Sensor placement optimization in buildings

Simone Bianco, Francesco Tisato

DISCo - Department of Informatics, Systems and Communication
University of Milano-Bicocca,
Viale Sarca 336, ed. U14
20126 Milano, Italy

ABSTRACT

In this work we address the problem of optimal sensor placement for a given region and task. An important issue in designing sensor arrays is the appropriate placement of the sensors such that they achieve a predefined goal. There are many problems that could be considered in the placement of multiple sensors. In this work we focus on the four problems identified by Hörster and Lienhart.[1] To solve these problems, we propose an algorithm based on Direct Search, which is able to approach the global optimal solution within reasonable time and memory consumption. The algorithm is experimentally evaluated and the results are presented on two real floorplans. The experimental results show that our DS algorithm is able to improve the results given by the most performing heuristic introduced in.[1] The algorithm is then extended to work also on continuous solution spaces, and 3D problems.

Keywords: Sensor placement, sensor arrays, pose optimization

1. INTRODUCTION

In this work we address the problem of optimal sensor placement for a given region and task. The problem of sensor placement arises in several areas of computer vision, computer graphics, and robotics, such as localization, tracking, surveillance, object or environment reconstruction, and image based rendering. An important issue in designing sensor arrays is the appropriate placement of the sensors such that they achieve a predefined goal. There are many problems that could be considered in the placement of multiple sensors. In this work we focus on four problems identified by Hörster and Lienhart:[1]

- Given the number of sensors of one type and their specific parameters, determine their positions and poses in space such that coverage is maximized.
- Given several types of sensors, their parameters and specific costs as well as the maximum total price of the sensor array, determine the sensor types and positions/poses that maximize coverage in the given space.
- Given the fixed positions and respective types of a number of sensors determine their optimal poses such that coverage is maximized.
- Given a minimally required percentage of coverage determine the sensor array with minimum cost that satisfies this coverage constraint.

Although the most general case of the sensor layout problem for a given volume of interest is a 3D problem, we focus here on its 2D instantiation: the problem is formulated in terms of planar regions that are typical of a building floorplan. Given a floorplan, the problem is then to efficiently compute a sensor layout such that certain task-specific constraints are met. Different computational geometry algorithms exist if the floor plan is approximated as a collection of simple polygons.[2–4] However, in our work, we allow the polygon to have holes that represent potential visibility-occluding entities in the floorplan, e.g., columns, separator wall partitions, etc. Furthermore different classes of sensors may behave differently in presence of these entities. The visibility

Further author information: (Send correspondence to S.B.)
S.B.: E-mail: simone.bianco@disco.unimib.it, Telephone: +39 02 6448 7871

Image Processing: Machine Vision Applications V, edited by Philip R. Bingham, Edmund Y. Lam, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 8300, 830003 · © 2012 SPIE-IS&T · CCC code: 0277-786X/12/$18 · doi: 10.1117/12.911021

algorithms for simple polygons[4,5] are not applicable in the case of polygons with holes. To solve the four problems considered, we propose two different optimization-based algorithms: the former for discrete problem spaces, the latter for continuous ones. The terms discrete and continuous are referred to the parameter space. All the four problem considered can be formally formulated in a similar way. Let V be an arbitrary connected volumetric region. Let T be the given task and let C be the set containing all the constraints required by T. Let the vector P_i represent the parameters that define the sensor i. The problems can be thus formulated as:

$$\arg\min_{\mathbf{P}} F(\mathbf{P}) \text{ subject to } C \text{ given } V. \tag{1}$$

Where $\mathbf{P} = [P_1, \ldots, P_n]$, n is the number of sensors to be placed in V, and $F(\cdot)$ is the cost or objective function to be minimized. The different algorithms proposed are experimentally evaluated and compared to state of the art algorithms for camera placement[1,6] and experimental results are presented. The results show that the algorithms work well and are suited for different practical applications. Furthermore it is shown how the proposed algorithm can be easily extended to solve the more general 3D sensor layout problem.

2. THE OPTIMIZATION ALGORITHM

The proposed approach is based on Direct Search algorithm (DS) for the case of a continuous search space, and on an our extension of it to deal with discrete spaces. DS is a derivative-free method for solving optimization problems.[7] DS is based on a sequential examination of trial solutions involving comparison of each trial solution with the "best" obtained up to that time together with a strategy for determining what the next trial solution will be.[8]

Given an initial guess, the DS algorithm computes a sequence of points that approaches an optimal solution. At each step, the algorithm selects and compares a set of points (the *mesh*) around the current best point. The mesh is computed as a scalar multiple of a set of vectors (the *pattern*) centered at the current best point. If a point in the mesh that improves the objective function at the current best point is found, the new point becomes the current best point at the next step of the algorithm.

Formally, suppose that we want to solve an optimization problem of the form "minimize $f(x)$ subject to $x \in \Omega$", where Ω is the set of all possible solutions to our problem. At each iteration k of the implemented DS algorithm, we have the current iterate $\mathbf{x}_k$, a set D_k of n vectors which identify the search directions (i.e. the pattern), and a step-length parameter Δ_k. Usually the pattern D_k is the same for all iterations. For each direction $\mathbf{d}_{k,i} \in D_k$, $i = 1, \ldots, n$ we set $\mathbf{x}_i^+ = \mathbf{x}_k + \Delta_k \mathbf{d}_{k,i}$ and we examine $f(\mathbf{x}_i^+)$. If $\exists \mathbf{d}_{k,i} \in D_k : f(\mathbf{x}_i^+) < f(\mathbf{x}_k)$, we set $\mathbf{x}_{k+1} = \mathbf{x}_i^+$ and $\Delta_{k+1} = \alpha_k \Delta_k$; otherwise, we set $\mathbf{x}_{k+1} = \mathbf{x}_k$ and $\Delta_{k+1} = \beta_k \Delta_k$.

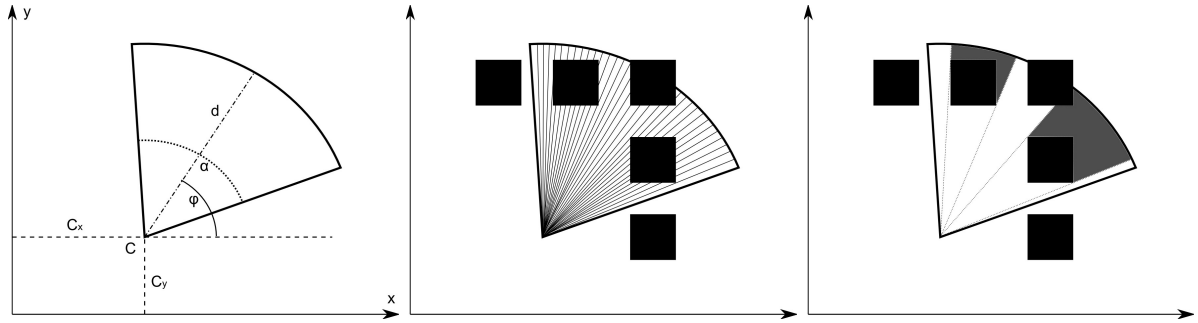
The main difference of the standard, continuous, implementation of the DS algorithm and our implementation to deal with discrete spaces is the use of the weights α_k and β_k. Usually the functions to be minimized take values in continuous spaces, and thus the weights are chosen as follows: $\alpha_k > 1$ and $0 < \beta_k < 1$. This means that the mesh expands when fitter solutions are found and contracts when they are not. In order to deal with functions taking value on an discrete domain, we have decided to choose the weights as follows: $\alpha_k = \alpha = 1$ and $\beta_k = \beta_{k-1} + 1$ with $\beta_0 = 1$. This means that when fitter solution are found, the mesh reduces to the pattern and when they are not the mesh expands. Thus, the neighborhood of the current solution is first explored and if no fitter solutions are found, the search area expands.

The pattern adopted is the simplest one: for the case of 2D floorplans, the pattern is $D_k = \{\pm i, \pm j\}$, where i and j are respectively the versors of the x and y directions. For the case of 3D floorplans, the pattern is $D_k = \{\pm i, \pm j, \pm k\}$, where i and j are as before, and k is the versor of the z direction.

3. THE VISIBILITY ALGORITHM

The sensors are modeled as having a sensing range (i.e. a maximum distance at which they can sense), and a sensing angle (i.e. they are directional). Each sensor can be then uniquely identified by the quintuplet $(c_x, c_y, \varphi, d, \alpha)$, where the couple (d, α) identifies the sensor characteristics, while the triplet (c_x, c_y, φ) identifies the sensor position and orientation (see Fig. 1(a)). In this work we will focus our attention on visual sensors, but the same methods apply to other classes of sensors too.

In order to solve the problems described in Sec. 1, for each possible camera we have to compute its field-of-view, taking into account the eventual static occlusions. To this end, we have developed an algorithm inspired by ray casting:[9] the sector representing the ideal field of view in absence of occlusions is sampled at equal angles and the rays are generated starting from the center $C = (c_x, c_y)$ (Fig. 1(b)). Each ray is then scanned from the center C of the sector to the arc, and we stop if we find an obstacle. The union of all the rays forms the camera field-of-view (Fig. 1(c)).



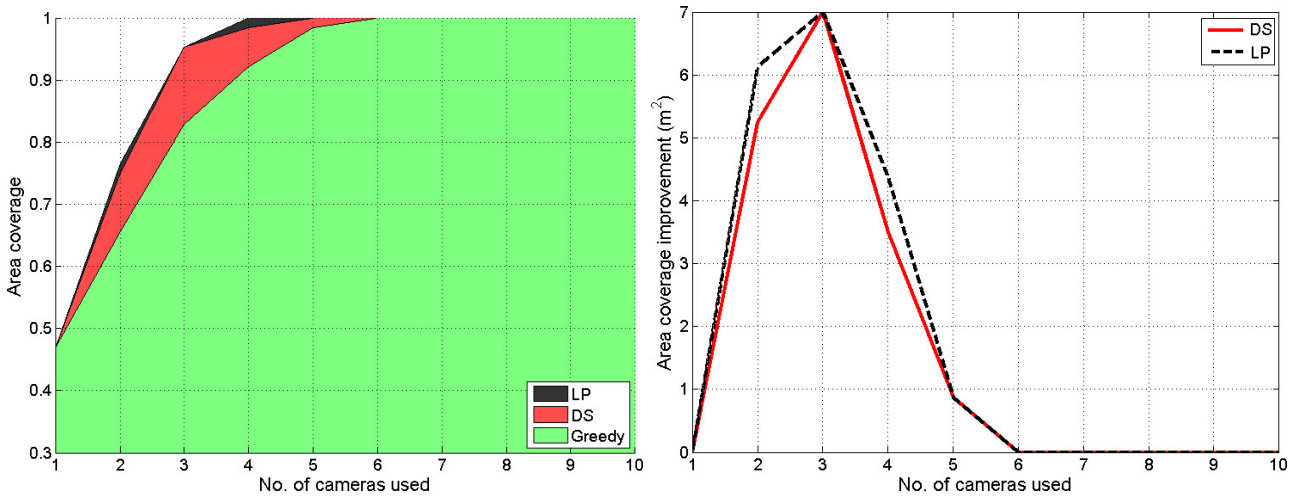
(a) (b) (c)

Figure 1: The visibility algorithm. a) Graphical representation of the quintuplet that uniquely identifies each sensor used. b) Computation of the sensor field-of-view in presence of occlusions (black squares) through ray casting. c) Resulting field-of-view, with projected shadows in gray.

4. EXPERIMENTAL RESULTS

4.0 Experiment 0

In this experiment we want to compare the performance of the Greedy algorithm[1] and the proposed DS algorithm with respect to the optimal solution obtained modeling the problems as Binary Integer Programming (BIP) problems.[1,6,10,11] In particular we investigate how the solutions found by the Greedy and DS algorithms approximate the global optimal solution given solving the BIP model.[12] The BIP model is solved using the freely available *lpsolve* package.[13] Due to the limitations of freely available solvers in handling large scale problems,[1] in order to keep small the number of variables and constraints in the BIP model, the floorplan used is a simple square room with a total area of $56.25m^2$. Following[6] the floorplan is represented as an occupancy grid formed by square cells with single resolution. Each such cell coincides with a pixel, so that each pixel corresponds to an area of about 56 cm^2. In this experiment we have used a discrete search space: the cameras were allowed to be placed on a regular grid with nodes 15 pixel apart (i.e. almost 1.875 m); the possible poses $\varphi \in [0, 2\pi]$ were sampled at $\pi/4$ angles. The camera parameters chosen are $(d, \alpha) = (15m, 65°)$.



(a) (b)

Figure 2: Comparison between the global optimal solution given by the solution of the BIP model (LP), and the Greedy and DS solutions for the problem 1 on a square room. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the LP solution over the Greedy solution with respect to the number of cameras used (dotted black line). Area coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used (solid red line).

In Table 1 we report a brief summary of the type of solution that the BIP, Greedy and DS algorithm give (i.e. global optimum or approximate solution), and the kind of problems that they can handle.

Table 1: Brief summary of the type of solution that the BIP, Greedy and DS algorithm give, and the kind of problems that they can handle.

Method	Global Optimum/Approx. solution	Discrete problems	Continuous Problems	3D problems	Large-scale problems
BIP	•/–	•	–	•	◦
Greedy	–/•	•	–	•	•
DS	–/•	•	•	•	•

◦ depends on the solver used

Given the limitations of freely available solvers in handling large scale problems, and the similar performance showed in this experiment by the BIP and DS solution, in the next experiment we will compare only the Greedy and DS algorithms. The Greedy has been chosen since, among the heuristics presented in,[1] it was the one that best approximated the optimum solution given by the BIP model.

4.1 Experiment 1

In this experiment, we test instances of all the four problems as described in Sec. 1. The floorplan used represents one of the laboratories of our department and has a total area of about 282.43 m^2. Following[6] the floorplan is represented as an occupancy grid formed by square cells with single resolution. Each such cell coincides with a pixel, so that each pixel corresponds to an area of about 56 cm^2. In this experiment we have used a discrete search space: the cameras were allowed to be placed on a regular grid with nodes 25 pixel apart (i.e. almost 1.875 m); the possible poses $\varphi \in [0, 2\pi]$ were sampled at $\pi/8$ angles. Two different kinds of cameras are employed: an high-quality one, with $(d, \alpha) = (15m, 65°)$ at the cost of 100$, and a low-quality one, with $(d, \alpha) = (11.5m, 45°)$ at the cost of 60$. The parameter d has been determined using the equation for the length of the projection of a real world object on the image plane of a camera,[14] the camera specifications, and the minimum required spatial resolution of $0.25 pixels/mm$.[6] The occupancy grid generated is reported in Fig. 3. The nodes of the regular grid on which the cameras were allowed to be placed are reported in Fig. 4.

The comparison of the results obtained by the Greedy algorithm proposed in[1] and the Direct Search algorithm proposed, for the four problems considered, are respectively reported in Fig. 5-9.

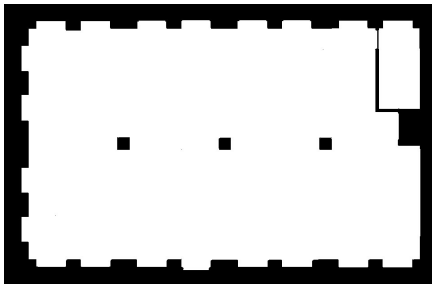


Figure 3: Floorplan used for Experiment 1 (LAB).

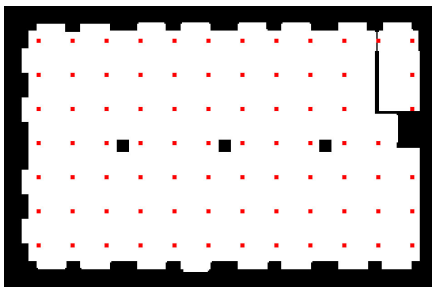
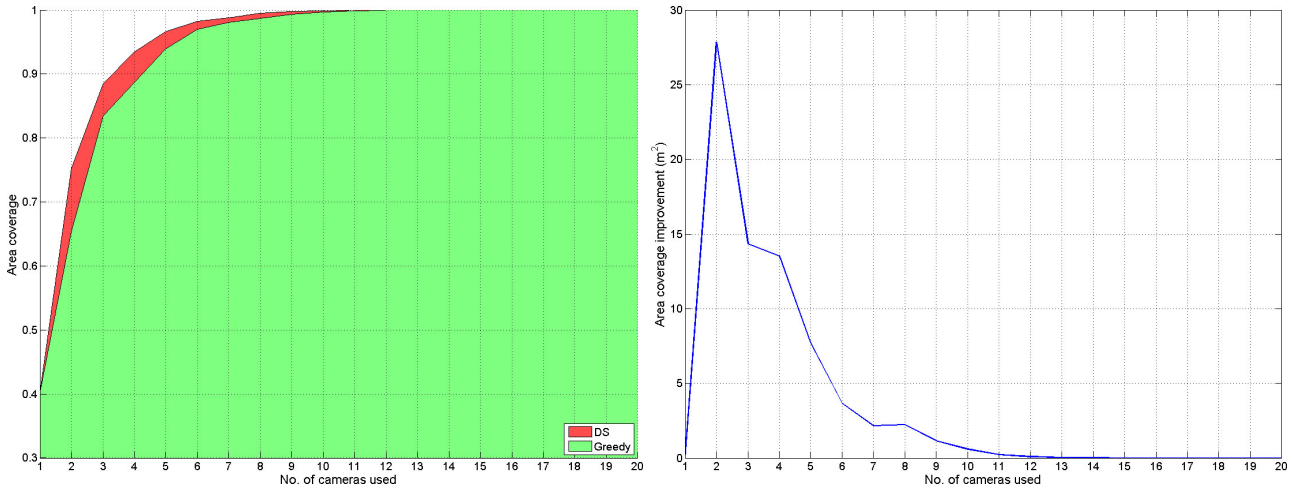


Figure 4: Nodes of the regular grid on which the cameras are allowed to be placed for the LAB floorplan used for Experiment 1.

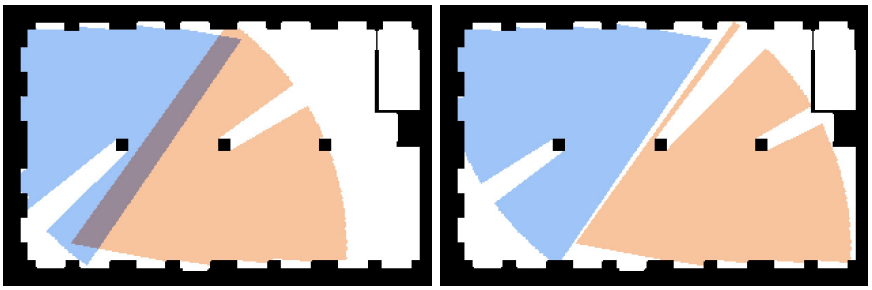


(a) (b)

Figure 5: Comparison between the Greedy and DS solutions for the problem 1 on the LAB floorplan. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used.

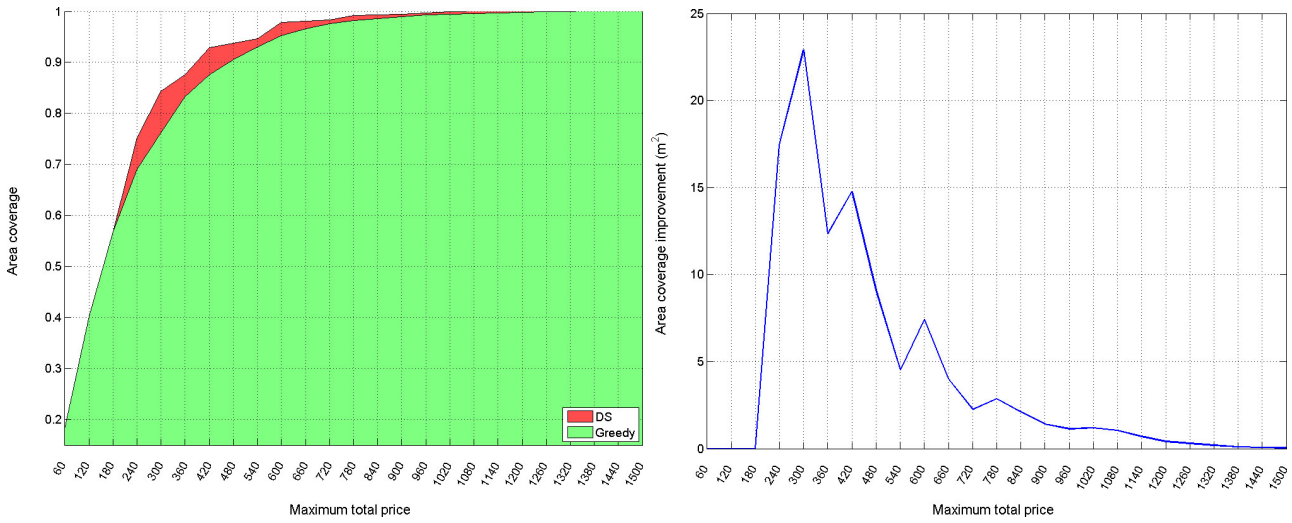
As an example, we report in Fig. 6 the solutions given by the Greedy and the DS algorithm for the case of two cameras. The field-of-view (FOV) of the first camera placed is drawn in red, the FOV of the second camera

placed is drawn in blue. It is possible to notice how the Greedy solution (Fig. 6(a)) shows an overlap between the FOV of the two cameras placed, and that the FOV of the second camera is obstacled in great part by the walls of the room. On the other side, the DS solution (Fig. 6(b)) shows no overlap between the two FOVs, and the FOV of the second camera is much less obstacled, resulting in a higher area coverage. In fact, the Greedy solution with two cameras covers the 65.36% of the area, 40.29% of which covered by the first camera and 25.07% by the second one. The DS solution instead covers the 75.23% of the area, 37.55% of which covered by the first camera and 37.68% by the second one.



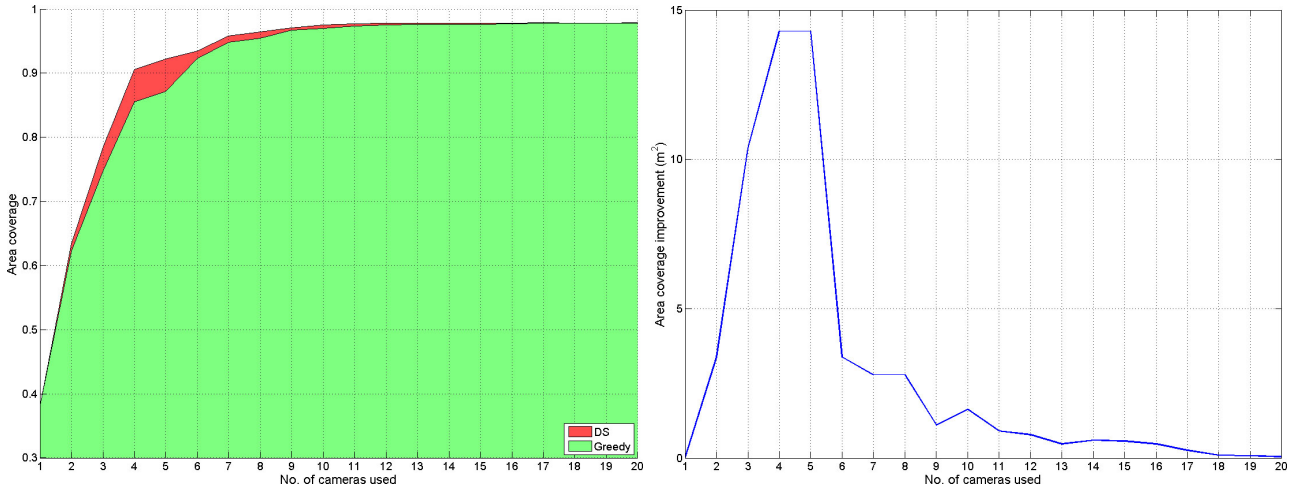
(a) (b)

Figure 6: Solutions given by the Greedy (a) and the DS algorithm (b) for the case of two cameras for the problem 1 on the LAB floorplan. FOVs of the cameras placed: the FOV of the first camera is colored in red, the second camera's one in blue.



(a) (b)

Figure 7: Comparison between the Greedy and DS solutions for the problem 2 on the LAB floorplan. a) Maximum area coverage with respect to the maximum total price of the sensor array. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the maximum total price of the sensor array.



(a) (b)

Figure 8: Comparison between the Greedy and DS solutions for the problem 3 on the LAB floorplan. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used.

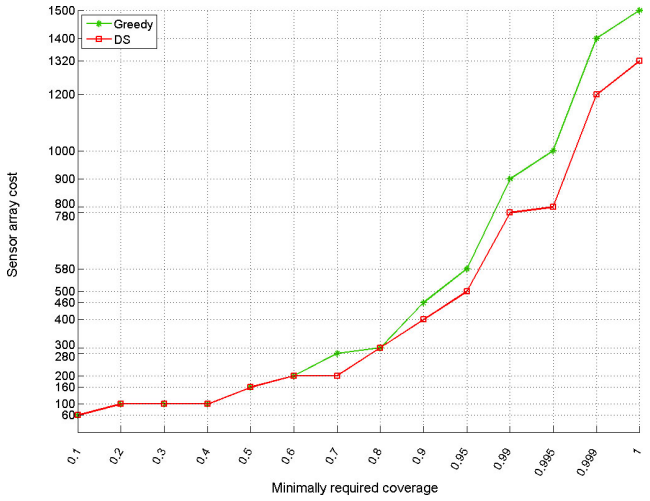


Figure 9: Comparison between the Greedy and DS solutions for the problem 4 on the LAB floorplan: minimum sensor array cost with respect to the minimally required area coverage.

4.2 Experiment 2

In this experiment, we test the instances of all the four problems as described in Sec. 1. The floorplan used represents the left wing of our department and has a total area of about 822.35 m^2. The same settings as is Sec. 4.1 have been used here. The occupancy grid generated is reported in Fig. 10. The nodes of the regular grid on which the cameras were allowed to be placed are reported in Fig. 11.

The comparison of the results obtained by the Greedy algorithm proposed in[1] and the Direct Search algorithm proposed, for the four problems considered, are respectively reported in Fig. 12-15.

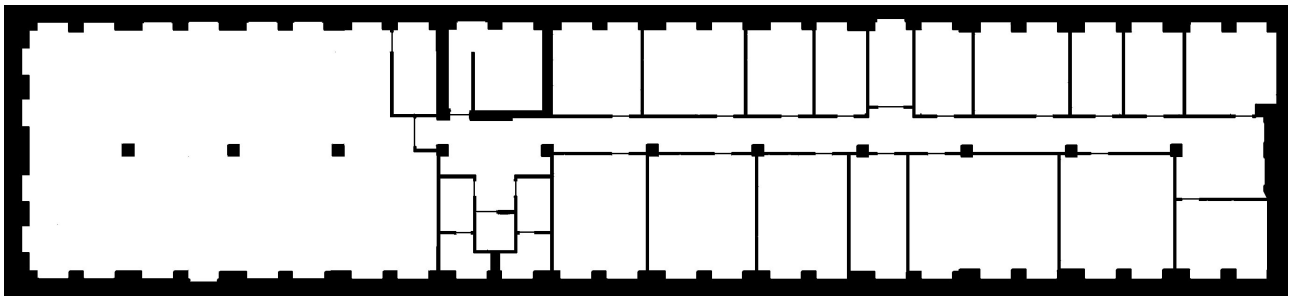


Figure 10: Floorplan used for Experiment 2 (HL).

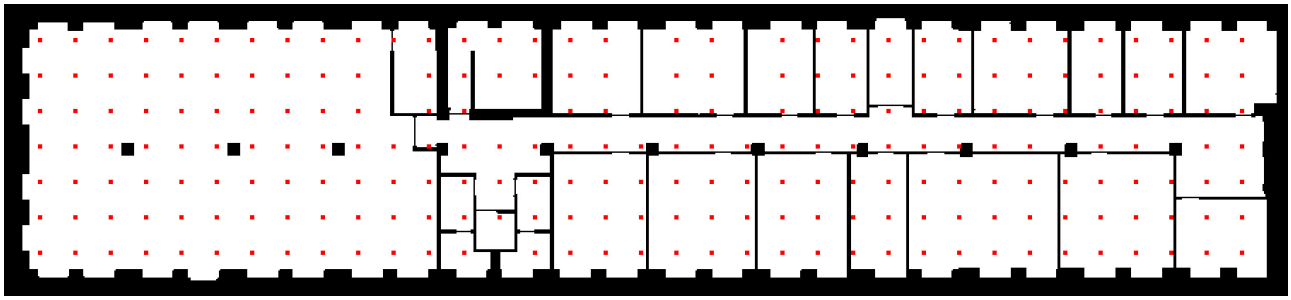
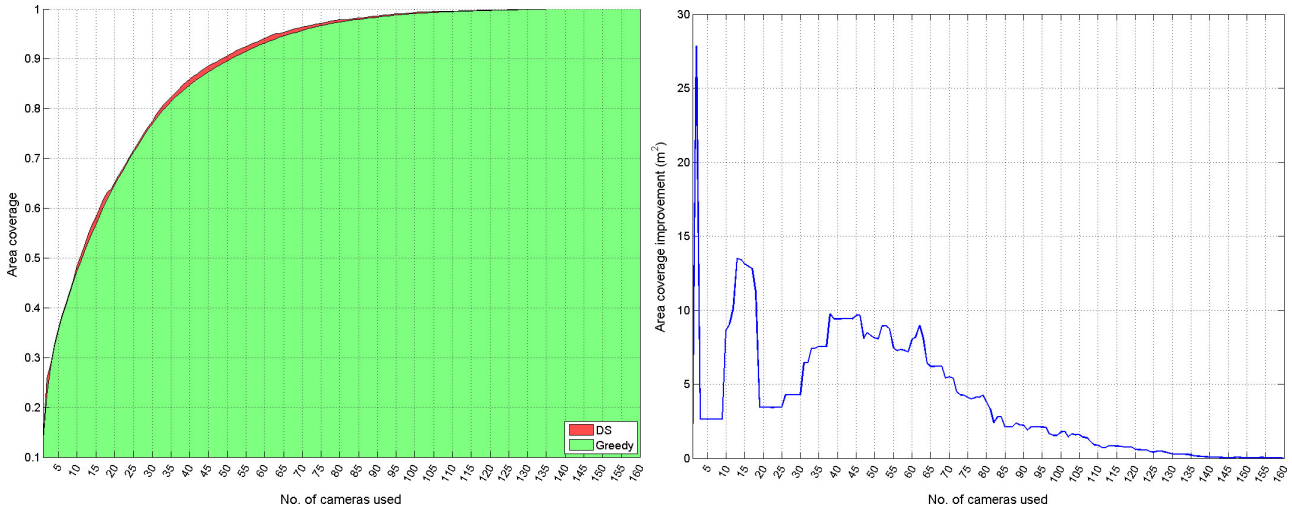
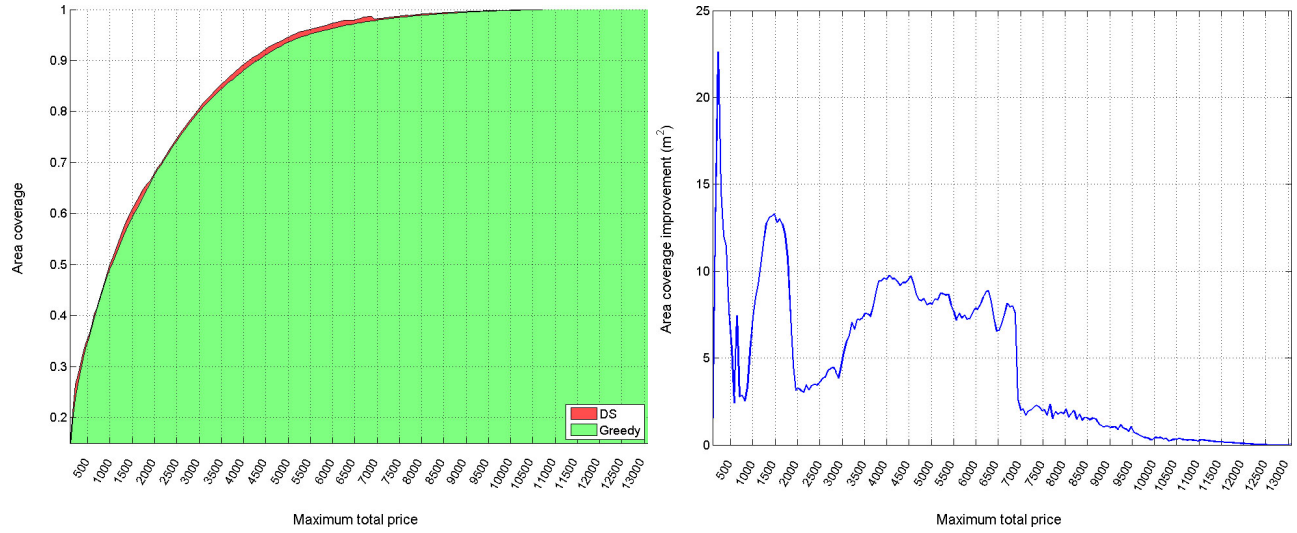


Figure 11: Nodes of the regular grid on which the cameras are allowed to be placed for the HL floorplan used for Experiment 2.



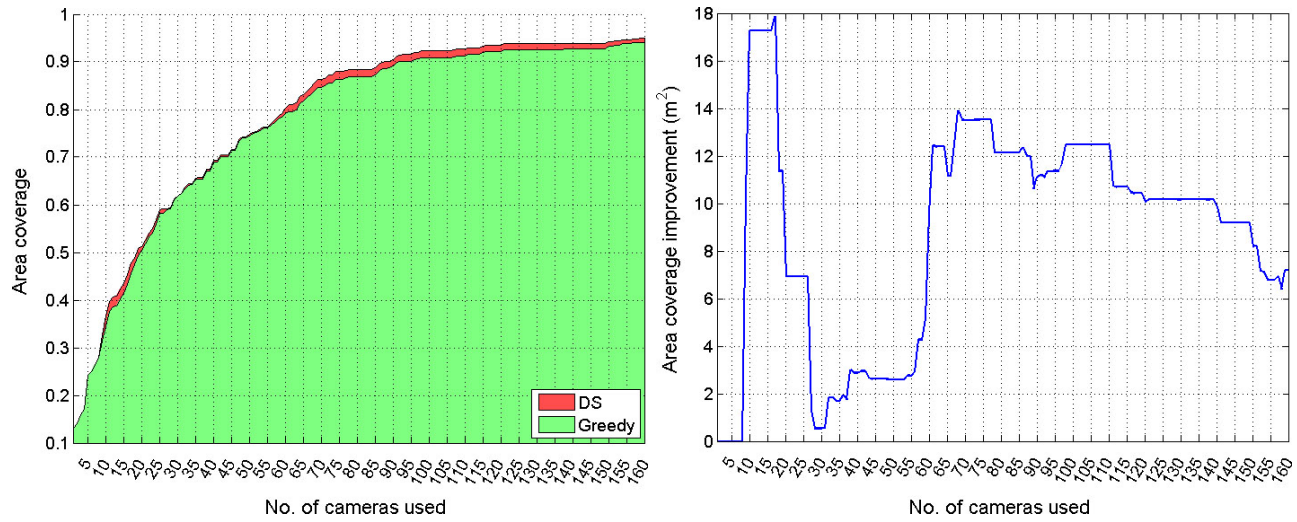
(a) (b)

Figure 12: Comparison between the Greedy and DS solutions for the problem 1 on the HL floorplan. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used.



(a) (b)

Figure 13: Comparison between the Greedy and DS solutions for the problem 2 on the HL floorplan. a) Maximum area coverage with respect to the maximum total price of the sensor array. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the maximum total price of the sensor array.



(a) (b)

Figure 14: Comparison between the Greedy and DS solutions for the problem 3 on the HL floorplan. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used.

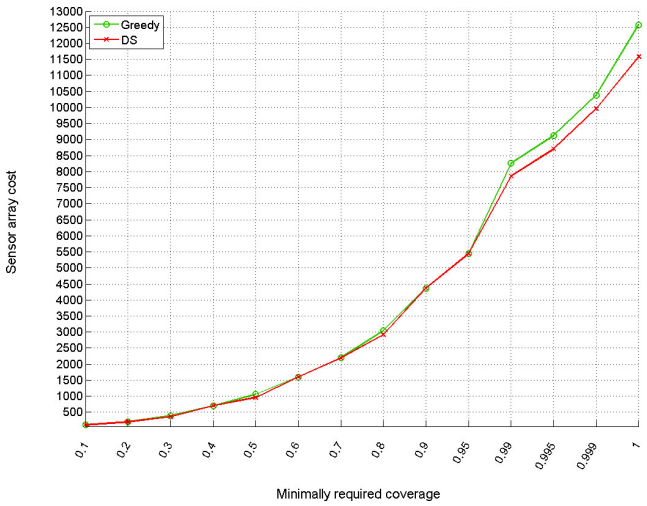
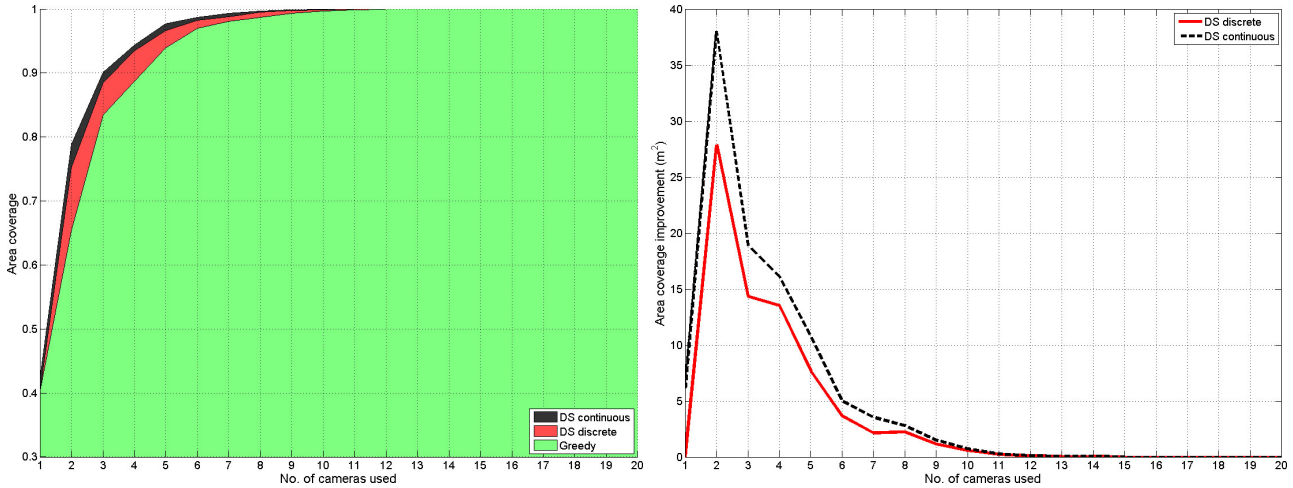


Figure 15: Comparison between the Greedy and DS solutions for the problem 4 on the HL floorplan: minimum sensor array cost with respect to the minimally required area coverage.

4.3 Experiment 3

In this experiment, the instance of the first problem as described in Sec. 1. The floorplan used is the same of Sec. 4.1, and represents one of the laboratories of our department. Differently from Sec. 4.1, we consider here a continuous search space: this means that the camera can be placed on every cell of the occupancy grid, and the pose α can be $\alpha \in [0, 2\pi]$, $\alpha \in \mathbb{R}$.

The comparison of the results obtained by the Greedy algorithm proposed in[1] and the Direct Search algorithm proposed, are reported in Fig. 16. Since the Greedy algorithm can not work with continuous search spaces, it is run with the same configuration of Sec. 4.1. For completeness, the results obtained by the discrete version of the Direct Search algorithm of Sec. 4.1 are also reported.



(a) (b)

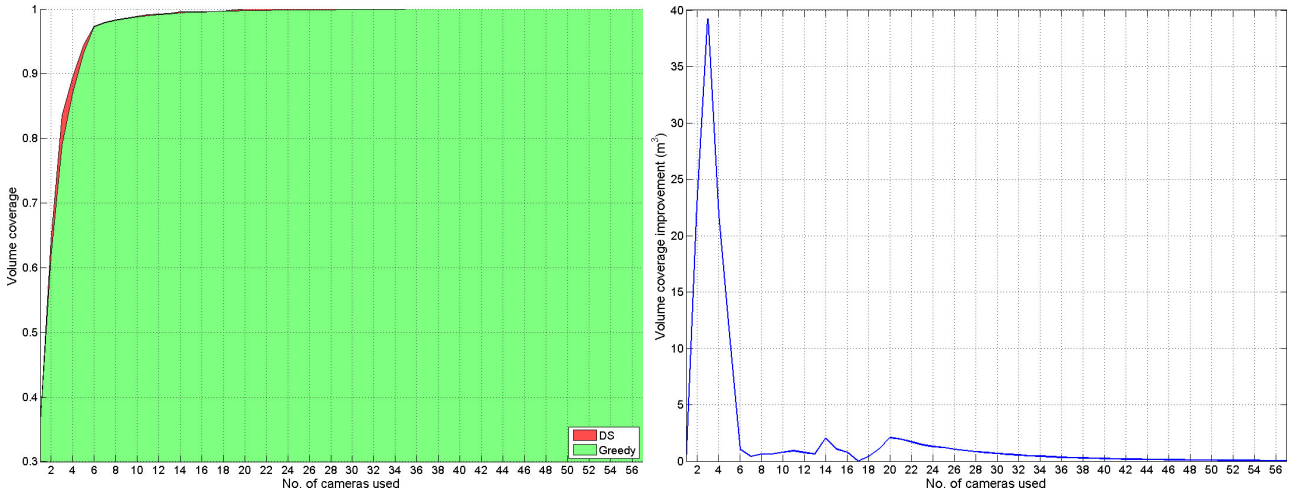
Figure 16: Comparison between the Greedy and DS solutions for the problem 1 on the LAB floorplan, for the continuous case. a) Maximum area coverage with respect to the number of cameras used. b) Area coverage improvement of the discrete DS solution over the Greedy solution with respect to the number of cameras used (solid red line); area coverage improvement of the continuous DS solution over the Greedy solution with respect to the number of cameras used (dotted black line).

4.4 Experiment 4

In this experiment, the instance of the first problem as described in Sec. 1. Differently from Sec. 4.1, we consider here a 3D map. The coverage that we want to maximize is thus no more a 2d area coverage, but a 3D volume coverage.

The floorplan used represents one of the laboratories of our department and has a total volume of about 847.29 m^3. The floorplan is represented as an occupancy grid formed by cubic cells with single resolution. Each such cell coincides with a voxel, so that each voxel corresponds to a volume of about 421 cm^3. In this experiment we have used a discrete search space: the cameras were allowed to be placed on the ceiling on a regular grid with nodes 25 pixel apart (i.e. almost 1.875 m); the possible poses $\varphi \in [0, 2\pi]$ were sampled at $\pi/4$ angles. Since we are dealing with the 3D case, we have another pose to consider, i.e. the tilt: this is defined as the angle with the z axis, and in our setup is sampled in the range $\theta \in [\pi/2, \pi]$ at $\pi/4$ angles.

The comparison of the results obtained by the extension for the 3D case of the Greedy algorithm proposed in[1] and the Direct Search algorithm proposed, are reported in Fig. 17.



(a) (b)

Figure 17: Comparison between the Greedy and DS solutions for the problem 1 on the 3D representation of the LAB floorplan. a) Maximum volume coverage with respect to the number of cameras used. b) Volume coverage improvement of the DS solution over the Greedy solution with respect to the number of cameras used.

4.5 Experiment 5

In this experiment, we want to see how the improvement of our DS method over the Greedy changes varying the size of the room in which we want to place the sensors. The instance of the problem considered is the first one described in Sec. 1. The room considered is a simple one: it is a square room with varying side length L. The side length L has been chosen such that the ratio d/L - where d is the sensing range of the sensor (see Fig. 1(a)) - varies approximately in the range $[0.5, 4]$. Different sensing angles have been also considered. The results are reported in Fig. 18, where the colored strips contains the values comprised between the 5^{th} and the 95^{th} percentile, and the dotted white line is the average across sensing angles.

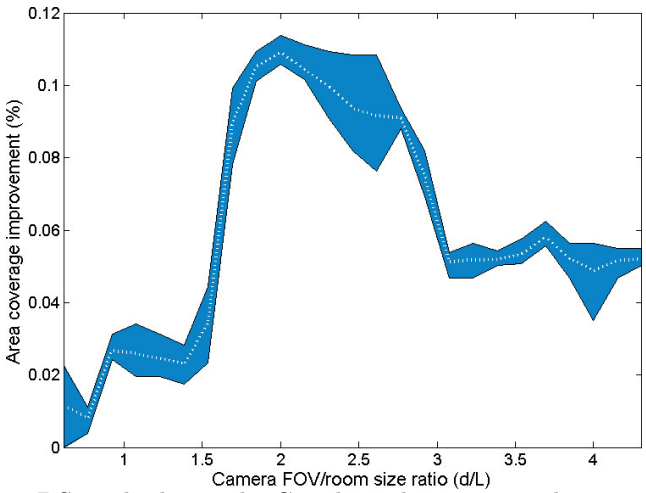


Figure 18: Improvement of our DS method over the Greedy with respect to the size of the room in which we want to place the sensors. The colored strips contains the values comprised between the 5^{th} and the 95^{th} percentile, and the dotted white line is the average across sensing angles.

5. CONCLUSIONS

In this paper we have addressed the problem of optimal sensor placement for a given region and task. An important issue in designing sensor arrays is the appropriate placement of the sensors such that they achieve a predefined goal. There are many problems that could be considered in the placement of multiple sensors. In this work we focus on the following four problems: i) maximizing coverage subject to the given number of sensors; ii) maximizing coverage subject to the maximum total price of the sensor array; iii) optimizing sensor poses given fixed locations; iv) minimizing the cost of the sensor array given a minimally required percentage of coverage.

To solve these problems, we have proposed an algorithm based on Direct Search, which is able to approach the global optimal solution within reasonable time and memory consumption. The algorithm has been experimentally evaluated and the results on two real floorplans have been presented. The algorithm has then been extended to work also on continuous solution spaces, and 3D problems. The experimental results show that our DS algorithm is able to improve the results of approximated algorithms in the state of the art.

As future work we plan to extend the proposed DS algorithm to deal with areas of different importance, and non-static sensors.

REFERENCES

[1] Hörster, E. and Lienhart, R., "On the optimal placement of multiple visual sensors," in [*Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks (VSSN'06)*], 111–120 (2006).

[2] Chvatal, V., "A combinatorial theorem in plane geometry," *Journal of Combinatorial Theory Series* **18**, 39–41 (1975).

[3] Fisk, S., "A short proof of chval's watchman theorem," *Journal of Combinatorial Theory Series* **24**, 374 (1978).

[4] O'Rourke, J., [*Art Gallery Theorems and Algorithms*], Oxford, New York (1987).

[5] Gindy, H. and Avis, D., "A linear algorithm for computing the visibility polygon from a point," *Journal of Algorithms* **2**, 86–197 (1981).

[6] Erdem, U. M. and Sclaroff, S., "Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints," in [*Proceedings of the OMNIVIS Workshop*], (2004).

[7] Kolda, T. G., Lewis, R. M., and Torczon, V., "Optimization by direct search: New perspectives on some classical and modern methods," *SIAM Review* **45**, 385–482 (2003).

[8] Hooke, R. and Jeeves, T. A., "Direct search solution of numerical and statistical problems," *Journal of the ACM* **8**, 212–229 (1961).

[9] Appel, A., "Some techniques for shading machine renderings of solids," in [*Proceedings of the AFIPS spring joint computer conference*], **32**, 37–45 (1968).
[10] Hörster, E. and Lienhart, R., "Calibrating and optimizing poses of visual sensors in distributed platforms," *ACM Multimedia Systems Journal, Special Issue on Multimedia Surveillance System* **12**, 195–210 (2006).
[11] Chakrabarty, K., Iyengar, S. S., Qi, H., and Cho, E., "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transactions on Computer* **51(12)**, 1448–1453 (2002).
[12] Williams, H., [*Model building in mathematical programming*], J. Wiley, New York, second edition (1985).
[13] Berkelaar, P. N. M. and Eikland, K., "lpsolve: Open source (mixed-integer) linear programming system." Eindoven University of Technology, http://groups.yahoo.com/group/lp_solve/files/version5.5/ (last access on 13 dec. 2011).
[14] Abrams, S., Allen, P. K., and Tarabanis, K., "Computing camera viewpoints in an active robot cell," *International Journal of Robotics Research* **18**, 267–285 (1999).