is summarized as a four-step iterative process [Karaboga and Basturk 2007].

—Step 1: initialization. An initial population of food sources (feasible solutions) is created randomly by the employed bees and their qualities are determined by employing an appropriate fitness function of the objective to be optimized.
—Step 2: employed bee phase. These bees share the information of the food sources with onlooker bees and then go back to their remembered food source and search for a new food source in the neighborhood of the old one. The new food source will be remembered if it has higher quality value.

—Step 3: onlooker bee phase. Each onlooker bee selects a food source from all food sources reported by the employed bees probabilistically depending on the quality of the food source. The onlooker then chooses a new food source from the neighborhood of the selected food source.
—Step 4: scout bee phase: When a food source is abandoned by the bees due to reaching a predefined criteria, a new food source is randomly determined by a scout bee. Scouts have low search costs and result in a low average food source quality.
—The best food source is remembered and Steps 2–4 are repeated until convergence or a stopping criteria is reached.

Chrysostomou et al. [2012] and Chrysostomou and Gasteratos [2012] proposed a variant of ABC to compute the optimal multicamera topology for crowd analysis. The authors introduced the concepts of elite bees and elite sites. In the onlooker phase, the food sources are ranked according to their preference level by the onlooker bees (fitness value) and the top ranked ones are marked as sites for further exploration. Among the sites, the top few of them are named elite sites with the bees that discovered them marked as elite bees. A site is effectively a food source with a search boundary defined (patch size). For each site, extra bees are recruited for detailed exploitation of the site (search within the neighborhood of a good solution). More bees are recruited for elite sites than normal sites as an elite site represents a higher confidence in finding the optimal solution. A patch size is defined for each site to restrict the exploitation process to be within a boundary. If there are bees that have not been assigned an exploitation task, they become a scout bee which searches for a food source randomly in the search space. The best food source of each site are grouped together to form part of a new population. The other part of the new population is formed by the food source discovered by the newly assigned scout bees who again randomly probe in the problem space. It is crucial in any search process that the procedures of exploration and exploitation must be carried out in conjunction with each other. Compared to the standard ABC, this modified version has two benefits. First, it allows more refined exploitation of the neighborhood of a feasible solution. Second, employing a set of dedicated scout bees allows the search space to be better explored, especially in camera placement problems where the search space is usually large. The steps are listed in the metaheuristic in Appendix B, Algorithm 2.

6.4.2. Particle Swarm Optimization (PSO). The second category of SI algorithms that have found their use in camera placement problems is the PSO [Conci and Lizzi 2009; Konda and Conci 2012, 2013b; Xu et al. 2010, 2011; Morsly et al. 2010, 2012; Zhou and Long 2011; Xu et al. 2013]. PSO is a class of more popular optimization algorithms intended to simulate social behavior representing the movement of organisms such as a flock of birds or a school of fish [Kennedy and Eberhart 1995; Kennedy 1997]. In PSO, each feasible solution is a particle and a swarm of particles are moved in the search space in a similar fashion as birds move in flocks to find food. Artificially, each particle acts according to simple functions over its current position (i.e., current feasible solution), a randomized velocity, its own best-known position, and the swarm's best-known position. Same as ABC (and GA, SA, TDSA), PSO is also a metaheuristic and can search large spaces of nonlinear, nondifferentiable objective functions with a cost of not being able to guarantee optimality. Listed in Appendix C, Algorithm 3 is the basic variant of PSO, which has been used to optimize the utility of a network of cameras, that is, Max-Utility problem [Conci and Lizzi 2009; Konda and Conci 2012, 2013b; Zhou and Long 2011; Xu et al. 2011]. If there are constraints that have to be satisfied while optimizing the objective, one can either include them as penalty functions to the objective or use feasibility operators, which are operators that make a particle feasible when it violates a constraint [Xu et al. 2010, 2013].

In its generic form (Appendix C, Algorithm 3) PSO can only be used to solve formulations with a known number of cameras. Another variant of PSO, namely, Binary Particle Swarm Optimization (BPSO), developed by Kennedy and Eberhart [1997], has been used in cases when the decision variable is a binary vector with an unknown number of cameras. The generic PSO adjusts trajectories through controlling each particle with a velocity. In BPSO, velocity is redefined as changes of probabilities of selecting a particular candidate camera. Morsly et al. showed that their proposed BPSO-IP (BPSO-Inspired Probability) achieves a more superior performance than many alternatives in the camera placement context [Morsly et al. 2010, 2012].

6.5. Genetic Algorithms (GA)

Besides PSO and ILP based approaches, another class of well-used methods is the GA. A GA is a metaheuristic optimization algorithm that mimics the process of natural selection through using techniques inspired by natural evolution, such as mutation and crossover [Goldberg 1989]. As a population based algorithm, a GA maintains a population of candidate solutions (termed chromosome) at all times and the population evolves toward a better solution as iteration count increases. Generic GAs represent the solutions using binary vectors, which is a desired property for discrete camera placement formulations. A standard GA proceeds as follows.

—First, a generation of candidate solutions is randomly selected from the problem search space.
—Then the algorithm starts to iterate until certain stopping criteria are met.
—In each iteration, a subset of candidate solutions are selected as seed to breed a new generation of solutions, which is of the same size as the current generation: The candidate solutions are first evaluated using a fitness function that allows them to be ranked. The top few solutions are selected as parents for the next generation. A number of operations can be performed on the parents to achieve this and typical ones include crossover and mutation. Crossover selects more than one parent chromosome from the current seeds and combines them to form a chromosome belonging to the new generation. A mutation operation alters one or more values in a chromosome of the current seed to form a new chromosome. Although mutation and crossover are the two main operations commonly performed in GAs, other operations can be defined as well. This process of generation evaluation continues until stopping criteria are met. As an example, in a camera placement context, each chromosome is a set of cameras and a crossover operation can be defined as a swapping of a subset from one set with a subset of the same size from another set of cameras [van den Hengel et al. 2009].

GA algorithms have been used widely in camera placement [David et al. 2007; Feng et al. 2011; van den Hengel et al. 2009; Indu et al. 2009; Kim et al. 2008; Murray et al. 2007; Olague and Mohr 2002; Yao et al. 2008, 2010] due to a number of preferable characteristics. First, GAs do not make assumptions on the objective function and the search space, such as the existence of derivatives and continuity. Secondly, they provide a good balance between exploitation and exploration through the use of random processes and genetic operators such as mutation and crossover. Thirdly, the generic GAs represent the solution binary vectors which are extremely convenient to use in the discrete formulations of camera placement problems. Fourthly, GAs are population based, meaning it will not only find the best approximation to the optimum, it will also list meaningful candidates. Lastly, the convergence behavior of GAs to the optimum is not only supported by the natural selection processes they attempt to mimic but has also been theoretically explained by the schemata theorem [Holland 1975].

6.6. Markov Chain Monte Carlo (MCMC)

MCMC is a class of sampling based methods, including algorithms such as Metropolis-Hastings (MH) sampling [Metropolis et al. 1953], Gibbs sampling [Gelfand and Smith 1990], and Simulated Annealing [Kirkpatrick 1984]. The aim of these methods is to obtain a set of samples from a probability distribution through the construction of a Markov chain that has the desired distribution as the equilibrium distribution. When the chain is initially started from a random position, it will experience varying statistical properties but after a certain number of steps (burn-in period) the chain will reach a steady state where the statistic property no longer alters (i.e., equilibrium position). Any subsequent samples in the chain can be used as a sample of the original distribution. MH and Gibbs sampling are two classical MCMC algorithms for obtaining such Markov chains. The original use of MH and Gibbs sampling is to compute integrals of complicated functions by summing up samples drawn from the function that has been perceived as a distribution. In a natural way, one can use MH and Gibbs sampling for optimization by simply treating the objective function as a distribution from which a set of samples are to be drawn. Then as the Markov chain is constructed (i.e., as each samples is drawn consecutively), the best sample seen is kept as the best approximation to the optimal. This sampling strategy is completely random and uniform and therefore may take an excessively long time to find a good approximation.

A strategy named Simulated Annealing (SA) [Kirkpatrick 1984] is purposely designed for finding the optima by sampling only around regions where peaks exist. Instead of sampling from the original objective function, SA algorithms introduce an variable named temperature to the objective function. The temperature is initially set as a high value and will decrease according to a cooling schedule. At each temperature, SA algorithms sample from a distribution proportional to the modified objective function at the temperature value. As the temperature cools down, samples drawn will be increasingly clustered around the optima. This annealing process tries to mimic the annealing process: as a heated material slowly cools down, its molecules will line up in a minimum energy pattern provided that the cooling process is sufficiently slow. SA algorithms mimic this process and have been proven to converge [Henderson et al. 2003; Granville et al. 1994]. In camera placement, SA algorithms have mostly been used for the cases when the number of cameras is known [Zhao 2011; Mittal and Davis 2008; Debaque et al. 2009]. Algorithm 4 in Appendix D presents the pseudocode of the algorithm.

SA algorithms in their basic form have the restriction that the number of cameras must be known and cannot be a variable to be inferred from the data. To address this, Liu et al. [2012, 2014] proposed to solve the camera placement problem in a TDSA framework [Brooks et al. 2003; Andrieu et al. 2000]. The key difference is that in Liu et al.'s work, the number of cameras is included as a penalty term appended to the objective function. This, when combined with a TDSA algorithm, allows the number of cameras, as well as the parameters of each camera to be jointly optimized. The TDSA algorithm is essentially an extension to conventional SAs in that TDSA incorporates a move selection step, which randomly selects the move to generate the next candidate state in the Markov chain. Instead of having a single move, as in SAs, to alter the parameters of a current state, Liu's TDSA includes two additional moves, birth and death, which alter the dimension of the current state to form a new candidate. This strategy effectively explores a large number of dimensions and selects the optimal approximation that consists of the optimal number of cameras (optimal model dimension) along with the optimal parameters of each camera.

SA (and TDSA) is used in camera placement problems due to a number of reasons. First, it treats the objective function as a probability which does not have to be linear, convex, or differentiable. Secondly, it outputs a list of alternative solutions which are meaningful in certain circumstances. Lastly, although SA is inspired from a natural process, its convergence has been proven [Henderson et al. 2003; Granville et al. 1994]. SA algorithms, however, suffer from a number of drawbacks. Despite being effective, they can be quite inefficient. The amount of iterations is determined jointly by the number of temperature decrements and the number of samples in the Markov chain drawn at each temperature. The cooling schedule therefore must be carefully chosen to obtain a balance of solution quality and processing time. However, this can be a difficult task and the choice may be case dependent. Due to this reason, often in simulated annealing algorithms such as Liu et al. [2014], conservative cooling schedules are used to allow greater tolerance to variations in the problem complexity at the cost of much longer processing time.

6.7. Summary and Remarks

In this section, we have reviewed a number of optimization techniques found in the camera placement literature. These are ILP (BIP, in particular), SDP, greedy, GA, ABC, PSO, SA, and TDSA. Strictly speaking, BIP and SDP are formulations since they define standard structures for problems to be represented. They are included in the discussion of optimization because they serve as entry points to standard optimization procedures (e.g., Branch and Bound) that no longer require problem context. On the other hand, all the other methods discussed, including greedy, GA, ABC, PSO, SA, and TDSA, are metaheuristic methods, and the application of them must be accompanied with the context of the problem. For example, the crossover operation in GA and the birth step in TDSA all need to be properly defined within a specific context and the performance of the algorithm is sensitive to how these functions are defined.

Among all algorithms studied, BIP methods are the only methods that output true optimal solutions. However, the Branch and Bound algorithm, which is used in almost all camera placement approaches to directly solve BIP formulations, has a worst-case complexity of $O(2^N)$, and it has been shown in various cases that the large search space of a reasonably sized camera placement problem cannot be adequately addressed at all by the algorithm. SDP on its own may be solved to optimality, but a SDP is only obtained through relaxing the constraints of the original BIP problem, and the effect of the relaxation is still unknown. Thus, the quality of the approximated solution cannot be guaranteed. A greedy heuristic is the most intuitive alternative to BIP methods as it selects the camera that offers the optimal performance in each iteration. If the problem exhibits optimal substructures, then the set of cameras selected in this manner is globally optimal, but this is not the case in most camera placement problems. All the other metaheuristics share a number of desirable features in common. Most importantly, these methods do not have strict requirements for the objective function, constraints, and the search space, such as the existence of derivatives and continuity. Secondly, they provide a good balance between exploitation and exploration. ABC algorithms achieve this through the use of employed bees and scout bees; GAs use random processes and genetic operators such as mutation and crossover to perform exploitation and exploration. Lastly, all these methods produce not only the best solution found but also a short list of alternatives, which can also be practically useful. These three reasons make them preferable choices in real-world camera placement problems as opposed to traditional optimization methods such as hill climbing. One drawback of these methods is that they lack a confidence measure with respect to the true optimal solution. The goodness of a particular method is often demonstrated using restricted examples that are solvable using BIP methods. Table IV presents a summary of all surveyed papers.

Table IV. Summary of Recent Camera Placement Approaches

	Disc. or Cont.			Num. of Cam.			Focus			
	Disc.	Cont.	NA	Known	Unknown	NA	Form.	Opti.	Main Objective	Opti. Method
Mittal and Davis [2004, 2008]	✓			✓			✓		dynamic occlusion	SA
Bodor et al. [2005, 2007]		✓		✓			✓		path observability	hill-climbing
Ercan et al. [2006]	✓			✓			✓		target localization	SDP
Erdem and Sclaroff [2006]	✓				✓		✓	✓	coverage, PTZ operation	BIP
Angella et al. [2007]	✓			✓			✓		coverage	heuristic
David et al. [2007]	✓				✓		✓		coverage	BIP, GA
Gupta et al. [2007a]	✓				✓		✓		dynamic occlusion and confusion	greedy heuristic
Murray et al. [2007]	✓				✓		✓		coverage, backup coverage	NA
Zhao and Cheung [2007] and Zhao et al. [2009]	✓				✓		✓	✓	tag visibility	BIP, greedy heuristic
Al Hasan et al. [2008]	✓				✓		✓		stereo cameras	BIP, greedy heuristic
Kim et al. [2008]	✓				✓		✓	✓	backup coverage	multiobjective GA
Mostafavi and Dehghan [2011]	✓				✓		✓		target coverage from multiple orientations, dynamic occlusion	BIP
Yabuta and Kitazawa [2008]	✓				✓		✓		coverage	BIP
Yao et al. [2008, 2010]	✓				✓		✓		target hand-off safety margin	GA
[Gonzalez-Barbosa et al. 2009]	✓				✓		✓		coverage	NA
[Conci and Lizzi 2009]			✓			✓	✓		coverage	PSO
Fehr et al. [2009]			✓			✓	✓		coverage, foreshortening and resolution	NA
Van den Hengel et al. [2009]			✓			✓	✓	✓	coverage and backup coverage	GA
Horster and Lienhart [2009]	✓			✓	✓		✓	✓	coverage	BIP, greedy heuristic
Indu et al. [2009]		✓		✓			✓	✓	PTZ operation, priority points	GA
Morsly et al. [2010, 2012]	✓				✓		✓	✓	coverage	PSO and variants
Munishwar and Abu-Ghazaleh [2010, 2011, 2013]	✓			✓			✓	✓	amount of targets	centralized and distributed heuristics
Zhao [2011]	✓				✓		✓	✓	tag visibility	BIP, greedy heuristic, SDP, SA
Chrysostomou and Gasteratos [2012]	✓				✓			✓	multiple	ABC
Chrysostomou et al. [2012]	✓					✓		✓	coverage, FoV, resolution, frame rate	ABC
Liu et al. [2012, 2014]		✓		✓	✓		✓	✓	coverage	TDSA
Konda and Conci [2012, 2013b]	✓			✓			✓	✓	coverage, target coverage, illumination	PSO
Natarajan et al. [2012, 2014]	✓			✓			✓		PTZ camera control, target motion, resolution, number of targets	greedy heuristics
Konda and Conci [2013a, 2014]			✓	✓			✓		PTZ camera control, scene entropy, coverage, targe coverage and resolution/zoom	deterministic

7. CONCLUSION AND FUTURE WORK

Camera networks have become complex systems capable of obtaining extensive video information for intelligent processing tasks, such as target localization, identification, and tracking. In all cases, it is of vital importance that the optimal camera configuration (i.e., optimal location, orientation, etc.) is determined before cameras are deployed, as the cost of modifications of the network can be expensive; a suboptimal placement solution will adversely affect intelligent video surveillance and video analytic algorithms; and the optimal configuration may provide substantial savings on the total number of cameras required to achieve the same level of utility.

There exists a large body of work trying to tackle this problem with varying capabilities, approaches, and degrees of success. In this article, we have attempted to summarize all the recent approaches to camera placement in a structured manner. Current research trends in camera placement have taken two directions:

(1) addressing specific user requirements and/or
(2) developing an effective optimization strategy.

Specific focus has been given in this review to the formulations of the various objectives and methods used to solve them. A number of fundamental aspects of current camera placement approaches including camera and environment models, simplification, and quantization techniques have been reviewed. We have also presented categorization of the camera placement problem, that is, discrete or continuous depending on whether the number of cameras is known a priori. Finally, we have summarized most, if not all, recent papers (post 2000) on the topic. We believe that our review can serve as a first point of entry for readers wishing to start research into this area or engineers who need to implement a camera placement system in practice.

There are a number of areas where current work on camera placement can be extended. From a practical stem point, there is a lack of a unified framework in which all the discussed design variables can be effectively incorporated. End-users will benefit from such a framework since they can directly see the effect of addition or removal of a particular design variable without having to redesign the system. From the theoretical perspective, camera placement researchers tend to focus on the practical aspects of the topic, resulting in discrepancies in understanding of the theoretical implications of the design variables as well as the methods of computing goodness of the approximated results. If the discrepancies, such as the effects of occlusion on the optimization process, can be better addressed, more effective and efficient optimization may be derived as a consequence.

Cameras are evolving with the advances in embedded processing and wireless communication technologies. Wireless smart cameras have been developed to serve as extensions to current fixed camera networks. As these camera networks are naturally large in scale and often need redeployment, there is a significant demand of an efficient and effective automatic technique to select the optimal camera configurations in order to maximize the utility of these large scale camera networks. However, current solutions to camera placement are mostly designed as an offline optimization for centralized camera network; there will be, or perhaps have already been, distributed camera placement algorithms developed to satisfy this growing need. The difficulty in designing a distributed algorithm, however, is to estimate the goodness of the obtained results in comparison to the true optimum, which can be extremely difficult to find for these large and sometimes dynamic networks.

As cameras are becoming smarter and more and more pervasive, there is a rising trend in concerns for privacy intrusion, leading to the introduction of privacy-aware cameras. Privacy-aware cameras are essentially smart cameras capable of onboard processing to protect identities of individuals while perserving behavioral information. Algorithms executed on these cameras are either global protections based on algorithms such as downsampling and blurring or object based. Object based approaches first identify sensitive regions on the image using computer vision methods, such as face detection, and then apply image processing techniques such as blanking, pixelation, etc., to ensure these regions are protected before the images leave the cameras. As Winkler and Rinner [2014] point out, system utility of a camera degrades as privacy protection level increases, thus for engineers needing to design a camera placement approach for a privacy-aware camera network, one fundamental consideration will be how to best utilize camera coordination to compensate the loss of system utility as a result of privacy protection.

Camera placement is still far from being claimed as a solved topic. While current solutions are mainly targeting practical aspects of the problem, leaving behind a number of theoretical discrepancies including understanding of the optimality of the obtained results as well as effects of varies design variables. The problem is also evolving along with the advances in cameras, the spread of camera networks, shifts in demand, and the rising concerns of privacy. All of these represent endless opportunities for camera placement engineers and researchers to explore.

REFERENCES

Besma R. Abidi, Nash R. Aragam, Yi Yao, and Mongi A. Abidi. 2009. Survey and analysis of multimodal sensor planning and integration for wide area surveillance. *ACM Computing Surveys* 41, 1, Article 7 (Jan. 2009), 36 pages.

Steven Abrams, Peter K. Allen, and Konstantinos Tarabanis. 1999. Computing camera viewpoints in an active robot work-cell. *International Journal of Robotics Research* 18 (1999), 267–285.

Mohammad Al Hasan, Krishna K. Ramachandran, and John E. Mitchell. 2008. Optimal placement of stereo sensors. *Optimization Letters* 2, 1 (2008), 99–111.

Khaled Amriki and Pradeep K. Atrey. 2011. Towards optimal placement of surveillance cameras in a bus. In *2011 IEEE International Conference on Multimedia and Expo (ICME)*. 1–6.

Christophe Andrieu, Nando de Freitas, and Arnaud Doucet. 2000. Reversible jump MCMC simulated annealing for neural networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 11–18.

F. Angella, L. Reithler, and F. Gallesio. 2007. Optimal deployment of cameras for video surveillance systems. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS'07)*. 388–392.

Eric Becker, Gutemberg Guerra-Filho, and Fillia Makedon. 2009. Automatic sensor placement in a 3D volume. In *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assisstive Environments (PETRA'09)*. ACM, New York, NY, Article 36, 8 pages.

Robert Bodor, Andrew Drenner, Paul Schrater, and Nikolaos Papanikolopoulos. 2007. Optimal camera placement for automated surveillance tasks. *Journal of Intelligent and Robotic Systems* 50, 3 (2007), 257–295.

Robert Bodor, Paul Schrater, and Nikolaos Papanikolopoulos. 2005. Multi-camera positioning to optimize task observability. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS'05)*. 552–557.

Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. 1999. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Inc., New York, NY.

S. P. Brooks, N. Friel, and R. King. 2003. Classical model selection via simulated annealing. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65, 2 (2003), 503–520.

Paolo Burelli, Luca Gaspero, Andrea Ermetici, and Roberto Ranon. 2008. Virtual camera composition with particle swarm optimization. In *Smart Graphics*, Andreas Butz, Brian Fisher, Antonio Krueger, Patrick Olivier, and Marc Christie (Eds.). Lecture Notes in Computer Science, Vol. 5166. Springer, Berlin, 130–141.

S. Y. Chen and Y. F. Li. 2004. Automatic sensor placement for model-based robot vision. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 34, 1 (2004), 393–408.

Xing Chen. 2009. *Improved Detection and Tracking of Objects in Surveillance Video*. Ph.D. dissertation. Queensland University of Technology.

Xing Chen and James Davis. 2008. An occlusion metric for selecting robust camera configurations. *Machine Vision and Applications* 19, 4 (2008), 217–222.

Dimitrios Chrysostomou and Antonios Gasteratos. 2012. Optimum multi-camera arrangement using a bee colony algorithm. In *2012 IEEE International Conference on Imaging Systems and Techniques (IST)*. 387–392.

Dimitrios Chrysostomou, Georgios Ch. Sirakoulis, and Antonios Gasteratos. 2012. A bio-inspired multi-camera topology for crowd analysis. In *1st International Workshop on Pattern Recognition and Crowd Analysis*.

V. Chvatal. 1975. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B* 18, 1 (1975), 39–41.

N. Conci and L. Lizzi. 2009. Camera placement using particle swarm optimization in visual surveillance applications. In *2009 16th IEEE International Conference on Image Processing (ICIP)*. 3485–3488.

Marcelo C. Couto, Cid C. Souza, and Pedro J. Rezende. 2008. Experimental evaluation of an exact algorithm for the orthogonal art gallery problem. In *Experimental Algorithms*, Catherine C. McGeoch (Ed.). Lecture Notes in Computer Science, Vol. 5038. Springer, Berlin, 101–113.

Marcelo C. Couto, Pedro J. de Rezende, and Cid C. de Souza. 2011. An exact algorithm for minimizing vertex guards on art galleries. *International Transactions in Operational Research* 18, 4 (2011), 425–448.

George B. Dantzig. 1963. *Linear Programming and Extensions*. Princeton University Press, New Jersey.

Dipankar Dasgupta. 1999. An overview of artificial immune systems and their applications. In *Artificial Immune Systems and Their Applications*, Dipankar Dasgupta (Ed.). Springer, Berlin, 3–21.

Pierre David, Vincent Idasiak, and F. Kratz. 2007. *A Sensor Placement Approach for the Monitoring of Indoor Scenes*. Lecture Notes in Computer Science, Vol. 4793. Springer, Berlin, 110–125.

Benoit Debaque, Rym Jedidi, and Donald Prevost. 2009. Optimal video camera network deployment to support security monitoring. In *12th International Conference on Information Fusion (FUSION'09)*. 1730–1736.

Simon Denman, Alina Bialkowski, Clinton Fookes, and Sridha Sridharan. 2011. Determining operational measures from multi-camera surveillance systems using soft biometrics. In *2011 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*. 462–467.

Joachim Denzler, Matthias Zobel, and Heinrich Niemann. 2002. On optimal camera parameter selection in Kalman filter based object tracking. In *Pattern Recognition*, Luc Gool (Ed.). Lecture Notes in Computer Science, Vol. 2449. Springer, Berlin, 17–25.

Marco Dorigo, Vittorio Maniezzo, and Albert Colorni. 1996. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 26, 1 (1996), 29–41.

Ali O. Ercan, Danny B. Yang, Abbas El Gamal, and Leonidas J. Guibas. 2006. Optimal placement and selection of camera network nodes for target localization. In *2nd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'06)*. Springer-Verlag, Berlin, 389–404.

Ugur Murat Erdem and Stan Sclaroff. 2004. Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints. In *Proceedings of OMNIVIS Workshop*.

Ugur Murat Erdem and Stan Sclaroff. 2006. Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements. *Computer Vision and Image Understanding* 103, 3 (2006), 156–169.

Emre Ertin, John W. Fisher, and Lee C. Potter. 2003. Maximum mutual information principle for dynamic sensor query problems. In *2nd International Conference on Information Processing in Sensor Networks (IPSN'03)*. Springer-Verlag, Berlin, 405–416.

Duc Fehr, Loren Fiore, and Nikolaos Papanikolopoulos. 2009. Issues and solutions in surveillance camera placement. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*. 3780–3785.

Guodong Feng, Min Liu, Xuemei Guo, Jun Zhang, and Guoli Wang. 2011. Genetic algorithm based optimal placement of PIR sensor arrays for human localization. In *2011 International Conference on Mechatronics and Automation (ICMA)*. 1080–1084.

Steve Fisk. 1978. A short proof of Chvátal's watchman theorem. *Journal of Combinatorial Theory, Series B* 24, 3 (1978), 374.

Shachar Fleishman, Daniel Cohen-Or, and Dani Lischinski. 1999. Automatic camera placement for image-based modeling. In *7th Pacific Conference on Computer Graphics and Applications, 1999*. 12–20, 315.

Clinton Fookes, Frank Lin, Vinod Chandran, and Sridha Sridharan. 2012. Evaluation of image resolution and super-resolution on face recognition performance. *Journal of Visual Communication and Image Representation* 23, 1 (2012), 75–93.

Alan E. Gelfand and Adrian F. M. Smith. 1990. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association* 85, 410 (1990), 398–409.

David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning* (1st ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

Jose-Joel Gonzalez-Barbosa, Teresa Garcia-Ramirez, Joaquin Salas, Juan-Bautista Hurtado-Ramos, and Jose-de-Jesus Rico-Jimenez. 2009. Optimal camera placement for total coverage. In *IEEE International Conference on Robotics and Automation (ICRA'09)*. 844–848.

Vincent Granville, Mirko Krivanek, and Jean-Paul Rasson. 1994. Simulated annealing: A proof of convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16, 6 (1994), 652–656.

Ned Greene, Michael Kass, and Gavin Miller. 1993. Hierarchical z-buffer visibility. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'93)*. ACM, New York, NY, 231–238.

Abhinav Gupta, Anurag Mittal, and Larry S. Davis. 2007a. COST: An approach for camera selection and multi-object inference ordering in dynamic scenes. In *IEEE 11th International Conference on Computer Vision (ICCV'07)*. 1–8.

Abhinav Gupta, Anurag Mittal, and Larry S. Davis. 2007b. COST: An approach for camera selection and multi-object inference ordering in dynamic scenes. In *ICCV'07*. 1–8.

Darrall Henderson, Sheldon H. Jacobson, and Alan W. Johnson. 2003. The theory and practice of simulated annealing. In *Handbook of Metaheuristics*, Fred Glover and Gary A. Kochenberger (Eds.). International Series in Operations Research & Management Science, Vol. 57. Springer US, 287–319.

John H. Holland. 1975. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, MI.

Eva Horster and Rainer Lienhart. 2006. On the optimal placement of multiple visual sensors. In *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks*. ACM, 111–120.

Eva Horster and Rainer Lienhart. 2009. *Multi-Camera Networks: Concepts and Applications*. Elsevier, Chapter 5, Optimal Placement of Multiple Visual Sensors.

Y.-C. Hsieh, Y.-C. Lee, and P.-S. You. 2011. The optimal locations of surveillance cameras on straight lanes. *Expert Systems with Applications* 38, 5 (2011), 5416–5422.

S. Indu, Santanu Chaudhury, Nikhil R. Mittal, and Asok Bhattacharyya. 2009. Optimal sensor placement for surveillance of large spaces. In *3rd ACM/IEEE International Conference on Distributed Smart Cameras. (ICDSC'09)*. 1–8.

Dervis Karaboga and Bahriye Basturk. 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization* 39, 3 (2007), 459–471.

Dervis Karaboga, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga. 2012. A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review* (2012), 1–37.

N. Karmarkar. 1984. A new polynomial-time algorithm for linear programming. *Combinatorica* 4, 4 (1984), 373–395.

Giorgos D. Kazazakis and Antonis A. Argyros. 2002. Fast positioning of limited-visibility guards for the inspection of 2D workspaces. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002.*, Vol. 3. 2843–2848.

James Kennedy. 1997. The particle swarm: Social adaptation of knowledge. In *IEEE International Conference on Evolutionary Computation, 1997*. 303–308.

James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks, 1995*, Vol. 4. 1942–1948.

James Kennedy and Russell Eberhart. 1997. A discrete binary version of the particle swarm algorithm. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation.*, Vol. 5. 4104–4108.

Kamyoung Kim, Alan T. Murray, and Ningchuan Xiao. 2008. A multiobjective evolutionary algorithm for surveillance sensor placement. *Environment and Planning B: Planning and Design* 35, 5 (2008), 935–948.

Scott Kirkpatrick. 1984. Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics* 34, 5 (1984), 975–986.

Krishna Reddy Konda and Nicola Conci. 2012. Camera positioning for global and local coverage optimization. In *2012 6 International Conference on Distributed Smart Cameras (ICDSC)*. 1–6.

Krishna Reddy Konda and Nicola Conci. 2013a. Optimal configuration of PTZ camera networks based on visual quality assessment and coverage maximization. In *2013 7th International Conference on Distributed Smart Cameras (ICDSC)*. 1–8.

Krishna Reddy Konda and Nicola Conci. 2013b. Global and local coverage maximaization in multi-camera networks by stochastic optimization. *Infocommunications Journal* 2013 (2013).

Krishna Reddy Konda and Nicola Conci. 2014. Real-time reconfiguration of PTZ camera networks using motion field entropy and visual coverage. In *Proceedings of the International Conference on Distributed Smart Cameras (ICDSC'14)*. ACM, New York, NY, Article 18, 8 pages.

K. N. Krishnanand and D. Ghose. 2009. Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. *Swarm Intelligence* 3, 2 (2009), 87–124.

Ailsa H. Land and Alison G. Doig. 1960. An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society* (1960), 497–520.

Monique Laurent. 2001. A comparison of the Sherali-Adams, Lovsz-Schrijver and Lasserre relaxations for 0-1 programming. *Mathematics of Operations Research* 28 (2001), 470–496.

D. T. Lee and Arthur K. Lin. 1986. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory* 32, 2 (1986), 276–282.

Bennett Liles. 2005. PTZ cameras. *Sound and Video Contractor* 23, 6 (June 2005), 46–51.

Junbin Liu, Clinton Fookes, Tim Wark, and Sridha Sridharan. 2012. On the statistical determination of optimal camera configurations in large scale surveillance networks. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part I (ECCV'12)*. Springer-Verlag, Berlin, 44–57.

Junbin Liu, Sridha Sridharan, Clinton Fookes, and Tim Wark. 2014. Optimal camera planning under versatile user constraints in multi-camera image processing systems. *IEEE Transactions on Image Processing* 23, 1 (Jan. 2014), 171–184.

Liang Liu, Xi Zhang, and Huadong Ma. 2010. Optimal node selection for target localization in wireless camera sensor networks. *IEEE Transactions on Vehicular Technology* 59, 7 (Sept. 2010), 3562–3576.

Aaron Mavrinac and Xiang Chen. 2013. Modeling coverage in camera networks: A survey. *International Journal of Computer Vision* 101, 1 (2013), 205–226.

C. McCool, V. Chandran, S. Sridharan, and C. Fookes. 2008. 3D face verification using a free-parts approach. *Pattern Recognition Letters* 29, 9 (2008), 1190–1196.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics* 21, 6 (1953), 1087–1092.

Anurag Mittal and Larry Davis. 2004. *Visibility Analysis and Sensor Planning in Dynamic Environments Computer Vision - ECCV 2004*. Lecture Notes in Computer Science, Vol. 3021. Springer Berlin/Heidelberg, 175–189.

Anurag Mittal and Larry S. Davis. 2008. A general method for sensor planning in multi-sensor systems: Extension to random occlusion. *International Journal of Computer Vision* 76, 1 (2008), 31–52.

Siti Kamaliah Mohd Yusoff, Abas Md Said, and Idris Ismail. 2011. Optimal camera placement for 3D environment. In *Software Engineering and Computer Systems*, Jasni Mohamad Zain, Wan Maseribt Wan Mohd, and Eyas El-Qawasmeh (Eds.). Communications in Computer and Information Science, Vol. 180. Springer, Berlin, 448–459.

Eduardo Monari and Kristian Kroschel. 2009. A knowledge-based camera selection approach for object tracking in large sensor networks. In *3rd ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC'09)*. 1–8.

Eduardo Monari and Kristian Kroschel. 2010. Dynamic sensor selection for single target tracking in large video surveillance networks. In *2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 539–546.

Yacine Morsly, Nabil Aouf, Mohand Said Djouadi, and Mark Richardson. 2012. Particle swarm optimization inspired probability algorithm for optimal camera network placement. *IEEE Sensors Journal* 12, 5 (2012), 1402–1412.

Yacine Morsly, Mohand Said Djouadi, and Nabil Aouf. 2010. On the best interceptor placement for an optimally deployed visual sensor network. In *2010 IEEE International Conference on Systems Man and Cybernetics (SMC)*. 43–51.

Seyed Akbar Mostafavi and Mehdi Dehghan. 2011. Optimal visual sensor placement for coverage based on target location profile. *Ad Hoc Networks* 9, 4 (2011), 528–541. Multimedia Ad Hoc and Sensor Networks.

Vikram P. Munishwar and Nael B. Abu-Ghazaleh. 2010. Scalable target coverage in smart camera networks. In *4th ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC'10)*. ACM, New York, NY, 206–213.

Vikram P. Munishwar and Nael B. Abu-Ghazaleh. 2011. Target-oriented coverage maximization in visual sensor networks. In *Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access (MobiWac'11)*. ACM, New York, NY, 175–178. DOI:http://dx.doi.org/10.1145/2069131.2069164

Vikram P. Munishwar and Nael B. Abu-Ghazaleh. 2013. Coverage algorithms for visual sensor networks. *ACM Transactions on Sensor Networks* 9, 4, Article 45 (July 2013), 36 pages. DOI:http://dx.doi.org/10.1145/2489253.2489262

Alan T. Murray, Kamyoung Kim, James W. Davis, Raghu Machiraju, and Richard Parent. 2007. Coverage optimization to support security monitoring. *Computers, Environment and Urban Systems* 31, 2 (2007), 133–147.

Prabhu Natarajan, Pradeep K. Atrey, and Mohan Kankanhalli. 2015. Multi-camera coordination and control in surveillance systems: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications* 11, 4, Article 57 (June 2015), 30 pages. DOI:http://dx.doi.org/10.1145/2710128

Prabhu Natarajan, Trong Nghia Hoang, Kian Hsiang Low, and Mohan Kankanhalli. 2012. Decision-theoretic coordination and control for active multi-camera surveillance in uncertain, partially observable environments. In *2012 6th International Conference on Distributed Smart Cameras (ICDSC)*. 1–6.

Prabhu Natarajan, Trong Nghia Hoang, Yongkang Wong, Kian Hsiang Low, and Mohan Kankanhalli. 2014. Scalable decision-theoretic coordination and control for real-time active multi-camera surveillance. In *International Conference on Distributed Smart Cameras (ICDSC'14)*. ACM, New York, NY, Article 38, 6 pages.

Brad Nelson and Pradeep K. Khosla. 1994. Integrating sensor placement and visual tracking strategies. In *IEEE International Conference on Robotics and Automation, 1994*, Vol. 2. 1351–1356.

Gustavo Olague and Roger Mohr. 2002. Optimal camera placement for accurate reconstruction. *Pattern Recognition* 35, 4 (2002), 927–944.

Joseph O'Rourke. 1987. *Art Gallery Theorems and Algorithms*. Oxford University Press, Inc., 282 pages.

Siva Ram, K. R. Ramakrishnan, P. K. Atrey, V. K. Singh, and M. S. Kankanhalli. 2006. A design methodology for selection and placement of sensors in multimedia surveillance systems. In *4th ACM International Workshop on Video Surveillance and Sensor Networks*. ACM, 121–130.

Sumantra Dutta Roy, Santanu Chaudhury, and Subhashis Banerjee. 2004. Active recognition through next view planning: A survey. *Pattern Recognition* 37, 3 (2004), 429–446.

Stefan Rudolph, Sarah Edenhofer, Sven Tomforde, and Jörg Hähner. 2014. Reinforcement learning for coverage optimization through PTZ camera alignment in highly dynamic environments. In *International Conference on Distributed Smart Cameras (ICDSC'14)*. ACM, New York, NY, Article 19, 6 pages.

David Ryan, Simon Denman, Clinton Fookes, and Sridha Sridharan. 2011. Textures of optical flow for real-time anomaly detection in crowds. In *2011 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*. 230–235.

Davide Scaramuzza. 2014. Omnidirectional camera. In *Computer Vision*, Katsushi Ikeuchi (Ed.). Springer US, 552–560.

William R. Scott, Gerhard Roth, and Jean-Francois Rivest. 2003. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys* 35, 1 (March 2003), 64–96.

Sabesan Sivapalan, Daniel Chen, Simon Denman, Sridha Sridharan, and Clinton Fookes. 2011. Gait energy volumes and frontal gait recognition using depth images. In *2011 International Joint Conference on Biometrics (IJCB)*. 1–6.

Petr Slavik. 1997. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms* 25, 2 (1997), 237–254.

L. Snidaro, R. Niu, P. K. Varshney, and G. L. Foresti. 2003. Automatic camera selection and fusion for outdoor surveillance under changing weather conditions. In *IEEE Conference on Advanced Video and Signal Based Surveillance, 2003*. 364–369.

Stanislava Soro and Wendi Heinzelman. 2007. Camera selection in visual sensor networks. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS'07)*. 81–86.

Peter Sturm, Srikumar Ramalingam, Jean-Philippe Tardif, Simone Gasparini, and João Barreto. 2011. Camera models and fundamental concepts used in geometric computer vision. *Foundations and Trends in Computer Graphics and Vision* 6, 1 (Jan. 2011), 1–183.

Tomokazu Takahashi, Osanori Matsugano, Ichro Ide, Yoshito Mekada, and Hiroshi Murase. 2006. Planning of multiple camera arrangement for object recognition in parametric eigenspace. In *18th International Conference on Pattern Recognition (ICPR'06)*, Vol. 1. 603–606.

Konstantinos A. Tarabanis, Peter K. Allen, and Roger Y. Tsai. 1995. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation* 11, 1 (1995), 86–104.

Konstantinos A. Tarabanis, Roger Y. Tsai, and Peter K. Allen. 1994. Analytical characterization of the feature detectability constraints of resolution, focus, and field-of-view for vision sensor planning. *CVGIP: Image Understanding* 59, 3 (1994), 340–358.

Anton van den Hengel, Rhys Hill, Ben Ward, Alex Cichowski, Henry Detmold, Chris Madden, Anthony Dick, and John Bastian. 2009. Automatic camera placement for large scale surveillance networks. In *2009 Workshop on Applications of Computer Vision (WACV)*. 1–6.

Lieven Vandenberghe and Stephen Boyd. 1996. Semidefinite programming. *SIAM Review* 38 (1996), 49–95.

Bang Wang. 2011. Coverage problems in sensor networks: A survey. *ACM Computing Surveys* 43, 4, Article 32 (Oct. 2011), 53 pages.

Hanbiao Wang, G. Pottie, K. Yao, and D. Estrin. 2004. Entropy-based sensor selection heuristic for target localization. In *3rd International Symposium on Information Processing in Sensor Networks (IPSN'04)*. 36–45.

Qiang Wang, Jing Wu, and Chengnian Long. 2013. On-line configuration of large scale surveillance networks using mobile smart camera. In *2013 7th International Conference on Distributed Smart Cameras (ICDSC)*. 1–6. DOI:http://dx.doi.org/10.1109/ICDSC.2013.6778238

Thomas Winkler and Bernhard Rinner. 2014. Security and privacy protection in visual sensor networks: A survey. *ACM Computing Surveys* 47, 1, Article 2 (May 2014), 42 pages. DOI:http://dx.doi.org/10.1145/2545883

Jingxin Xu, S. Denman, S. Sridharan, and C. Fookes. 2012. Activity analysis in complicated scenes using DFT coefficients of particle trajectories. In *IEEE 9th International Conference on Advanced Video and Signal-Based Surveillance (AVSS'12)*. 82–87.

Yichun Xu, Bangjun Lei, Shuifa Sun, Fangmin Dong, and Chilan Chai. 2010. Three particle swarm algorithms to improve coverage of camera networks with mobile nodes. In *2010 IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*. 816–820.

Yi-Chun Xu, Bangjun Lei, and EmileA. Hendriks. 2013. Constrained particle swarm algorithms for optimizing coverage of large-scale camera networks with mobile nodes. *Soft Computing* 17, 6 (2013), 1047–1057.

Yi-Chun Xu, Bangjun Lei, and Emile A. Hendriks. 2011. Camera network coverage improving by particle swarm optimization. *Journal on Image and Video Processing*, Article 3 (Jan. 2011), 10 pages.

K. Yabuta and H. Kitazawa. 2008. Optimum camera placement considering camera specification for security monitoring. In *IEEE International Symposium on Circuits and Systems (ISCAS'08)*. 2114–2117.

Yi Yao, Chung-Hao Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi. 2008. Sensor planning for automated and persistent object tracking with multiple cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 1–8.

Yi Yao, Chung-Hao Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi. 2010. Can you see me now? Sensor positioning for automated and persistent surveillance. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 40, 1 (2010), 101–115.

Mohamed Younis and Kemal Akkaya. 2008. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Networks* 6, 4 (2008), 621–655.

Jian Zhao. 2011. *Camera Planning and Fusion in a Heterogeneous Camera Network*. Ph.D. dissertation. University of Kentucky.

Jian Zhao and S. S. Cheung. 2009. Optimal visual sensor planning. In *IEEE International Symposium on Circuits and Systems (ISCAS'09)*. 165–168.

Jian Zhao and S. C. S. Cheung. 2007. Multi-camera surveillance with visual tagging and generic camera placement. In *1st ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC'07)*. 259–266.

Jian Zhao, Sen-Ching S. Cheung, and Thinh Nguyen. 2009. *Multi-Camera Networks: Concepts and Applications*. Elsevier, Chapter 6, Optimal Visual Sensor Network Configuration.

Pu Zhou and Chengnian Long. 2011. Optimal coverage of camera networks using PSO algorithm. In *2011 4th International Congress on Image and Signal Processing (CISP)*, Vol. 4. 2084–2088.

Received July 2014; revised January 2016; accepted January 2016