Optimum Camera Placement Considering Camera Specification for Security Monitoring

Kenichi Yabuta and Hitoshi Kitazawa
Department of Electrical and Electronic Engineering, Tokyo University of Agriculture and Technology,
2–24–16, Naka-cho, Koganei-shi, Tokyo, 184–8588 Japan
Email: kyabuta@m.ieice.org, kitazawa@cc.tuat.ac.jp

***Abstract*— We present an optimum camera placement algorithm. We are motivated by the fact that the installation of security cameras is increasing rapidly. From the system cost point of view, it is desirable to observe all the area of interest by the smallest number of cameras. We propose a method for deciding optimum camera placement automatically considering camera specification such as visual distance, visual angle, and resolution. Moreover, to reduce the number of cameras, we divide the scene into regions and weight them according to their importance of observation. Then, we calculate camera locations which cover all the important regions and as many other regions as possible by considering the trade off between a number of observed regions and the number of cameras required. We formulate this problem as a set covering problem.**

I. Introduction

Public video surveillance by security camera is becoming more and more important for investigation and deterrent of crimes. Therefore, the installation of security camera is increasing in both indoor and outdoor environments such as streets, parks, buildings, and stores. Consequently, there arises a new problem of deciding position, direction, and visual angle of camera, in order to cover all the area by a minimum number of cameras.

This problem is well known as the *Art Gallery Problem (AGP)* [1]. However, it is assumed that the sensor has unlimited visibility, that is, infinite visual distance and 360-degree visual angle in the AGP. In fact, visibility of camera is limited by its visual distance and visual angle. A. T. Murray et al. have proposed new methods to calculate the optimum camera placement using the set covering problem [2], [3]. However, in their works, camera positions and pan-tilt-zoom parameters are decided separately. Moreover, the camera placement locations are limited.

In this paper, we propose a new algorithm for optimum camera placement. We automate solving the camera placement considering camera specification by formulating a set covering problem. Our algorithm does not limit the camera locations in order to indoor surveillance, because the cameras can be installed anywhere including ceilings. Moreover, we reduce processing time and number of cameras by weighting necessary regions for surveillance.

II. Flow of Camera Placement Algorithm

Our algorithm consists of "scene segmentation into regions," "visibility test," "set covering problem formulation," and "solving set covering problem" as shown in Fig 1. These processes are explained in detail in the following sections.

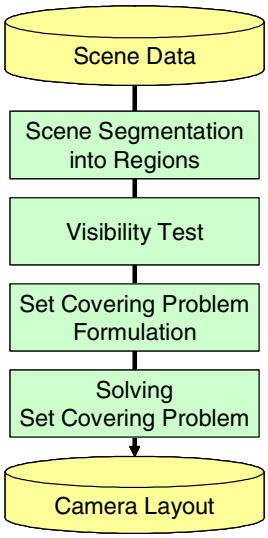

Fig. 1. Flow of proposed optimum camera placement algorithm.

III. Scene Segmentation into Regions

The scene to be observed is divided into rectangular regions. We show the algorithm of scene segmentation in this section. Figure 2 shows the process flow.

a) Data input: First, we input the top view scene data. The scene data consist of area data and block data. We approximate area and blocks by two dimensional rectangles, in this paper. Figure 2 (a) shows an input scene.

b) Line extension: We segment regions by extending peripheral lines of input blocks. Both horizontal and vertical lines are extended to the end of area as shown in Fig. 2 (b). We call the generated rectangle a "*region*," in this paper. If a region is larger than predefined value, the region is subdivided.

c) Region merge: In order to reduce the computational time, the unnecessarily divided regions are merged. As shown in Fig. 2 (b) and (c), if both ends of the separating line L between two regions do not connect to a block corner, we remove the separating line L and merge the two regions. However, if the width or height of the merged region becomes larger than the predefined value this process is avoided.

978-1-4244-1684-4/08/$25.00 ©2008 IEEE

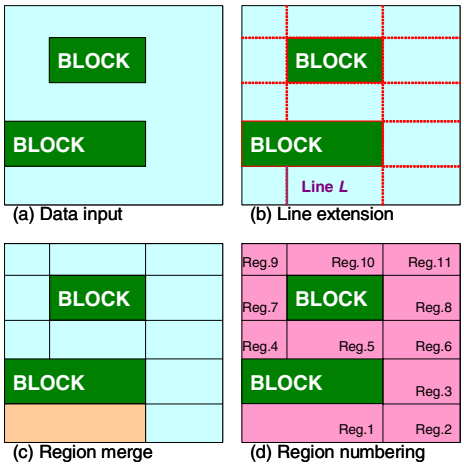

Fig. 2. Region segmentation process. (a) Data input, (b) Line extension, (c) Region merge, (d) Region numbering.

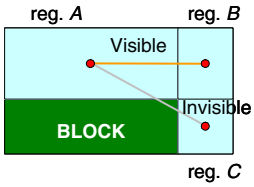

Fig. 3. The geometrical visibility analysis.

d) Region numbering: All regions are numbered sequentially as shown in Fig. 2 (d).

IV. Visibility Test

The visibility between each region, which is called "*geometrical visibility*" in this paper, is analyzed. The geometrical visibility is decided by the line segment connecting region centers as shown in Fig. 3. The line segment *AB* crosses no block. Therefore, region *A* is visible from region *B* and vice versa. The line segment *AC* crosses a block. Therefore, region *A* is invisible from region *C* and vice versa. We analyze this geometrical visibility between all combinations of two regions. In this paper, the geometrical visibility is decided only by center points of regions. If we increase the number of test points and put the test points not only on the center of regions but also on the other positions just like the corners, we can enhance the precision of visibility.

A camera is placed at the center of a region directing a certain direction. In addition, a camera has its own visual angle. It is possible that a camera have all combination of three parameters, that is, location, direction, and visual angle. We call this potential camera "camera candidate." The camera direction is rotated step by step with a given angle value. When a camera candidate is placed as shown in Fig. 4, the observed regions are parts of geometrical regions which are visible from the camera candidate. We call these specific regions visible from a camera candidate "*camera visibility*."

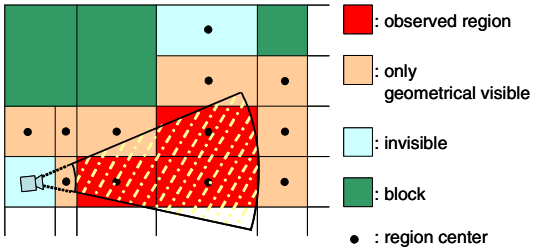

Fig. 4. Observed regions from camera candidate.

V. Camera Placement

In this section, we formulate a set covering problem to solve the camera placement problem. We use CPLEX [4] solver to solve the set covering problem based on a linear programming relaxation scheme. Notation for stating the problem formally is as follows:

$$\begin{aligned} i &= \text{index of camera candidates, } i = 1, 2, \ldots, n \\ j &= \text{index of observed regions, } j = 1, 2, \ldots, m \end{aligned}$$

$$x_i = \begin{cases} 1 & \text{If camera candidate } i \text{ is placed.} \\ 0 & \text{otherwise.} \end{cases}$$

$$p_{ij} = \begin{cases} 1 & \text{If camera candidate } i \text{ can observe region } j. \\ 0 & \text{otherwise.} \end{cases}$$

A. All Region Observation (ARO)

First, we consider the simplest condition. The number of cameras is minimized on condition that all regions should be observed. The linear programming equations are as follows:

Minimize

$$Z = \sum_{i=1}^{n} x_i \tag{1}$$

Subject To:

$$\sum_{i=1}^{n} p_{ij} x_i \geq 1 \quad \forall j \tag{2}$$

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

The objective 1 is to minimize the number of placed cameras. Constraints 2 require all regions are observed by at least one camera. Constraints 3 specify that the x_i is binary.

B. Weighted Region Observation (WRO)

By using ARO algorithm, a large number of cameras are required to cover all the regions for the complicated scene. In security monitoring, we need to select regions which should be observed for efficient monitoring. This region is called "*essential region*" in this paper. We can manually select the essential regions according to the importance for surveillance such as doorway, front of showcase. However, we automate selecting essential regions by tracing the flow of person for simplifying experiments in this paper. In order to trace the flow of person, it is sufficient to observe "*cross-regions*" where a

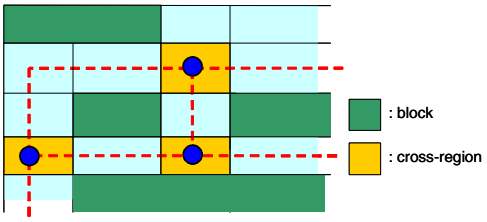

Fig. 5. Result of thinning regions and detecting cross-region.

pass forked into several regions. We detect the cross-regions by using thinning algorithm as shown in Fig. 5. The red lines are the result of thinning region, and the blue round nodes are the intersections of the lines. The region which includes the node is cross-region, that is, essential region.

The cameras should observe all essential regions and as many other regions as possible. If the number of observed regions can be increased by M by adding a new camera, the camera is added. Here, we introduce new parameters s and y, which are

$$s_j = \begin{cases} 1 & \text{If region } j \text{ is essential region.} \\ 0 & \text{otherwise.} \end{cases}$$

$$y_j = \begin{cases} 1 & \text{If region } j \text{ is not essential region} \\ & \text{and region } j \text{ is observed.} \\ 0 & \text{otherwise.} \end{cases}$$

The liner programming equations are as follows:

Minimize

$$Z = \sum_{i=1}^{n} x_i - w_c \sum_{j=1}^{m} (1 - s_j)\, y_j \tag{4}$$

Subject To:

$$\sum_{i=1}^{n} p_{ij} x_i \geq 1 \text{ if } s_j = 1 \tag{5}$$

$$-y_j + \sum_{i=1}^{n} p_{ij} x_i \geq 0 \text{ if } s_j = 0 \tag{6}$$

$$x_i \in \{0,1\} \quad \forall i \tag{7}$$

$$y_j \in \{0,1\} \quad \forall j \tag{8}$$

The objective 4 is to minimize number of placed cameras and maximize number of observed regions. The addition of cameras is controlled by the constant w_c. If the number of new observed regions is increased by M, a new camera is added. On the other hand, if it is less than M, no camera is added. Therefore, the constant w_c should satisfy $1 - w_c \times M < 0$ and $1 - w_c \times (M-1) > 0$. Hence, the w_c is defined as,

$$w_c = \frac{1}{M} + w\prime, \tag{9}$$

where $w\prime$ is a small constant. In our experiment, the M is 5, and the constant $w\prime$ is 0.001, thus the constant w_c is 0.201.

TABLE I
CAMERAS SPECIFICATION.

	visual angle [deg]	visual distance [m]
camera 1	15	3–38
camera 2	30	2–19
camera 3	45	1–12
camera 4	70	1–7
camera 5	90	1–5

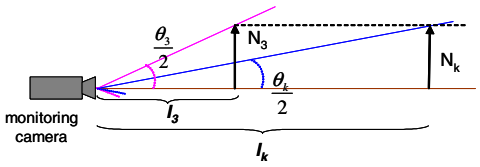

Fig. 6. Decision of camera specification. l is visual distance, θ is visual angle, and N is resolution.

Constraints 5 require if region j is essential region, at least one camera should observe it. Constraints 6 allow the y_j becomes 1, if camera candidate i covers region j, and it is not essential region. Constraints 7 and 8 specify that the x_i and the y_j are binary, respectively.

VI. EXPERIMENTAL RESULTS

In this section, experimental results of proposed camera placement algorithm are shown. The experimental data is $20[m] \times 15[m]$ and it includes 20 blocks. The number of regions is 107 as the result of scene segmentation.

In this experiment, we use 5 types of cameras. Table I shows the specification of cameras. When camera type is single, only *camera 3* is used. The visual distance $l_3 = 12$, and the visual angle $\theta_3 = 45$. In order to keep the same resolution at the farthest position, the visual distance of other cameras are calculated by

$$\frac{l_3}{N_3} \tan\frac{\theta_3}{2} = \frac{l_k}{N_k} \tan\frac{\theta_k}{2} \; (k = 1, 2, 4, 5), \tag{10}$$

where N is a resolution: number of pixels in a line. In this experiment, all cameras have an identical resolution.

Table II shows the results of the number of cameras and processing time on a Pentium 4 3GHz PC. The camera layout is calculated by ARO and WRO, and the camera type is single and multiple. The step of rotation is 5 degrees for the camera visibility analysis in Section IV. In the case of WRO, the number of cameras was decreased nearly 50% compared with that of ARO. Moreover, when camera type is multiple, the number of cameras can be decreased, in spite of increasing the number of covered regions. Figures 7 (a) and 7 (b) demonstrate the camera layout of ARO-single algorithm and WRO-multiple algorithm, respectively.

Figure 8 shows correlation of processing time with number of regions. The set covering problem has exponential complexity. However, in order to estimate execution time, we

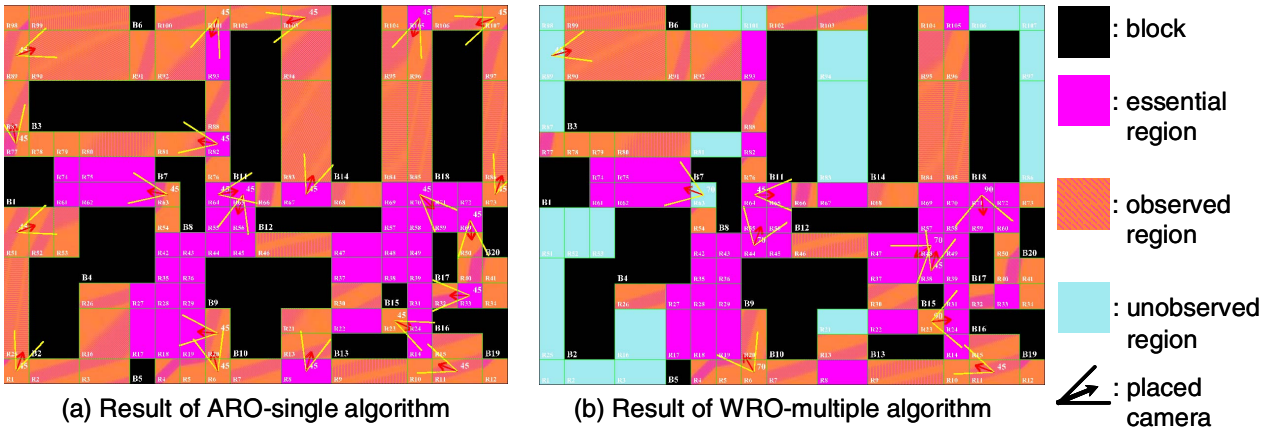

Fig. 7. Result of optimum camera placement. (a) ARO-single algorithm, (b) WRO-multiples algorithm.

TABLE II
OPTIMUM CAMERA PLACEMENT RESULTS: NUMBER OF OBSERVED REGIONS, NUMBER OF CAMERA CANDIDATE, NUMBER OF PLACED CAMERAS, AND PROCESSING TIME.

	Camera type	Observed regions	Camera candidate	Placed cameras	Processing time [sec]
ARO	single	107	7704	22	0.500
ARO	multiple	107	38520	20	1.063
WRO	single	82	7704	11	0.328
WRO	multiple	85	38520	10	0.844

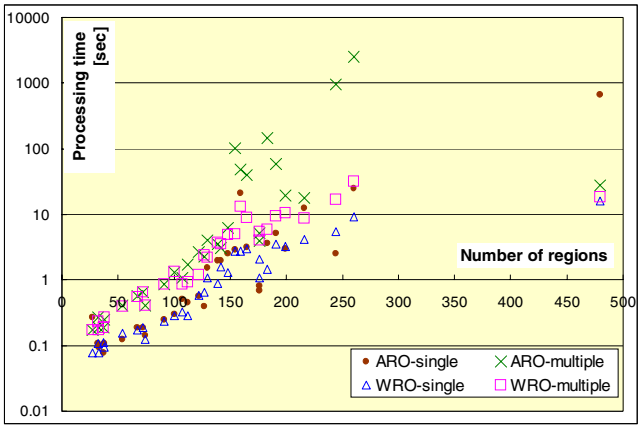

Fig. 8. Correlation of processing time with number of regions.

calculate the average complexity. The average complexities of WRO-multiple algorithm and ARO-multiple algorithm are approximately $O\left(m^{2.12}\right)$ and $O\left(m^{3.01}\right)$, respectively, where m is the number of regions. From table II, it is seen that WRO decreases processing time considerably.

Figure 9 shows the number of placed cameras in each four algorithm. According to the average result, the WRO-multiple algorithm can reduce the number of placed cameras by 60%

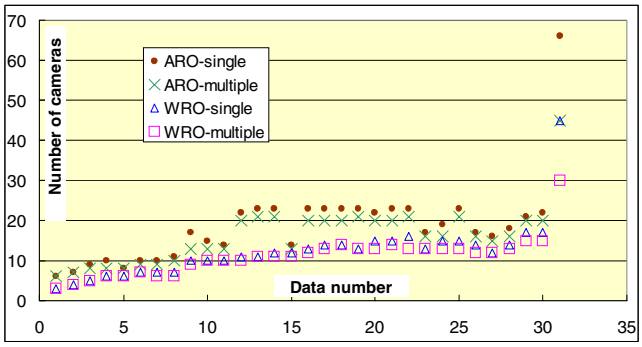

Fig. 9. Number of placed cameras in each four algorithm.

comparing ARO-single algorithm. The number of cameras is further reduced by using multiple types of cameras.

VII. CONCLUSION

We have presented a novel algorithm for an optimum security camera placement considering camera specification. Our algorithm automatically decides position, direction, and visual angle of cameras for efficient monitoring. To reduce the number of required cameras, our algorithm weights observed regions and allocates multiple types of cameras. Future work should focus on enhancing this algorithm to optimize the camera locations considering network and power supply connections.

REFERENCES

[1] J. O'Rourke, *Art Gallery Theorems and Algorithms*, ser. The International Series of Monographs on Computer Science. New York, NY: Oxford University Press, 1987.
[2] A. T. Murray, K. Kim, J. W. Davis, R. Machiraju, and R. Parent, "Coverage optimization to support security monitoring," *Computers, Environment and Urban Systems*, vol. 31, no. 2, pp. 133–147, 2007.
[3] F. Janoos, R. Machiraju, R. Parent, J. W. Davis, and A. Murray, "Sensor configuration for coverage optimization for surveillance applications," in *Proc. SPIE*, vol. 6491, 2007.
[4] (2007) The ILOG CPLEX website. [Online]. Available: http://www.ilog.com/products/cplex/