

Automated Placement of Cameras in a Floorplan to Satisfy Task-Specific Constraints

Ugur Murat Erdem and Stan Sclaroff
Department of Computer Science, Boston University
{merdem,sclaroff}@cs.bu.edu

Abstract

In many multi-camera vision systems the effect of camera locations on the task-specific quality of service is ignored. Researchers in Computational Geometry have proposed elegant solutions for some sensor location problem classes. Unfortunately, these solutions utilize unrealistic assumptions about the cameras' capabilities that make these algorithms unsuitable for many real-world computer vision applications: unlimited field of view, infinite depth of field, and/or infinite servo precision and speed. In this paper, the general camera placement problem is first defined with assumptions that are more consistent with the capabilities of real-world cameras. The region to be observed by cameras may be volumetric, static or dynamic, and may include holes that are caused, for instance, by columns or furniture in a room that can occlude potential camera views. A subclass of this general problem can be formulated in terms of planar regions that are typical of building floorplans. Given a floorplan to be observed, the problem is then to efficiently compute a camera layout such that certain task-specific constraints are met. A solution to this problem is obtained via binary optimization over a discrete problem space. In preliminary experiments the performance of the resulting system is demonstrated with different real floorplans.

1 Introduction

Computer vision in video sensor networks has become a very hot research topic in recent years. The practical need for such systems is increasing, while the associated hardware costs are decreasing. The aggregate video sensor network, depending on the specific system design and architecture, should be made fault-tolerant to camera “drop out” – for instance, the occasional failures of cameras, temporarily obstructed camera views, etc. As in cellular telephone networks, the aim is to have as much coverage as possible within a pre-defined region, with an acceptable level of quality-of-service. In video sensor networks, the layout of video sensors should assure a minimum level of image qual-

ity needed to satisfy certain task-specific requirements – for instance, sufficient image resolution, depth of field, servo speed for pan-tilt-zoom cameras, etc.

2 Related Work

In the field of Computational Geometry, extensive progress has been made in solving optimal “guard” location problems for a polygonal area. For instance, [16, 6, 11] give a nice introduction to the family of static guard problems known as *Art Gallery Problems* (AGPs). In the AGP, the task is to determine a minimal number of cameras and their static positions, such that all points in the polygon are observed. Even though efficient algorithms exist giving a lower bound for AGPs with simple polygons [16], the exact solution is proven to be NP-Hard.

A variant of the AGP is known as *Watchmen Tours* where guards are allowed to move inside the polygon [5, 9, 13]. The objective is to find an optimal number and route for guards guaranteeing the detection of some intruder with an unknown initial position and unlimited speed. Suzuki, *et al.* introduce another variant of watchman problem termed *boundary search* where guards are allowed to move only along the boundary of the polygon [17]. In a similar vein, *Floodlight Illumination Problems* deal with the illumination of planar regions by some light sources having prescribed angles [4, 10]. The AGP is an instance of the Floodlight Illumination Problem where light sources have 360° angles.

To our knowledge, the current solutions to the AGP and its variants employ unrealistic assumptions about the cameras' capabilities that make these algorithms unsuitable for many real-world computer vision applications: unlimited field of view, infinite depth of field, and/or infinite servo precision and speed. One main aim of our work is to bridge the existing gap between the highly theoretical, well-established computational geometry and more realistic requirements of computer vision with real video cameras.

Active Vision is the area of computer vision dealing with task-based vision and camera control [2]. An interesting problem of active vision is determining the *next best view* – finding the next optimal camera placement and orientation

given the acquired visual data history for the scene under exploration [1, 14].

Active vision methods have also been proposed for surveillance applications. The most basic surveillance task may be stated as detection and tracking of some targets in some area of interest. For instance [8, 3] use a peripheral sensor to detect the position of a moving object(s) and drive a foveal sensor to gather detailed images of the target(s). Mikic, *et al.* [15] build a camera network for an intelligent room with static and active cameras. They also employ orientation-based active camera selection criteria.

There is also related work with robot motion control for video surveillance. For instance, Cortes, *et al.* [7] use gradient descent to compute optimal locations for mobile sensing networks given some utility function over a convex polygon.

It is important to note that in these active vision systems no consideration is given to the *off-line* selection and placement of the cameras to improve the *on-line* system performance. The system in this paper aims to find a camera layout for a region of interest, that satisfies certain task-specific constraints.

3 Problem Definition

In this paper we pose the problem of optimal camera placement for a given region and vision task. We focus on the camera placement problem, where the goal is to determine optimal positioning and number of cameras for a region to be observed, given a set of task-specific constraints, and a set of possible cameras to use in the layout. This camera placement takes place *off-line*, for cameras that will be mounted on surfaces in an area of interest to support the task-specific requirements of *on-line* computer vision systems. In the most general (and most challenging) case, the region to be observed by cameras may be an arbitrary volumetric shape. It may be an open space or a delimited environment, or a blend of both, *i.e.*, outdoors *vs.* indoors. The region may include holes that are caused, for instance, by columns and trees, or furniture in a room that can obstruct potential camera views. It may contain an arbitrary number of static and/or dynamic objects. Furthermore, the region itself may change in time, *i.e.*, furniture or walls may be added, removed, or moved in a floor plan. Finally, one can choose from an arsenal of different types of cameras that could be used in satisfying the requirements for the specified video sensing task(s).

3.1 Cameras

For the sake of completeness we first outline some optical camera parameter definitions. We then describe three major video camera types employed in surveillance and compare

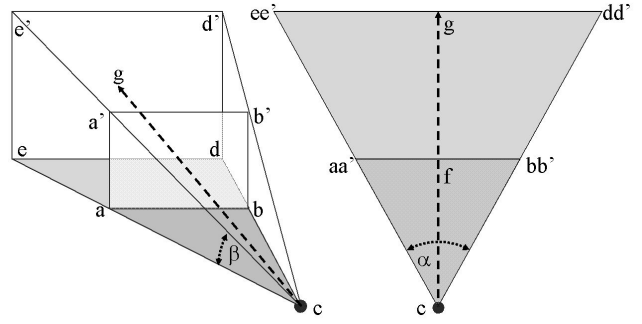


Figure 1: Field of View and Depth of Field. α and β are respectively azimuth and latitude of the Field of View. c is the camera, \overline{cg} is the optical axis, the frustum defined by the planes $abde$ and $a'b'd'e'$ is the Depth of Field.

them with respect to their optical parameters. All video camera sensors observe visible spectra. Two crucial parameters for the current work are:

- **Field of View (FoV):** The maximum volume visible from a camera. As shown in Fig. 1, the FOV is determined by the apex angles (azimuth and latitude) of the pyramidal visible region emanating from the camera. This pyramid is also known as the *viewing frustum*, and can be skewed by oblique projection.
- **Depth of Field (DoF):** Depth of field is the amount of distance between the nearest and farthest objects that appear in acceptably sharp focus in an image, as shown in Fig. 1. It is determined by the camera's aperture and the focal length.

There are many types of video cameras available. They differ in the sensor element, lens type, servo capabilities, etc. The following three are frequently used in computer vision research and applications:¹

- **Fixed Perspective Camera:** Once mounted in place, these cameras have a fixed position, orientation, and focal length.
- **PTZ (Pan-Tilt-Zoom) Camera:** These cameras can rotate around their horizontal (Tilt) and vertical (Pan) axis using servos. Some also have an adjustable focal length (Zoom) limited by some range. They are mounted in a fixed position in the environment.
- **Omnidirectional Camera:** These cameras have 360° horizontal FoV, as opposed to a pyramidal FoV. Despite their FoV range, they may suffer from lens ab-

¹One additional camera type not included in this list is a mobile camera (mounted on a moving platform or robot). This type is not included, because we focus on the *off-line* sensor layout problem.

beration effects due to small focal length and/or convex mirrors used in the setup [3, 8].

Let the vector π_i represent the parameters that define camera i . It will contain two sub-vectors: $\pi_i^{\mathbf{I}}$, the intrinsic parameters like focal length, and $\pi_i^{\mathbf{E}}$, the extrinsic parameters that define the location and orientation of the camera with respect to world coordinate system.

Furthermore we will refer to the layout phase of the vision system as *off-line* and to the runtime phase as *on-line*. For all three camera types the location parameters are variable during the *off-line* phase, i.e. we can place them freely. Table 1 shows a more structured comparison of the three camera types. We label as ‘‘OFF’’ any parameters that are adjustable *off-line* but must remain fixed during *on-line* phase. We label as ‘‘ON’’ any parameters adjustable during the *on-line* phase.

TYPE	ZOOM	FOV	DoF	ORIENT.	LOCATION
Fixed	OFF	OFF	OFF	OFF	OFF
PTZ	ON	ON	ON	ON	OFF
Omni	OFF/ON	OFF	OFF/ON	OFF	OFF

Table 1: A comparison of the three basic video camera types considered in our problem.

3.2 Camera Placement Problem

Camera placement is an optimization problem by definition. Let \mathbf{V} be an arbitrary connected volumetric region. If \mathbf{V} is not connected then its connected parts can be treated as individual regions. Let \mathcal{T} be the given task and let \mathcal{C} be the set containing all the constraints required by \mathcal{T} . These may include spatial (i.e., coverage) constraints of \mathbf{V} , temporal (i.e., foveation) constraints for active cameras, quality-of-service (i.e., resolution) constraints, etc. The challenge is to find where to place minimal number of cameras in \mathbf{V} satisfying \mathcal{C} . This can be stated in a more compact form as:

$$\underset{\Pi}{\operatorname{argmin}} \quad |\mathbf{Z}| \text{ subject to } \mathcal{C} \text{ given } \mathbf{V} \quad (1)$$

where $\Pi = \{\pi_1 \dots \pi_{|\mathbf{Z}|}\}$, \mathbf{Z} is the set of cameras to be placed in \mathbf{V} and $|\cdot|$ is the cardinality operator. Note that this definition is an abstraction and different problem instances can be obtained by plugging-in different constraints, objectives and tasks.

Let us give two problem instances that are consistent with the definition.

Problem 1: Given a volumetric area \mathbf{V} , find the minimum cardinality and placement for a camera set \mathbf{Z} such that $\forall p \in \mathbf{V}$ is visible from some camera $c_i \in \mathbf{Z}$ in less than time T . In a more compact form:

$$\underset{\Pi}{\operatorname{argmin}} \quad |\mathbf{Z}| \text{ s.t. } \forall p \in \mathbf{V} \exists c_i : \Lambda(\pi_i, p) \leq T \quad (2)$$

where $c_i \in \mathbf{Z}$, and $\Lambda(\pi_i, p)$ gives the maximum time required to foveate camera c_i with parameters π_i on point p .

Problem 2: Given a volumetric area \mathbf{V} , find the minimum cardinality and placement for a camera set \mathbf{Z} such that $\forall p \in \mathbf{V}$ can be put in DoF of some camera $c_i \in \mathbf{Z}$ with some minimum spatial resolution.

$$\underset{\Pi}{\operatorname{argmin}} \quad |\mathbf{Z}| \text{ s.t. } \bigcup_{i=1}^{|\mathbf{Z}|} \Omega(\pi_i, r) = \mathbf{V} \quad (3)$$

where $\Omega(\pi_i, r) = \{p \in \mathbf{V} : \text{point } p \text{ is visible from camera } c_i \text{ with spatial resolution greater than } r\}$.

These are only a few examples of problem instances of the general camera placement problem given in Eqn. 1.

3.3 Problem Simplification

Although the discovery of an algorithm that can solve the most general case of the camera layout problem for a given volume of interest is highly-desirable, it may prove quite challenging if not intractable. We therefore focus on a more manageable subclass of this general problem that can be formulated in terms of planar regions that are typical of a building floor plan, e.g., Fig. 2. We will then approximate the region by a polygon. This is a valid assumption since most buildings and floor plans consist of polygonal shapes or can be approximated by a collection of polygons. Given a floor plan to be observed, the problem is then to efficiently compute a camera layout such that certain task-specific constraints are met. As will be shown, a solution to this problem can be obtained via binary optimization over a discrete problem space.

There exist efficient algorithms developed in computational geometry for operations involving polygons, such as convexity determination, area finding, triangulation, etc. In this work we will also assume \mathbf{V} to be a simple polygon. A polygon \mathbf{P} is said to be simple if the only points of the plane belonging to two polygon edges of \mathbf{P} are the polygon vertices. A simple polygon can not have holes and it has a well defined interior and exterior region. These properties allow us to have efficient algorithms including a linear time visibility algorithm [12] which computes the visible subregion of \mathbf{P} from a point p inside \mathbf{P} .

4 Approach

The main idea is to convert the constraint set \mathcal{C} and the task \mathcal{T} to satisfy the following canonical 0-1 optimization model

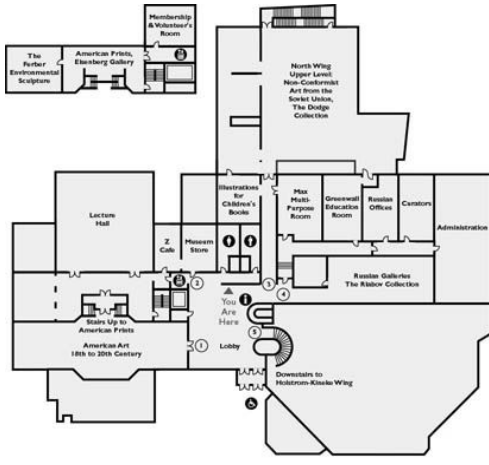


Figure 2: A typical floor plan

[18]:

$$\min \mathbf{c}\mathbf{x} \text{ s.t. } \mathbf{A}\mathbf{x} \geq \mathbf{b} \quad \mathbf{x} \in \{0, 1\} \quad (4)$$

where \mathbf{A} is an $m \times n$ matrix whose i^{th} row elements are coefficients of i^{th} linear inequality constraint, \mathbf{b} is a $m \times 1$ vector whose i^{th} element is the right-hand-side coefficient of constraint i , \mathbf{c} is $1 \times n$ vector whose i^{th} element is the cost associated with i^{th} element of \mathbf{x} which is a $n \times 1$ vector containing n decision variables.

In the most general sense, the constraints given by \mathbf{A} and \mathbf{b} define a convex polytope $Poly$ in n dimensional space which contains all the feasible solution points for the given optimization problem. The 0-1 programming tries to find a binary vector \mathbf{x}^* which gives the minimum cost function value overall $Poly$. Let Q be the set containing all possible binary combinations of \mathbf{x} . Let $Q^* \subset Q$ containing only the elements of Q found inside $Poly$. Then 0-1 programming can be explained as *looking for $\mathbf{x}^* \in Q^*$ giving the minimum cost function value.*

In our approach we first sample Π over a feasible interval constrained by the polygon \mathbf{P} , the camera specifications and the task requirements, for example over location, orientation, focal length, DoF, FoV etc. For each sample point we find the associated grid occupancy vector for the polygon \mathbf{P} . Note that if we let each sample point's associated grid occupancy vector be a column of \mathbf{A} and the grid occupancy vector of \mathbf{P} be \mathbf{b} , then the $Poly$ defined by $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ contains all the feasible combinations of sample points satisfying full coverage of \mathbf{P} . Then we may apply 0-1 programming to find the best set of sample points giving the minimum cost value. In other words the solution to the 0-1 model (Eqn. 4) constructed as explained is then the solution to our original camera location problem.

It is important to note that the solution of the discrete problem will depend on the number of samples, the sam-

pling method and the resolution of the grid occupancy which are input parameters to the algorithm. If sampled uniformly, the higher the number of sample points is, the closer is the solution of the discrete optimization to the continuous (global) optimal. Similarly, the higher the resolution of the grid occupancy, the closer is the solution to the continuous one. However, generally relatively lower density samples are sufficient to obtain a near-optimal solution.

We obtain the 0-1 model representation from a given camera location problem following these steps:

Step 1 Find a representation for the given constraints as a spatial coverage. This is the most crucial phase of the solution. If there exists a way to represent \mathcal{C} as a spatial coverage problem then it is also possible to solve it using the proposed method. Solutions for **Problem 1** and **Problem 2** differ mainly in this representation. These will be given in detail in the section 5.

Step 2 Represent the polygonal region \mathbf{P} as an occupancy grid. Let $\mathbf{OG}(\mathbf{P})$ be the $h \times w$ binary matrix whose $(i, j)^{\text{th}}$ element is 1 if grid cell p with coordinates (j, i) is inside \mathbf{P} and 0 otherwise. Let us call $\mathbf{OG}(\mathbf{P})$ the occupancy grid of \mathbf{P} . Note that $h \times w$ is directly proportional to the resolution of the occupancy grid and it is an input parameter for the algorithm.

Step 3 Choose n samples from Π given the camera specs and \mathbf{P} . Let \mathbf{s}_k be the k^{th} sample. Note that n is an input parameter for the algorithm. For instance, depending on the task constraints, Π can be sampled over different focal lengths (multiple camera lenses), different camera orientations, locations, aperture etc.

Step 4 For each \mathbf{s}_k find the occupancy of its spatial coverage representation given \mathcal{C} . Let \mathbf{S}_i be the $h \times w$ binary matrix whose $(i, j)^{\text{th}}$ element is 1 if cell grid p with coordinates (j, i) is inside the spatial coverage of \mathbf{s}_k and 0 otherwise.

Step 5 Construct the 0-1 model (Eqn. 4). Let \mathbf{A} be:

$$\begin{aligned} \mathbf{A}_{(m=h \times w, n)} &= \{\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_i \dots \mathbf{a}_n\} \\ \mathbf{a}_i &= \{\mathbf{q}_1^i \mathbf{q}_2^i \dots \mathbf{q}_j^i \dots \mathbf{q}_w^i\}^T \\ \mathbf{q}_j^i &= \{j^{\text{th}} \text{ column of } \mathbf{S}_i\}^T \end{aligned}$$

Let \mathbf{b} be:

$$\begin{aligned} \mathbf{b}_{(m=h \times w, 1)} &= \{\mathbf{e}_1 \mathbf{e}_2 \dots \mathbf{e}_j \dots \mathbf{e}_w\}^T \\ \mathbf{e}_j &= \{j^{\text{th}} \text{ column of } \mathbf{OG}(\mathbf{P})\}^T \end{aligned}$$

Let $\mathbf{c} = \{c_1 c_2 \dots c_j \dots c_n\}$ where c_j is the cost associated with the camera location \mathbf{s}_j . This may be the price, required bandwidth, consumed energy etc. of

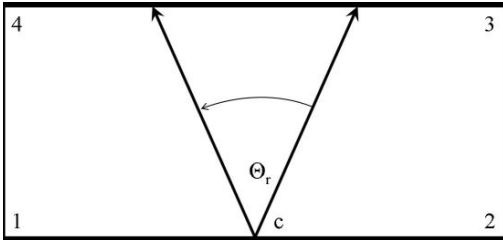


Figure 3: Illustration of the reachable region in the rectangle 1234 from camera location c .

the camera. When $c = \mathbf{1}_{1 \times n}$ the solution is for the minimum number of cameras.

At this point we have all the necessary components of Eqn. 4. The last step is to solve this model using one of the well-known methods (in our case “Branch-and-Bound” [18]). Let us denote the optimal solution of the model with \mathbf{x}^* . Note that the decision variable vector \mathbf{x}^* is also an indicator vector, i.e. if $x_i^* = 1$ then the camera location s_i is one of the optimal camera locations for the given problem instance. If $\mathbf{x}^* = \text{infeasible}$ then there is no camera location configuration satisfying \mathcal{C} given the current sample set s .

4.1 Correctness

The i^{th} constraint in Eqn. 4 is:

$$\mathbf{A}_{i,1} \times x_1 + \mathbf{A}_{i,2} \times x_2 + \dots + \mathbf{A}_{i,n} \times x_n \geq b_i \quad (5)$$

Note that $\mathbf{A}_{i,j}$ represents the coverage status of the *same* grid cell by j^{th} π sample: it is 1 if occupied, 0 otherwise. b_i represents the occupancy of the *same* grid cell by the polygon \mathbf{P} in the same way. So, the constraint in Eqn. 5 guarantees that the grid cell represented by b_i is covered by *at least* 1 camera. Since this is true for all $w \times h$ constraints, model 4’s feasible region is all the possible combinations $C(n, j)$ $j = 1 \dots n$ of s which cover \mathbf{P} given \mathcal{C} . Then the solution of this model is the combination of s which covers \mathbf{P} and gives the minimum value for the objective function $\mathbf{c}\mathbf{x}$.

5 Experiments

In this section we implement Problems 1 and 2 defined previously and give solutions using the proposed approach. We demonstrate the system using four floorplans taken from Fig. 2, using real camera specifications to determine the sampling in II. More specifically we will show how to represent the constraints of the problem definitions as area coverage constraints (Step 1 of our algorithm). The remaining steps will be the same for all problems.

For each implementation we choose the grid resolution ($w \times h$ in *Step 2*) as 100×100 . The number of sample points chosen is different for each floor-plan. The algorithm is implemented using MATLAB 6.5 R13 on a Pentium III 800 MhZ computer with 512MB memory. The minimum running time for the given experiments is 23 seconds and the maximum is 2 minutes 11 seconds. The main bottleneck for running time is our custom implementation of Branch-and-Bound algorithm which can be highly optimized.

Problem 1: Suppose the cameras are PTZ. All PTZ cameras have servo speed limitations. Let us denote the maximum horizontal angular speed of a given PTZ camera with v_h . Recall that T is the time constraint (Eqn. 2). Assume that cameras can only be placed along the perimeter of \mathbf{P} . Consider the worst case scenario. Suppose at some given point in time camera c is foveated towards the minimum angle given its location. If c is located along an edge E , then its orientation corresponding to its minimum horizontal angle will be along E . Let E be parallel to x -axis without loss of generality. Then the minimum horizontal angle will be 0. Camera c can then foveate up to orientation $\theta_1 = T \cdot v_h$. So, it can foveate at any direction $\theta \leq \theta_1$ without violating T . Now consider the other extreme, c pointing to its maximum horizontal angle, π . It can foveate down to orientation $\theta_2 = \pi - T \cdot v_h$ in time T for the same reason. The intersection of the two regions formed by orientations spanning $[0, \theta_1]$ and $[\theta_2, \pi]$ is the feasible coverage for the worst case scenario and it is defined by $\theta_r = \theta_1 + \theta_2 - \pi$ [Fig. 3].

Now we are able to represent the constraint defined by time threshold T as a coverage constraint. The camera to be placed is Sony EVI-D30 PTZ camera with $v_h = 80^\circ/\text{sec}$. Let $T = 1.5 \text{ sec}$, then

$$\begin{aligned} \theta_1 &= \theta_2 = T \cdot 80^\circ/\text{sec} = 120^\circ. \\ \theta_r &= \theta_1 + \theta_2 - 180^\circ = 60^\circ \end{aligned}$$

We sample five uniform camera locations per edge on the polygon \mathbf{P} . The solutions for four floorplans taken from Fig. 2 are shown in Fig. 4.

Problem 2: In this problem the constraint is on the spatial resolution. We want to be able to servo and focus some camera c_i to put some point $p \in \mathbf{P}$ in c_i ’s DoF with spatial resolution greater than r . Note that there is no constraint on foveation time. Suppose we want to place PTZ cameras. There is a range of focal lengths achievable by a particular camera. This corresponds to a corresponding feasible range of DoF for a camera c_i . The loci of this DoF range form a circular band around the center of projection of c_i . There is also a maximum focal length and distance from center of projection corresponding to the required minimum spatial resolution r . The loci of points falling inside this maximum distance form a disk centered at the camera location. The

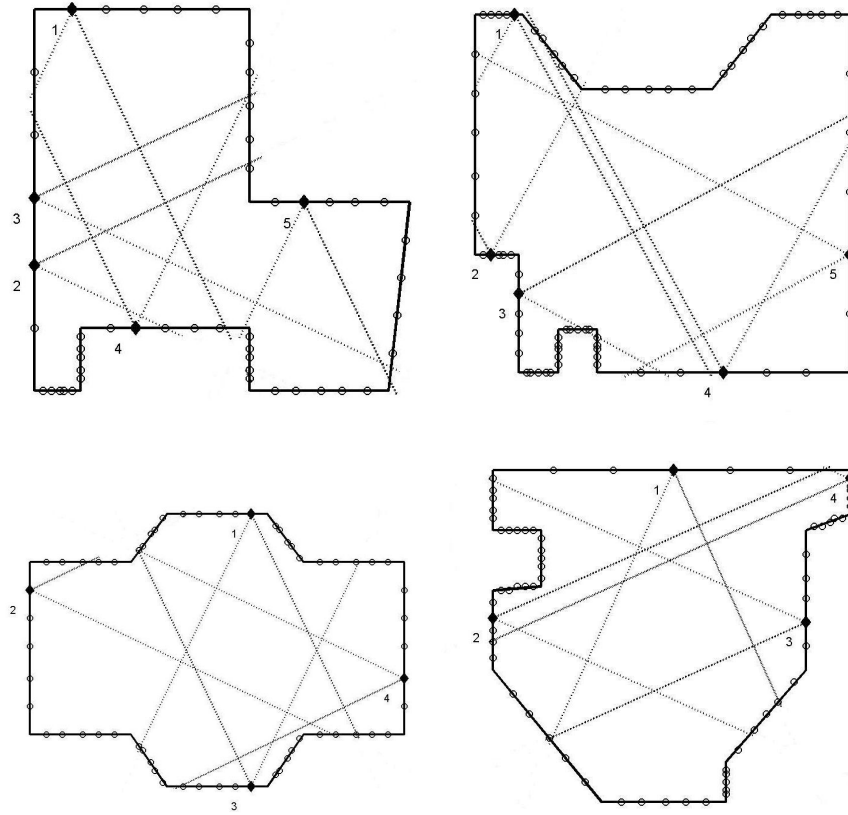


Figure 4: Solution for problem 1 with $T = 1.5 \text{ sec}$ and $v_h = 80^\circ/\text{sec}$ using four real floorplans taken from Fig. 2. Circles are sampled points, filled markers are the optimal locations and dashed lines represent reachable regions for each optimal camera location. The number of samples are five per edge for all polygons.

feasible region covered by c_i given r then becomes the intersection of the DoF circular band and the disk. Suppose the minimum spatial resolution is at 5000mm from the camera with a 35mm focal length. Running our algorithm with these parameters produces the camera locations that satisfy these constraints for the layouts in Fig. 5.

6 Conclusion

We formulate a general task-based camera location problem and give three instances along with a solution based on discrete solution space, polygonal area and binary optimization technique. We also show that if the task based constraints of the vision system are reducible to area coverage then these constraints may also be satisfied by the solutions of our proposed method.

The results in the preliminary study are encouraging. In the immediate future we plan to conduct further experiments on a wider range of problem instances including (a) the Art Gallery Problem with fixed focal length lenses

common of inexpensive surveillance systems and (b) surveillance with mixed camera setups that include omnidirectional and foveating PTZ cameras. Performance analysis and increased efficiency of the proposed method are also our short-term goals. In the-long term our main goal is to search for a method using continuous solution space as opposed to the discrete one, giving globally optimal solutions with less overhead.

References

- [1] Tal Arbel and Frank P. Ferrie. Entropy-based gaze planning. *Image and Vision Computing*, 19(11):779–786, September 2001.
- [2] Ruzena Bajcsy. Active perception. In *Proceedings of the IEEE*, pages 996–1005, 1988.
- [3] Jorge Batista, Paulo Peixoto, and Helder Araujo. Real-time active surveillance by integrating peripheral motion detection with foveated tracking. In *IEEE Work-*

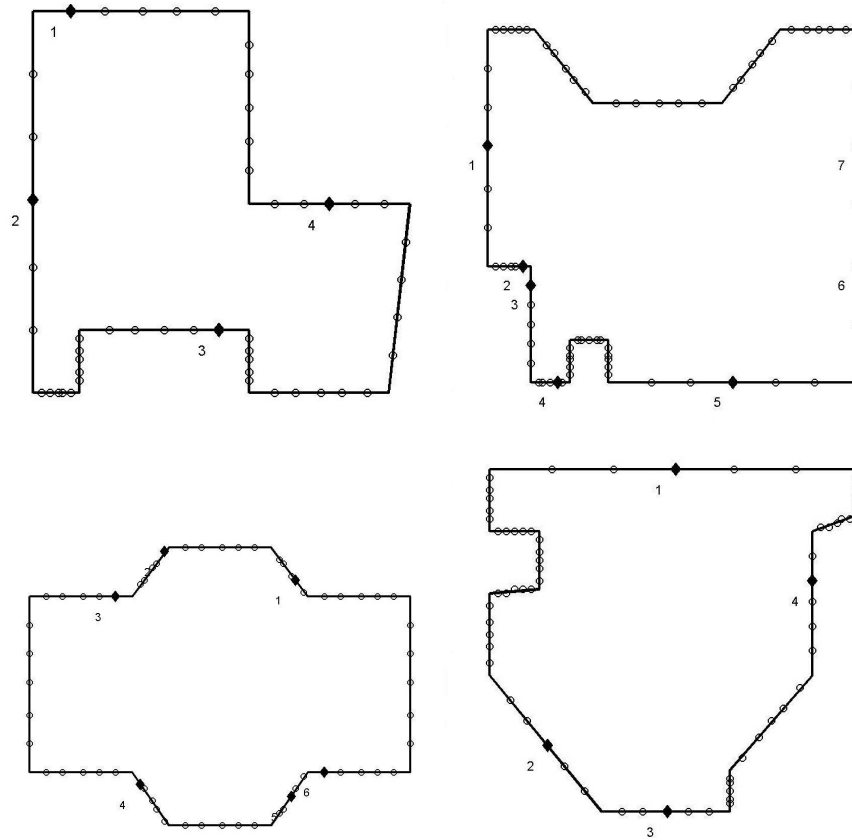


Figure 5: Solution for problem 2 with maximum distance 5000mm using four real floorplans. Circles are sampled points, filled markers are the optimal locations. The number of samples are five per edge for all polygons.

shop on *Visual Surveillance*, Monbay, India, January 1998.

- [4] Prosenjit Bose, Leonidas J. Guibas, Anna Lubiw, Mark H. Overmars, Diane L. Souvaine, and Jorge Urrutia. The floodlight problem. *International Journal of Computational Geometry and Applications*, 7(1/2):153–163, 1997.
- [5] Svante Carlsson, Bengt J. Nilsson, and Simeon C. Ntafos. Optimum guard covers and m-watchmen routes for restricted polygons. In *Workshop on Algorithms and Data Structures*, pages 367–378, 1991.
- [6] Vasek Chvatal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory Series*, 18:39–41, 1975.
- [7] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. In *IEEE Conference on Robotics and Automation*, pages 1327–1332, Arlington, VA, May 2002.

[8] Y. Cui, S. Samasekera, Q. Huang, and M. Greiffenhagen. Indoor monitoring via the collaboration between a peripheral sensor and a foveal sensor. In *IEEE Workshop on Visual Surveillance*, pages 2–9, Monbay, India, January 1998.

[9] Alon Efrat, Leonidas J. Guibas, Sarel Har-Peled, David C. Lin, Joseph S. B. Mitchell, and T. M. Murali. Sweeping simple polygons with a chain of guards. In *Symposium on Discrete Algorithms*, pages 927–936, 2000.

[10] Vladimir Estivill-Castro, Joseph O’Rourke, Jorge Urrutia, and Dianna Xu. Illumination of polygons with vertex lights. *Information Processing Letters*, 56(1):9–13, 1995.

[11] S. Fisk. A short proof of Chvatal’s watchman theorem. *Journal of Combinatorial Theory Series*, 24:374, 1978.

- [12] H. El Gindy and D. Avis. A linear algorithm for computing the visibility polygon from a point. *Journal of Algorithms*, 2:186–197, 1981.
- [13] Leonidas J. Guibas, Jean-Claude Latombe, Steven M. LaValle, David Lin, and Rajeev Motwani. A visibility-based pursuit-evasion problem. *International Journal of Computational Geometry and Applications*, 9(4/5):471–, 1999.
- [14] J. Maver and R. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Trans. Pattern Anal. Machine Intell.*, 15(5):417–433, May 1993.
- [15] I. Mikic, K. Huang, and Mohan M. Trivedi. Activity monitoring and summarization for an intelligent meeting room. In *Workshop on Human Motion*, pages 107–112, 2000.
- [16] Joseph O’Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, New York, 1987.
- [17] Ichiro Suzuki, Yuichi Tazoe, Masafumi Yamashita, and Tiko Kameda. Searching a polygonal region from the boundary. *International Journal of Computational Geometry and Applications*, 11(5):529–553, 2001.
- [18] Laurence A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.