Global and local coverage maximization in multi-camera networks by stochastic optimization

Krishna Reddy Konda and Nicola Conci

***Abstract*—In this paper we present a camera positioning and reconfiguration algorithm for complex indoor environments. The algorithm initially optimizes the global coverage of the selected environment in order to maximize the visibility on the entire area. Reconfiguration of the devices is then performed after the camera installation, in order locally optimize coverage according to the application requirements.**

Both initial coverage optimization and reconfiguration are achieved using Particle Swarm Optimization (PSO). The proposed solution has been validated in different setups, also taking into account occlusions and blocking introduced by the presence of obstacles. The achieved results confirm the viability of the approach in both positioning and reconfiguration, also in presence of considerably complex environmental geometries.

I. Introduction

The reduction in price of video sensors and the ever increasing need for security are significantly contributing to the diffusion of video surveillance systems. The large availability of different types of cameras and lenses let the user customize the sensing infrastructure to achieve the desired area coverage not only by choosing the number of sensors to be deployed, but also selecting their features in terms of field-of-view, resolution, frame rate, indoor/outdoor or night/day operating modes. The large availability of different types of devices and the corresponding number of parameters that can be tweaked provide on the one hand a higher degree of flexibility, but at the same time complicate the setup procedures of the acquisition system, often resulting in a suboptimal configuration that could lead to a reduced efficiency of the entire sensing infrastructure [1].

The availability of automatic planners to choose the optimal positioning of the sensors would be of great help for the security personnel, by improving the quality of coverage, minimizing the number of sensors and the black spots, and including in the optimization model also the presence of obstacles, areas subject to privacy constraints, and other personalization factors.

This kind of instruments would be in fact particularly suitable to plan the deployment of fixed installations (buildings, offices, public spaces), but also in need of temporary deployments (e.g., sports events, fairs, exhibitions) where a fast and efficient planning would be highly desirable.

More in general, the goal of a camera planner is to guarantee the maximum coverage of the observed space, minimizing occlusions, and obtaining the best possible visibility of the areas of major interest.

The problem of coverage maximization can be easily described through the so-called *art gallery problem* (AGP), where the minimum number of guards is to be determined for a given area [2]. A variant of the AGP is known as the *Watchmen Tour Problem* (WTP), where guards are allowed to move inside the polygon [3]. The objective is to determine the optimal number of guards and their route to guarantee the detection of an intruder. However, both AGP and WTP related algorithms are unsuitable for most real-world camera placement applications. Consequently, more sophisticated algorithms have to be adopted to take into account the most important parameters related to the surrounding environment, which include constraints on observability [4], but also camera and illumination parameters. A short overview on the most relevant literature is reported hereafter.

Earlier solutions belong to the so-called *Generate and Test* approaches, in which all the constraints and models are incorporated into a simulation model. The HEAVEN system [5] is one of the earliest tools using such approach. HEAVEN uses a spherical representation to model the sensor configuration. A geodesic dome is created around the object, tessellating the sphere with an icosahedron that is further subdivided in a hierarchical fashion by recursively splitting each triangular face into four new faces. A similar system called ICE (Illumination Control Expert) [6], includes also the planning of illumination sources.

Synthesis approaches model the constraints as analytic functions, and formulating the problem in terms of satisfaction of constraints. Each requirement generates a geometric constraint, which is satisfied in the 3D domain of admissible locations. The admissible domains obtained are then intersected to each other in order to determine locations that satisfy all constraints simultaneously. An early work in this area is proposed by Cowan and Kovesi [7], in which camera locations are generated also with respect to illumination planning. In [8] the authors propose a camera placement algorithm based on a binary optimization technique and using polygonal spaces presented as occupancy grids. In [9] camera views are optimized so as to provide the highest resolution of objects and motions in the scene. The optimal view is defined by the application scenario (e.g., motion recognition, visual metrology).

Expert Systems address instead the high-level aspects of the problem, informing, for instance, about whether front or back illumination is more appropriate for the particular object and the corresponding features to be observed. An expert system is primarily used as an advising tool. A good

Manuscript Submitted October 7, 2012, revised February 14, 2013.

Krishna Reddy Konda and Nicola Conci are with the Department of Information Engineering and Computer Science, University of Trento, Via Sommarive 5, Trento, 38123, Italy, {konda, conci}@disi.unitn.it

example is the system LIGHTING ADVISOR [10], which provides advice regarding the best lighting configurations in given circumstances.

Among the most recent proposals in camera positioning and reconfiguration algorithms, the work by Mittal and Davis [11] presents a probabilistic framework for object visibility in a multi-sensor environment. Piciarelli et al. [12] address the problem of camera networks reconfiguration, by adjusting pan, tilt, and zoom. They use the Expectation Maximization algorithm, in order to maximize the coverage of salient portions of the observed scene, where the saliency is identified by motion activity maps. In [13], the authors propose a framework for target coverage based on the spatial decomposition of the network and optimizing the solution for individual partitions. Erdem and Sclaroff [14] maximize the visibility in multi-camera networks based on radial sweep. In [15] event based network re-configuration of cameras is presented, even though positioning and reconfiguration is foreseen for moving cameras, which is not in line with generic surveillance scenarios, where the absolute position of cameras is generally fixed. Similarly, a reconfiguration algorithm for continuous tracking is proposed in [16]. However, the camera model used in the paper may result too simplistic since it does not consider various camera parameters that might vary based on reconfiguration. Dieber et al. [17] propose an algorithm, which self optimizes the positioning of mobile aerial cameras based on a simple camera model, which assumes fixed coverage for all the cameras. In this paper we propose an automatic camera positioning tool, which aims at addressing both the *global* and the *local* coverage issues, where by *local* we define critical areas in the area like doors, windows, objects of interest. The work stems from an early solution to the optimization problem, presented in [18]. The main novel aspects of this work consists of two main items. The first one is given by the concept of *quality of view* for the camera model, by defining the optimal region in the image plane for the observation of specific objects and targets. Secondly, we also address the issue of sensors reconfiguration after initial positioning, achieved by adjusting the camera parameters on the basis of the requirements imposed by the events occurring in the scene.

II. Particle Swarm Optimization

Particle Swarm Optimizer (PSO), developed by Kennedy and Eberhart [19], is a robust stochastic search technique based on the movement and intelligence of swarms. It has demonstrated to be effective in solving complex non-linear multidimensional discontinuous problems in a variety of fields [20]. Unlike other multiple-agent optimization procedures such as Genetic Algorithms (GA) [21], PSO is based on the cooperation among the agents rather than their competition. Three main advantages of the PSO over the GA can be identified. In the first place, PSO requires a reduced algorithmic complexity, since it considers only one simple operator that is the particles velocity updating, while the GA uses three genetic operators and the best configuration among several options of implementation needs to be chosen. Then, PSO parameters are easier to calibrate and to manipulate than the GA ones, whose optimal values must be evaluated among various operators. Finally, PSO has a major ability to prevent the stagnation of the optimization process than GA, thanks to a more significant level of control of its parameters [22][23].

During PSO optimization procedure, the particles of the swarm iteratively change their positions in the solution space, searching for the best location. The solution space is defined by selecting the parameters to be optimized and assigning them the corresponding range of variation. Consequently, each parameter models a particular dimension of the solution space, and each location in the solution space corresponds to a particular trial solution. The goodness of the trial solutions is evaluated by means of a suitable fitness function, which provides the link between the optimization algorithm and the physical world.

A short pseudocode of the PSO algorithm is shown hereafter, and the corresponding equations to compute and update the velocity of the particles are shown in (1) and (2), respectively. In the algorithm, at every iteration i, $F(j)$ represent the current fitness value for particle j, $pBest$ is the minimum fitness value obtained so far for particle j, and $gBest$ is the overall global best among all the particles.

```
input: Number of Particles
input: Number of Iterations
InitializeParticles;
for i ← 1 to Number of Iterations do
    for j ← 1 to Number of Particles do
        F(j) = Fitness(j);
        if F (j) <pBest(j) then
            pBest(j) ← F(j);
        end
    end
    gBest = min(pBest(j));
    for j ← 1 to Number of Particles do
        CalculateVelocity(j);
        UpdateVelocity(j);
    end
end
```

Algorithm 1: Pseudocode of the PSO.

$$v(j) = v(j) + c_1 * rand() * (pBest(j) - F(j) + c_2 * rand() * (gBest - F(j)) \quad (1)$$

$$F(j) = F(j) + v(j) \quad (2)$$

In (1) c_1 and c_2 are configuration parameters defined in literature, and $rand()$ is a random number defined in $[0, 1]$.

III. The proposed approach

In our approach we propose to optimize the coverage using a global and a local model as an analytic function of the camera parameters (positional degrees of freedom, pan, tilt and zoom), where the global component is defined to take the environment configuration into account, while the local

component considers the position of objects of interest in the scene.

Initially, global optimization is achieved finding a suitable positioning of the sensors in the environment. Once the position of the cameras have been fixed, in case of need (e.g., presence of new obstacles, objects, changes in the environment) reconfiguration is run in order to maximize the local coverage.

A. Global coverage

In order to model global coverage, we have defined two different conditions to be met: *pixel density* and *quality of view*.

Pixel density refers to the fact that the information obtained from an image or a video is dependent on the number of pixels per surface area of the environment. Pixel density can be modeled as a function of the field-of-view and the resolution of the camera. In order to model the pixel density, we propose to represent the camera as a point source and each pixel of the CCD as the corresponding ray that emerges from the point source. Accordingly, far away areas in the environment receive less rays, corresponding to lower pixel density, whereas areas closer to the camera will be intersected by a higher number of rays, thus achieving higher resolution. An example about the mapping of the grid in the floor plan to the camera CCD is illustrated in Fig. 1.

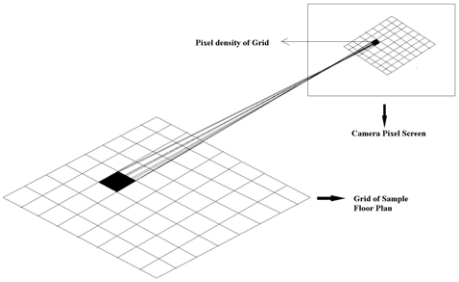


Fig. 1. Pixel Mapping. Each area of the floor plan is captured by a number of pixels that depends on the distance from the camera.

The *quality of view* of a target refers instead to the relationship between feature detectability of the target and the distance from the camera. Given a sensor resolution, the recognition of an object will strongly depend on its distance from the camera: if it is too distant details will be unintelligible; if it is too close, the whole object might not be visible entirely due to the limited field-of-view of the camera. To take into account this parameter, we start from the pixel density information defined in the previous paragraph, so as to model the visibility constraint as a Gaussian distribution that is computed along the ray emerging from the camera (see Fig. 2. The optimal distance is located at the center of the Gaussian, and needs to be specified according to the size and type of the objects to be monitored (e.g., humans, cars), such that they are clearly visible in case they enter the field-of-view of the camera.

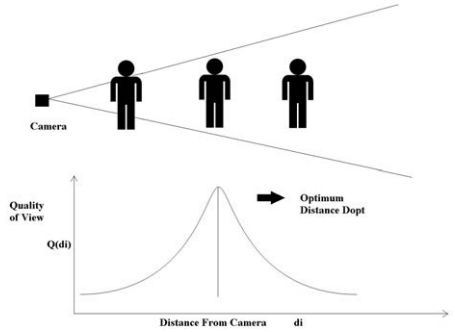


Fig. 2. Quality of view. The optimal distance for observation depends on the kind of objects to be monitored in the specific scene.

B. Local Coverage

As far as the local coverage is concerned, each area of interest (doors, windows, statues, paintings, other objects, etc.) is modeled as a negative exponential function of global coverage at the local target location, and it is expressed as a function normalized between zero and one, as shown in (3). As we can see from the equation the function reaches its maximum when the object is at the cell which has the maximum global coverage. Further details about the parametrization of all the above mentioned elements are provided in next sections.

$$T_k = 1 - exp\left(-C_G\right) \tag{3}$$

C. Camera Model

The parameterization of the camera model, as discussed above, includes both aspects of pixel density, and quality of view.

All simulations we will present are carried out on the ground plane, thus discarding the vertical dimension. From the optimization point of view, the extension to the third dimension is straightforward. However, it is worth noting that in most scenarios the height term is less relevant, because it is common sense to position the cameras either on the ceiling or on the walls, and at the same height. This limits on the one hand the accessibility to non authorized users, and on the other hand it improves the visibility of the area to be monitored.

The quality of view of an object is modeled as a Gaussian distribution, evaluated along the ray emerging from each pixel of the camera, as shown in (4):

$$Q(d_i) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(d_i - D^{opt})^2}{2\sigma^2}\right) \tag{4}$$

where d_i is the distance of the cell hit by the i-th ray, and D^{opt} is the optimum distance, at which the objects of interest has maximum quality of view. In the current scenario D^{opt} is chosen as a constant.

In order to determine the areas that are visible in the map, we have to define a metric of visibility for the quality of view function. A given cell in a grid is said to be covered if the total quality of view, as defined by summation of $Q(d_i)$ of individual rays that pass through cell, is greater that a certain threshold. The threshold value has been empirically set to 0.5.

As far as the reconfiguration is concerned, and considering that we model the environment as a 2D plane, the camera parameters we consider for reconfiguration are *pan* and *zoom*, ignoring the *tilt* component. Pan is defined as the horizontal orientation of the camera, and zoom is in general expressed as a combination of two components, namely *optical* and *digital zoom*.

Optical zoom corresponds to imposing a change in the focal length of the camera. Using optical zoom enables us to capture a picture from a near view without any change in image quality. Mapping optical zoom into our model, basically implies to shift the quality of view function by the corresponding amount of zoom applied. For instance if the optical zoom applied is of a factor N the (4) is modified as in (5).

$$Q(d_i) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(d_i - N * D^{opt}))^2}{2\sigma^2}\right) \tag{5}$$

Digital zoom allows instead for a closer view on the target by decreasing the image resolution on the object of interest and resampling. This operation works reasonably well when the object is within a limited distance from the camera and becomes less effective as soon as the object of interest is moves far away. In this second case, and in order to be effective, digital zoom is to be used in combination or in cascade with optical zoom.

In our model we simulate zoom by decreasing the field of view of the camera, while keeping the number of rays that correspond to pixels in the camera constant. The rays intersecting the object of interest will then increase.

D. Algorithm

The proposed algorithm can be described in six steps, as explained here after.

Step 1 - Determine the solution space. If the task at hand is initial positioning of cameras, we need to identify the solution space, consisting on the perimetral and internal walls, and defined in terms of position and maximum orientation span. In case the provided map also includes the presence of target objects of interest, given the number of cameras, the algorithm will optimize the position of the devices, by either focusing more on global or local coverage depending on the input requirements.

Similarly, if the task at hand is reconfiguration, the solution space consists of all pan and zoom configuration allowed for the devices, given their initial position.

Step 2 - Compute Global coverage. In order to calculate the global coverage, the environment map is divided into a grid of $N \times N$ pixels. The granularity of the grid is chosen depending on the map scale, as well as on the desired accuracy in positioning. The finer the grid, the more accurate will be the result, at a cost of a higher computational complexity. For each camera, the pixel density (i.e., the number of rays passing through each cell of the grid) is computed. While estimating the number of rays, obstructions caused by the obstacles are also taken into account. The higher the number of rays that cover a grid cell, the higher the pixel density, as calculated in (6):

$$C(m,n) = \sum_{i=0}^{R} Q(d_i) \tag{6}$$

where $C(m,n)$ is the final quality of view metric obtained for a specific cell, and m and n give the location of the cell in the map. As can be seen from (6), this metric will weight the quality of view function measured as in (4) considering the number of rays that intersect the cell (R). Conversely, we can say that the number of pixels occupied by a particular cell in the video frame is directly proportional to the number of rays that pass through that cell in the grid.

We then label the cell as "visible" only if the quality of view is higher than a predefined threshold. The global coverage is estimated as the number of visible cells divided by total number cells in the grid (7).

$$C_G = \frac{Cells_{visible}}{Cells_{total}} \tag{7}$$

Step 3 - Include Local coverage. For a given camera position the local coverage is given by (5). Accordingly, the overall local coverage is given by (8):

$$C_T = \frac{1}{T} \sum_{k=0}^{T} T_k \tag{8}$$

where T is the total number of target objects and T_k is the target coverage given by (3) for the corresponding target.

Step 4 - Fitness Function. We need now to define a suitable fitness function that will be used by the PSO algorithm as a target for the optimization procedure. The proposed fitness function combines both global and local coverage, and each term can be weighted according to the users' preferences and the application requirements (9).

$$F(C_G, C_T) = (1 - C_G) * w + (1 - C_T) * (1 - w) \tag{9}$$

In (9), C_G represents the global coverage, and C_T models the local (target) coverage; w is the weight used to balance the trade-off between global and local coverage, respectively. We can notice from (9) that, as soon as the global and local coverage approach 1 (maximum coverage), the fitness function converges to zero.

Step 5 - PSO. PSO is applied to the solution space defined in *Step 1*. At each iteration, the particles position and velocity are updated, until convergence. Convergence is usually achieved when the fitness function reaches a minimum, or when a termination criterion is fulfilled (e.g., maximum number of iterations).

Step 6- Check for changes in the environment In case after the initial camera deployment, changes in the environmental conditions occur (thus requiring a reconfiguration of the network) the algorithm can be run again and the optimization will in this case only focus on the pan and zoom parameters.

E. Fitness function

The fitness function is the most critical element of the whole algorithm, as it determines the behavior of the particle

swarm and also the convergence characteristics of the PSO algorithm. Typically an ideal fitness function should have following characteristics

- it should get minimized when the intended output is maximized;
- minimization should be applicable even if only one of the involved parameters is varied while keeping all other parameters constant;
- each and every parameter that is involved in the optimization process should get due representation in the fitness function.

In line with the above requirements our fitness function has been formulated in (9). As can be seen, the fitness function is the weighted summation of local and global coverage. The overall algorithm and the computation of the fitness function is shown by pseudo code in Algorithm 2.

input : Map I divided into $N \times M$ cells of equal size
input : Camera Resolution
output: Fitness Value

Initialize Camera Positions from PSO;
Initiate number of rays from camera based on video resolution;

QOV ; % Quality of View of the Cell

QOL ; % Intensity of light perceived by the cell

goodCells = 0 ; % Number of cells with good coverage

C = 0; % Obtained Coverage for each cell

for $i \leftarrow 1$ **to** N **do**
  for $j \leftarrow 1$ **to** M **do**
    pixelDensity $\leftarrow$`RayIntersect(`i,j`);`
    QOV $\leftarrow$ `QualityOfView(`i,j`);`
    QOL $\leftarrow$`LightIntensity ;` % Constant
    $\mathsf{C}(i,j) = \mathsf{pixelDensity} * \mathsf{QOV} * \mathsf{QOL}$;
    if $\mathsf{C}(i,j) >= \mathsf{C}_{th}$ **then**
      goodCells ++ ;
    end
  end
end

$\mathsf{C}_g \leftarrow \frac{\mathsf{goodCells}}{N*M}$; *% Compute Global Coverage*

$k = \mathsf{T}$;
while $k \neq \emptyset$ **do**
  % Compute Target Coverage
  $\mathsf{T}(k)$ = `Gaussian(`D,C *(m,n)*,QOL`)` ;
  $\mathsf{C}_T = \mathsf{C}_T + \frac{\mathsf{T}(k)}{T}$;
  k - - ;
end
% The final output is a combination of Global and Local coverage with weights w_G and w_T
$F(\mathsf{C}_G, \mathsf{C}_T) \leftarrow w * (1 - \mathsf{C}_G) + (1 - w) * (1 - \mathsf{C}_T)$;
Return $F(C_G, C_T)$;

Algorithm 2: Fitness Calculation

IV. Scenarios

Without loss of generality and considering that cameras are usually positioned at the same height, simulations are performed in two dimensions, thus discarding the height coordinate. Moreover, the number of rays corresponding to the pixels is downsampled by a factor 4, in order to make the computational complexity tractable. Considering a standard camera resolution of 640x480 pixels, this implies using 160 rays emitted by each camera, which still represents a fairly dense sampling of the space. The field-of-view is fixed in the range between 5 and 90 degrees and optimum distance for quality of view is fixed at 40 pixels in the map, which corresponds to about 10 meters in the real environment.

In order to assess the validity of our approach, we tested the algorithm on three different maps. As far as the simulation procedure is concerned, we initially determine the cameras position on the map, assuming that no object is present. This is equivalent to optimizing only with respect to global coverage.

After the initial setup, ten different objects of interest are placed in the map. At this point the algorithm is required to re-align the cameras, keeping the positioning of the sensors fixed. This implies that in determining the new camera parameters, only local coverage is considered, thus setting $w = 0$.

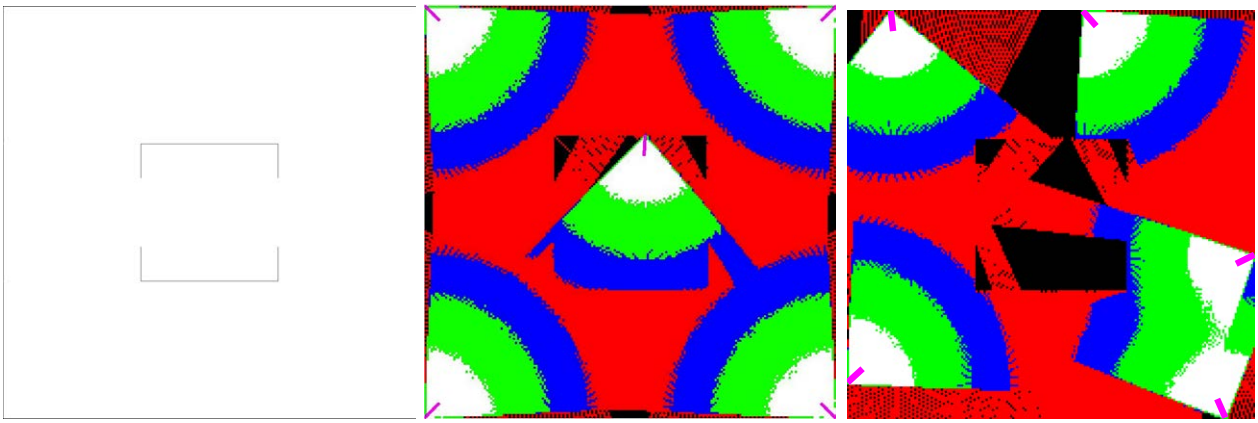
The environment maps used for the testing are shown in the left column of respective Fig. 4-6. Cameras can be positioned along internal and perimeter walls of the environment. In the picture we also show the positioning of the targets that will be introduced after the initial setup of the camera infrastructure is found.

V. Experimental Results

As explained in Section IV, we will present the results obtained in the selected scenarios by first illustrating the quality of the global coverage achieved in the initial positioning, and then focusing on reconfiguration for local (target) coverage.

Initially, the environment in which the cameras have to be deployed, do not include targets. Hence, the goal of initial placement is global coverage maximization. In order to do so, we assume that initially cameras are zoomed out (maximum field of view). At this stage, the aim of the algorithm is to find the best position to achieve optimum global coverage. In the maps that will be presented, different colors are used to illustrate the quality of the coverage in over the entire map. For global coverage, areas with maximum coverage (i.e. when $C(m,n)$ is greater than 100) are represented in white, and areas which fall in the range $10 <= C(m,n) < 100$ are represented in green. Blue indicates areas, which satisfy $0.5 <= C(m,n) < 10$. Red areas represent zones of the environment, which are visible to cameras but fall below our visibility threshold of 0.5, and black areas are not visible to cameras because they fall out of the field-of-view, or due to the presence of obstacles.

It is worth mentioning that the coverage will in general hardly reach 100%, since the coverage function decreases with the distance of the pixel from the camera position. As an example, to show the effectiveness of the global coverage optimization we have applied our algorithm on a simple map,



(a) (b) (c)

Fig. 3. Global coverage maximization. Sample map (a), optimal coverage (60%), proposed method (52%).

shown in Fig. 3, for which the optimal positioning (about 60%) is known. Through our simulation and after 50 iterations, we could achieve 52%, demonstrating that PSO turns out as a good alternative to deterministic algorithms.

After the initial positioning is completed, targets are randomly distributed over the map. The goal of reconfiguration is to maximize target coverage; however, in most surveillance scenarios camera deployment is fixed and does not allow repositioning after installation, unless PTZ cameras are used. Hence, according to our model the only reconfigurable parameters are pan, and zoom. The algorithm is re-run considering the absolute position of the cameras fixed, thus optimizing target coverage. The input maps used for the three test cases are shown in Fig. 4-6. Initial positioning is performed with the zoom set to 1x, while pan is a free parameter that can be adjusted to obtain maximum coverage. As far as the first map is considered, the coverage level obtained after the initial placement of cameras is shown in Fig. 4(b). In the figure, the environment is overlapped for convenience with the output coverage map obtained from the algorithm. The percentages of global coverage C_G are shown in Table I.

However, we can notice that although the overall coverage percentage is good, this is not optimized when the targets are deployed, and reconfiguration is required.

TABLE I
GLOBAL AN LOCAL (TARGET) COVERAGE FOR THE THREE MAPS USED FOR TESTING.

Map	Config.	C_G	Green	Blue	Red	Black	C_T
1	Initial	0.51	5	1	3	0	0.60
	Final	0.50	5	3	1	1	0.87
2	Initial	0.54	2	1	5	1	0.58
	Final	0.44	7	1	0	1	0.78
3	Initial	0.51	2	3	3	2	0.54
	Final	0.48	7	1	0	2	0.80

As can be seen from the figures in Table I and as depicted in Fig. 4(c), after reconfiguration, C_T has increased quantitatively and qualitatively. In fact, initially three targets are left out with average coverage of 0.60, while after reconfiguration the average coverage has increased to 0.87 by almost 50% with only 2 targets left out.

In a similar fashion, we have conducted the the initial positioning and the re-alignment of the cameras also on the remaining two maps. The obtained maps after initial global coverage optimization are depicted in Fig. 5(b) and Fig. 6(b), respectively. Similarly, the final camera setup after reconfiguration is shown in Fig. 5(c) and Fig. 6(c), respectively. Similarly to the previous experiment, numerical results are presented in Table I.

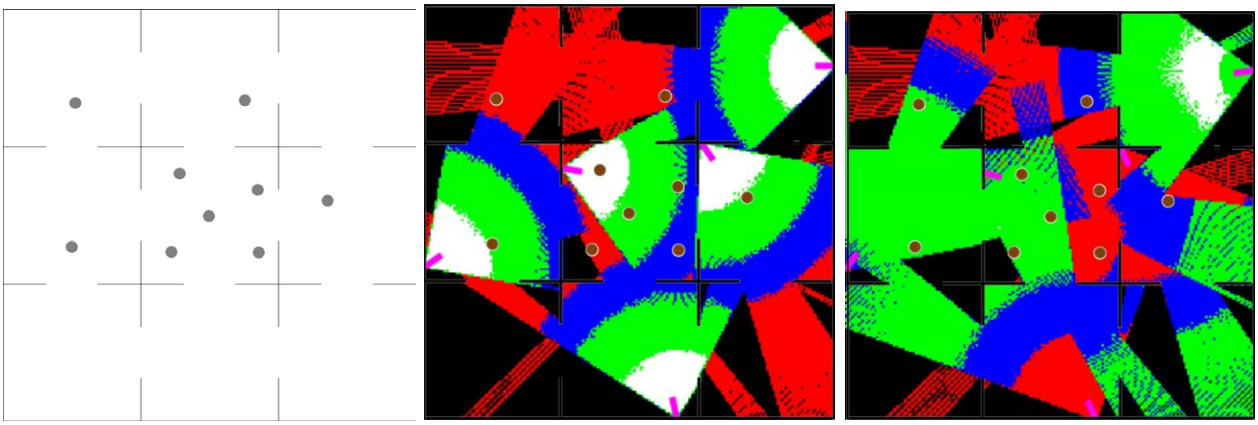
From the experiments it is reasonable to say that the performance of the algorithm is consistent across different environments, maintaining in all cases more than 50% improvement in the target coverage.

VI. CONCLUSIONS

In this paper we have proposed a tool for automatic positioning and reconfiguration in multi-camera networks using the PSO algorithm. The algorithm aims at dealing with both global and local coverage, that can be set with tunable weights. The coverage is defined as the fulfillment of different quality parameters, including pixel density, quality of view, and the presence of obstacles. The experimental validation carried out on different environmental setups demonstrate the efficiency of the algorithm in achieving a good coverage of the observed space, also in presence of obstacles and targets.

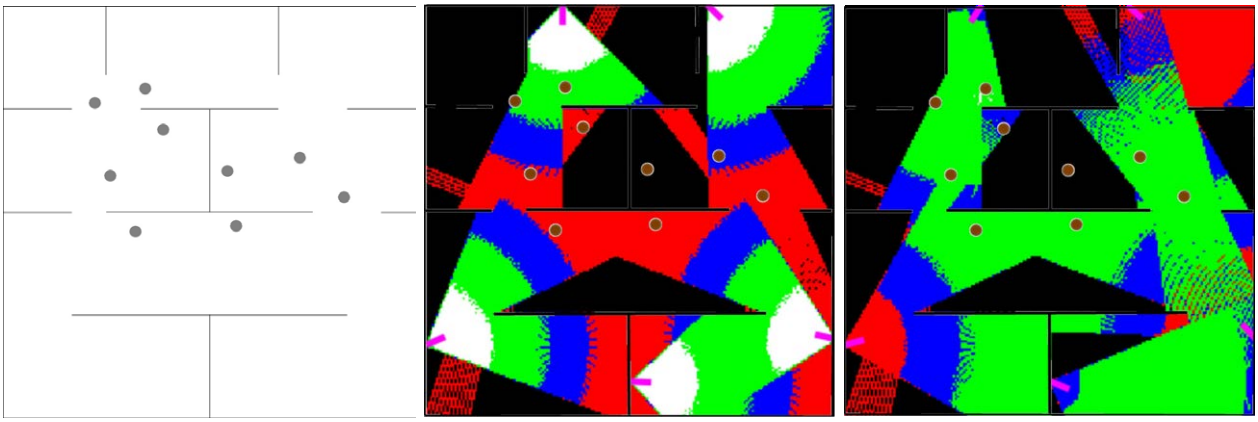
REFERENCES

[1] Hoerster, E., Lienhart, R.: Optimal placement of multiple visual sensors. In: Multi-Camera Networks. (2008) 117–138
[2] Chvatal, V.: A combinatorial theorem in plane geometry. (1975) 39–41
[3] Carlsson, S., J.Nilsson, B., C.Ntafos, S.: Sweeping simple polygons with a chain of guards. (1991) 367–378
[4] Shafer, S.: Automation and calibration for robot vision systems. Technical report, Technical Report CMU-CS-88-147, Carnegie Mellon University (1988)
[5] Sakane, S., Ish, M., Kakikura, M.: Occlusion avoidance of visual sensors based on a hand-eye action simulator system: Heaven. Advanced robotics **2** (1987) 149–165
[6] Sakane, S., Niepold, R., Sato, T., Shirai, Y.: Illumination setup planning for a hand-eye system based on an environmental model. Advanced Robotics **6** (1991) 461–482
[7] Cowan, C., Bergman, A., Nitzan, D.: Automatic placement of vision sensors. In: 1990 NSF Manufacturing System Research Conference. (1990) 389–395



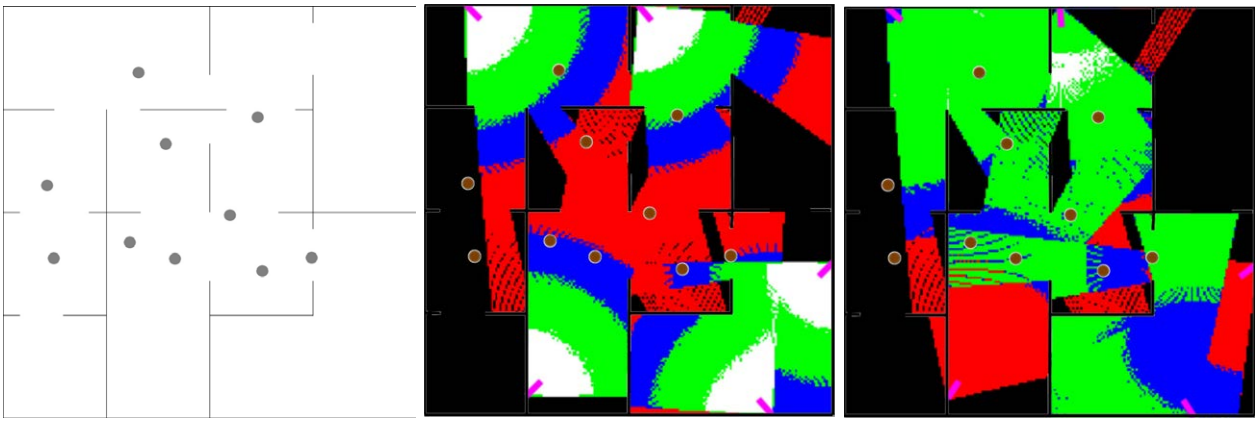
(a) (b) (c)

Fig. 4. Map 1 (a), initial positioning (b), and reconfiguration (c).



(a) (b) (c)

Fig. 5. Map 2 (a), initial positioning (b), and reconfiguration (c).



(a) (b) (c)

Fig. 6. Map 3 (a), initial positioning (b), and reconfiguration (c).

[8] Erdem, U., Sclaroff, S.: Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints. In: OMNIVIS Workshop. (2004)

[9] Bodor, R., Drenner, A., Schrater, P., Papanikolopoulos, N.: Optimal camera placement for automated surveillance tasks. Journal of Intelligent & Robotic Systems **50** (2007) 257–295

[10] Grimson, W.: Sensing strategies for disambiguating among multiple objects in known poses. Robotics and Automation, IEEE Journal of **2** (1986) 196–213

[11] Mittal, A., Davis, L.: A general method for sensor planning in multi-sensor systems: Extension to random occlusion. International Journal of Computer Vision **76** (2008) 31–52

[12] Piciarelli, C., Micheloni, C., Foresti, G.: Occlusion-aware multiple camera reconfiguration. In: Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras, ACM (2010) 88–94

[13] Munishwar, V., Abu-Ghazaleh, N.: Scalable target coverage in smart camera networks. In: Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras, ACM (2010) 206–213

[14] Erdem, U., Sclaroff, S.: Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements. Computer Vision and Image Understanding **103** (2006) 156–169

[15] Wittke, M., del Amo Jimenez, A., Radike, S., Grenz, C., Hahner, J.: Enra: Event-based network reconfiguration algorithm for active camera networks. In: Distributed Smart Cameras (ICDSC), 2011 Fifth ACM/IEEE International Conference on, IEEE (2011) 1–6

[16] Lemon, B., Kulathumani, V.: Local reconfiguration for simultaneous coverage and tracking in a large scale camera network. In: Technologies for Homeland Security (HST), 2011 IEEE International Conference on, IEEE (2011) 117–122

[17] Dieber, B., Micheloni, C., Rinner, B.: Resource-aware coverage and task assignment in visual sensor networks. Circuits and Systems for Video Technology, IEEE Transactions on **21** (2011) 1424–1437

[18] Conci, N., Lizzi, L.: Camera placement using particle swarm optimization in visual surveillance applications. In: Image Processing (ICIP), 2009 16th IEEE International Conference on, IEEE (2009) 3485–3488

[19] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: IEEE Int. Neural. Networks Conf. Volume 4. (1995) 1942–1948

[20] Eberhart, R.C., Shi, Y.: Evolving artificial neural networks. In: 1998 Int. Conf. Neural. Networks and Brain. (1998)

[21] Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addison-wesley (1989)

[22] Robinson, J., Rahmat-Samii, Y.: Particle swarm optimization in electromagnetics. IEEE Trans. on Antennas and Propagation **52** (2004) 397–407

[23] Eberhart, R.C., Shi, Y.: Particle swarm optimization: developments, applications and resources. In: Congress on Evolutionary Computation. (2001) 81–86

Krishna Reddy Konda received his B.Tech in Electronics and Communications from Jawaharlal Nehru Technological University, India, in 2005, and the M.E degree in Systems and Signal Processing from Osmania University, India, in 2007. In 2007-2009 he was a researcher in convergence S/W Research Division at Electronics at the Telecommunication Research Institute, Daejeon, South Korea. In 2009-2011 he worked in various organizations like Larsen and Toubro India, Hellosoft India, and Squid Design Systems. He is currently pursuing his Phd at University of Trento, Italy since 2011. His research interests include video compression, machine vision, and video analytics.

Nicola Conci (S'04-M'08) received the Ph.D in 2007 from the University of Trento, Italy. In 2007 he was visiting student at the Image Processing Lab. at University of California, Santa Barbara. In 2008-2009 he was post-doc researcher in the Multimedia and Vision research group at Queen Mary, University of London (UK). Since 2009 he is assistant professor at the University of Trento, Italy. His research interests are related to machine vision for behavior understanding and human computer interfaces. In 2006 he received the Best Student Paper Award at the international ACM conference Mobimedia held in Alghero (Italy).