Optimal Arrangement of Ceiling Cameras for Home Service Robots Using Genetic Algorithms

Stefanos Nikolaidis and Tamio Arai

Abstract—**In the near future robots will be used in home environments to provide assistance for the elderly and challenged people. As home environments are complicated, external sensors like ceiling cameras need to be placed on the environment to provide the robot with information about its position. The pose of cameras influences the area covered by the cameras, as well as the error of the robot localization. We examine the problem of the finding the arrangement of ceiling cameras at home environments that maximizes the area covered and minimizes the localization error. Genetic algorithms are proposed for the single and multi-objective optimization problem. Simulation results indicate that we can obtain the optimal arrangement of cameras that satisfies the given objectives and the required constraints.**

I. Introduction

RECENTLY, improved medical care and high living standards have contributed to a dramatic rise of the median age of the population in most developed countries. The use of robots in home environments can play a major role in providing assistance for the elderly and reducing labor costs.

Home environments are complicated environments, consisting of doors, walls and furniture. Such environments are full of obstacles that limit the movement of the robot. In order to execute home service tasks, as well as for safety reasons, a robot needs to have precise information about its own position. Due to the complexity of the environment, external sensors placed on the environment are used for robot localization. Although there has been much research on robot and object localization techniques with different types of sensors [1], in most of past research sensors were placed randomly or intuitively. However, the position of sensors affects greatly the error of the localization process. Where to fix sensors is an important requirement, when we design a robotic environment.

Past research on camera placement was mostly limited to the placement of only one sensor [2, 3], two sensors [4] or to the special case of marker detection [5]. The problem of camera placement is closely related to the guard placement problem (AGP). The guard placement problem is the problem of determining the minimum number of guards covering the interior of an art gallery and it is known to be an NP-hard problem [6]. It is addressed by the art gallery theorem [7]. In the AGP all guards are assumed to have identical properties. In this study, however, we consider cameras with different field-of-views. Furthermore, we consider not only the area covered, but also the uncertainty of the position of a robot moving at that area during localization process, due to sensor constraints. Occluded area in complicated 3D environments is also considered.

In Section II we define the optimization problem of optimal camera placement. In Section III we describe a methodology to estimate the localization error due to image resolution error, as well as the occlusion by obstacles in the room. In Section IV and V we study the camera placement optimization problem for the single and multi-objective case respectively. All experiments are presented in Section VI. Finally, conclusions are drawn in Section VII.

II. The Optimal Camera Placement Optimization Problem

A. Assumptions

We consider the case of a robot executing home-service tasks at a room in a home environment. For simplification purposes, we assume that the 3*D* map of the room is known. As 3*D* map, we mean the position and CAD model of doors, walls and room furniture. We assume the presence of only one robot in the room. As the floor of the room is flat, we can assume that the robot is moving at a known and fixed height. When optimizing the pose of each camera, we search for its optimal 2*D* position on the ceiling, pan and tilt angle. All simulations are performed at a virtual 3*D* room, built using the OpenGL library [8], which is an exact 3*D* model of the Kanagawa House Square Experimental Room (Fig. 1). This room is a part of the Universal Design Project [9].

B. Definition of the Optimization Problem

When placing cameras on the room for robot localization, we want to use as few cameras as possible, as the cost of the cameras is high. In other words, given a fixed number of cameras, we want these cameras to cover as much area as possible, so that the robot is visible and therefore localized by the cameras in that area. The area covered by the cameras is influenced by the pose of the cameras. Therefore, given a fixed number of cameras, we need to find the arrangement of cameras that results in the maximized area covered. Due to the complexity of the home environment, as well as for

Manuscript received March 11, 2009.

This study was performed through Special Coordination Funds for Promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology, the Japanese Government

Stefanos Nikolaidis and Tamio Arai are with the Department of Precision Engineering, School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 Japan

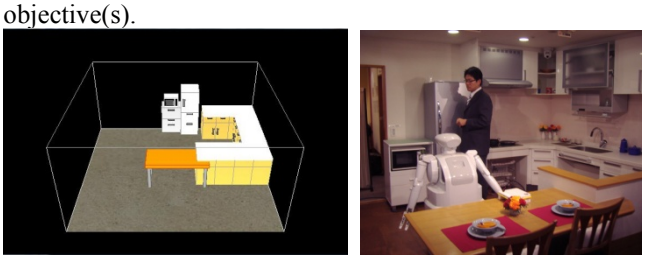
Stefanos Nikolaidis presently belongs to Square Enix CO., Ltd., Research and Development Division, Shinjuku Bunka Quint Bldg. 3-22-7 Yoyogi, Shibuya-ku, Tokyo 151-8544, Japan

(e-mails:nikolaid@square-enix.com, arai-tamio@robot.t.u-tokyo.ac.jp).

safety reasons, the robot should be localized with a certain precision. In other words, robot localization error should be as small as possible. Therefore, robot localization error should be also considered, together with the area covered, when placing the cameras.

In the following sections, we consider two cases for the optimization problem, a single and a multi-objective one. For the single-objective case, only the visible to total area ratio of the room is considered as evaluation function for the optimization problem. For the multi-objective case, the ratio of visible to total area of the room and the average localization error of the visible area are both considered.

Accordingly, the optimization problem is defined as follows: The optimization problem is, given the map of the room, the required objectives and the given constraints, to find the optimal arrangement of cameras, which satisfies the necessary constraints and maximize / minimize the required objective(s).



(a) (b)

Fig. 1. (a) Virtual 3D room environment and (b) Real room environment at Kanagawa House Square.

III. Image Resolution Error and Occlusion Estimation

In order to be able to estimate the average localization error of the area where a robot executes home service tasks, we need to estimate this error for each sampled point of the area, for a given camera arrangement. Errors that affect the accuracy in robot localization are mainly:

1) image resolution errors (quantization errors)
2) image processing errors
3) camera calibration errors

Among these errors, image resolution errors are focused mainly on this paper, because these errors can be reduced by appropriate arrangement of cameras. Image processing errors depend mainly on software and camera calibration errors are not related to camera arrangement.

A. Image Resolution Error

At 3D localization process, most vision-based tracking systems perform some sort of triangulation on rays from two or more cameras. Some target feature is detected on the image and the ray defined by its 2D location and the camera center is back-projected into 3D space. The intersection point of rays from multiple cameras defined by the same feature is the 3D location of the feature. However, one can only determine the 2D location of a feature on an image to a certain precision, as limited by the image resolution, as shown in Fig. 2. Assuming the robot is navigating at a known height, one camera is enough for robot localization. In that case, the uncertainty caused by the image resolution of the camera during the localization process, can be estimated as follows.

Let us assume a plane Π and a point $P(X, Y, Z)$, visible by a camera C. The height of the camera with reference to the plane Π is assumed to be known. Then, the point P is projected to a pixel $P'(x, y)$ at the image plane of the camera. Assuming that the intrinsic and extrinsic parameters of the camera are known, we can calculate the area Ω at the plane Π corresponding to the same pixel at the image plane. This area represents the uncertainty of the robot location, when the robot is detected at pixel P' in the image plane (Fig. 3).

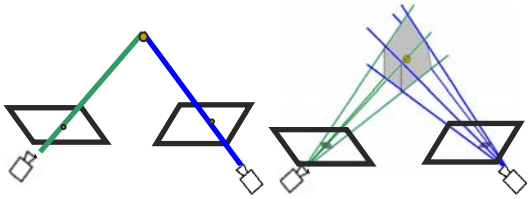


Fig. 2. (a) 3D localization with triangulation. (b) 3D position uncertainty due to image resolution.

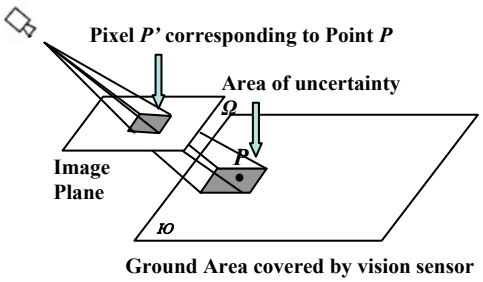


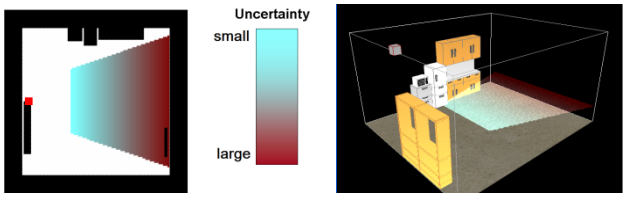
Fig. 3. Uncertainty caused by image resolution, when the robot is known to be moving at a known height. The robot can be anywhere inside the area Ω, yet it will be detected at the same pixel P'.

Given a certain camera arrangement, we can sample the area the robot is moving with a large number of points and calculate the uncertainty for each point. Therefore, the uncertainty distribution of the whole visible area can be estimated. In Fig. 4, the visible area is calculated from the projection of the viewing frustrum of the camera at a specific height. The uncertainty distribution of that area is also illustrated.

Hence, our goal is to arrange the cameras in such a way that the average uncertainty of the whole area is minimized. If two or more cameras observe the same area, the camera that results at the minimum uncertainty is selected, and this minimum value is adopted. This is reasonable, as after we design the camera arrangement we can know which camera is optimal for the localization process at each arbitrary point. This is illustrated in Fig. 5, for the case of three cameras.

Even if a point at the area of the room is inside the field of view of the camera, it may be invisible to the ceiling cameras due to occlusion. When calculating the average localization

error of the area, it is important to find the regions that are occluded by obstacles and therefore are not visible by the cameras.



(a) (b)

Fig. 4. (a) Uncertainty distribution of the visible area at a specific height (2D cut of the room at a specific height).(b) The uncertainty distribution illustrated at the 3D visual environment.

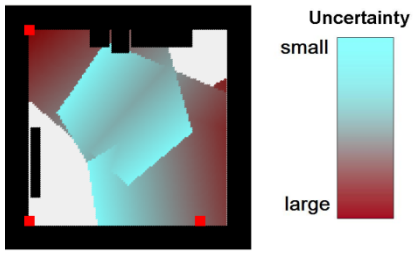
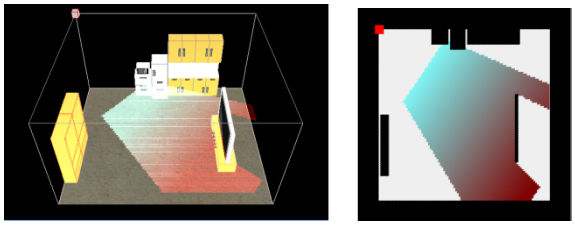


Fig. 5. Uncertainty distribution for the case of three cameras with overlapping viewing frustrums.

B. *Occlusion Estimation at 3D Environment*

We use an algorithm proposed by [10] that, given the 3D map of the room, computes if some or all points of a feature of interest are occluded by environment objects. Convex and concave polyhedral with or without holes and the viewing model of projective projection are employed. We assume that the objects in the environment are at a known position and can be represented by known CAD models. Then, the boundaries of these polyhedral objects are considered to be comprised of one or more polygonal faces, and the occluded region of an area of interest due to the environment as a whole is equal to the union of the component occluded regions generated by the individual object faces. In this method, the problem of computing the occluded region between a convex occluding polygon and a convex feature polygon is employed as a component step. A concave or multiply connected object face is decomposed (i.e. partitioned) into convex polygons. As an example of convex partitioning, it is known that any simply or multiply connected polygonal domain can be triangulated [7]. The occluded region is the union of the component occluded regions generated between all pairs of convex polygons, one taken from the convex polygons of the object face and one from the convex polygons of the feature. Computing the occluded region in the case of a convex occluding polygon and a convex feature is explained in detail in [11].

As an application of the above algorithm, we consider as area of interest a plane parallel to the floor of the 3D room of Fig. 1 and at a specific height above the floor of the room. Then, we place a television in the middle of the room, and, we illustrate the part of the area of interest that is visible from a ceiling camera fixed at the top left of the room, considering the occlusion caused by the presence of the television (Fig. 6).



(a) (b)

Fig. 6. (a) Virtual 3D room environment and (b) 2D cut of the room at a specific height and projection of the viewing frustrum of the camera at a specific height considering occlusion from the television in the middle of the room.

IV. Single-Objective Optimization

A. *Formulation of the Optimization Problem*

The first step is to maximize / minimize the required objectives in the environment with a fixed number of n cameras. In the single-objective case, we fix the cameras so that the ratio of the visible to total room area is maximum.

Assuming that the total area that the robot executes home-service tasks is sampled by a rectangular grid of N points $\mathbf{P}_i$ of equal distance, the visible to total area ratio is defined as follows:

$$F_1 = \frac{\sum_{i=1}^{N} f(\mathbf{P}_i)}{N}, \text{ where } f(\mathbf{P}_i) \rightarrow \{0, 1\} \quad . \tag{1}$$

The function $f(\mathbf{P}_i)$ is one if the point $\mathbf{P}_i$ is visible by one or more cameras or zero if it is not visible by any camera. The point is not visible by any camera if it is outside the FOV (Field Of View) of the camera, or if it is occluded by another obstacle.

B. *Methodology*

A genetic algorithm is proposed to address the single-objective case of optimal camera placement.

1) *Genetic Algorithms Overview*

Since the appearance of GAs (Genetic Algorithms) in 1975 [12], GAs have been used in solving many optimization problems successfully. GAs are stochastic, parallel search algorithms based on the mechanics of natural selection and the process of evolution. GAs perform a multidirectional search by maintaining a population of potential solutions and encourage information formation and exchange between these solutions. A selection mechanism based on the fitness is applied to the population and the individuals strive for survival. The fitter ones have more chance to be selected and to reproduce offsprings by genetic operators such as crossover and mutation. The process is repeated and the population is evolved generation by generation. After many generations, the population converges to solutions of good quality, and the best individual has good chance to be the optimal or near-optimal solution.

2) Problem Representation

In this study each individual of the generation of the genetic algorithm is encoded as a binary string. Each string contains information about the position (x, y), pan angle (θ), and tilt angle (ϕ) of each camera. We encode with 7 bits the position in x-axis, 7 bits the position in y-axis, 6 bits pan angle and 6 bits the tilt angle of the camera. Therefore, the total length of each individual is $26 \times n$ bits, where n is the number of cameras.

3) Genetic Operators

We use the following genetic operators:

(a) *Selection*: A proportionate stochastic selection is used for the selection of the individuals [13]. In other words, each individual is chosen for reproduction with a probability proportionate to its fitness. As fitness of each individual we consider the visible to total area ratio (Eq. (1)) resulting from the encoded camera arrangement of that individual.

(b) *Crossover*: After two individuals (called *parents*) are selected, a single-point crossover is done. A crossover - point is selected randomly and the data of the two parents before that point is interchanged [13]. The role of crossover operator is to combine data of selected individuals in order to create new, better solutions.

(c) *Mutation*: We implement mutation by changing the value of an arbitrary bit of an individual with some probability. Mutation is essential in maintaining a large diversity of solutions.

(d) *Elitism*: We select a number of the best individuals of the generation and copy them to the next generation without performing any genetic operation on them. This is because we want to keep a proportion of the best of these solutions to the next generations intact, without having them recombined or mutated.

4) Genetic Algorithm Flow

We illustrate the algorithm flow of the genetic algorithm in the activity diagram of Fig. 7. As a solution we choose the individual from the final generation that has the best fitness.

V. Multi-Objective Optimization

A. Formulation of the Optimization Problem

In this study, as optimization criteria we consider the visible to total area ratio F_1 and the average localization error of the visible area F_2. F_1 is given by Eq.(1) (Section IVA), whereas F_2 is defined by Eq.(2). We assume again that the total area in which the robot executes home-service tasks is sampled by a rectangular grid of N points $\mathbf{P}_i$ of equal distance. The function $g(\mathbf{P}_i)$ returns the uncertainty of the position of the center of the robot, if the center is located at point $\mathbf{P}_i$. The uncertainty is estimated as the area corresponding to the same pixel $\mathbf{P}_i$ at the image plane of the camera, as explained in Section III-B.

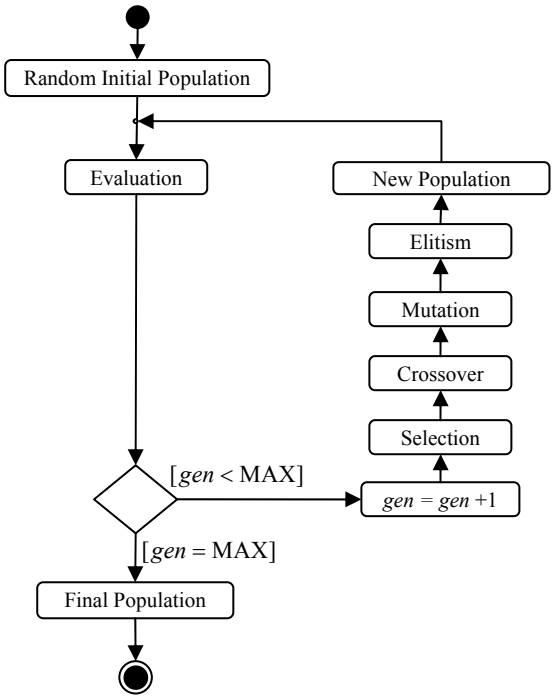


Fig. 7. Activity diagram of genetic algorithm for the single-objective case. *gen* is the current generation index and MAX is the maximum generation number.

$$F_2 = \frac{\sum_{i=1}^{M} g(\mathbf{P}_i) \cdot f(\mathbf{P}_i)}{\sum_{i=1}^{N} f(\mathbf{P}_i)}, \; f(\mathbf{P}_i) \rightarrow \{0, 1\}. \qquad (2)$$

The function $f(\mathbf{P}_i)$, like in (Eq. (1)) is 1 if the point is visible by at least one of the cameras and 0 elsewhere. Therefore, the function calculates the localization error of the visible area. For the trivial case of a solution with no visible point $\mathbf{P}_i$, we reject the solution without calculating F_2.

We want to maximize F_1 and minimize F_2 :

$$F_1 \rightarrow \max, \; F_2 \rightarrow \min \qquad (3)$$

Furthermore, we assume to have the additional constraints:

$$F_1 \geq K_1, \; F_2 \leq K_2 \; . \qquad (4)$$

It is clear that there is an objective conflict between F_1 and F_2. In other words, optimal solutions for each objective individually are different from the optimal solutions for both objectives.

B. Methodology

1) Overview

Traditional multiobjective optimization approaches usually combine all objectives to form a scalar fitness function by using a weighted aggregation approach, the method of distance functions, or the Min-Max formulation [14]. Then, any optimization algorithm, like the steepest descent or the genetic algorithm can then optimize this scalar fitness function. However, these classic approaches can be

very sensitive to the precise aggregation of goals and tend to be ineffective and inefficient. Furthermore, these approaches lead to one solution only. However, there may be a set of multiple solutions that maximizes / minimizes the given objectives and satisfies the given constraints. In real-world situations, we are usually interested in finding more than one optimal solutions that satisfy the given constraints. This is because decision makers often need different alternatives in decision making.

For these reasons, we use the NSGA (Nondominated Sorting Genetic Algorithm) [14], which is a multi-objective pareto optimization algorithm, which returns a set of optimal solutions, called *pareto optimal set* or *pareto front*. These solutions are all optimal solutions and represent the best trade-off between conflicting design goals, in our case the ratio of visible to total area and the average localization error.

2) Pareto Optimization

In order to analyze the flow of the NSGA algorithm, we first need to give some basic definitions of pareto optimization, specified for the camera placement. In this problem, we want to maximize objective function $F_1(\mathbf{x})$ and minimize objective function $F_2(\mathbf{x})$, as described in Eq. (3)). Here, $\mathbf{x}$ is a decision variable vector, consisting of $4n$ variables representing the camera arrangement of n cameras: $\mathbf{x} = [x_1, y_1, \vartheta_1, \phi_1, ..., x_n, y_n, \vartheta_n, \phi_n]$, where (x_i, y_i) is the $2D$ position of camera-i on the ceiling, and ϑ_i and ϕ_i its pan and tilt angle. The goal space is the two-dimensional space with $F_1(\mathbf{x})$ on one axis and $F_2(\mathbf{x})$ on the other. We define that a solution $\mathbf{x}^{(1)}$ *dominates* a solution $\mathbf{x}^{(2)}$ if

$\left[F_1(\mathbf{x}^{(1)}) \geq F_1(\mathbf{x}^{(2)}) \text{ and } F_2(\mathbf{x}^{(1)}) < F_2(\mathbf{x}^{(2)}) \right]$, or

$\left[F_1(\mathbf{x}^{(1)}) > F_1(\mathbf{x}^{(2)}) \text{ and } F_2(\mathbf{x}^{(1)}) \leq F_2(\mathbf{x}^{(2)}) \right]$.

From a set of p solutions, any solution $\mathbf{x}^{(i)}$ that is not dominated by any other solution is called *nondominated* solution. The solutions that are nondominated within the entire search space are denoted as *pareto optimal* and constitute the *pareto optimal set* or *pareto front*. Pareto optimization algorithms search for the entire set of pareto optimal solutions. Then, the solutions that satisfy the required constraints are selected. From these solutions, the designer can then choose the one that bests fits his interests.

3) The NSGA algorithm

Compared to a typical genetic algorithm (Section IV-B)., the NSGA is different basically in the way the selection operator works Before the selection is performed, the population is ranked on the basis of an individual's nondomination, as it is described in the previous Section. The nondominated individuals are assumed to constitute the first nondominated front in the population and are assigned a large dummy fitness value. The same fitness value is assigned to give an equal reproductive potential to all these nondominated individuals. In order to maintain diversity in the population, these classified individuals are then *shared* with their dummy fitness values. Sharing is achieved by performing selection operation using degraded fitness values which are obtained by dividing the original fitness value of an individual by a quantity proportional to the number of individuals around it. This causes multiple optimal points to co-exist in the population. After sharing, these nondominated individuals are ignored temporarily to process the rest of population in the same way to identify individuals for the second nondominated front. This new set of points is then assigned a new dummy fitness value which is kept smaller than the minimum shared dummy fitness of the previous front. This process is continued until the entire population is classified into several fronts.

The sharing in each front is achieved by calculating a sharing function value between two individuals in the same front. We used the sharing function in Eq. (5).

$$Sh(d_{ij}) = \begin{cases} 2 - \dfrac{d_{ij}}{\sigma_{share}}, & \text{if } d_{ij} < \sigma_{share} \\ 1 & \text{otherwise} \end{cases} . \quad (5)$$

In the above equation, the parameter d_{ij} is the phenotypic distance between two individuals i and j in the current front and σ_{share} is the maximum phenotypic distance allowed between any two individuals to become members of a niche. A parameter *niche count* is calculated by adding the above sharing function values for all individuals in the current front and dividing the sum with the number of individuals at that front. Finally, the shared fitness value of each individual is calculated by dividing its dummy fitness value with its niche count.

Figure 8 shows a flow chart of the NSGA algorithm. The algorithm is similar to the simple GA (Fig. 7, Section IV-B), except for selection operation, which is done by classifying the nondominated fronts, as well as the sharing operation.

With the above technique, NSGA performs a ranking classification according to the nondominance of the individuals in the population and a distribution of the nondominated points is maintained using a niche formation technique. Therefore, distinct nondominated points are found in the population.

VI. EXPERIMENTS

A. Single-Objective Optimization

In this Section, we use the genetic algorithm described in the Section IV-B, in order to find the optimal arrangement of n cameras in a virtual 3D room environment.

The system is set up as the following:

1. We use $n = 3$ cameras.
2. The environment is a 4.90[m] x 4.55[m] x 2.3[m] virtual 3D room as shown in Fig. 1(a).
3. Each simulated camera has a zoom lens of 3[mm], a horizontal viewing angle of 42 [deg] and a 4/3 image width – height ratio.
4. The height at which we assume the robot is executing home service tasks, and at which the visible and the total area are calculated, is set to 0.75 [m].

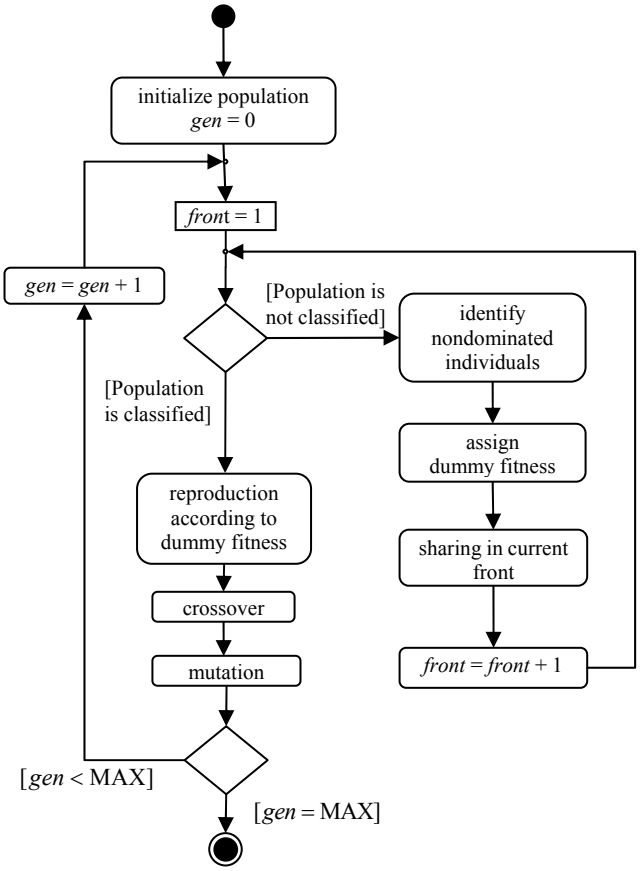


Fig. 8. Activity diagram of genetic algorithm for the multi-objective case. *gen* is the current generation index, *front* the current front index and MAX is the maximum generation number.

The parameters of the genetic algorithm are:

1. Maximum Generation Number: 1000
2. Population size: 100
3. Elitism rate: 0.1
4. Crossover probability: 0.85
5. Mutation probability: 0.01

The evolution of the best and average fitness are illustrated in Fig. 9. For the case of three cameras, it takes about 47 minutes with a Pentium D CPU (3.20 GHz) for the algorithm to converge. We can see that for the case of three cameras, the algorithm quickly converges to the visible ratio of 1. The solution that has the best fitness at the final generation is illustrated in Fig. 10.

We then change the number of cameras and apply the GA algorithm for different number of cameras. As solution, we choose the individual from the last generation which has the largest fitness value. As can see from Figure 11, if there is a required threshold for the visible ratio of the room area of 0.90, we need at least two cameras, if it is 0.95 we need at least three cameras. We optimize the pose of cameras for the same evaluation criteria using the *steepest descent method.* The steepest descent method was used for the camera placement optimization problem in [15]. Results are included in Fig. 11.

It is obvious that the GA has a better performance than the steepest descent method. The superiority of the genetic algorithm is expected, as the steepest descent method converges to local minima and its results depend on the initial arrangement of the cameras, whereas GA is a global-search heuristic which scans a large search-space.

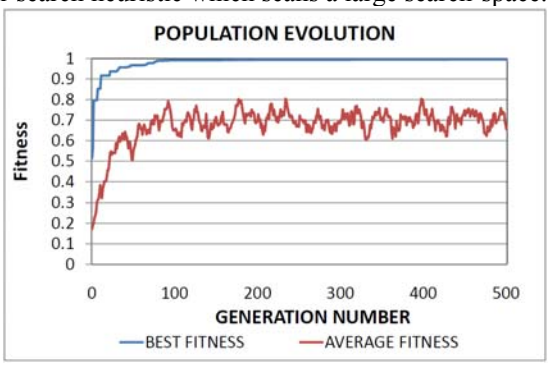


Fig. 9. Best and average fitness evolution of genetic algorithm for three cameras

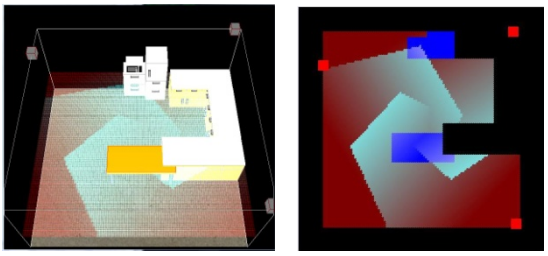


Fig. 10. Converged solution of the genetic algorithm for three cameras

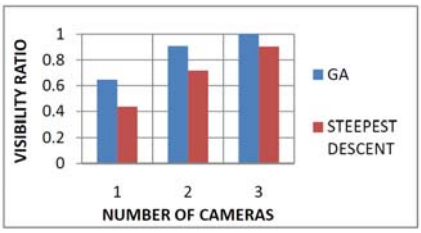


Fig. 11. Results of GA and steepest descent for different number of cameras.

B. Multi-Objective Optimization

We apply the NSGA algorithm for the problem of optimal camera arrangement. The environment set up is the same as that for the single-objective case (Section VI, Part A).

Apart from the objective functions F_1 and F_2, we have the additional constraints $F_1 \geq K_1$, $F_2 \leq K_2$. In this experiment, we chose K_1 and K_2 so that

$$K_1 = 0.75,\ K_2 = 70\ [\text{mm}^2] \qquad (6)$$

In the NSGA Algorithm, d_{ij} in the sharing function of Eq. (5) is the phenotypic distance between two individuals i and j in the current front. In this study d_{ij} is given by:

$$d_{ij} = \sqrt{({}^{i}F_1 - {}^{j}F_1)^2 + ({}^{i}F_2 - {}^{j}F_2)^2} \qquad (7)$$

In the above equation ${}^{i}F_1$ is the value of objective function F_1 for individual i. Objective F_1 is defined as the ratio of visible to total area and has a range of [0, 1], whereas objective F_2 is defined as the average localization error of the visible area. Therefore, we normalize the value of F_2 by a maximum value M, so that the values of function F_2 are approximately of the same order as with F_1, and therefore terms $({}^{i}F_1 - {}^{j}F_1)^2$ and $({}^{i}F_2 - {}^{j}F_2)^2$ in Eq. (7) contribute equally to the value of d_{ij}.

The parameters of the NSGA genetic algorithm are:

1. Maximum Generation Number: 1000
2. Population size: 1000
3. Crossover probability: 0.85
4. Mutation probability: 0.01
5. At the sharing function of Eq. (5), we chose the normalizing value M of F_2 to be 100 [mm^2] and the value of σ_{share} to be 0.02

In Fig. 12, the initial and final population of the NSGA are illustrated. The horizontal axis represents the average localization error (F_2 objective), whereas the vertical axis the ratio of visible to total area (F_1 objective). The population is "pushed" to the upper left of the two axes, as the genetic algorithm is minimizing the average localization error and maximizing the ratio of visible to total area.

In Fig. 13, the final generation and its pareto front are illustrated. We can see that the NSGA algorithms converges to a set of optimal solutions (pareto front) which cover a wide range of visible ratio (F_1) and localization error (F_2) values. In case we have some additional constraints we choose from the pareto front the solutions that satisfy the required constraints. For the constraints of Eq. (6), solutions that satisfy the required constraints are inside the shaded area of Fig. 13.

We isolate the shaded area of Fig. 13 and keep only the pareto-optimal solutions in Fig. 14. From these solutions, the designer can select the solution that best fits the design of the house environment or other factors. We randomly select one solution (circled solution in Fig. 14) and illustrate it in Fig. 15.

Discussion

The greatest advantage of the pareto multi-objective algorithm is that with one execution we can acquire a set of pareto-optimal solutions that satisfy the required constraints. Therefore, the designer can select the best solution from a number of several alternatives. On the other hand, if we solved the optimization problem by using a scalar fitness function (weighted aggregation approach) we would get only one optimal solution with one execution, which would be very sensitive to the selection of the weights of the evaluation function [14]. The drawback of the implemented algorithm is the large computational load, as the ranking procedure is proven to be of $O(MN^2)$ complexity, with M the number of objectives and N the population size [16]. Given the parameters listed in Section V-B, it took 5 hours and 13 minutes for the algorithm to converge for the case of three cameras, using with a Pentium D CPU (3.20 GHz) processor.

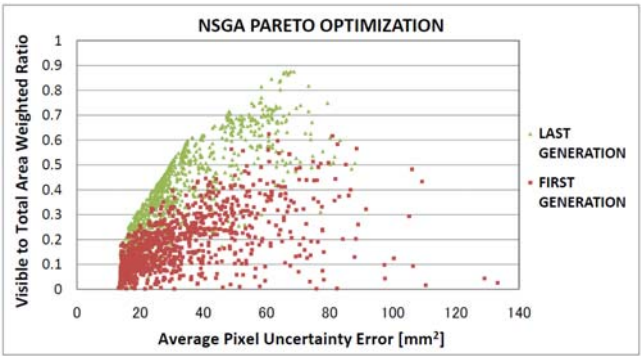


Fig. 12. Initial and final generation of NSGA algorithm

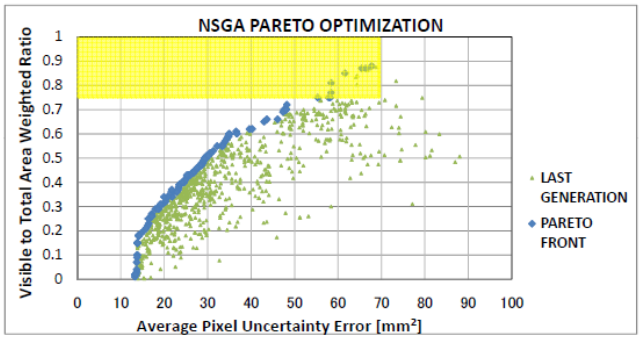


Fig. 13. Final generation and its pareto front. The yellow area represents the sub-region of the performance space that satisfies the constraints given by Eq. (6).

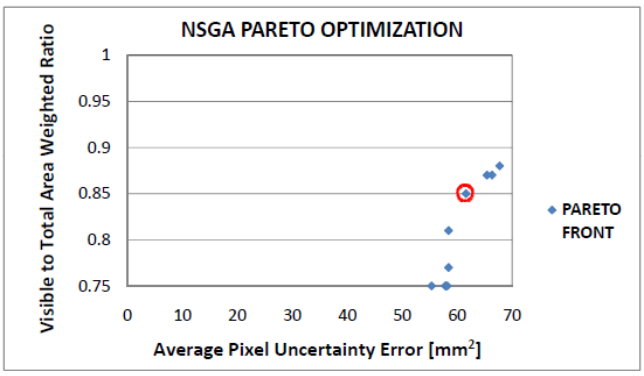
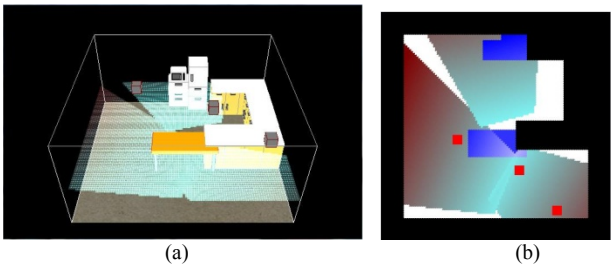


Fig. 14. Set of pareto-optimal solutions that satisfy the constraints given by Eq. (6) and selected final solution (red circle)



(a) (b)

Fig. 15. Visualization of selected solution (a) virtual 3D room environment (b) top view

VII. Conclusion

In this study, the problem of optimal camera arrangement has been examined. In particular, we:

- explained the need for precise robot localization at a home environment and verified the importance of the camera pose at the robot localization process
- divided the problem of optimal camera placement into two sub-objectives:
  a) place the cameras in order to maximize the area covered (single-objective case)
  b) place the cameras in order to maximize the area covered and minimize the average localization error of the visible area (multi-objective case)
- approached sub-objective (a) by a genetic algorithm and quantitively compared its performance with the steepest descent method used in past research [15].
- approached sub-objective (b) by the NSGA genetic algorithm.

The qualitive and quantitative results obtained from this study can be summarized as follows:

- To our knowledge, it is the first time that a four-degree optimization for a non-fixed number of cameras is achieved.
- For the single-objective case, we used the genetic algorithm to find the position on the ceiling, pan and tilt angle of the cameras so that the robot could be visible at 100% of the area of a complex home environment with obstacles
- For the multi-objective case, we found a set of optimal solutions that minimized the objective conflict between the maximization of the area covered and the minimization of the robot localization error. Then, we selected the set of solutions that satisfied the required constraints. As an example, we presented a solution where the robot is visible at 85% of the area and the average localization error is 65 [mm^2].

In the future, we intend to generalize the optimization problem of sensor arrangement by including different kinds of sensors, such as range sensors, RFID (Radio Frequency Identification) technology etc..

This research has been part of the Universal Design Project [9] and the results from this study have been used for the arrangement of the ceiling cameras at the u-RT (ubiquitous Robot Technology) Kanagawa House Square Experimental Room.

Appendix

In this study we assumed that the robot is tracked and localized by one or more ceiling cameras through image processing. To prove the validity of our assumption, we implemented a system of robot 3D localization with multiple ceiling cameras at a real home environment. We used a red Roomba floor-cleaning mobile robot (made by iRobot) and four pan/tilt cameras (SNC-RZ50N made by Sony) attached on the ceiling. Color information of the robot and of a marker placed on top the robot are stored in a database. Then, we use a slightly modified version of the CamShift algorithm [17], to online track the robot using the color information stored in the database. For stable lighting conditions, we could continuously track and localize the robot with a small localization error. Further details of the localization system developed are beyond the scope of this paper.

References

[1] S. Thrun et al, *Probabilistic Robotics*, MIT Press. 2005.

[2] Triggs B, Laugier C, "Automatic camera placement for robot vision tasks", *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1732-1737, 1995.

[3] Kececi F, Tonko M, Nagel H-H, Gengenbach V: "Improving visually servoed disassembly operations by automatic camera placement", *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2947-2952, 1998.

[4] Lee, J.H., Akiyama, T., Hashimoto, H., "Study on optimal camera arrangement for positioning people in intelligent space", *IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, pp.220-225, 2002.

[5] He X, Benhabib B, Smith K C, Safaee-Rad R: "Optimal camera placement for an active-vision system", *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1, pp. 69-74, 1991.

[6] D. T. Lee, "Computational Complexity of Art Gallery Problems", *IEEE Transactions on Information Theory*, Vol. 32, Issue 2, pp. 276-282, March 1986.

[7] O'Rourke J, *Art Gallery Theorems and Algorithms*, Oxford University Press, New York, 1987..

[8] Richard Wright *et al.*, *OpenGL SuperBible*: *Comprehensive Tutorial and Reference*, 4th Ed., Addison-Wesley Professional, 2007.

[9] K. Ohara et al., "Ubiquitous Robotics with Ubiquitous Functions Activate Module", *Proc. INSS 2005*, pp. 97-102, 2005.

[10] Tarabanis K.A., Tsai R.Y., Allen P.K.: "The MVP sensor planning system for robotic vision tasks", *IEEE Transactions on Robotics and Automation*, Vol. 11, pp. 72-85, Feb. 1995.

[11] K. Tarabanis and Tsai, R. Y., "Computing occlusion-free viewpoints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, pp. 279-292, March 1996.

[12] J**.** Holland, *Adaptation in natural and artificial systems. Ann* Arbor University of Michigan Press, 1975.

[13] Michael D.Vose, *The Simple Genetic Algorithm*, The MIT Press, 1999.

[14] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithm," *J. Evolutionary Computation*, Vol. 2, pp. 221-248, 1995.

[15] Stefanos Nikolaidis, Ryuichi Ueda, Akinobu Hayashi, Tamio Arai, "Optimal Camera Placement Considering Mobile Robot Trajectory," *IEEE International Conference on Robotics and Biomimetics*, 2008 (to appear).

[16] Deb K., Pratap, A., Agarwal, S., Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II**",** *IEEE Transactions on Evolutionary Computation*, Vol. 6, Issue 2, pp. 182-197, April 2002.

[17] Bradski, G. R., "Computer vision face tracking for use in a perceptual user interface", *Intel Technology Journal,* 2nd Quarter, 1998.