MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Sensor Placement Tool for Rapid Development of Video Sensor Layouts

Garaas, T.W.

TR2011-020 April 2011

Abstract

The arrangement of video sensors in closed-circuit television (CCTV) systems, for instance can have drastic effects on the efficiency and cost of the final system. In the present work, I describe a tool designed for rapid construction of simulated video sensor layouts that allows quantification of sensor coverage and cost estimation to be determined prior to installation; thus, avoiding costly changes during or after the installation. Most previous work in this area either considers sensor coverage only in a 2D space or requires significant preparation to achieve accurate results in 3D. In the present work, I describe an implementation of a novel coverage-analysis algorithm that uses the geometry of image formation to cast rays from simulated video sensors through the monitored area to estimate sensor coverage at every 3D location. Visualization techniques of the acquired sensor coverage data are additionally presented.

Simulation for Archietecture and Urban Design

Sensor Placement Tool for Rapid Development of Video Sensor Layouts

Tyler W. Garaas

Mitsubishi Electric Research Laboratories
201 Broadway, Cambridge, MA, USA, 02139
garaas@merl.com

Keywords: sensor placement, coverage problem, security monitoring, visibility analysis, ray casting

Abstract

The arrangement of video sensors – in closed-circuit television (CCTV) systems, for instance – can have drastic effects on the efficiency and cost of the final system. In the present work, I describe a tool designed for rapid construction of simulated video sensor layouts that allows quantification of sensor coverage and cost estimation to be determined prior to installation; thus, avoiding costly changes during or after the installation. Most previous work in this area either considers sensor coverage only in a 2D space or requires significant preparation to achieve accurate results in 3D. In the present work, I describe an implementation of a novel coverage-analysis algorithm that uses the geometry of image formation to cast rays from simulated video sensors through the monitored area to estimate sensor coverage at every 3D location. Visualization techniques of the acquired sensor coverage data are additionally presented.

1. INTRODUCTION

The use of closed-circuit television (CCTV) systems, on which I focus in the current work, has seen a rapid increase in the past decade. For instance, an estimated 4.2 million CCTV cameras are deployed in the UK alone as of 2004 (McCahill et al. 2004). Aside from the privacy and effectiveness conversations that are now in the spotlight due to the massive deployment of CCTV systems in public spaces, novel research regarding sensor placement, networking, and event detection are rapidly becoming more commonplace.

When designing a CCTV system, or indeed any video sensor layout, many factors need to be considered. As the monitoring region expands to cover greater and greater area – sometimes reaching neighborhood- or city-wide coverage – these factors can become critical in controlling cost and ensuring the effectiveness of the CCTV system. For instance, issues such as spatial resolution and angular coverage of certain 3D locations can mean the difference between successful deployment of a face-recognition algorithm, and egregious overlap in sensor coverage can increase system cost dramatically.

1.1. Previous Work

Research related to coverage analysis has roots in a computational geometry problem commonly referred to as *the art gallery problem*. In the art gallery problem, the goal is to place a minimum number of guards such that all wall-positions are observable by at least one guard; see O'Rourke (1987) for a review.

Most of the research regarding the art gallery problem and its variants has ignored certain, significant real-world constraints inherent in the placement of video sensors. As such, some researchers have extended the art gallery problem to include such considerations as limited field of view, visibility, and spatial resolution. Erdem and Scarloff (2004) present an algorithm for the placement of static and active video sensors that incorporates such considerations.

While the large majority of work related to video sensor coverage and placement has focused on 2D spaces (e.g., Erdem and Scarloff, 2004; Yabuta and Kitazawa, 2008), some have recently considered the problem in 3D spaces. Murray et al. (2007) describe a process for placing sensors while considering occlusion in 3D, but base their work on the assumed existence of a visibility algorithm. Becker et al. (2009) describe a method for placing cameras in a 3D environment based on the visibility of individual 3D locations. However, this work does not consider the quantification of visibility, but treats coverage of the locations binarily as either visible or not visible. Finally, van den Hengel (2009) present a method for placement of video sensors in a 3D environment based on the visibility of 'marker locations' that meet a minimum criterion – e.g., number of pixels that observe the marker. This work also does not describe quantitative coverage analysis – aside from the definition of a minimum criterion – and, furthermore, the 3D model must be manually annotated with meta-data, which appears to be a slow process.

Perhaps the previous work most closely related to the present work, is a commercial product known as VideoCAD (http://www.cctvcad.com), which includes many functionalities to aid in the creation of CCTV systems. However, while VideoCAD does provide an interface for designing CCTV layouts, it does not allow quantitative analysis of sensor coverage.

1.2. Overview

In the present work, I describe *Sensor Placement Tool* (SPtool, a non-commercial, internal tool used by Mitsubishi Electric), which facilitates rapid construction of 3D environments and effective video sensor layouts. To start, users construct a 3D representation of the environment they wish to monitor by adding and manipulating 3D models using GUI controls common in 3D modeling and CAD software. Since 3D modeling software has become commonplace through products such as Google Sketchup, Blender, and 3DS Max among many others, details regarding the construction of 3D environments are omitted here. Figure 1 illustrates the standard layout and GUI components of SPtool as well as a sample environment constructed with the software.

Once constructed, users add and orient virtual video sensors within the 3D environment. Analysis and visualization of the sensor coverage – a measure indicating how much or how well the environment is observed by the sensor(s) – is automatically performed to aid users in determining the most effective sensor layout.

In the following section, I describe the coverage analysis and visualization algorithms implemented in SPtool, and in Section 3, I provide a discussion of SPtool and potential extensions for making the techniques discussed here more applicable to sensor placement tasks in general.

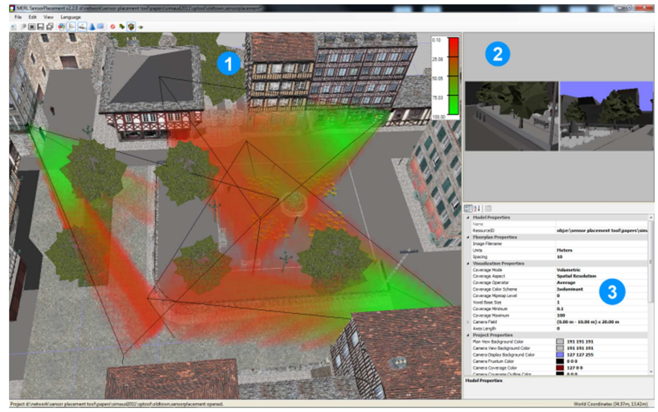

Figure 1. Illustration of the SPtool GUI and the components that comprise it. Component 1 is the primary view window, where users interact with the environment and video sensors. Component 2 provides a virtual view from sensors within the environment. Finally, component 3 is a properties grid, where users can customize the parameters of selected 3D models, selected video sensors, environment parameters, or the sensor coverage analysis and visualization.

2. CONSTRUCTING THE SENSOR LAYOUT

Following construction of the environment, users place and orient video sensors in the environment using simple GUI operations. Parameters of the camera(s) (e.g., sensor width or image dimensions) and lens (e.g., focal lengths) are manipulated using the properties grid.

2.1. Coverage Computation

Previous approaches to computing coverage by a set of video sensors have largely computed coverage as the percentage of a 2D floorplan observed by one or more sensors. This can lead to drastic errors between the computed coverage and the intuitive sense of coverage due to such considerations as partial occlusions (e.g., cubicles) or monitoring height (e.g., it may be critical that the sensors selectively observe the faces of people in the monitored area). As such, I have developed a method to compute 3D sensor coverage that automatically handles such considerations, which I briefly describe in the following paragraphs.

Figure 2 illustrates the rough steps of the algorithm used in the present work to compute sensor coverage. The first step in coverage analysis, is to compute a bounding box that entirely encloses the target environment (Figure 2a; environment geometry is omitted for clarity). The bounding box is then divided into a regular grid of voxels (Figure 2b; voxel sizes are exaggerated for clarity), which store raw sensor coverage data for computing quantitative measures of sensor coverage. In order to determine the raw data stored in the voxel, I employ a straightforward ray-casting algorithm based on the geometry of image formation.

As is done with traditional ray-casting rendering techniques, a single ray is projected (Figure 2c), for each pixel, from the focus of the sensor, through the environment, to the point where it either intersects an object in the environment or exits the bounding box. If the ray were part of a rendering algorithm, it would perform scattering and integration of light to determine each pixel's final color value. However, it is only necessary here to determine which voxels it traverses (Figure 2d) before either intersecting an object or exiting the environment. Many algorithms exist for tracing a ray through a regularly sampled volume (e.g., Cleary and Wyvill 1988). A nice side effect of using this algorithm for computing coverage, is that occlusions are implicitly handled, as contributing rays are eliminated as they come into contact with objects in the environment.

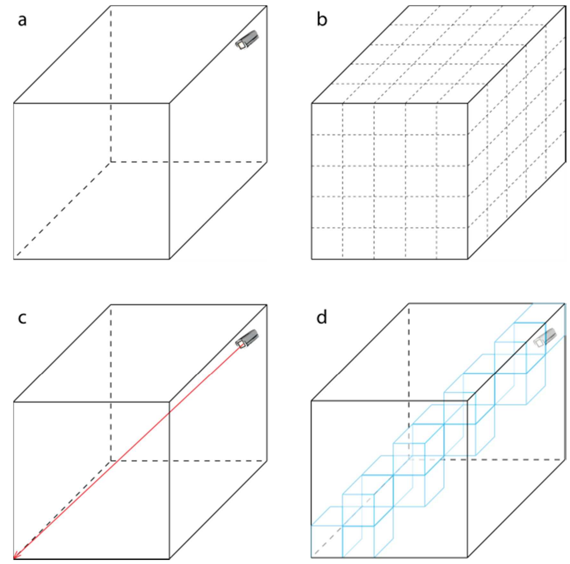

Figure 3. Sequence of steps comprising coverage analysis in SPtool. (a) A bounding box of environment, which contains a single video sensor, is computed first. (b) The bounding box is divided into a regular grid of voxels, which store the raw coverage data. (c, d) Rays from video sensors are traversed through the voxels, contributing to data stored in each voxel they intersect.

2.2. Coverage Visualization

Raw coverage data stored in each voxel are then used to compute a coverage measure indicating how well the voxel is covered by the sensors in the layout. In particular, SPtool includes three such measures described individually below.

- **Spatial Resolution** – number of rays intersecting a voxel; analogous to the number of pixels that observe the voxel.
- **Camera Count** – number of cameras that have at least one pixel observing the voxel.
- **Angular Coverage** – range of angles from which the voxel is observed; see Huang and Tseng (2003) for related work.

Once the coverage has been quantized into one of these measures, it can be mapped into a normalized value by having the user supply the minimum and maximum acceptable coverage (i.e., values normalized to the range [0-1] using this min and max are considered acceptable coverage). To indicate the level of coverage to the user, I employ two general techniques. In the simplest technique, voxels within the acceptable range are rendered as semi-transparent, color-mapped boxes, which I will refer to as the *volumetric* visualization. An alternative, *projective* visualization technique involves creating a coverage overlay, where the color that is mapped to the overlay at each individual pixel indicates the level of coverage along that pixel's line of sight.

Each of these visualizations has strengths and weaknesses when conveying sensor coverage to the user. The volumetric visualization conveys sensor coverage of individual voxels within the environment. Generally, the user can manipulate the view to determine the level of coverage for all 3D locations; however, this can be quite time consuming and often, this level of detail is not needed. In contrast, the projective visualization combines the coverage from multiple voxels along a single line of sight to create a synthetic overlay. This visualization can be used to confer a simple, high-level overview of the level of coverage achieved over a large area, and is best used when viewing the environment from afar – particularly useful for orthographic projections of the environment.

3. DISCUSSION

SPtool provides a useful interface for accomplishing one primary task, quickly developing effective video sensor layouts for a target environment. A user familiar with SPtool can build a moderately complex environment and video sensor layout and estimate its effectiveness (eg, coverage of monitored areas) and cost in well under one

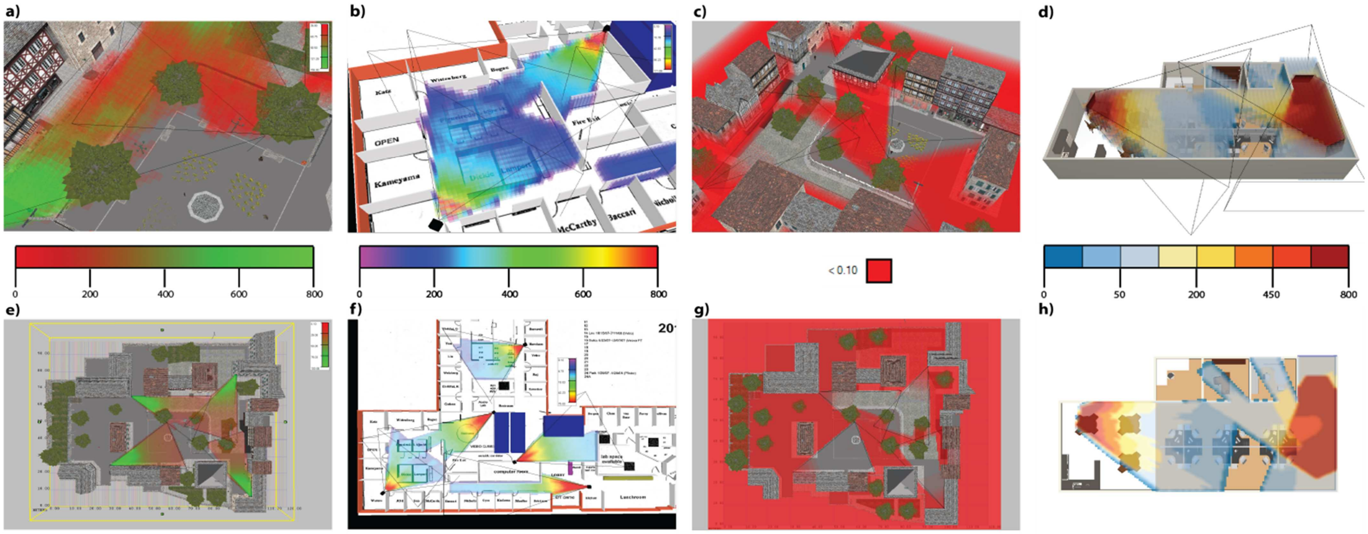

Figure 3. Examples of 3D environments and sensor coverage produced in SPtool. a-d) Volumetric visualization of coverage. e-h) Projective visualization of coverage. Panels **c** and **g** illustrate 'holes' in the coverage, where coverage does not meet the minimum coverage level set by the user.

hour. Increases in the required level of accuracy and environment detail will, of course, affect development time.

In general, though, the techniques described here are not limited to surveillance networks alone or to the implementation described here. For instance, the algorithm for determining sensor coverage presented here is applicable to any sensor that can have its sensing elements reasonably extended into space. For instance, SPtool has additionally been used for creating LIDAR sensor layouts in addition to traditional video sensor layouts, and may be applicable to pervasive sensors, such as passive infrared sensors.

Furthermore, the methods presented here provide an excellent starting point for constructing more complex surveillance tools. For instance, in previous work regarding coverage analysis using 2D representations of the environment, (Erdim and Scarloff 2004; Murray et al. 2007; Yabuta and Kitazawa 2008), the computed coverage was used to compute an optimal sensor layout without user intervention. Since these algorithms operate over a 2D space, the computed sensor layouts can only be considered an approximation. Voxel representations of coverage, such as the one employed in the present work, form an ideal basis over which coverage, which is already stored per voxel, can be optimized using only modest revisions to the 2D optimization algorithms.

References

BECKER, E., GUERRA-FILHO, G. AND MAKEDON, F. 2009. AUTOMATIC SENSOR PLACEMENT IN A 3D VOLUME. *WORKSHOP ON APPLICATIONS OF COMPUTER VISION (WACV).*

CLEARY, J. G. AND WYVILL, G. 1988. ANALYSIS OF AN ALGORITHM FOR FAST RAY TRACING USING UNIFORM SPACE SUBDIVISION. *THE VISUAL COMPUTER 4*, 65-83.

ERDEM, U. M. AND SCARLOFF, S. 2004. AUTOMATED CAMERA LAYOUT TO SATISFY TASK-SPECIFIC AND FLOORPLAN-SPECIFIC COVERAGE REQUIREMENTS. *COMPUTER VISION AND IMAGE UNDERSTANDING 103*, 156-169.

VAN DER HENGEL, A., HILL, R., WARD, B., CICHOWSI, A., DETMOLD, H., MADDEN, C., DICK, A. AND BASTIAN, J. 2009. AUTOMATIC CAMERA PLACEMENT FOR LARGE SCALE SURVEILLANCE NETWORKS. *2ND INTERNATIONAL CONFERENCE ON PERVASIVE TECHNOLOGIES RELATED TO ASSISTIVE ENVIRONMENTS (PETRA).*

HUANG, C. AND TSENG, Y. 2003. THE COVERAGE PROBLEM IN A WIRELESS SENSOR NETWORK. IN *ACM INTERNATIONAL WORKSHOP ON WIRELESS SENSOR NETWORKS AND APPLICATIONS (WSNA)*, 115-121.

MCCAHILL, M., NORRIS, C., AND WOOD, D. 2002. EDITORIAL. THE GROWTH OF CCTV: A GLOBAL PERSPECTIVE ON THE INTERNATIONAL DIFFUSION OF VIDEO SURVEILLANCE IN PUBLICLY ACCESSIBLE SPACE. *SURVEILLANCE & SOCIETY 2*, 110-135.

MURRAY, A. T., KIM, K., DAVIS, J. W., MACHIRAJU, R., AND PARENT, R. 2007. COVERAGE OPTIMIZATION TO SUPPORT SECURITY MONITORING. *COMPUTERS, ENVIRONMENT AND URBAN SYSTEMS 31*, 133-147.

O'ROURKE, J. 1987. *ART GALLERY THEOREMS AND ALGORITHMS*. NEW YORK: OXFORD UNIVERSITY PRESS.

YABUTA, K. AND KITAZAWA, H. 2008. OPTIMUM CAMERAPLACEMENT CONSIDERING CAMERA SPECIFICATION FOR SECURITY MONTORING. *IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS*, 2114-211.