Dynamic Camera Positioning and Reconfiguration for Multi Camera Networks

Krishna Reddy Konda

UNIVERSITY OF TRENTO - Italy

Department of Information Engineering and Computer Science

Advisor: Dr Nicola Conci

February 2015

Abstract

The large availability of different types of cameras and lenses, together with the reduction in price of video sensors, has contributed to a widespread use of video surveillance systems, which have become a widely adopted tool to enforce security and safety, in detecting and preventing crimes and dangerous events. The possibility for personalization of such systems is generally very high, letting the user customize the sensing infrastructure, and deploying ad-hoc solutions based on the current needs, by choosing the type and number of sensors, as well as by adjusting the different camera parameters, as field-of-view, resolution and in case of active PTZ cameras pan,tilt and zoom. Further there is also a possibility of event driven automatic realignment of camera network to better observe the occurring event. Given the above mentioned possibilities, there are two objectives of this doctoral study. First objective consists of proposal of a state of the art camera placement and static reconfiguration algorithm and secondly we present a distributive, co-operative and dynamic camera reconfiguration algorithm for a network of cameras. Camera placement and user driven reconfiguration algorithm is based realistic virtual modelling of a given environment using particle swarm optimization. A real time camera reconfiguration algorithm which relies on motion entropy metric extracted from the H.264 compressed stream acquired by the camera is also presented.

Keywords [Distributed camera networks,Coverage maximization,Video analytics,H.264 compressed domain]

"An idea that is developed and put into action is more important than an idea that exists only as an idea"

-Gautama Buddha-

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter we introduce the history and various applications associated with video based surveillance. After the introduction we present the relevancy of camera placement and reconfiguration for video surveillance systems. A brief overview of the research in the area of camera placement and configuration is presented. Brief introduction of the proposed solution and its innovative aspects are also discussed.

1.1 Video based surveillance

Traditionally humans have been used for the job of visual surveillance until the cusp of 20th century. Human based collaborative surveillance system is a truly distributive surveillance system as each node acts as independent yet collaborative part of the system. However since the start of the 20th century video cameras are increasingly being used for surveillance, which is quite cost effective and rigorous when compared with direct deployment of humans. The first surveillance system was installed in Germany to watch over the launch of rockets, it was designed and developed by noted german engineer Walter Bruch. In US first closed circuit surveillance system was developed by Vericon in 1949. Early surveillance systems were just used for monitoring as there was no way of recording and storing the data.

The development of reel-to-reel media enabled the recording of surveillance footage. These systems required magnetic tapes to be changed manually, which was a time consuming, expensive and unreliable process, with the operator having to manually thread the tape from the tape reel through the recorder onto an empty take-up reel. Due to these shortcomings, video surveillance was not widespread. VCR technology became available in the 1970s, making it easier to record and erase information, and use of video surveillance became more common. Development of digital video compression and advancement in storage technology using semiconductor devices has made video storage quite inexpensive. Owing to this fact surveillance footage up to a month is being stored in most of the deployed systems. Digital video is generally recorded in one of the video compression standards formulated by ITU-T.Most of the present day data is stored in H.264 [70] compressed format. Video based surveillance is used in many areas of application of which we list out few of the important.

Crime prevention An analysis published by researchers from Northeastern University and Cambridge University in 2009, discusses the impact of video surveillance on crime rate under various scenarios [69]. There was up to 51 percent decrease in crime rate in parking lots, up to 23 percent decrease in case of public transport system and up to 7 percent in general public spaces. As we can see from the study, video surveillance helps in crime reduction just by deployment of the system. There is also a possibility of surveillance footage giving us the evidence for any form of the crime that has been committed.

Industrial process Continual monitoring of work environment is essential in certain industries, where in circumstances can easily deteriorate becoming dangerous for the workers involved. In fact industries dealing in

dangerous chemical substances are required to install video surveillance under the mandate of the law.

Critical area monitoring As a part of day to day life and also some administrative duties, there are certain critical situations and tasks where there will be utmost need for caution and transparency. Some of the situations include critical areas in defence establishments, which house technological know how of the systems, and also critical financial transactions in banks and economic establishments. These kind of areas require round the clock surveillance by the stakeholders. Video surveillance is highly helpful in such situations. Markets and banks use constant video surveillance to overcome these issues.

A typical surveillance monitoring split screen after the deployment of cameras in commercial establishment looks like as in Fig.1.1. Given the variety and the complexity of environment and video feed from various cameras, there is a definite need for intelligent and collaborative framework to address the deployment, data gathering and summarization of the video streams from various cameras.

1.2 Camera planning

Given the critical tasks that a video surveillance system performs in modern day scenarios, coverage of the environment involved is critical for many of these tasks. In any given urban scenario, the number of sensors deployed as a part of surveillance system is high. Hence a network of sensors deployed has many possible configurations depending on the various combinations of camera parameters. Depending on the placement and coverage, the cost of the system deployed may vary dramatically. Centralized planning of the camera network placement and configuration will result in consider-

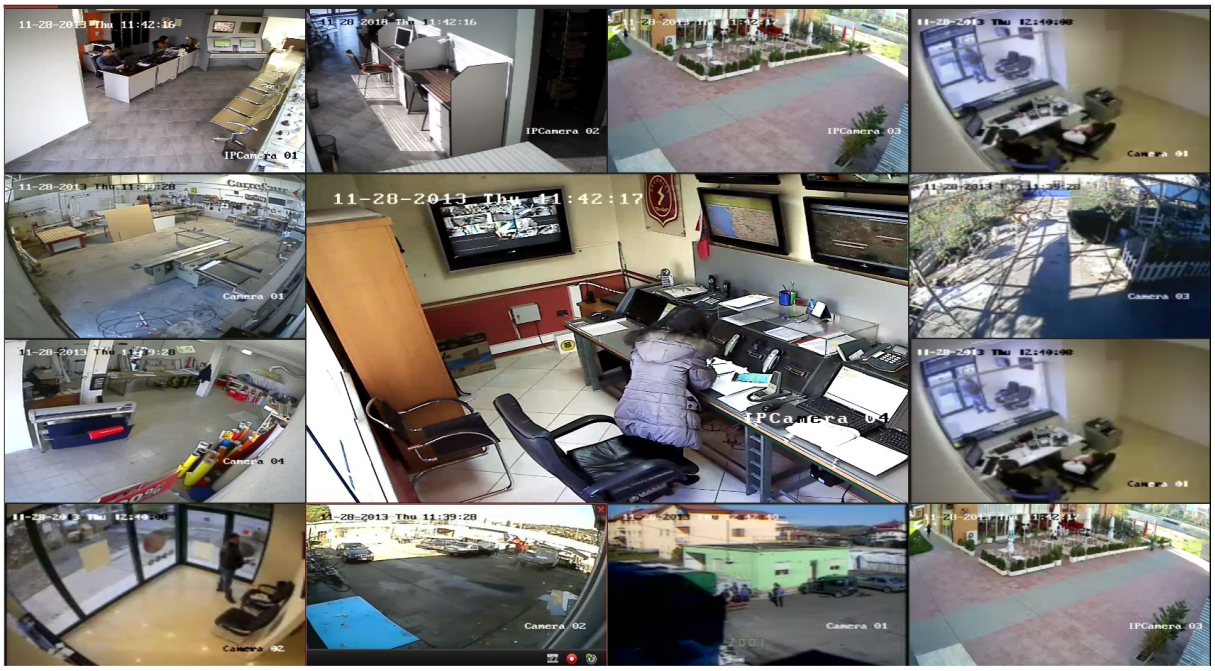


Figure 1.1: A typical surveillance split screen

able increase in efficiency of the deployed system in terms of coverage and quality of coverage. However in most scenarios the camera parameters are set manually, depending on the individual judgement of the personnel involved. Another important factor overlooked is camera placement with respect to the light placement. Such a deployment generally leads to the sub optimal video quality along with large number of blind spots in the surveillance environment. The availability of a planning tool for the automatic set-up of the sensing infrastructure given the map of the environment to be covered, would allow optimizing the configuration of the camera network, minimizing both the number of sensors and the black spots. This procedure would be useful to carefully plan fixed installations but also to achieve fast and efficient planning of temporary deployments (e.g., sports events, fairs, exhibitions), where a rapid design is required, and spaces are typically reconfigurable, as in presence of removable walls, pieces of furni-

ture, equipment, and illumination systems. Furthermore, automatic tools can be designed so as to take into account additional constraints, such as the presence of obstacles, areas that are more difficult to reach with cables, areas subject to privacy constraints etc.

1.3 Camera reconfiguration

Once the deployment of the system has been optimally achieved, there is a need for continuous monitoring of the area. As there is always chance of change in the environment or in the system itself. For example after the deployment is done, occlusions may occur partially or completely covering the view of one or more cameras. In such scenario there would be a definite blind spot in the area being previously covered by the camera. Similar issues could arise in presence of camera malfunctioning. There are also cases where in environment changes dynamically. For example in surveillance scenarios, there may be a high likelihood of people moving in one particular area at certain time and another area might have higher likelihood at another time. In such scenario cameras need to reconfigure themselves to observe people rather than pointing to areas with no movement. Reconfiguration becomes much more important and flexible when we are dealing with PTZ cameras. A system capable of detecting such changes and reconfiguring itself, would require far less number or cameras and also would capture more information of the same event at better resolution. Hence there is a need for smart camera network which reconfigures itself based on the changing environment. The advancement in camera production technology and the increased sophistication of the devices, significantly contributed to the diffusion of pan-tilt-zoom (PTZ) cameras, often replacing, or complementing the camera networks, usually consisting of ordinary static cameras. In fact, the capability of repositioning the sensors

to satisfy specific coverage requirements, tremendously increases the flexibility of the network. The freedom to change the camera pan and tilt also after the physical deployment, may also help in simplifying the topology of the network, since with a reduced number of devices capable of being reconfigured, it is possible to satisfy the requirements in coverage that would otherwise imply the use of a large number of static cameras to perform the same task. PTZ cameras can also be utilized to design an intelligent distributed smart camera network, in which information is shared between the cameras in order to perform collective tasks like reconfiguration, detection and tracking. These aspects do not only increase the monitoring capabilities of the network, but they also contribute to a better observation of the events that take place in the area, as a reconfigurable camera system can be utilized to track moving objects by continuously changing the camera parameters according to the rules defined by the system architecture.

For the reasons mentioned above, cameras reconfiguration is an active and relevant area of research. In this work we present a cooperative and distributed camera reconfiguration algorithm for PTZ camera networks, using motion field entropy and visual coverage as the metrics to be optimized. In fact, while motion field entropy can be used to represent the status variations of the monitored environment, it is however important to guarantee the maximum visual coverage of the space. Furthermore, camera reconfiguration algorithms require to be as reactive as possible, reducing the computational burden to analyze the video, and satisfying real-time operation. In order to achieve this, we base our algorithm on the H.264 [70] stream, directly provided by the camera, and carry out the analysis in the compressed stream, instead of acting in the traditional pixel domain. To reduce the computational load required by the algorithm, we utilize some existing features available in the H.264 bit stream, so as to skip the video decoding and video feature extraction.

1.4 The solution

There are two principal objectives of this doctoral study. One of the objectives is to propose a novel, low complexity camera placement and static re-configuration algorithm for multi camera network and the other is to propose a generic, smart, distributive,low complexity and dynamic reconfiguration algorithm based on video analytics of the events that occur in deployed environment.Overview of the system that we propose to develop is shown in Fig.1.2

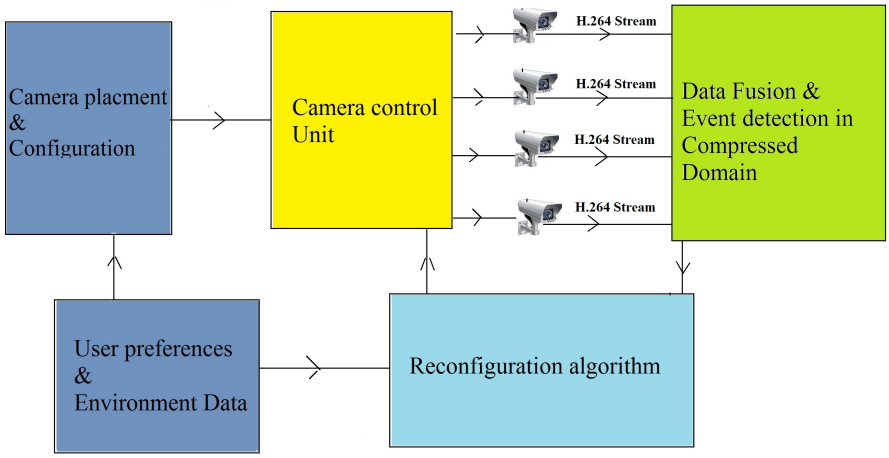


Figure 1.2: Overview of the proposed system

1.4.1 Camera placement and static reconfiguration

Camera placement generally involves arriving at an optimal locations and parameters for the given number cameras in the network, so that coverage of the network is maximized and the blind spots in the deployed environment are minimized. Both planning and reconfiguration are achieved in the virtual domain by modelling both camera and environment. Optimal solution should be obtained, given a set of user constraints pertaining to possible installation locations of the camera. Static reconfiguration takes

into account the static or long term changes that occur in the environments. Examples are camera malfunction and occlusions.

The Research in this areas is broadly classified into three categories.

- *Generate and Test Approach:* In Generate and Test approaches all the constraints and models are incorporated into a test simulation model and solution is obtained by evaluating all possible configurations.
- *Synthesis Approach:* In case of synthesis approach we model the given constraints as analytic functions. Despite being the most popular approach of the three, the task is somewhat complicated as it involves so many parameters, and modelling a constraint in such high dimensional space is difficult. In this approach various models are used to describe different constraints, and then these models are solved for various camera parameters using iterative solutions.
- *Expert Systems:* The Expert Systems approach addresses the high-level aspects of the problem in which a particular viewing and illumination technique is chosen from a set of previous data which has been accumulated over time. For instance, whether front or back illumination is more appropriate for the particular object and the features to be observed. An expert system is primarily used for advice for a certain set of tasks.

Considering the range of applications for which multi camera networks are used we aimed at developing an algorithm, which is more generalist and will work for range of applications with only a slight change in functionality. We would also like to mention that the proposed algorithm will be a combination of all three major lines of research in this area. As the solution space and the static reconfiguration would be prompted by user inputs, the user would also be able restrict the solution space according to the application for which he intends to deploy, in line with the principles of

expert systems. We want to model the aspects that are to be optimized as analytic functions as is the case of synthesis approach. Similarly, we would like to optimize the coverage by generating and simulating the camera network response to the environment using virtual modelling technique, which is in line with 'generate and test approach'.

Innovative aspects In general most of the present day camera planning algorithms, concentrate on coverage maximization with a simplistic camera model which has uniform coverage and also assume uniform lighting. However we feel that such a description of the problem is an oversimplification. The innovative aspects of our approach are listed below.

1. *Camera model* We propose a three dimensional, realistic and innovative camera model based on ray projection, with number of rays being proportional to the resolution of the camera. Such a representation allows us to measure the coverage in terms of pixel density.

2. *Distortion modelling* Optimization is also focused on minimizing perspective distortion as seen by the camera, made possible by the ray projection camera model.

3. *Environment model* We also model the environment to its slightest detail using modelling parameters like colour, texture, reflection coefficient, and diffusion coefficient using POVray software [50]

4. *Light modelling* We model the light sources, often ignored in coverage maximization problem. There is a possibility of modelling an array of light models like circular, area, parallel etc. The attenuation rate of light is also modelled based on the environment.

5. *Entropy* Along with coverage and distortion, entropy of the view as seen by the cameras is also made a subject of optimization in order

to get best possible views for the cameras. As the entropy or quality of view is highly dependent on illumination and focal length of the camera.

1.4.2 Dyanamic reconfiguration

Dynamic reconfiguration algorithm for the camera network helps in reducing the cost and also improving the efficiency of the system as discussed in Section 1.3. The requirements of the algorithm are as follows

1. *Real time operability* Any dynamic system requires a real time response, hence we need propose a real time video event detection and information gathering algorithm. Which also requires low bandwidth requirement in order to avoid delays.

2. *Distributive algorithm* We intend to propose a solution which is robust to sensor failures, hence the algorithm needs to be distributive. Distributive architecture can also be easily implemented using network of low power embedded devices rather than a high complexity centralized computational unit.

3. *Generic* Since the video surveillance systems are deployed for variety of applications, the solution should be as general as possible. Hence the reconfiguration algorithm should not be biased towards any particular surveillance application.

Innovative aspects Apart from being distributive, low complexity and generic, the algorithm has following innovative aspects.

1. *Compressed domain* The proposed algorithm completely operates in compressed domain, thereby eliminating the need for decoding the

video stream. This reduces the complexity as there is no feature extraction, which is generally computationally very intensive. Compressed stream also brings down the bandwidth requirement.

2. *Motion entropy* In this thesis we propose a generic quantitative measure for information in video directly derived from compressed video bit stream. It will be the basis of the camera reconfiguration and camera handoff. This metric can be used in many ways to achieve various computer vision tasks like tracking, segmentation, anomaly detection etc.

3. *Scalability* The Proposed algorithm is robust to sensor malfunction and is easily scalable to any number of cameras without any further addition of infrastructure or hardware.

4. *Customizability* Although proposed system is generic and is not biased towards any computer vision application. User can easily train the system to achieve desired result in many computer vision application areas.

5. *Deployment* Algorithm relies on motion field as a basis for many of the derived metrics. Which is extracted from the compressed bit stream. Owing to the low complexity, the algorithm can be easily embedded in the video encoder platform inside the camera hardware thereby eliminating any requirement of additional hardware.

1.5 Thesis organization

Thesis is organized into six chapters, with each chapter describing the research problems which constitute the doctoral study. Chapter 2 describes the camera planning part of the thesis, state of the art, proposed algorithms, and validation are discussed. Chapter 3 in turn describes the

problem of light planning, proposed solution and its validation. Chapter 4 entirely constitutes the discussion about video analytics, state of the art and its relevance to the smart camera reconfiguration. We also propose a generic metric for visual information in compressed domain. Proposed metric is validated by utilizing it for object detection,segmentation and fall detection. Chapter 5 discusses the reconfiguration of camera networks. State of the art, proposed algorithm based on generic information metric and its validation are also discussed. In chapter 6 we present the summary of the study along with the many possible future works which we intend to undertake.

Chapter 2

Camera Planning

In this chapter we discuss about the camera planning and static reconfiguration. In order to serve various surveillance environments, both 2D and 3D camera planning models are presented. 2D model is more suited for large environments and is of less complexity, whereas 3D model is more accurate and is suited for complex indoor environments.

As discussed in the previous chapter, most of the algorithms in the state of the art consider camera planning as a simplistic coverage maximization problem. Camera models used are also uniform and fixed, such an assumption is highly over simplistic. Further such a model would be only valid for static cameras. Rendering these algorithms inapplicable for PTZ cameras which are increasingly replacing the static cameras. Many of other critical parameters like lighting and other radiometric properties are also overlooked. As a solution to these problems we present two camera models for planning based on the size of environment.

2.1 State of the art

A recent survey by Liu et al [40] summarizes the historical developments in the area of camera networks planning and coverage modeling, along with an overview of the research field also highlighting the open problems and

challenges. In this section we will briefly introduce how the problem of coverage maximization has been faced in literature, presenting the most important milestones in this research area.

A very simple model to describe the problem of coverage maximization is the so-called *art gallery problem* (AGP), where the minimum number of guards is to be determined for a given area [12]. A variant of the AGP is known as the *Watchmen Tour Problem* (WTP), where guards are allowed to move inside the area perimeter [7]. The objective in this case is to determine the optimal number of guards and their route to guarantee the detection of an intruder with an unknown initial position. Both approaches can be useful to understand the problem, but are generally not suitable for the application in real-world scenarios, when a real deployment of sensors is required. Therefore, more sophisticated algorithms have to be adopted to take into account the most important elements of the surrounding world, which include constraints on observability [62], but also camera parameters (field of view, focal length), and illumination parameters.

The HEAVEN system by Sakane et al [59] was proposed with the goal of modeling the coverage exploiting a simulation tool. HEAVEN uses a spherical representation to model the sensor configuration. To this aim, 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. This process can be implemented at different levels of detail, depending on the available computational resources. The viewing sphere is centered on the object and its radius is equal to a distance (chosen a priori) from the sensor to the target object.

Another category of methods, defined in literature as *synthesis* approaches, aim to model the coverage constraints as analytic functions, and formulating the problem in terms of satisfaction of constraints. The pro-

cess typically turns out to be more complex as it involves many parameters, resulting in a high-dimensionality space, in which the optimization has to be performed. Nevertheless, and in case the computational cost is not the first priority, modeling constraints as analytic functions can bring some advantages. In fact, each requirement generates a geometric constraint, which in turn is satisfied in a domain of admissible locations in the three-dimensional space. The set of locations generated by each constraint can then be intersected in order to determine the ones that best satisfy all constraints simultaneously. An early work in this area is proposed by Cowan et al [13], in which camera locations are generated also with respect to illumination planning.

A considerable portion of the present day research is carried out following these principles, also thanks to the ever-increasing available computational power, different models can be defined for the relevant camera parameters using iterative solutions. For example in [21] Erdem et al propose a camera positioning algorithm based on a binary optimization technique and using polygonal spaces presented as occupancy grids. Bodor et al [5] propose an algorithm to optimize views so as to provide the highest resolution of objects and motion in the scene.

Among the most recent proposals in camera positioning and reconfiguration algorithms, the work by Mittal et al. [45] presents a probabilistic framework for object visibility in a multi-sensor environment. In the work by Piciarelli et al. [52], the authors address the problem of camera networks reconfiguration, by adjusting pan, tilt, and zoom. Expectation Maximization is then used to maximize the coverage of salient portions of the observed scene, identified by motion activity maps. In [47] Munishwar et al propose a framework for target coverage based on the spatial decomposition of the network and optimizing the solution for individual partitions. A recent work on camera planning by Morsley et al [46] dis-

cusses how an iterative approach can be used to solve a complex problem like camera positioning, stressing the fact that also for a limited number of cameras, hundreds of thousands of configurations are possible. Similarly, Song et al [63] describe a generalized framework for multi-camera tracking using the Kalman consensus algorithm, along with reconfiguration using game theory approaches.

In the work by liu et al [37] the authors propose a generalized statistical framework for optimal deployment of large scale camera networks. Although the proposed solution is evaluated in a simulated environment it gives another perspective of the deployment problem, especially with respect to large camera networks with special focus on user constraints. Similarly in [41] authors propose a camera view quality criteria based on application specific and environment specific functions However the algorithm is targeted at quality of view rather than for coverage. In [11] Cheng et al try to optimize the visual sensor networks with respect to the barriers encountered in the coverage, however this is yet another paper which addresses the optimization problem in application specific sense. State of the art in this area is quite rich with lot of works evaluating the problem in many unique perspectives, however there are many glaring deficiencies which render most of the algorithms inapplicable in real scenario. The deficiencies are summarized below.

1. *Simplistic camera model*: Camera model used in most of the cases very simplistic, it is generally a triangle in 2D based algorithms or pyramid and cone in most 3D algorithms. Such an assumption overlooks the fact that object which is near to the camera has much more detail than the one very far away even though both are covered. Also it overlooks the another important aspect of black spots near the camera location. Further the camera is assumed to be static with a fixed parameters. This assumption is obsolete as the PTZ cameras have become more

common, which have many configurations for a given location.

2. *Light Modelling* All the works in this area assume uniform illumination. Such an approximation might as well be valid during the day, but is not at all acceptable during the night. Illumination is one of the most critical aspects of any video surveillance system, as it has dramatic impact on accuracy and performance of most video processing and computer vision algorithms. Hence for a successful deployment of the surveillance system a proper estimation of lighting is absolutely necessary.

3. *Simplistic environment model* Most of the works represent the environment as a simple 2D map with lines representing objects or obstacles. Such an assumption is simply un viable as they cannot correctly describe the sensor response of the camera to that of the deployed environment. Thereby completely ignoring the qualitative aspect of the video that has to be recorded.

4. *Distortion* There are two types of distortions in a video, one is lens distortion while the other is perspective distortion. Lens distortion is easily rectified using interpolation and other techniques, however perspective distortion cannot be eliminated as it results from the view angle of the camera. None of the state of the art address this problem.

5. *Reconfiguration* Most of the systems proposed seem to think that camera placement as one time task. Camera locations arrived at, by such algorithms might work very efficiently but will at once become obsolete if any of the camera in the network fails. Such a situation would require complete re installation of cameras, with algorithm providing new locations for remaining set of cameras. Therefore there is a definite need for algorithm which provides a better configuration by

optimizing other parameters apart from location.

2.2 2D modelling

In very large environments, blind spot that typically appears on the immediate front of the camera is largely negligible. Most of the state of the art algorithms which employ 2D camera model are useful in such scenario However there are certain glaring inadequacies which have been addressed by our 2D camera and environment model.We propose to model global and local coverage as analytic functions of the camera parameters. While global coverage aims at maximizing the visibility of the entire area, local coverage focuses the attention on a limited number of *hot spots* or objects of interest. For the optimization procedure, we adopt the PSO algorithm. PSO [16] demonstrated to be effective in solving complex non-linear multidimensional discontinuous problems in a variety of fields [17]. Unlike other multiple-agent optimization procedures such as Genetic Algorithms (GA) [24], PSO is based on the cooperation among the agents rather than their competition. Another motivation that lead us to the choice of PSO is due to the random deployment of obstacles, as well as the topology of the environments, making it difficult to find appropriate mathematical formulations. 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 giving them a certain range of variation. Consequently, each parameter corresponds to 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.

2.2.1 Global and Local Coverage

In order to model global coverage, we consider three conditions that need to be satisfied, namely *pixel density*, *quality of view*, and *light intensity*.

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 modelled 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 floor plan to the camera CCD is illustrated in Fig. 2.1.

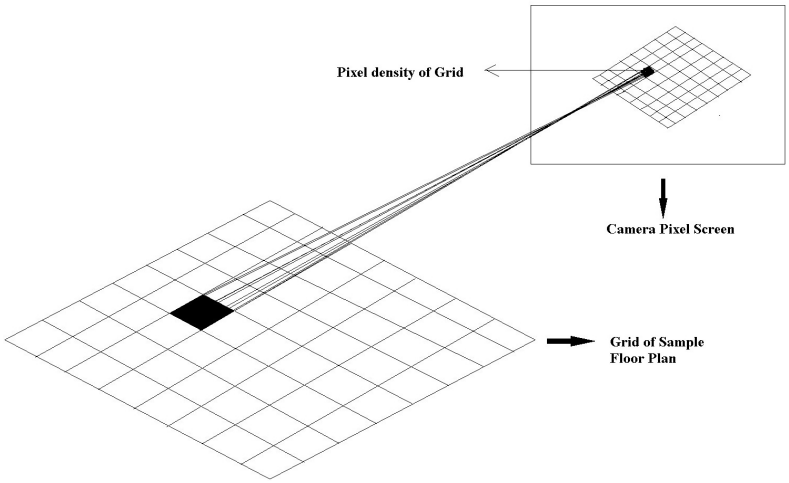


Figure 2.1: Pixel Mapping of Grid. Each area of the floor plan is captured by a number of pixels that depends on the distance from the camera.

The *quality of view* of a target is related to the relationship between feature detectability and distance of the target. 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 limited field-of-view of the camera. To take into account this parameter, we model the visibility constraint as a Gaussian distribution that is computed along the ray emerging from the camera. Fig. 2.2 shows the visibility of the object modeled a Gaussian distribution. The optimal distance is located at the center of the Gaussian, and needs to be specified according to the size of the objects to be monitored (human, cars, etc.).

The *light intensity* at the object location will also affect quality of the captured image, and it is therefore a very important parameter to be included in the model. According to the standard decay of the light intensity, we model it as an inverse square function of the distance from the light source. This model is not exhaustive for illumination modeling, since it does not take into account for example, of reflections and shadows. The term is meant to model the light intensity of specific spots in the environment, so as to change the cameras positioning also according to this parameter.

As far as the local coverage is concerned, each area of interest (doors, windows, statues, paintings, other objects, etc.) is modeled as a Gaussian function of distance from the camera. The mean of the Gaussian defines the optimal distance of the specific target from the camera, namely where the quality of view for that particular target is maximum. This value is related to the size of the target as well as its relevance in the scene. We have also planned to include light intensity as a parameter in target coverage, measuring the expected intensity of light at the target location. For instance, if a target falls under an area of low light intensity, the camera has to be placed closer to it.

Further details about the parametrization of all the above mentioned

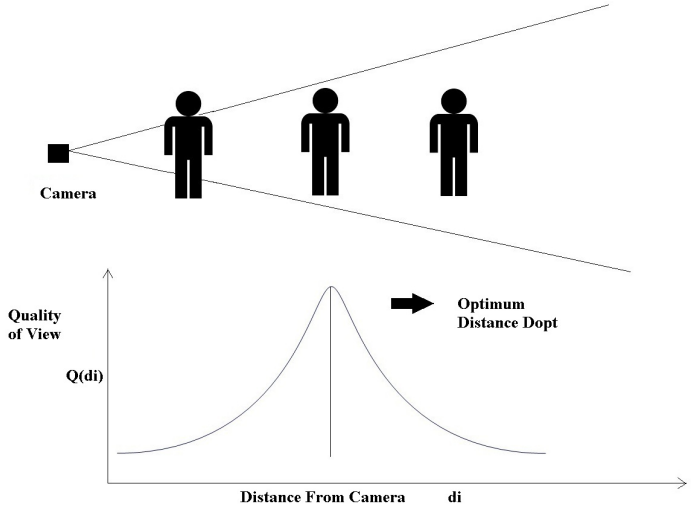


Figure 2.2: Quality of view. The optimal distance for observation depends on the objects of interest for the specific scene.

elements are provided in next sections.

2.2.2 Camera Model

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

2.2.3 Assumptions

All simulations we present are carried out on the $(X - Z)$ ground plane, thus discarding the vertical dimension (Y). From the optimization point of view, the extension to the third dimension is straightforward. However, it is worth noting that it is common sense to position the cameras either on the ceiling or on the walls and usually at the same height, to first limit the accessibility to non authorized users, and then to improve the visibility on the scene. Under these assumptions, it is possible to approximate the optimal camera positioning also regardless of the Y dimension. However, from an algorithmic point of view the algorithm is scalable and it would

only require the definition of an additional constraint.

Equations and Constants

As we have pointed out in the previous section, the quality of view of an object is modeled as a Gaussian distribution positioned along the ray emerging for each pixel of the camera, as given in Eq. 2.1

$$Q(d_i) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(d_i - D^{opt})^2}{2\sigma^2}\right) \tag{2.1}$$

where d_i is the distance of the cell on the i-th ray hitting it, and D^{opt} is the optimum distance at which the object has maximum quality of view. In the current scenario D^{opt} is chosen as a constant, which depends on the application and varies on the purpose of camera deployment.

The second term we need to define is related to the information about the light intensity at a given location, defined as in Eq. 2.2

$$L(d_j) = \frac{P}{d_j^2} \tag{2.2}$$

where P is the power of the j-th light source with $j \in \{1 \dots S\}$, and d_j is the distance of the point from the light source.

As for the local coverage, the model is in Eq. 2.3:

$$T_k = exp\left(-\frac{(d_k - D_k^{opt})^2}{2\sigma_{d_k}^2 * \sum_{j}^{S'} L(d_j)}\right) \tag{2.3}$$

where d_k is the distance of the k-th target from the Nearest Visible camera, σ_{d_k} is the variance of the target coverage, and D_k^{opt} is the optimum distance from camera for the k-th target where the quality of view is maximum. L_{d_j} is the light intensity at the target location given by summation of the contribution from all the light sources ($S' \leq S$) hitting the target.

While calculating the contribution of each light source, blockage of light due to the presence of obstacles is also taken into account. Smaller values for σ_{d_k} will keep the cameras strongly focused on the targets, while higher values for σ_{d_k} will relax the constraint, accepting targets to be also decentralized in the field-of-view of the camera. The parameter is in general related to the environment size, as well as to the coverage requirements.

In order to determine the areas that are visible in the map, we have to fix a threshold for visibility for the quality of view function. This threshold is calculated in accordance with [20]. According to the standard 100 Lumens is the required amount of light for casual observance of surroundings. Lumens is a SI derived unit for the luminous flux, which is different from radiant flux as it also takes into consideration human eye sensitivity. Now we classify a given cell in a grid to be covered if and only if the total luminous flux in the cell is at least 100 Lumens, and the total quality of view as defined by summation of $Q(d_i)$ of individual rays that pass through cell, is at least 0.5. Since the scale used here is 4 pixels per meter, 100 Lumens corresponds to 1 watt per square meter according to most commercial light manufacturers [1].

Algorithm

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

Step 1 - Determine the solution space. After reading the map, 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 targets, 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.

Step 2 - Calculate Global coverage. In order to calculate the global coverage, the environment map is divided into a grid of NxN pixels. The granularity of the grid is chosen depending on the map scale, as well as on the accuracy in positioning that we want to achieve. The finer the grid, the more accurate will be the result, at a cost of a higher computational complexity. For each camera, the number of rays that pass through each cell of the grid are 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 Eq. 2.4:

$$C(m,n) = \sum_{i=0}^{R} Q(d_i) * \sum_{j=0}^{S} L(d_j) \tag{2.4}$$

where $C(m,n)$ is the final quality of view metric obtained for a specific cell, while m and n give the location of the cell in the map. As we can see from Eq. 2.4, this metric will weight the quality of view function measured as in Eq. 2.1 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. The light intensity component is instead obtained by summing up the contributions of all light sources in that point (S).

We then label the cell as "visible" only if the quality of view is higher than a predefined threshold, fixed in accordance with a recommendation from the European standard for lighting levels based on activity [20]. The global coverage is estimated as the number of visible cells divided by total number cells in the grid (Eq. 2.5).

$$C_G = \frac{Cells_{visible}}{Cells_{total}} \tag{2.5}$$

Step 3 - Include Local coverage. For a given camera position the local coverage is given by Eq. 2.3. Accordingly, overall local coverage is given by Eq. 2.6:

$$C_T = \frac{1}{T} \sum_{k=0}^{T} T_k \tag{2.6}$$

where T is the total number of target objects.

Step 4 - Fitness Function. We need now to define a fitness function that will be used by the PSO algorithm as a target for the optimization. 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 (Eq. 2.7).

$$F(C_G, C_T) = (1 - C_G) * w + (1 - C_T) * (1 - w) \tag{2.7}$$

In Eq. 2.7 C_G represents the global coverage and C_T represents the local (target) coverage; w is the weight assigned by the user to balance the tradeoff between global and local coverage. We can notice from Eq. 2.7 that, as soon as the global and local coverage approach 100%, the fitness function converges to zero.

Step 5 - PSO. PSO is applied to the solution space defined in *Step 1.* At each iteration, the particle position and velocity is 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).

2.3 3D modelling

To further enhance the accuracy and to also include various other factors in camera planning we also present a 3D camera planning algorithm. This algorithm is highly useful in scenarios where a more accurate modelling of

the camera and environment are paramount. This model includes many additional features like radiometric properties in terms of environment model. Camera model is based on classic pin-hole camera model and is completely replicated to obtain visual quality and perceived distortion.

2.3.1 Contribution

In this sub section we highlight the novel elements of our approach compared to the solutions available in the literature. They can be summarized into four distinctive features, as described in the next paragraphs.

Camera modeling

Most algorithms for camera networks planning use a model for the camera that does not take into account the sensitivity in terms of quality of the captured information. This is for example the case of surveillance systems, which goal is to implement efficient algorithms capable of recreating the human observation for event detection and analysis tasks. We propose a realistic camera model, which simulates the camera view, and assesses the visual quality for a given environment, by analyzing the scene in a virtual domain. The advantage of using a virtual representation enables a highly reliable evaluation of the camera configuration, providing a quantitative metric, both in terms of visual quality and coverage. Pictorial representation of this ray projection camera model is shown in Fig.2.3

Illumination and radiometric properties

In computer vision applications, illumination is known to be a very critical element, when setting up the camera network. We propose to model the light sources with a proper attenuation and diffraction model, as it would be observed in a real scenario. This will be computed in accordance with

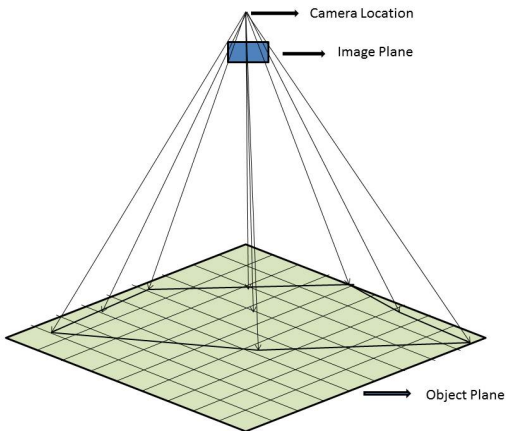


Figure 2.3: Ray projection on to the environment from focal point

the radiometric properties of the surfaces, namely their color and their reflection coefficient.

Distortion

Distortions can be introduced at different levels of the acquisition and processing stages. In particular, lens and perspective distortions are inherent in the captured videos, and can seriously affect the system performance, requiring image rectification. Such situations can be handled a priori, by selecting the configuration of the camera network that reveals the smallest level of distortion.

Scalability and user interaction

Most of the algorithms in the present state of the art do not consider the evolution of the environmental conditions over time, which should trigger a reconfiguration of the camera network. This is for example the case of sensors failures, introduction of new sensing equipment, changes in the environmental illumination, as well as spatial re-arrangements of pieces of furniture and objects to be monitored.

2.3.2 Metrics for visual quality assessment

Visual information

A good visibility of the observed scene is a fundamental step for a correct application of automatic analysis tools. Therefore, while planning the camera configuration, several factors have to be taken into account. For example, one has to position the cameras so that the image sensors are exposed to the correct amount of light. Videos captured under good illumination conditions will require less pre-processing and maximize the gathered information. However, gauging directly the sensor response of the camera is not a viable option, since planning should be done before the actual positioning of the sensorial equipment in the environment. In our system we deploy the virtual cameras in the synthesized domain. We successively use the image generated by the virtual model to assess the sensor response for various camera configurations. Considering that our virtual model also includes the configuration of the light sources, such an approximation turns out to be particularly realistic.

In order to measure the amount of information present in the captured image we propose to start from the histogram of the image, by determining the bins distribution. In fact, in presence of an under exposed image, the histogram will be biased towards the lower levels, while for an over exposed image, the histogram will likely be biased towards the higher intensity levels.

In order to quantify the quantity of information in the image, we compute the entropy, a metric that has already demonstrated to be effective in measuring the quality of the captured image [27]. For a given configuration of the i-th camera, and the corresponding captured image I_i, with $i \in \{1, \ldots, N\}$, let the intensity levels be $\Omega = (l_1 < l_2 < \ldots l_i \cdots < l_N)$, and the information content in the image be given by

$$H(\Omega) = \sum_{p_i \in \Omega}^{n} -p_i log p_i \tag{2.8}$$

In Eq. (2.8) p_i is the normalized region under each intensity level $l_i \in \Omega$, considering that the entropy is directly proportional to the information content of the image. We then maximize the image information, given a solution space, which consists in our case of all possible combination of the camera position (X, Y, Z), pan, tilt, and zoom:

$$H(\Omega)_{opt} = argmax \left(\sum_{r_i \in \Omega}^{n} -p_i log p_i \right) \tag{2.9}$$

where i represents a single instance of the solution space.

Instead of considering the standard RGB space, the analysis is carried out in the LAB color space, because of its property of being perceptually uniform, meaning that the same distance computed over different points of the color space, would correspond to an equal variation from a perceptual viewpoint.

Fig. 2.4 shows three sample images generated with our virtualization tool, at different levels of exposure, and the corresponding histogram distributions.

Distortion

There are basically two types of distortions, namely lens distortion and perspective distortion. Generally speaking, in most cases lens distortion can be corrected directly in the acquisition phase, and may not significantly affect the quality of the image. Perspective distortion, instead, primarily depends on the angle of view and focal length and can considerably deteriorate the visual quality of the collected data. Furthermore, perspective

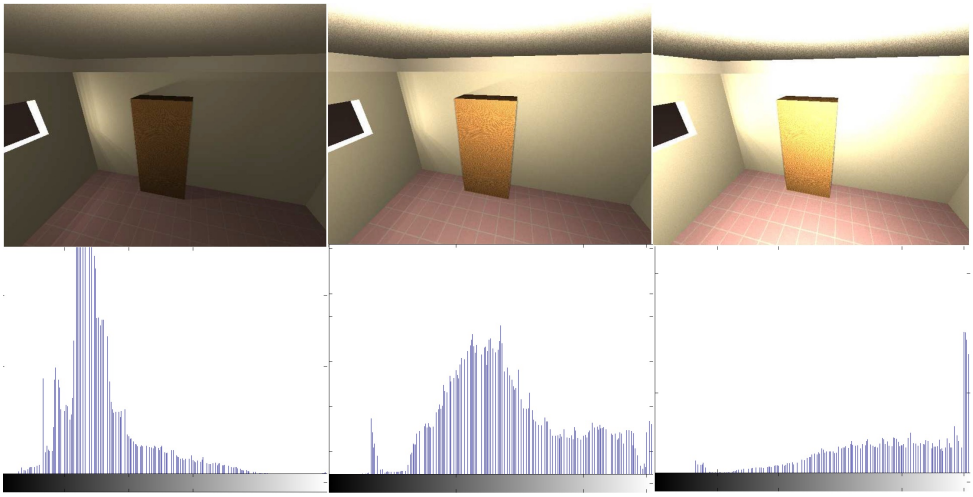


Figure 2.4: Images captured under different light exposure and corresponding histograms: under exposed (left), correctly exposed (center), and over exposed (right).

distortion is not uniform across the image, therefore while planning the camera network we have to make sure that the area of interest of a given environment falls under the portion of image, which exhibits the minimum amount of distortion. In Fig. 2.5 we show an example of the varying levels of distortion in an image. The image consists of three spheres positioned at different distances from the camera. All spheres have a radius of $0.5m$. We can observe from the figure that the sphere in the left most part of the image is subject to severe distortion and appears like an ellipsoid with a large difference between the major axis and minor axis. The level of distortion then progressively decreases, until, the sphere in the extreme right of the image preserves the original shape. It is clear from the observation that the camera should be positioned in such a manner that objects of interest is not distorted.

We propose to measure the perspective distortion using image registration. By projecting the image points onto the object of interest, and measuring the ratio of projected distance along the horizontal and vertical directions, we can estimate the ratio of the perspective distortion. In an

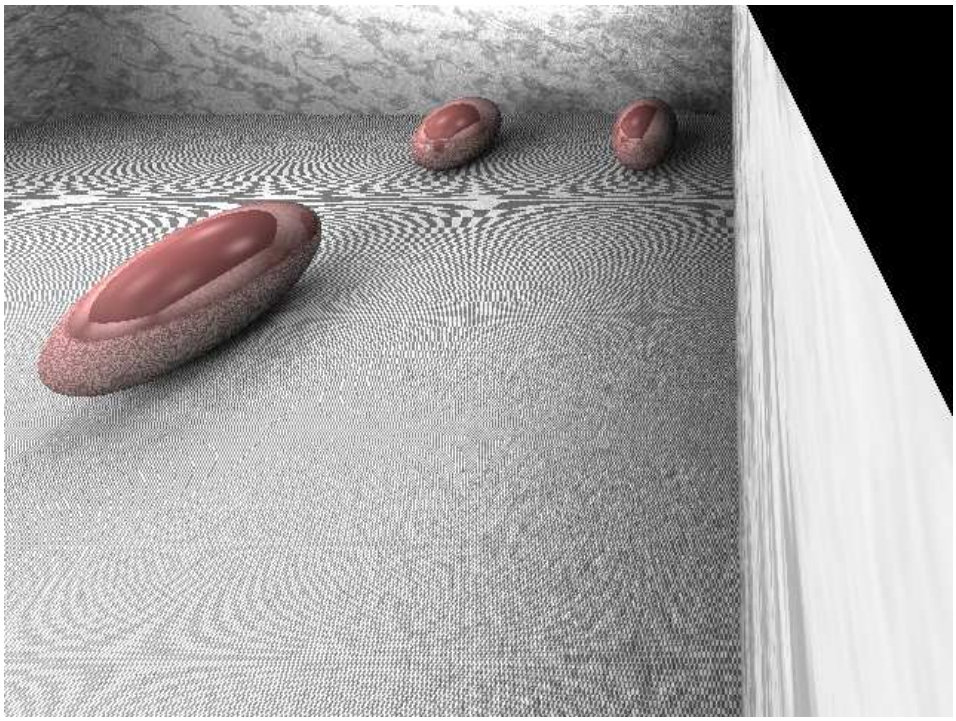


Figure 2.5: Example showing different levels of perspective distortion as captured by the camera.

ideal scenario, this ratio should be equal to the aspect ratio of the image. In order to estimate the distribution of the distortion we need to compute the mean and variance across the projection of the image plane on the area of interest (Fig. 2.6). The mean value of the distortion D can be defined as:

$$\mu_D = \frac{1}{A_i} * \oint \frac{X(x_i + dx_i) - X(x_i)}{(Z(y_i + dy_i) - Z(y_i)) * R} \tag{2.10}$$

where A_i is the area of the entire image, x_i and y_i correspond to the variations along X and Y on the 2D image plane, and R is the aspect ratio. In the discrete domain, the equation becomes:

$$\mu_D = \frac{1}{(N_w - 1) * (N_h - 1)} * \sum_{i=1,j=1}^{N_w-1,N_h-1} \frac{X(i+1) - X(i)}{Z(i+1) - Z(i)} \tag{2.11}$$

where N_w and N_h is the number of pixels in the horizontal and vertical directions in the 2D image plane, respectively. Similarly, variance can be calculated as:

$$\sigma_D^2 = \frac{1}{(N_w - 1) * (N_h - 1)} * \sum_{i=1,j=1}^{N_w-1,N_h-1} [\frac{X(i+1) - X(i)}{Z(j+1) - Z(j)} - \mu_D]^2 \tag{2.12}$$

After computing the mean and variance of the distortion, we can formulate a suitable cost function, resulting as the configuration that minimizes both terms:

$$D_{min} = argmin(\mu_D) + argmin(\sigma_D^2) \tag{2.13}$$

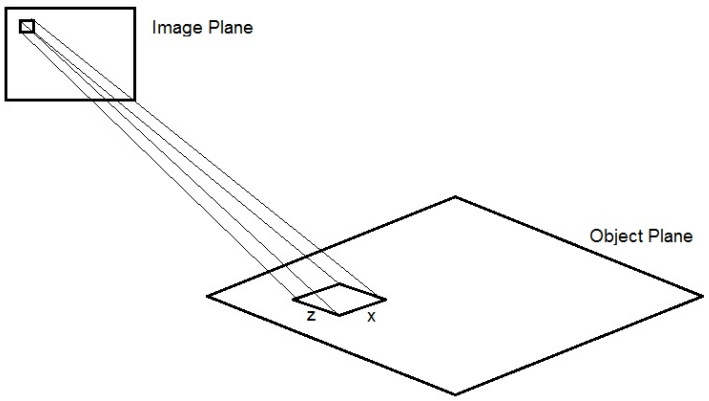


Figure 2.6: Projection of the area of interest on the image plane, to compute the distortion.

Coverage

With the term *coverage*, we refer to the portion of the environment visible by each camera. Coverage has been historically modeled through a fixed cone deriving from the field of view of the camera (2D or 3D), with uniform distribution of quality. We believe though that this assumption is too simplistic and may be valid only for very simple and static scenarios. In our case we present a model of the field of view, in which rays are projected from each pixel into the environment, thus assuming each pixel in the camera as a light emitter hitting a portion of the observed real world. Such a model should be as close as possible to the real world scenario, and allows tracking the changes in the distribution of ray projections according to the relative changes in the camera parameters. In order to compute the percentage of the area covered by each camera, the environment is divided into a grid of cells of equal square size. Each pixel is projected onto the real ground plane and classified as belonging to a specific cell. The cells, which have at least one ray intersecting them are classified as visible, and all others are considered invisible to the camera.

Depending on the inclination of the camera and on the focal length, the distribution of rays clearly changes. An optimal solution would require the

maximum spread in the distribution of rays, so as to maximize the number of visible cells. The number of rays intersecting the grid cell is a function of the cell location (X, Y, Z), focal length f, camera location x, y, z, pan θ and tilt ϕ:

$$N_{rays}(X, Y, Z) = F(X, Y, Z, x, y, z, \phi, \theta) \tag{2.14}$$

Also in this case we can formulate a cost function for the area covered in terms of the number of cells, which at least have one pixel projection on them, normalized by the total number of cells. The maximization of this condition implies:

$$Cov_{max} = argmax \left[\frac{Count(N_{rays}(X, Y, Z) >= 1)}{N_{cells}} \right] \tag{2.15}$$

2.3.3 Problem formulation and implementation

As for the metrics described in the previous paragraphs, entropy and distortion are camera-specific and mutually exclusive, i.e, they do not influence one with each other. Coverage, instead, has to be calculated collectively.

Initially, we formulate the cost function for both entropy and distortion for each individual camera. Entropy, mean, and variance of distortion are described in Eq. (2.8), Eq. (2.11), and (2.12). In order to normalize the cost the equations can be expressed as:

$$D_i = exp(-(\mu_D + \sigma_D^2)) \tag{2.16}$$

$$E_i = [1 - exp(-H(\Omega)_i)] \tag{2.17}$$

and for each camera i, these values are maximized when entropy is maximum and distortion is minimum. In order to merge these two metrics with the number of rays hitting the surface, the measure is multiplied by the normalized value of the number of rays. Unless specified differently,

entropy and distortion are equally important in the cost function. Taking all these factors into account the formulation becomes:

$$C_i(E_i, D_i) = \frac{1}{N_{rays}} * [D_i/2 + E_i/2] \tag{2.18}$$

Let there be N cameras that have to be positioned in the environment, and let the area covered by each camera be defined as A_i for the i-th camera. The total area covered by the network is given by:

$$\bigcup_{\forall i} A_i = \sum_{\forall i} A_i - \sum_{i<j} A_i \bigcap A_j + ... + (-1)^{n+1} \bigcap_{\forall i} A_i \tag{2.19}$$

which corresponds to the sum of the individual contributions in terms of coverage for the single cameras, minus the redundant area (overlap). Hence, our cost function should increase the overall visible area while minimizing the overlap between cameras field-of-view, as shown in Eq. (2.20):

$$C_V(\forall i) = \frac{1}{A_{env}} * \left[\sum_{\forall i} A_i - \sum_{i<j} A_i \bigcap A_j + ... + (-1)^{n+1} \bigcap_{\forall i} A_i \right] \tag{2.20}$$

where A_{env} is the total area of the environment. Combining the global coverage and individual camera measures we obtain the final cost function:

$$C_{final} = w_{task}(C_V(\forall i)) + (1 - w_{task}) * \left[\frac{1}{N} * (\sum_{i=0}^{N} C_i(E_i, D_i)) \right] \tag{2.21}$$

where $w_{task} \in \{0, \ldots, 1\}$ is application dependent and defined by the user.

2.3.4 Particle swarm optimization

Solving a problem involving six parameters describing the 3D geometry of the environment is a rather complex task, made even more critical due to the presence of random obstacles, and the use of traditional problem solving techniques like steepest descent is not a viable solution. Hence we propose to use a global optimization technique, namely the PSO, which has been already adopted in literature to solve camera planning problems [46] [74].

PSO [16], 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 [17]. Unlike other multiple-agent optimization procedures such as Genetic Algorithms (GA) [24], 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 GAs use three operators and the best configuration among several options of implementation needs to be chosen. Then, PSO parameters are easier to calibrate and to manipulate. Finally, PSO has a major ability to prevent the stagnation of the optimization process, thanks to a more significant level of control of its parameters [58] [18]. Further PSO has been widely used for coverage maximization in visual sensor networks [25, 71, 73]. Further details about the optimization algorithm, can be found in the relevant literature [16] [17].

2.3.5 Algorithm and implementation

As far as the camera model is concerned we have carried out the simulations using the classic pinhole camera model. The rays, which number is equal to the number of pixels in the sensor, are projected from the camera center

on the surface plane. The orientation of the image plane is defined by the pan and tilt of the camera (Fig. 2.7).

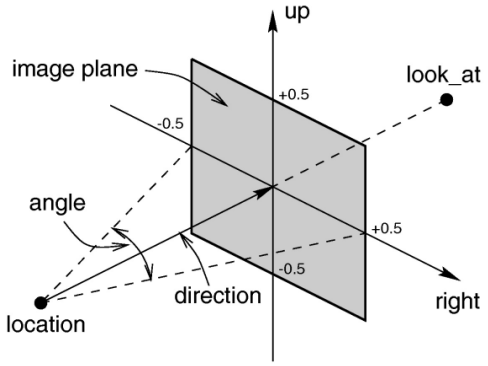


Figure 2.7: Model of the camera
[50]

In order to measure the visual information (as described in Section 2.3.2), we need a virtual environment to simulate the camera view, visualize and evaluate the obtained cameras position. For this purpose we use *Persistence of Vision*, or POV-Ray, a ray tracing software [50]. This software has a unique scene description language that can be used to replicate the objects in the real world, including their radiometric properties. Similarly, light sources can also be modeled, by specifying the attenuation along the travel distance. Matlab is then used to simulate the ray tracing and for image registration, starting from the simulated camera views obtained from POV-Ray, using the geom3D library [35].

The proposed algorithm, consisting of five main steps, can be described through the pseudocode shown in algorithm 1. Similarly pseudo code for subsequent particle swarm optimization is given by algorithm 2.

Step 1 - Determine the solution space. After identifying the perimetral and internal walls we need to determine the extent to which pan and tilt can be varied based on the cameras location on the walls. The environment

information is then converted into the POV-ray scene description language and the PSO is initialized. At each iteration of the algorithm, the particles are updated based on the fitness function.

Step 2 - Calculate entropy and distortion. Camera views are generated using POV-ray and the values for entropy and distortion are evaluated to obtain the camera measure in Eq. (2.18).

Step 3 - Calculate global coverage. The global coverage can then be computed, also taking into account the overlapping areas among the cameras. As described in the previous section, the surface area of the object of interest is divided into a grid of cells of equal size, and the coverage measure is calculated as described in Eq. 2.15.

Step 4 - Calculate configuration cost. The values obtained for each camera are then combined according to Eq. (2.21), where user or task-specific adaptation is also taken into account.

Step 5 - PSO. The fitness function is updated and the set of particles in the swarm optimization are updated until the termination criterion is reached.

2.4 Evaluation

This section deals with testing and evaluation of both 2D and 3D camera planning models. Test scenarios and obtained results are extensively discussed.

2.4.1 2D model

In this sub section we describe the scenarios used for the evaluation of camera planning and static reconfiguration, using 2D camera model. As mentioned in section 2.2 2D modelling of the environment is especially

input : Surface S of object of interest divided into $N \times M$ cells of equal size
input : Number of Cameras NC
input : Map I Description in POV-ray scene description language
input : Camera Resolution
output: Fitness Value

Initialize camera positions from PSO;
Initiate the number of rays for each camera based on the video resolution;
Generate camera view using POV-ray;

EOV ; % Quality of View of Cameras

DM ; % Mean of perspective distortion

DV ; % Variance of perspective distortion

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**
    for $k \leftarrow 1$**to** NC **do**
      pixelDensity ←RayIntersect(i, j, k);
      C(i, j) = C(i, j) + pixelDensity(k);
    end
    if C$(i, j) >= 0$ **then**
      goodCells ++ ;
    end
    if C$(i, j) > 1$ **then**
      goodCells– ;
    end
  end
end
for $k \leftarrow 1$**to** NC **do**
  EOV $(k) \leftarrow$ Entropyofview(k);
  DM $(k) \leftarrow$ Distortion_mean(k);
  DV $(k) \leftarrow$ Distortion_variance(k);
  CM = CM + NormalizedRayIntersectCount$(exp(-(\text{DM}(k) + \text{DV}(k)))/2 + (1 - exp(-\text{EOV}(k))/2)$;
end

$\text{C}_g \leftarrow \frac{\text{goodCells}}{N*M}$; % *Compute global coverage*
% *The final output is a combination of global coverage and camera measure with weight* w_G
$F(\text{C}_G, \text{CM}) \leftarrow 1 - [\text{C}_g * w_G + \text{CM}(1 - w_G)]$;
Return $F(\text{C}_G, \text{CM})$;

Algorithm 1: Fitness calculation

```
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 2: Pseudocode of the PSO.

relevant in large environments with large number cameras. It offers a low complexity alternative with only a slight reduction accuracy.

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. Uniform Illumination with a light intensity of 100 Lumens is considered for the purpose of simulation in order to decrease the complexity. Such an assumption is reasonable in any environment under the daylight conditions. The field-of-view is fixed in the range between 5 and 90 degrees and opti-

mum distance for quality of view is fixed at 40 pixels in the map, which corresponds to sbout 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.

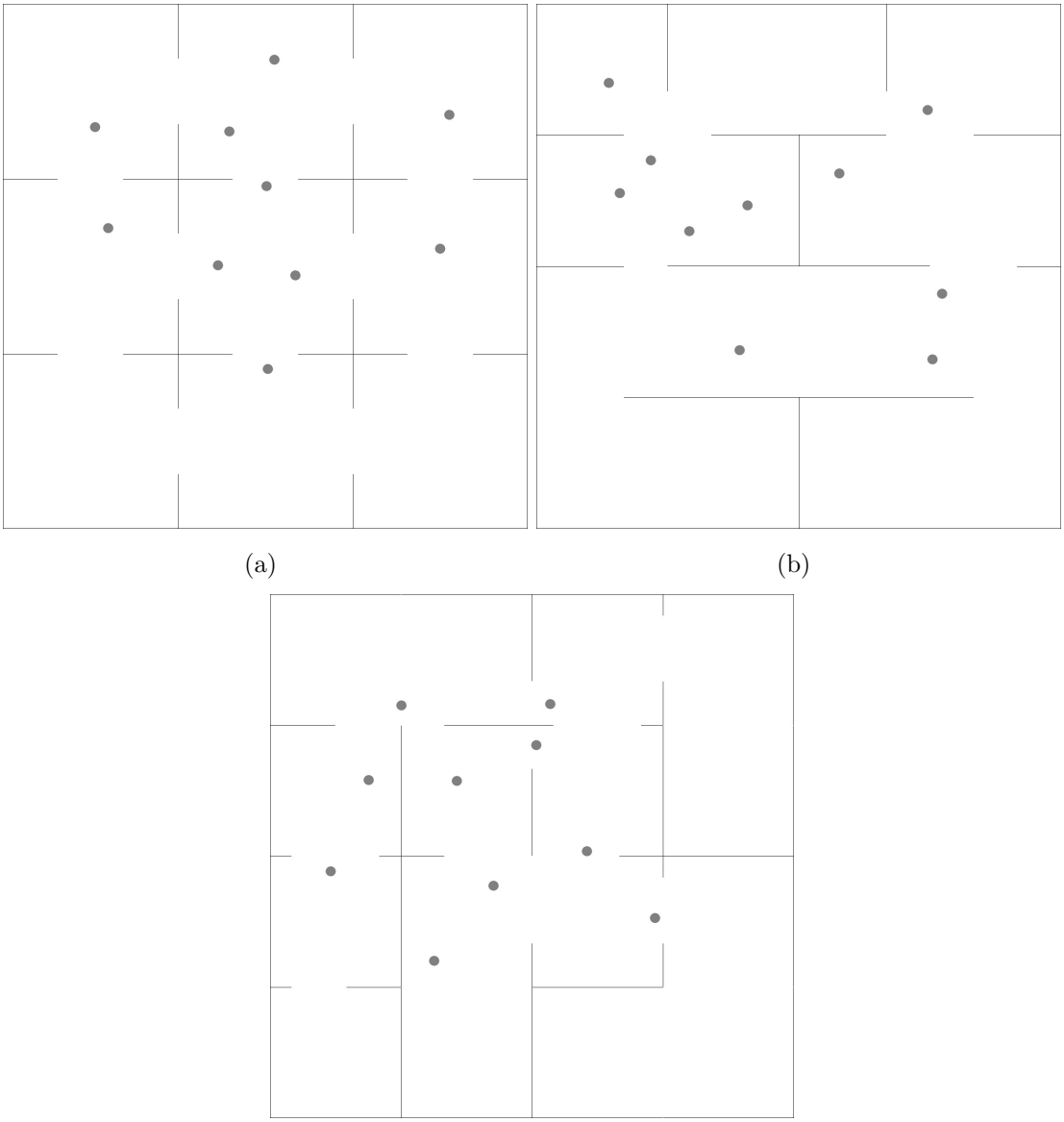
After the initial setup, nine different objects of interest are placed in the map. At this point the algorithm is required to re-align the cameras, keeping the positioning of the sensors fixed. This implies that in determining the new camera parameters, only local coverage is considered, thus setting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

The environment maps used for the testing are shown in Fig. 2.8. 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.

Experimental Results

As explained in Section 2.4.1, 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.

Initial Positioning 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 obtained maps, different colors are used to illustrate the



(a) (b)

(c)

Figure 2.8: Maps used for testing, with assignment of objects of interest.

quality of the coverage in over the entire map. Areas which have 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. Black areas are not visible to cameras due to the presence of obstacles.

Reconfiguration After the initial positioning is completed, 10 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 optical zoom. The algorithm is re-run considering the absolute position of the cameras fixed, thus optimizing target coverage.

Experiment 1 The input map to the algorithm is shown in Fig. 2.8(a). A uniform illumination of 10 Lumens is considered, and the initial positioning is performed with Optical and Digital Zoom both set to 1x, while pan is a free parameter along with the x-z positions that can be adjusted to obtain maximum coverage.The coverage map after the initial placement of cameras is shown in Figure 2.9(a). After the Initial positioning of the cameras, nine random targets are distributed over the entire environment map as we can see from Figure 2.9(a) and Table 2.1 these targets are not covered properly. The results obtained are shown in Figure2.9(b)

Table 2.1 summarizes the number of targets present in respective areas before and after reconfiguration. The initial positioning of cameras is completely based on global coverage whereas, reconfiguration is entirely based

Table 2.1: Target coverage for Map 1.

Configuration	White	Green	Blue	Red	Black	C_T	C_G
Initial	0	2	0	7	1	0.3883	0.5015
Final	0	6	2	0	1	0.7443	0.4193

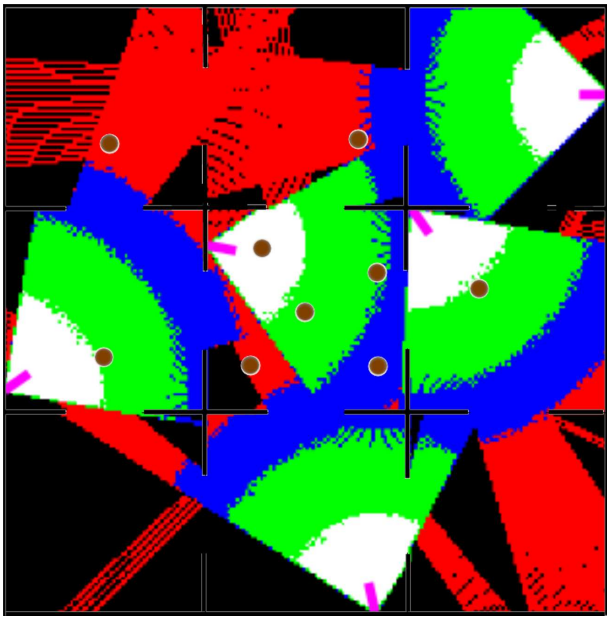
on local coverage. As can be seen from the figures in Table 2.1, C_T has increased both from quantitative and qualitative viewpoint. Initially only two targets were sufficiently covered but after realignment only one target has been left out and eight are covered. C_G in turn signifies the global coverage, as we can see from the table global coverage has decreased by about 20 percent after reconfiguration, this is on the expected lines since the reconfiguration is done entirely on the basis of target coverage.

Experiment 2 In the second experiment, the map of the environment is changed and all the other conditions are maintained the same. Initial coverage Map after the coverage based placement is given by Figure 2.10(a) and the final placement after the reconfiguration is shown in Fig. 2.10(b). Similar to the previous experiment results are presented in Table 2.2. As we can see from Table 2.2, initially only three targets were in the visible area but after the reconfiguration algorithm, a total of eight targets fall under the visible area. Expectedly target coverage C_T has increased from 0.5808 to 0.7764, since the number of targets covered has increased by 50 percent. Similarly there is a considerable reduction in global coverage.

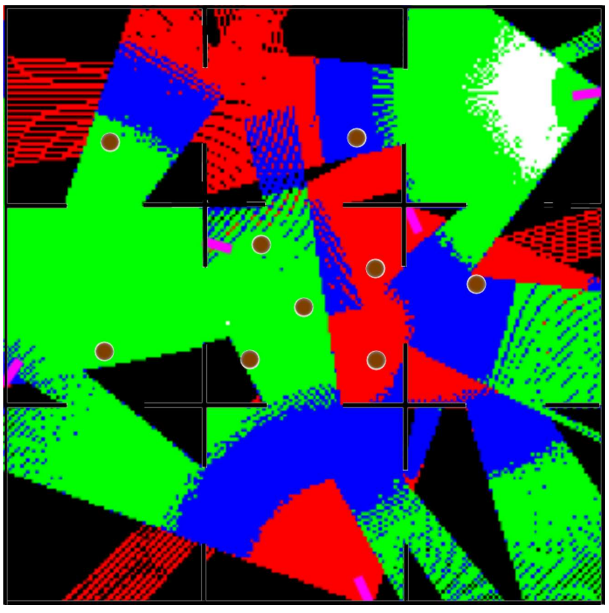
Table 2.2: Target coverage for Map 2.

Configuration	White	Green	Blue	Red	Black	C_T	C_G
Initial	0	2	1	5	1	0.5808	0.5433
Final	0	7	1	0	1	0.7764	0.4373

Experiment 3 In the final experiment we apply the algorithm on Map 3 under similar conditions and the initial and final coverage Maps are

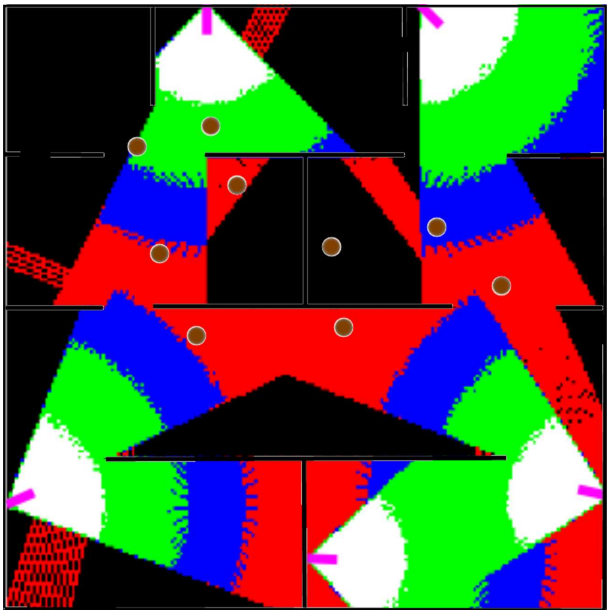


(a)

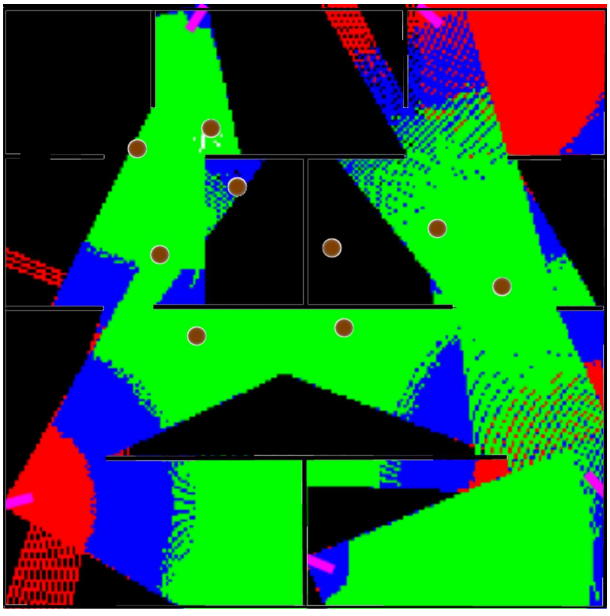


(b)

Figure 2.9: Map 1: (a) Initial positioning, and (b) after reconfiguration.



(a)



(b)

Figure 2.10: Map 2: (a) Initial positioning, and (b) after reconfiguration.

provided in Fig. 2.11. Table 2.3 summarizes the results obtained.

Initially there were only five targets in the visible area while all the other targets were not covered. However, after the reconfiguration a total of eight targets have become visible, while only two of them are not visible. The corresponding C_T has increased from 0.5461 to 0.8000. Decrease of the global coverage is in expected range,

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

Table 2.3: Target coverage for Map 3.

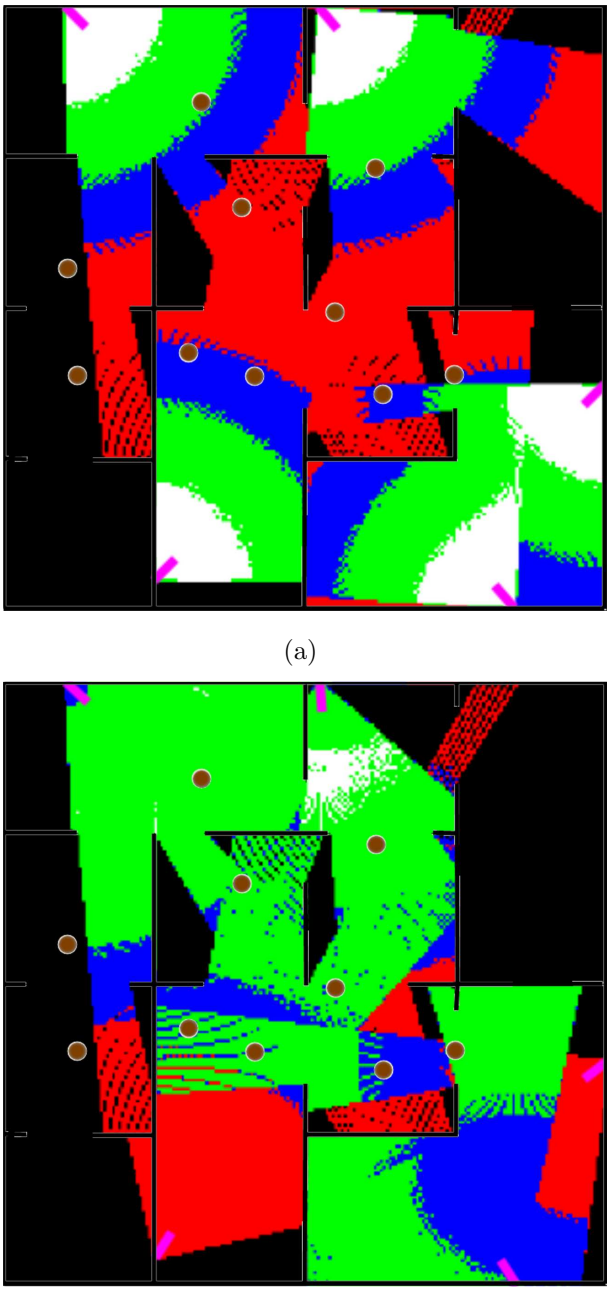
Configuration	White	Green	Blue	Red	Black	C_T	C_G
Initial	0	0	1	4	5	0.1973	0.5058
Final	0	4	3	2	1	0.6797	0.4806

2.4.2 3D model evaluation

In this sub section 3D model evaluation is presented, such a model is highly accurate and descriptive. It is very useful when there is no limit on computational complexity and requires highly accurate camera configuration.

Scenarios

In order to evaluate the performance of our optimization tool we present here the results obtained in two virtual scenarios. We also demonstrate the benefits of the solution when applied in a real context by comparing the camera deployment obtained by our method with the existing pre-installed cameras configuration. Eventually, we will apply common feature extraction and detection algorithms to demonstrate the improved efficiency of our solution. In order to correctly represent the environment, its description includes:



(a)

(b)

Figure 2.11: Map 3: (a) Initial positioning, and (b) after reconfiguration.

1. *Geometry* of all walls and floor in terms of planes, in a 3D coordinate system. In case an object cannot be described in geometrical terms, its nearest approximation is obtained as a combination of regular geometric shapes.

2. *Texture information* of the walls, ground plane, and other objects, described using the POV-ray scene description language.

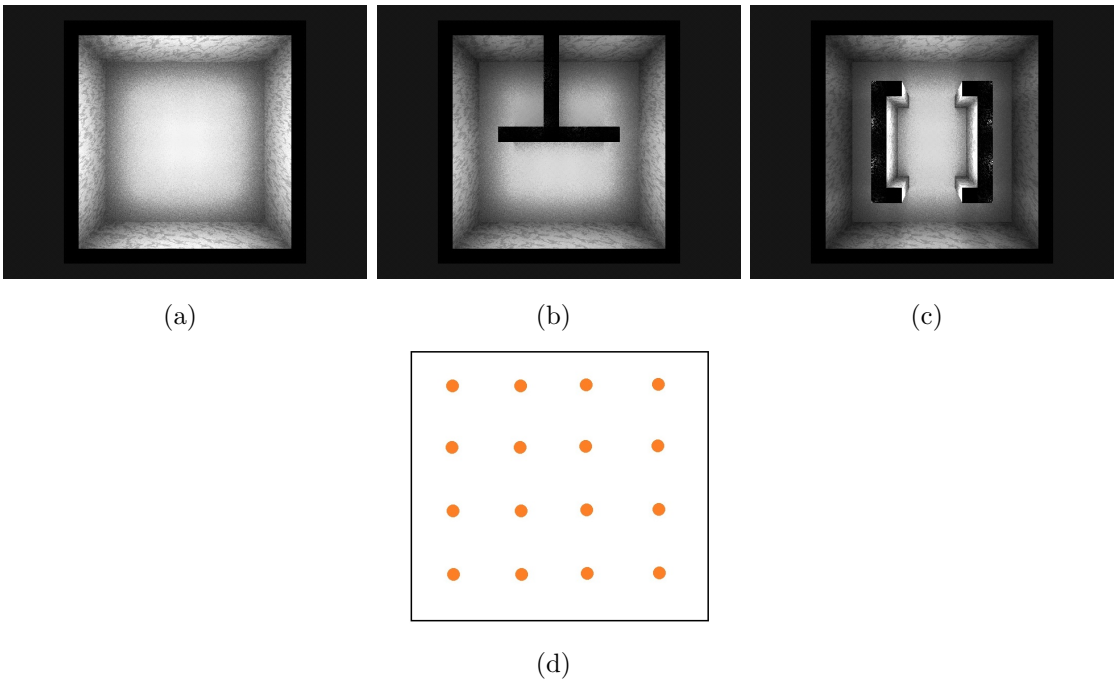
3. *Radiometric properties*, as diffusion of light over the surface and reflectivity.

4. *The configuration of the illumination sources* has also to be specified (approximately) in terms of color of the light, power, shape, location, and type (circular, cylindrical, parallel, spot etc.)

Virtual scenarios In order to test the algorithm, we will report here for the sake of demonstration two different realistic environments of medium size. The first environment (Fig. 2.12(a)) is a large empty hall of $15X15m^2$ in size, with walls that are $3m$ high. The room exhibits a texture in light gray. The diffusion of the walls is set to 0.5, and the reflection coefficient to 0.3. The remaining 0.2 corresponds to dissipation.

The second scenario, see Fig. 2.12(b), maintains the same properties of the first map, but in addition we introduce a T-shaped wall in the hall with the center of the T coinciding with origin of the environment reference system. The introduction of the obstacle severely alters the environment in terms of light spread and reflections.

In the third scenario, we introduce two C-shaped walls symmetrically placed in between the Map. The idea is to explore the performance of algorithms under different obstacle conditions. Two C-shaped walls are added and the overhead view of the environment is shown in Fig. 2.12(c)



(a) (b) (c)

(d)

Figure 2.12: Overhead view of the three maps and the lighting system.

As far as the lighting system is concerned, 16 white lights, each of them with a wattage of 100W, are uniformly distributed in the environment. We modeled the light source using so-called *area lights*, consisting of equidistant two dimensional array of point light sources whose cumulative wattage is equal to the value specified for the total light source.

Lights are placed at the height of $3m$ at a distance of $1.4m$ one from each other both along Z and X (Fig. 2.12(c)). For the attenuation of light we have used an inverse square function.

Real scenario To evaluate the validity of the proposed solution in a real scenario, we have selected a suitable test site consisting of an apartment designed to test assistive technologies for home automation. The apartment has an inbuilt surveillance system with video cameras already installed and calibrated. We propose to compare the existing surveillance system, against our deployment, designed according to the algorithm proposed .

For a fair comparison we simultaneously record with both systems a video showing people moving arbitrarily in the apartment. After recording we carry out both qualitative and quantitative evaluation. Due to the small size of the evaluation site, coverage is only a limited problem, hence our evaluation mainly focuses on the quality and the information content of the videos. All the details of the space are appropriately modeled, including color and texture. Even the natural light coming from the window is modeled using a small light source. The image of the real scenario and its virtual counterpart are shown in Fig 2.28 and 2.29, respectively.

Evaluation of the camera model

The metrics we have implemented to evaluate the camera positioning, are based on entropy and distortion.

Entropy essentially corresponds to the amount of information, or level of detail measured in the snapshot captured by the camera. It depends on many factors including occlusions, orientation, lighting. However, one element that greatly influences the level of detail in any image is the focal length of the camera. In order to demonstrate the validity of the entropy as a metric, we vary the focal length of the camera from $5mm$ to $85mm$ and track the variation of entropy both qualitatively and quantitatively. The environment used for testing (simulated using POV-ray) consists of a large room (sized $4.6 \times 4.85m^2$, with reflection coefficient 0.3, and a central lighting of 100W) with a sphere of radius $0.5m$ positioned in the center. The camera is placed image plane points towards the center of the sphere. The origin of the coordinate system is located at the center of the room.

Fig. 2.13 shows the variation of entropy with respect to the focal length in steps of $5mm$ from $5mm$ to $85mm$ for a total of 17 samples. The entropy measure is calculated according to Eq. 2.17. In order to explain the graph we have selected six images at various critical points in the graph, shown in

Fig. 2.14. The corresponding focal lengths and entropy values are reported in Table 2.4.

In Figure 2.14(a) most of the details of the environment are visible. Consequently the entropy measure obtained for the image is also high. As the focal length increases, we observe a progressive change in the entropy curve with occasional transients, mainly due to the change in the amount of details present in the captured snapshot. As can be seen from Figure 2.14(b-g) the variation of entropy is heavily dependent on the nature of the environment and the objects being observed. Hence, it is imperative that the camera system continuously re-adapts to optimize itself w.r.t entropy.

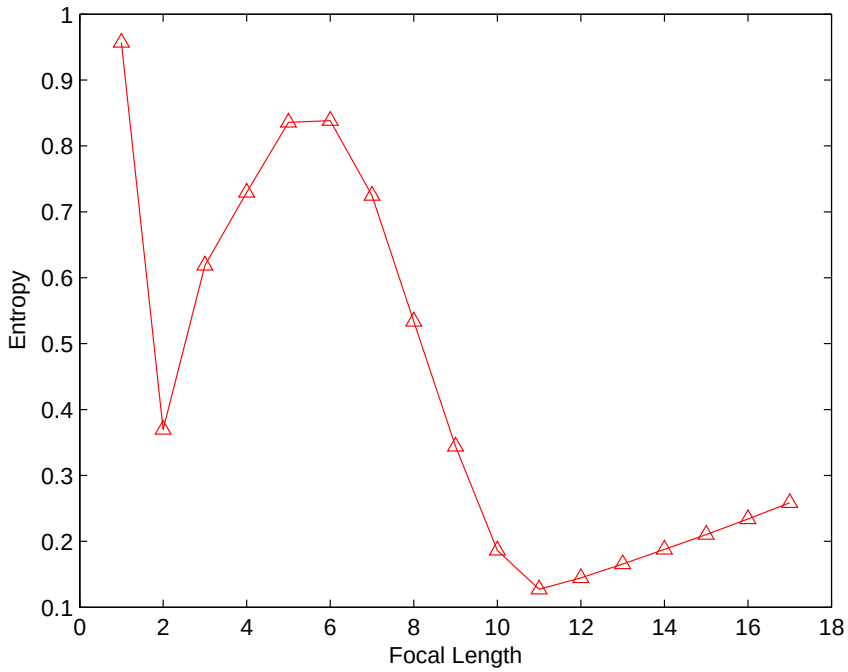
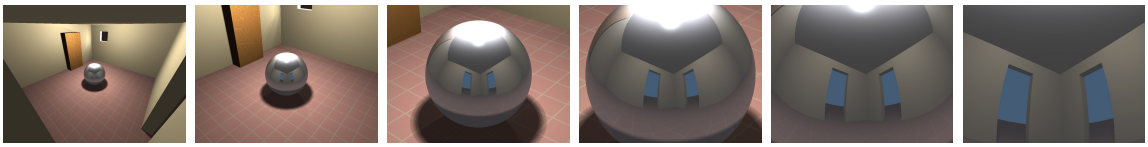


Figure 2.13: Variations in the entropy values obtained at different focal lengths.

In order to validate the measure of distortion, we present a sample image in Figure 2.15(a), where six spheres have been placed at various locations.



(a) (b) (c) (d) (e) (f)

Figure 2.14: Images obtained at different focal lengths.

Table 2.4: Entropy and focal length of images

Figure	Focal Length	Entropy Measure
2.14(a)	5 mm	0.9568
2.14(b)	10 mm	0.3697
2.14(c)	25 mm	0.8357
2.14(d)	40 mm	0.5337
2.14(e)	55 mm	0.1274
2.14(f)	85 mm	0.2584

The location of the camera for this particular simulation is the same as in the previous case, the focal Length is set to $4.2mm$. The locations of the spheres and mean distortions of each of them are listed in Table 2.5. Spheres are listed in the table following a clockwise arrangement starting from the one closest to the top left corner of the image. Using the camera model and the distortion metric, we have also divided the image into zones of distortion as shown in Figure 2.15. Red areas indicate the maximum distortion, while green areas exhibit the lowest level of distortion. Comparing the two figures 2.15(a,b) and also by looking at the table, we can clearly see that deformity of the spheres increases with the increase in distortion.

Results for virtual test cases

In order to test both the initial placement and reconfiguration we initially, position the predefined number of cameras according to the planning algorithm presented in Section 2.3.3. Considering the size of the environment, the number of cameras is fixed to four. After the initial positioning,

Table 2.5: Distortion and location of spheres

Location	Distortion
-1,0.25,1.5	0.6400
1,0.25,1.5	0.3828
-1,0.25,.25	0.7018
1,0.25,.25	0.5471
-1,0.25,-.75	0.7511
1,0.25,-.75	0.6560

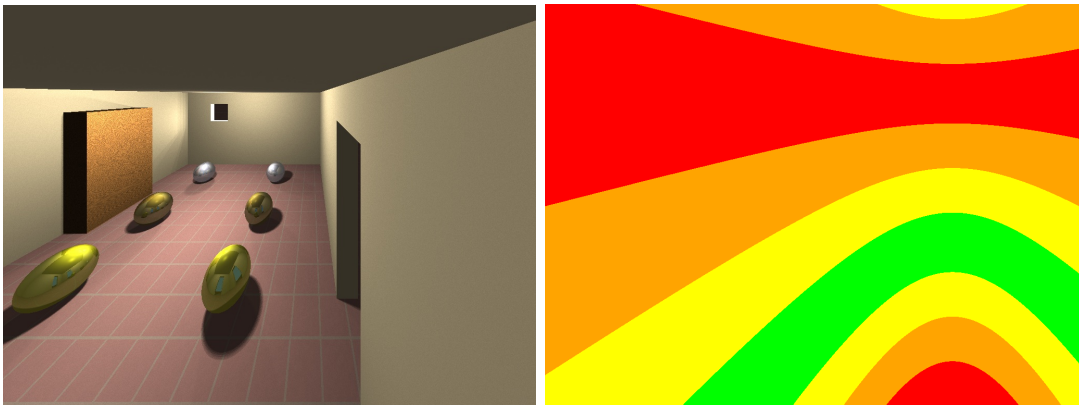


Figure 2.15: Sample image and distortion zones. Distortion is defined according to Eq. (2.10). Colors corresponds to different levels of distortion: red correspond to $\mu_D > 0.8$, orange $0.8 > \mu_D > 0.5$, yellow $0.5 > \mu_D > 0.2$, and green $0.2 > \mu_D > 0$

one of the cameras (selected randomly) is turned off and the algorithm is again applied to reconfigure in terms of PTZ. The absolute positions of the remaining three cameras remain the same, as they are already deployed (installed). Similarly to the previous scenario, another camera is turned off and the reconfiguration is performed for the remaining cameras in order to readjust the setup.

The radiometric constants used for the virtual environment are tabulated in Table 2.6.

Evaluation on Map1 In this scenario maximizing the coverage is a rather simple problem, due to the absence of obstacles. The main purpose of placement is then to avoid redundancy (i.e covering the same area with

Table 2.6: Environment radiometric constants

Property	Constant
Fade distance	0.2
Fade power	2
Diffusion	0.3
Turbulence	1

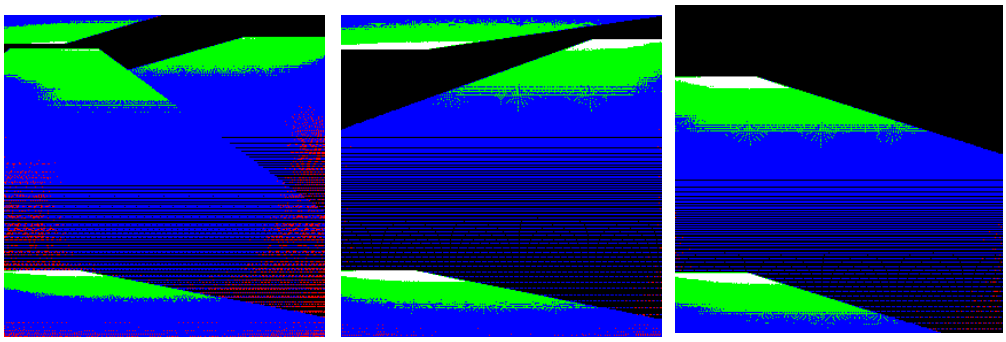
multiple cameras, unless specifically requested), while maximizing the quality of the captured data. Placement locations along the walls have been quantized at the rate of $0.5m$ both in X and Z. The focal length range has been varied from $5mm$ to $85mm$ in steps of $5mm$, while *pan* and *tilt* vary from 0° to 180° and 0° to 90°, respectively at steps of 1°.

Figure 2.17 shows the snapshots obtained from the cameras after the initial positioning. The total coverage obtained is 76.43%. As can be seen from Table 2.7, whenever the coverage of the individual camera is very small, it is compensated by low levels of distortion and high entropy, since our formulation assigns equal importance to both coverage and quality.

The final coverage map is shown in Figure 2.16(a). For the sake of visualization, a color code is used to describe the density of the rays that intersect the cells of the grids. Cells colored in *white*, correspond to more than 100 rays per cell, whereas *green* represents areas, which ray density is between 10 and 100. *Blue* regions have a ray density between 1 and 10, and *red* areas have only one ray per cell. We can also observe from the results that the area of intersection between FOVs of the individual cameras is limited, which is one of the objectives of optimization.

Table 2.7: Map1: Entropy and distortion after the initial setup.

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2085	0.5566	0.2278	15.18
2	0.8935	0.4832	0.1952	3.07
3	0.1489	0.4682	0.0715	40.17
4	0.2131	0.3314	0.0545	40.54



(a) (b) (c)

Figure 2.16: Map1: total coverage map after initial placement (a), reconfiguration 1 (b), and 2 (c).

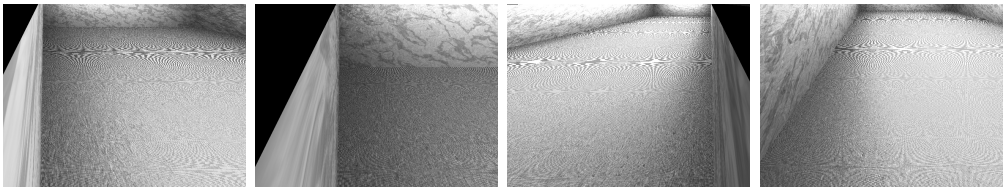


Figure 2.17: Map1: snapshots from four cameras.

After the initial positioning, we now turn off the fourth camera, simulating a malfunctioning. Due to the malfunction the system setup becomes sub-optimal. In order to improve its performance, reconfiguration has to be performed in terms of *pan*, *tilt*, and *zoom*, as the absolute positions are fixed. The algorithm is then applied again, and the configuration is updated. The resulting snapshots from the cameras are shown in Figure 2.18. Table 2.8 lists the individual distortion and entropy metrics of the cameras. The area covered reduces to 61%, compared to the 54%, which would have been the case, if reconfiguration was not run. The overall coverage map is provided in Figure 2.16(b). As can be seen from Table 2.8, camera 2 has extended the coverage area in order to compensate for the loss of one camera in the network. Subsequently, its covered area has doubled, while the entropy has halved.

Table 2.8: Map1: entropy and distortion after reconfiguration 1

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2124	0.5474	0.2139	14.87
2	0.4912	0.6216	0.3620	6.92
3	0.1479	0.4393	0.0661	40.13
4	OFF	OFF	OFF	OFF

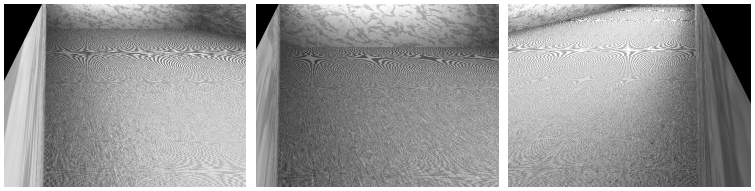


Figure 2.18: Map1: snapshots from the three cameras after reconfiguration 1.

The second reconfiguration implies turning off another camera in the system, to further test the resilience of the algorithm. Camera 3 is then turned off and reconfiguration is again performed. Figure 2.19 shows the obtained results. The total coverage for the camera network is shown in

Figure 2.16(c), while Table 2.9 showcases the quality metrics of the individual cameras. The overall coverage has reduced to about 45%, compared to 21.22%, the coverage obtained with only two cameras before reconfiguration.

Table 2.9: Map1: entropy and distortion after reconfiguration 2.

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2124	0.5474	0.2139	14.87
2	0.4912	0.6216	0.3620	40.42
3	OFF	OFF	OFF	OFF
4	OFF	OFF	OFF	OFF

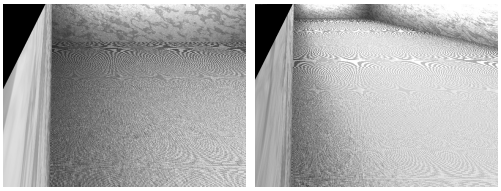
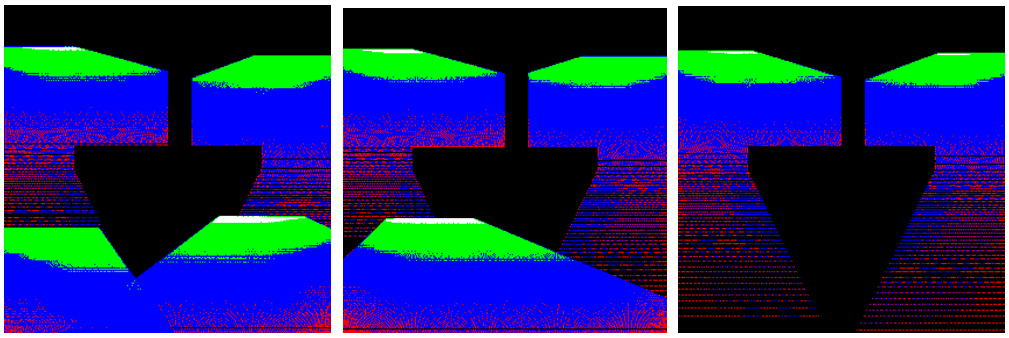


Figure 2.19: Map1: snapshots from two cameras after reconfiguration 2.

Evaluation for Map2 In this case Map 2 is subjected to the same sequence of tests as Map1. As discussed in Section 2.4.2, Map2 exhibits a T-shaped wall in between the map, which divides the environment into three rooms. All the configuration parameters used in the previous simulation are kept the same. Similarly to the previous case, the number of cameras that have to be placed initially is four. The obtained results are shown in Figure 2.21, and the corresponding metrics are reported in Table 2.10, showing a coverage equal to 62%, and minimum overlap across the views (Figure 2.20(a)).

Similarly to the routine adopted for Map1 we now turn off camera 4 and realign the camera parameters using the same algorithm (see Figure 2.22 and Table 2.11). The total coverage with three cameras is reduced to



(a) (b) (c)

Figure 2.20: Map2: total coverage map after initial positioning (a), reconfigurations 1 (b) and 2 (c).

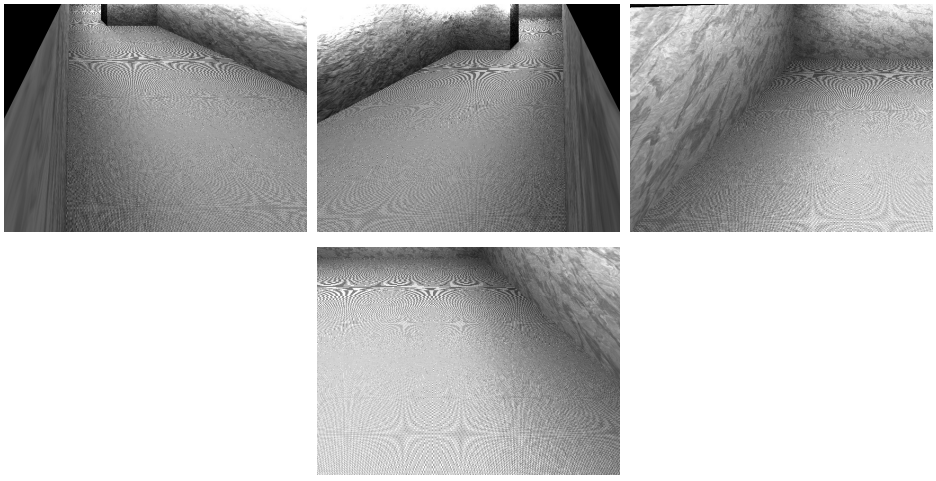


Figure 2.21: Map2. snapshots from four cameras

Table 2.10: Map2: entropy and distortion after the initial setup.

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2374	0.4545	0.1205	17.65
2	0.2305	0.3671	0.0442	17.85
3	0.2815	0.2584	0.0259	13.03
4	0.1279	0.3685	0.0733	23.03

about 61%, comparing well to 48% that would result in case no reconfiguration would be performed. Figure 2.20(b) presents the color map of the corresponding coverage after reconfiguration.

Table 2.11: Map2: entropy and distortion after reconfiguration 1.

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2446	0.4570	0.1255	17.01
2	0.2258	0.3822	0.0467	17.53
3	0.1170	0.3925	0.0634	26.85
4	OFF	OFF	OFF	OFF

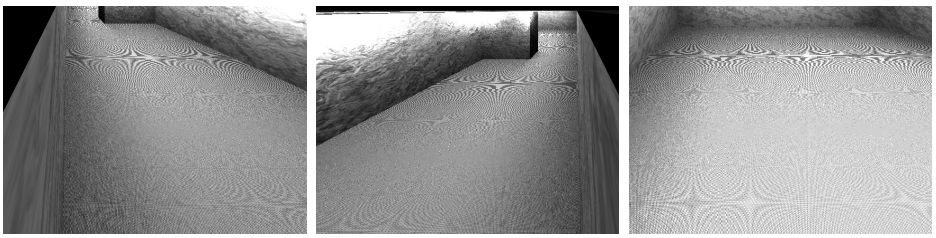


Figure 2.22: Map2: snapshots from three cameras after reconfiguration 1.

After turning off also the third camera (see Figure 2.23 and Table 2.12), the total coverage reduces to 36%, which is marginally better than almost 35%, without reconfiguration. The reason for a limited increment in coverage is due to the fact that the two cameras to be reconfigured are obstructed by walls. The corresponding coverage map is shown in Figure 2.20(c).

Table 2.12: Map2: entropy and distortion after reconfiguration 2.

Cam ID	Entropy	μ_D	σ_D^2	Coverage (%)
1	0.2124	0.5474	0.2139	18.45
2	0.4912	0.6216	0.3620	17.47
3	OFF	OFF	OFF	OFF
4	OFF	OFF	OFF	OFF

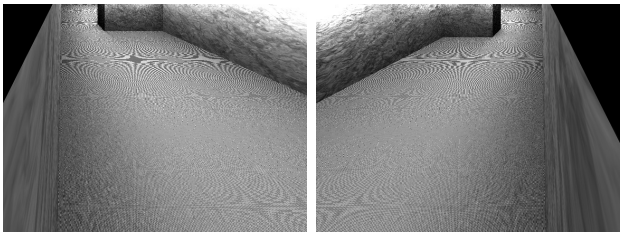


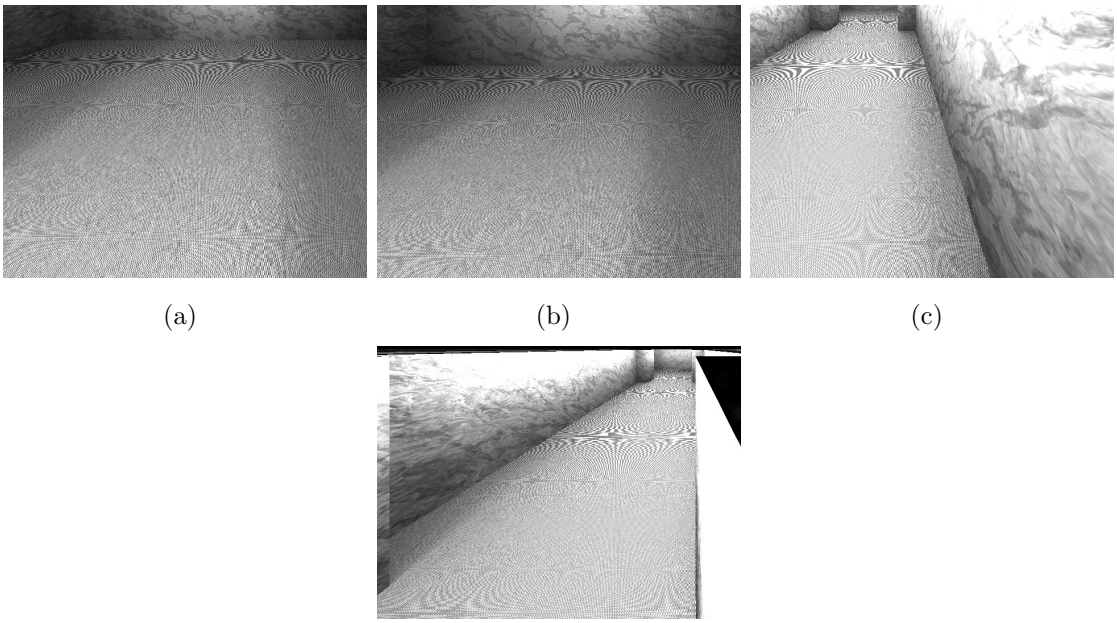
Figure 2.23: Map2: snapshots from two cameras after reconfiguration 2.

Evaluation for Map3 This environment consists of two symmetrically placed walls, in the middle of a large hall way, such arrangement leads to flooding of light in between the two walls and also between the hallway boundary and the walls. Such an arrangement is generally done for Museum or art gallery. We have chosen this case to demonstrate the utility of the algorithm in that context. Results obtained for the placement are shown in the Figures 2.24 and 2.25(a) and the quality metrics are presented in Table 2.13. If we observe the quality metrics of Camera 3 and 4, we can clearly see that entropy has clearly decreased due to the flooding of light and saturation of sensors. Coverage achieved is 48.53 percent.

Table 2.13: Entropy and Distortion for Map2 Initial Setup

Camera No.	$Entropy$	$Dist_{mean}$	$Dist_{Var}$	$Coverage$
1	0.3675	0.5888	0.2768	0.1323
2	0.3641	0.5045	0.1640	0.1197
3	0.0086	0.4201	0.0756	0.1929
4	0.0172	0.4341	0.1098	0.1786

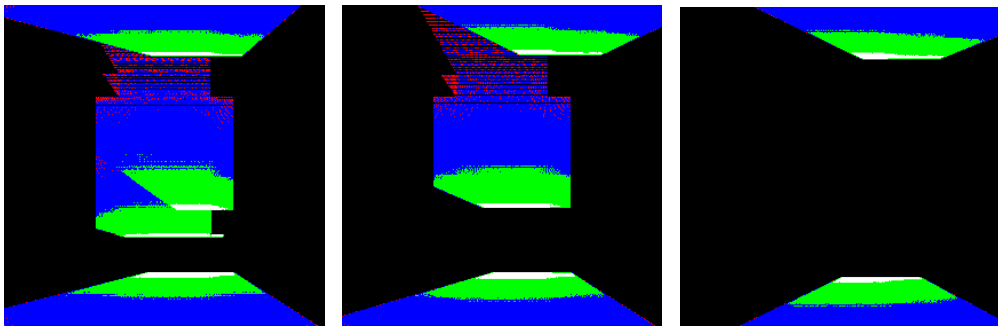
As part of reconfiguration Camera 4 is turned off and the other parameters of configuration are realigned according to the algorithm. Total coverage is about 41.92 percent while with out realignment its 42.27 percent. Coverage is virtually the same however there is marked increase in



(a) (b) (c)

(d)

Figure 2.24: Snapshots from four Cameras after initial placement



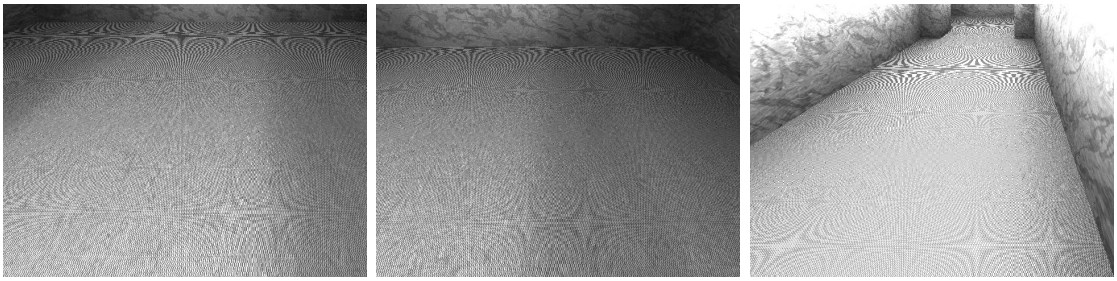
(a) (b) (c)

Figure 2.25: Map3: total coverage map after initial positioning (a), reconfigurations 1 (b) and 2 (c).

quality metrics with entropy increasing and distortion decreasing. so for negligible decrease in coverage the algorithm has increased the quality of coverage markedly.

Table 2.14: Entropy and Distortion for Map1 after Reconfiguration 1

Camera No.	*Entropy*	$Dist_{mean}$	$Dist_{Var}$	*Coverage*
1	0.3824	0.5872	0.2758	0.1244
2	0.5062	0.4549	0.1534	0.0978
3	0.0109	0.3551	0.0477	0.2129
4	OFF	OFF	OFF	OFF



(a) (b) (c)

Figure 2.26: Snapshots from three Cameras after Reconfiguration 1

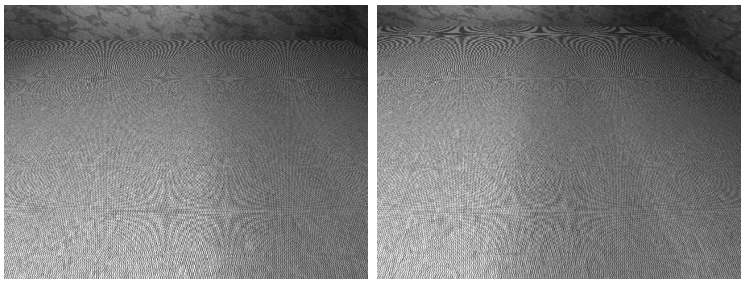
As per procedure camera 3 is also turned of and reconfiguration is performed, the results obtained are shown in Figures 2.27, 2.25(c) and Table 2.15 there is further decrease of coverage at 20 percent while with out reconfiguration its 22 percent however quality metrics have improved by large percentage as we can see from the Table 2.15.

Table 2.15: Entropy and Distortion for Map1 after Reconfiguration 1

Camera No.	*Entropy*	$Dist_{mean}$	$Dist_{Var}$	*Coverage*
1	0.4517	0.4524	0.1454	0.0956
2	0.5676	0.5406	0.2354	0.1034
3	OFF	OFF	OFF	OFF
4	OFF	OFF	OFF	OFF

Real-world evaluation

In the previous sections, the algorithm has been evaluated both in terms of camera model and simulation in two virtual environments. However,



(a) (b)

Figure 2.27: Snapshots from three Cameras after Reconfiguration 2

in order to prove the utility of the algorithm, a real world evaluation is necessary. In order to do so we propose to test the algorithm by actually designing a surveillance system for a real environment. The evaluation is carried out by comparing the quality of the videos obtained from the pre-existing installation and the planned cameras.

The tests have been carried out in a hall of dimensions $4.9 \times 4.6m^2$. The mentioned hall has a pre-existing camera system deployed and calibrated. Three cameras are positioned at three corners close to the ceiling subtending an angle of approximately 45° for both pan and tilt. The snapshots of the test environment are shown in Figure 2.28.

Figure 2.28: Panorama view of the test site.

In order to achieve the optimal planning of the cameras, the test site has to be replicated using POV-ray scene description language. The visualization of the test site in the virtual domain is shown in Figure 2.29.

After translating the environment into the virtual domain we apply our

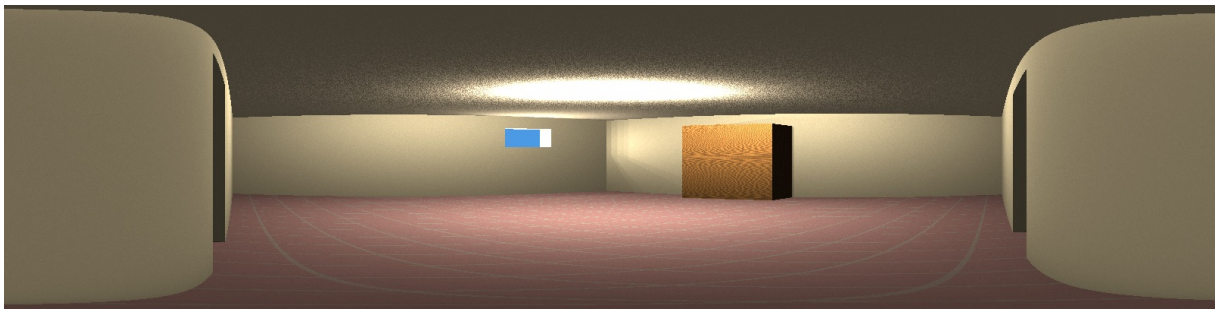


Figure 2.29: Panorama view of the test site as seen in the virtual environment.

algorithm to obtain the camera positions. Considering the limited size of the room, we have chosen to position two cameras and compare against the three pre-installed cameras. All devices used for validation are Axis 212 network cameras. The video snapshots obtained are shown in Figure 2.31. The snapshots taken from the three *unplanned* cameras are also provided for comparison in Figure 2.30.

As can be seen from the visual comparison, the pre-existing cameras suffer from bad illumination and high level of perspective and radial distortion. It can also be noticed that the field of view is not very effectively utilized, whereas in the planned configuration, the quality of illumination is improved and the perspective distortion reduced. The two cameras are complementary in terms of coverage, and still maintaining a very good level of detail of the room.

Comparison and evaluation In order to quantify the effectiveness of our algorithm, we asked two users to move randomly in the room. These actions are simultaneously captured from all five cameras at the frame rate of 15fps, for a total observation time of about 6 minutes. The comparison between the planned and unplanned videos is computed in terms of three measures, as reported hereafter.

1. *Variance of entropy* As entropy in an image represents the amount of

Figure 2.30: Snapshots taken from the unplanned cameras. The limitations of this setup are evident, as a large part of the images refer to non-relevant areas of the environment

Figure 2.31: Snapshots taken from the cameras positioned and configured according to our optimization algorithm. The attention is now concentrated on the ground floor, where relevant activities are more likely to occur.

information, when an event occurs, there will a subsequent change in the entropy of the video frames.

2. *HOG people detector* In order to test the effectiveness of the camera placement we apply a basic HOG cascade person detector, and compute the number of detections captured from both systems (planned and unplanned).

3. *STIP descriptor* We also propose to include in the comparisons, the number of extracted STIPs [34], widely adopted for action recognition and activity detection, in order to also account for the temporal component.

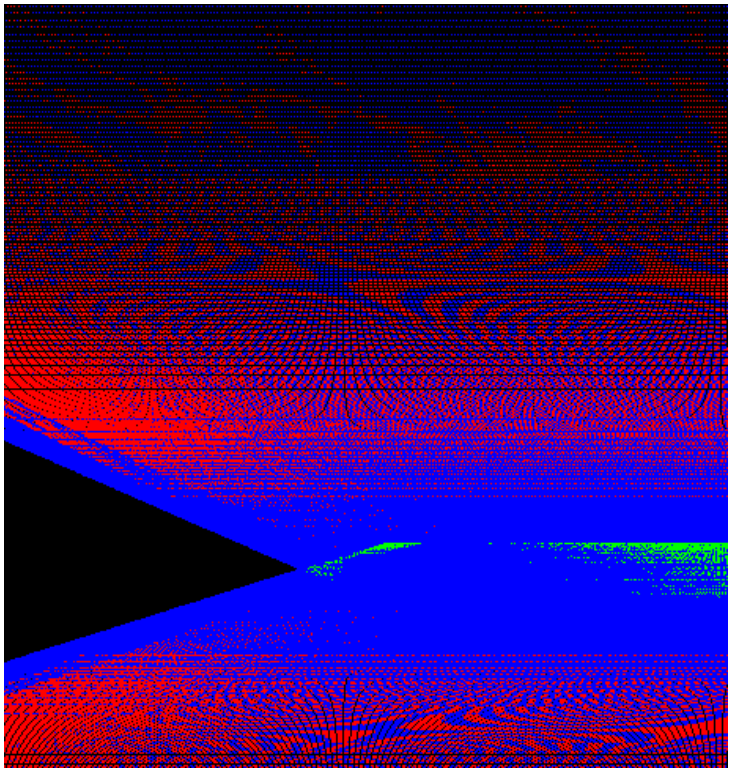


Figure 2.32: Total coverage map for the test site.

Entropy and variation In order to evaluate the entropy model we have compared the values obtained from the unplanned and planned configurations for all 5025 frames recorded by all cameras. The entropy variation has been plotted in Figure 5.6, which shows on the top the plot for the three videos recorded using the unplanned configuration. Camera 1, Camera 2, and Camera 3 are identified in blue, red, and cyan, respectively. Similarly, the second plot in Figure 5.6 showcases the entropy variation for the two videos that have been recorded using the planned cameras, shown in green and red, respectively. It is worth noting that plots report the entropy variation values, obtained after subtracting the mean value, for a better visualization. As can be seen from the figures, the plots obtained from the three unplanned cameras appear to be uncorrelated and exhibit a random nature among each other, whereas the plots for the planned cameras are highly correlated one with each other. Peaks in the plots are in this case correctly associated to the events occurring in the observed scene. The values obtained for mean and variance are reported in Table 2.16. As can be seen, the variance of the two planned cameras is identical, and significantly higher than that of unplanned cameras. The only exception can be noticed for Camera 3 in the unplanned configuration, since the camera faces the two doors in the test site, which contribute to increasing the entropy values.

HOG person detector Considering that the annotation of more than 5000 frames for each camera would be very time consuming, without loss of generality we have downsampled the video to 3 fps, run the HOG people detector and calculated the precision and recall. The obtained results are presented in Table 2.17. The total number of actual persons is reported in column 3, and the number of detections for the two persons are given in columns 4 and 5. Column 6 indicates instead the false detections. Preci-

sion, recall and F-measure are reported in columns 7, 8, and 9, respectively. As can be seen from the table, the two planned cameras outperform the results achieved by the unplanned cameras in most of the cases. On average, there is almost a 34% increase in the F-measure.

Table 2.16: Mean and variance of entropy across the frames.

Cameras		Mean	Variance
Unplanned	Camera 1	6.3274	0.0015
	Camera 2	6.4058	0.0018
	Camera 3	7.2077	0.0011
Planned	Camera 1	6.8157	0.0022
	Camera 2	6.7047	0.0022

Table 2.17: Comparison for the HOG person detector. GT refers to the ground truth, P1 and P2 report the number of detections for the two subjects, respectively, and FP reports the number of false detections.

Cameras		GT	P1	P2	FP		Precision	Recall	F-measure
Unplanned	Camera 1	1522	202	317	8		0.98	0.34	0.5
	Camera 2	1657	229	290	2		0.99	0.31	0.48
	Camera 3	1648	313	458	97		0.89	0.47	0.61
Planned	Camera 1	1442	283	365	11		0.98	0.45	0.66
	Camera 2	1610	454	548	6		0.994	0.62	0.75

STIP extraction STIPs are obtained for all the video 5025 frames. STIPs have been extracted simultaneously on all cameras, using a patch size of 5 and 3 levels in the pyramid. The results obtained are tabulated in Table 2.18. As can be seen from the table the number of STIPs obtained for the planned cameras is significantly higher when compared with the unplanned ones, showing on average an increase of 33% for the same activity.

Table 2.18: Number of STIPs obtained.

Cameras		STIPs	STIPs/frame
Unplanned	Camera 1	95083	18.92
	Camera 2	79540	15.82
	Camera 3	80166	15.90
Planned	Camera 1	112470	22.38
	Camera 2	114391	22.76

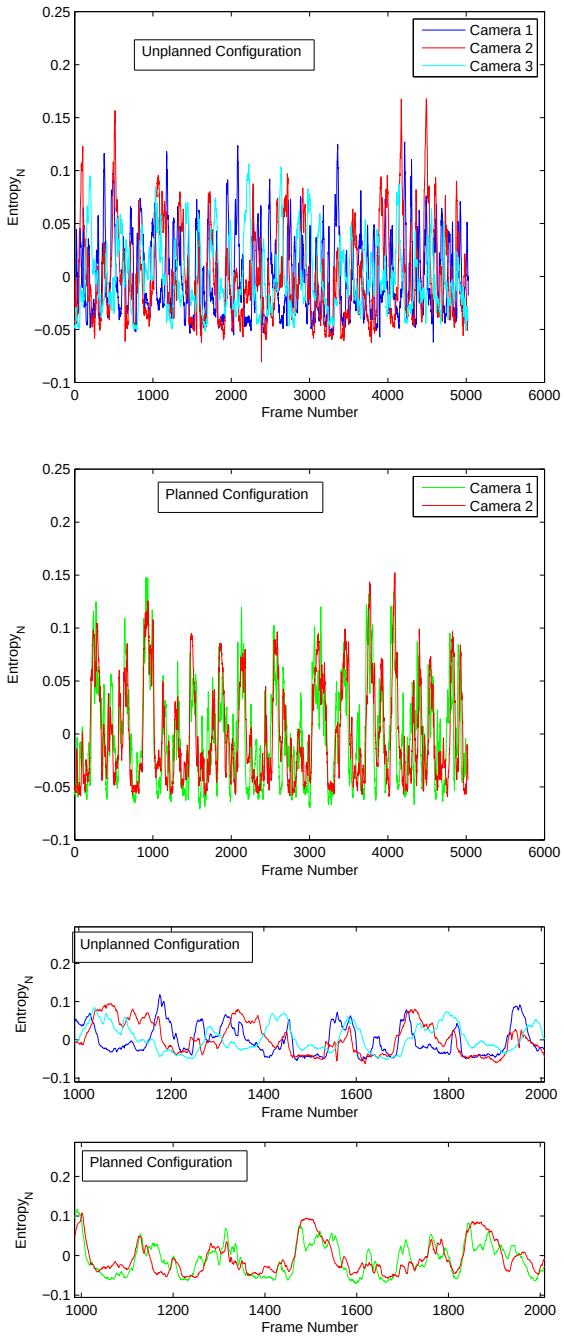


Figure 2.33: Plots reporting the difference in terms entropy variations for the unplanned and planned configurations. A zoomed segment from frame 1000 to frame 2000 is also shown in the last two plots. For the sake of visualization, the mean value of entropy has been subtracted from each sample.

Chapter 3

Light planning

Illumination is one of the most important aspects of any surveillance system. The quality of images or videos captured by the cameras heavily depends on the positioning and the intensity of the light sources in the environment. In this chapter we propose a novel 3D modelling of a given environment in a synthetic domain, combined with a generic quality metric, which is based on entropy measurement in a given image. The synthetic modelling of the environment allows us to evaluate the optimization problem a priori before the physical deployment of the light sources.

3.1 State of the art

Illumination is a very critical aspect in video and image acquisition, and the quality of the captured images is directly dependent on the exposure to light of the given scenario. Owing to this fact the success or failure of any video surveillance algorithm heavily depends on the positioning and the intensity of the light sources in the environment. A sub-optimal light placement will result in a limited performance of the surveillance systems. In particular two aspects need to be carefully taken into account: video quality, and adaptation.

Video quality. The sensors installed onboard video cameras are responsive to light intensity in specific wavelengths, and the response of the sensors is not linear across the range of responsiveness [30, 72]. In case of incorrect illumination, there is a chance that the sensor response falls in the non linear region. In such scenario quality and details of the scene may be lost, and may not be recoverable even after post processing. This will severely impact the efficiency of the algorithms.

Adaptation. The decrease in the cost of video cameras and the increasing efficiency of the algorithms for automatic scene analysis, has motivated the adoption of automatic video surveillance systems able to continually track the events, occurring in the observed scene, for activity monitoring and anomaly detection. These systems operate in real time with an intent to prevent untoward incidents. However, most descriptors used in computer vision like for example SIFT [39], HOG [14], SURF [4], are highly sensitive to the quality of illumination, since they essentially depend on the pixel intensity variation. Therefore, a proper planning of the illumination sources is of paramount importance.

Considering the issues mentioned above, positioning of light sources in a given environment is very important for a successful deployment of any computer vision algorithm. There has been a certain amount of work done in this area. Murase et al. [48] devise an optimal illumination based on the variation of contrast among the given set of objects in the obtained images, mainly for object recognition purposes. Similarly Ellenrieder et al [19] arrive at a suitable illumination considering the reflectivity of the surfaces involved. In this case the focus is on the reflectivity model of the surfaces. Further Borotschnig et al [6] present an efficient on line object recognition algorithm with a provision for reconfiguration which also takes into account the illumination changes. However these algorithms are rather task

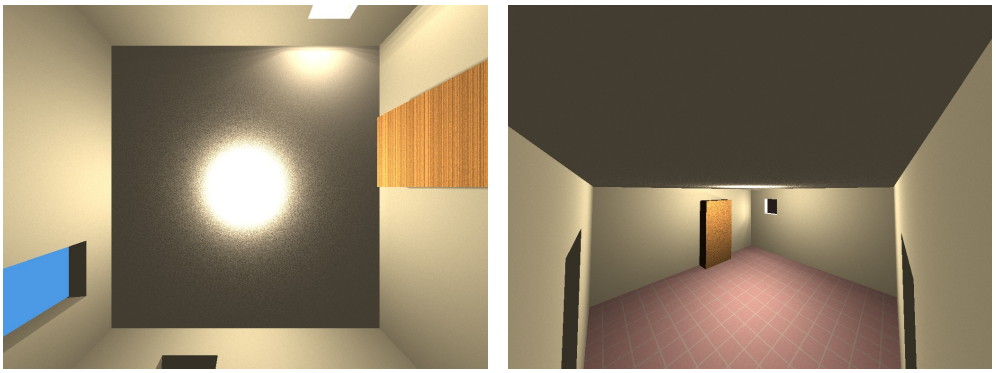
specific and consider the given illumination condition rather than treating light placement as the core problem. One of the first papers to address light source placement as serious problem in camera planning system is by Sakane et al. [60]. However, in this case, rather simplistic models are used for both camera and light sources, which make the approach unreliable in a real deployment. Reddy and Conci [55, 56] address the problem of the light sources using inverse square attenuation of light with a model of equal propagation of light in all directions.

3.2 Illumination and environment Models

In this section we describe the illumination and environment models used to model the light sources and the environment in the 3D virtual space. The discussion is very specific to illumination and radiometric properties.

3.2.1 Illumination model

The amount of light at any given area in a surveillance environment is a complex function of various radiometric phenomena like diffraction, diffusion, and reflection. In order to model the illumination in all these factors are taken into account in the our model. This is possible thanks to the POV ray software, which allows specifying the diffusion and reflection coefficients. By incorporating these factors we can model the light sources in a very accurate manner. In the POV ray software there is also a provision for selecting the nature of light, as for example the type of light sources (circular, cylindrical, point, parallel etc.). In real world light attenuates as it travels from the light source, and in our illumination model we adopted an inverse square attenuation. Figure 3.1(a) illustrates a sample light source, which has been simulated using our illumination model.



(a) (b)

Figure 3.1: Example of a light source (a), and of the environment (b) generated using the POV ray tracing software.

3.2.2 Environment model

As far as the environment is concerned, most of the present state of the art only consider the geometric features. We in turn propose to include all properties of the environment, which are likely to influence the illumination of the environment. These include color, reflection coefficient, nature of the surface of the walls etc. As mentioned earlier, the idea is to replicate the environment in the minutest detail possible. In addition to these properties we also take care of shadows and other illumination-related issues that may affect the environment. Figure 3.1(b) shows the snapshot of the replicated environment.

3.3 Proposed algorithm

After presenting the environment and illumination models, we introduce hereafter the optimization algorithm. Initially the whole environment where the cameras and light sources are to be fixed is given as an input the algorithm in the form of geometric description along with the coloring and radiometric properties in order for it be replicated in virtual domain. Similarly, also the light positions and the nature of light sources is also

given as input to the algorithm. It is then up to the PSO algorithm to arrive at a solution, which returns the configuration at maximum entropy.

3.3.1 Entropy calculation

A good visibility of the observed scene is a fundamental issue for a correct application of automatic analysis tools. Therefore, while planning the light source placement, several factors have to be considered, as the exposure to the correct amount of light, avoiding over and under-exposure. However, fitting the sensor response of the image sensors directly is not a viable option, since camera planning should be done before the deployment. Instead, in our system we deploy virtual cameras that replicate the performance of the camera in the real domain. We successively use the images generated by the virtual model to assess the sensor response.

In order to measure the amount of information present in the captured image, we propose to use the histogram of the image, by determining the bins distribution. In fact, in presence of an under-exposed image, the histogram will be biased towards the lower levels, while for an over exposed image, the histogram will be likely biased towards the higher intensity levels.

In order to quantify the image information content, we measure the entropy. For a given configuration I_i of the image I, with $i \in \{1, \ldots, N\}$ let the intensity levels be $\Omega = (l_1 < l_2 < \ldots l_i \cdots < l_N)$, and let the information content in the image given by

$$H(\Omega) = \sum_{p_i \in \Omega}^{n} -p_i log p_i \tag{3.1}$$

where p_i is the normalized region under each intensity level $l_i \in \Omega$ in the LAB color space, and considering that the entropy is directly proportional to the information content of the image as also suggested from the literature

[27]. After having established the metric to measure the visual information in an image, all we have to do is maximize the image information, given a solution space resulting as all possible configurations of the light sources in terms of positioning in (X, Y, Z). The entropy calculated according to (3.1) will become the basis of the fitness function to be used by the PSO.

3.3.2 Particle Swarm Optimization

For the optimization procedure, we adopt the PSO algorithm. PSO [16] demonstrated to be effective in solving complex non-linear multidimensional discontinuous problems in a variety of fields [17]. Unlike other multiple-agent optimization procedures such as Genetic Algorithms (GA) [24], PSO is based on the cooperation among the agents rather than their competition.

3.3.3 Algorithm

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

Step 1 - Determine the solution space. After identifying the roof and the perimeter walls of the environment, possible light deployment positions are identified and categorized as solution space. A set of particles positioned randomly is chosen as the initial configuration of light positions. The environment information is then converted into POV-ray scene description language and the PSO is initialized. At each iteration of the algorithm, the particles are updated based on the fitness function. Successively, the POV-ray scene description language is updated according to the camera location and the camera parameters.

Step 2 - Calculate Entropy. Camera views are generated using POV-ray and the entropy is computed as per Eq. (3.1).

Step 3 - Calculate configuration cost. The entropy calculated from the individual virtual cameras in the POV ray space is added up to obtain the configuration cost. The total entropy value is then normalized by expressing it as negative exponential function.

Step 4 - PSO. The fitness function is updated with the obtained negative exponential, and the set of particles in the swarm optimization are updated until the termination criterion is reached. The equation of fitness function is given in Eq (3.2) where $H(\Omega)_1$ and $H(\Omega)_2$ are the entropies obtained from the snapshots of the two cameras.

$$F(I) = \exp(-[H(\Omega)_1 + H(\Omega)_2]) \tag{3.2}$$

3.4 Testing and results

Test Scenario In order to test this algorithm we have modelled a large room with dimensions of $4.6X4.85$ with doors, windows, and furniture in the virtual space. Possible light placement locations are everywhere across the ceiling. In order to simplify the problem, the number of light sources is prefixed as four, with possible wattage from 10 to 50 watts. Two virtual cameras are deployed along the coordinates shown in Table 3.1, which have been generated by the camera placement algorithm proposed by Reddy and Conci [57]. The virtual snapshot for the environment is shown in Figure 3.1b. The reflection coefficient of the walls is fixed at 0.5 out of the maximum of 1. Light attenuation is modelled as an inverse square function. Similarly, light sources are modelled as circular lights facing downward from the ceiling. Given the environment information, the implemented algorithm is used to find the best possible light configuration. In order to eliminate the randomness of the PSO, the algorithm has been run 5 times and the snapshots of the best result obtained has been shown in

Figure 3.2. The location of the lights obtained after applying the algorithm are given in Table 3.2, and the quantitative results are shown in Table 3.3. Column 1 and 2 list individual entropies obtained for each camera, while column 3 shows the fitness function obtained as described by Eq. (1). As we can see from the results, except iteration 2, all other results have reasonable variance, in iteration 1, 2 and 4 the entropy has decreased drastically in case of camera 2.

Comparison To further evaluate the performance of the algorithm, we have compared the results of five iterations with a typical installation where four lights are deployed equidistantly on the ceiling, with the power of 25 watts, which is a typical case of an indoor installations of such magnitude. After performing the simulation we have obtained a value for the entropy of 6.8171 for Camera 1 and an entropy of 6.1785 for Camera 2 and the fitness function scored 0.2727. The snapshots obtained from two cameras are given in Figure 3.3. When we compare the obtained results with our optimization we can see that in terms of fitness values and individual entropies, four of the iterations are better than selecting an equidistant placement.Also we can see from the Figures 3.2 and 3.3 that there is better visual quality in case of algorithm based placement,where in walls are clearly visible while absent in equidistant placement. Similarly, if we consider the total light power wattage deployed, our solution also results in a reduce power consumption compared to the standard deployment.

Table 3.1: Camera Locations.

Camera	X	Y	Z	Pan	Tilt	Zoom
1	2.25	2.4	2.325	54.07	-46.02	0.3883
2	2.25	2.4	0.325	-45	-54.09	0.4193

Table 3.2: Light Locations.

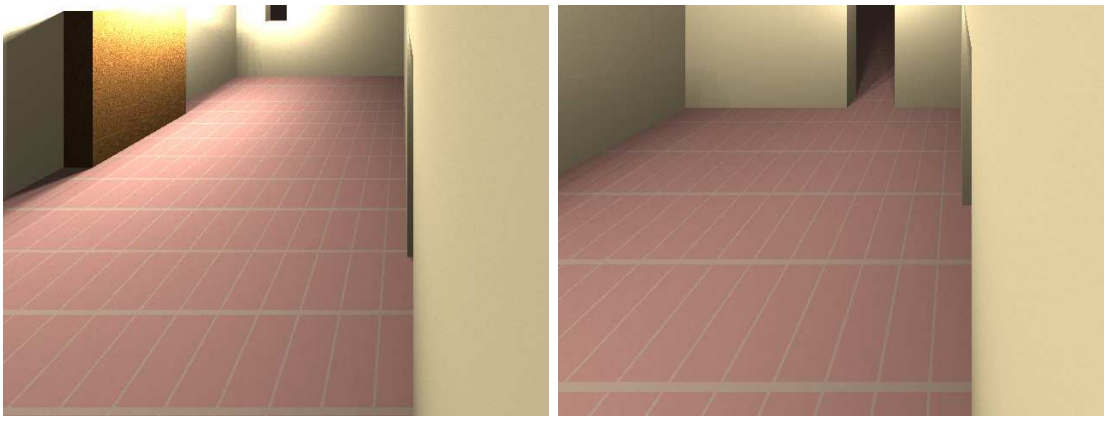
Iteration	Light 1	Light 2	Light 3	Light 4
1	[-1.84,2.45,0.4850]	[2.3,2.45,1.4550]	[1.38,2.45,0.97]	[-0.92,2.45,-1.94]
2	[-0.92,2.45,-0.9700]	[-0.92,2.45,1.4550]	[-2.3,2.45,2.425]	[1.38,2.45,0]
3	[-0.92,2.45,-0.9700]	[-0.92,2.45,1.4550]	[2.3,2.45,0.97]	[-1.84,2.45,-0.4850]
4	[1.38,2.45,0.97]	[-0.92,2.45,-1.94]	[-0.92,2.45,-1.4550]	[-1.84,2.45,-0.97]
5	[1.38,2.45,1.4550]	[1.38,2.45,-0.4850]	[0.46,2.45,1.94]	[0.92,2.45,-1.4550]

Table 3.3: Results.

Iteration	Camera 1	Camera 2	Fitness
1	6.7833	6.2912	0.2704
2	5.7491	5.7668	0.3161
3	7.4232	7.2145	0.2314
4	7.2070	6.0929	0.2645
5	7.2562	7.3565	0.2319

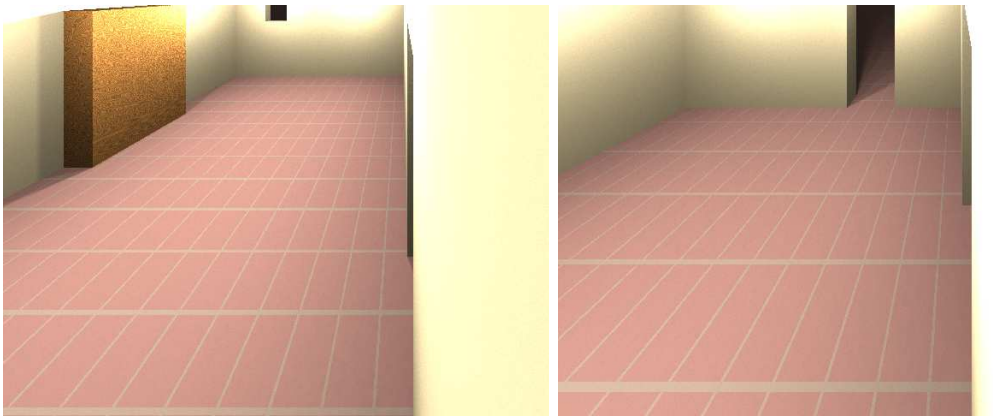
Table 3.4: Light Power in watts.

Iteration	Light 1	Light 2	Light 3	Light 4
1	21.15	26.85	39.75	37.8
2	25.6	39.10	39.45	33.8
3	36.85	10.75	17.8	33.95
4	15.2	33.7	41.25	10.2
5	13.95	12.9	33.65	33.85



(a) (b)

Figure 3.2: Snapshots from two cameras after Light placement



(a) (b)

Figure 3.3: Snapshots from two cameras Equidistant placement

Chapter 4

Video Analytics

In this chapter we describe the relevance of the video analytics in compressed domain to the current problem. Along with that we also present some state of the art, compressed domain techniques developed as a part of the study. These techniques have been partially used for quantifying the information video scene, along with video segmentation for object localization. These metrics and algorithms form a basis for event based real time reconfiguration.

4.1 State of the art

In present day computer vision scenario research in the area of event detection, action recognition etc. is quite mature [4, 8, 14, 34]. Accuracy of the algorithms is quite high and many have been commercialized into products [68]. However many of the algorithms are highly complex and operate off-line, another important factor is they operate in traditional pixel domain. Operation in pixel domain implies the video need to be decoded from its standard compressed format,in order for the algorithm to be applied. In the context of the present study, real time reconfiguration requires low complexity and video decoding adds another dimension to the complexity. Hence our aim is o present a generic low complexity video

event detection algorithm in compressed domain. Most of the work in event detection is concentrated towards specific applications rather than a generic algorithm. Prominent among the applications are video object segmentation and tracking, and some other applications like fall detection,face recognition, skin colour detection etc.

4.1.1 MPEG

Since MPEG 2 and MPEG 4 visual 2 are the early standards in video compression most of the early work done is in this domain [3, 32, 33, 42, 67]. The main characteristics of this domain are the uniform block size of 8×8, motion estimation at 8×8 block level unidirectional motion prediction and absence of half pel and quarter pel motion estimation. This resulted in low complexity and also low accuracy of the algorithms based in MPEG domain. However MPEG-4 visual had object based video encoding which was very useful for video object segmentation.some algorithms use a combination of DCT coefficients and motion vectors, while on the other hand the information embedded in the motion vectors is used exclusively. In [75] Yu et al present an algorithm based on motion vector clustering and background subtraction, applied in the DCT domain. In [3] Babu et al propose a technique entirely based on motion, temporal accumulation, and spatial interpolation of motion vectors. The authors in [22] apply a clustering technique on the motion field for object segmentation. Porikli et al. [53] use the spatial continuity of motion vectors and DCT coefficients to extract the objects moving in the scene.

4.1.2 H.264

H.264 is the video standard developed by Joint video team (JVT) Instituted by both MPEG and ITU-T, which has been incorporated into

MPEG-4 as part 10 profile. H.264 is presently the most widely used and efficient compression standard adopted across the world, owing to this fact most of the present day segmentation and tracking algorithms in compressed domain concern this standard. Important distinguishing features of H.264 include half pel and quarter pel motion estimation, motion estimation at 4×4 block level, variable block size and slice level encoding. This has two consequences for the algorithms, the complexity of processing the data increases manifold while also increasing the accuracy and precision and algorithms in this domain reflect this fact. Segmentation and tracking based on H.264 have been active area of research since past 4 to 5 years immediately after the wide adoption of this standard. One of the earliest and most cited work is [76] in this paper Markov random fields are used on motion fields to discover the similarity and there by segmenting the moving object. In [38] Liu et al utilize spatio temporal similarity of motion field to achieve segmentation of the objects. Most recent work in this is of Khatoonabadi et al [31] which utilizes markov random fields to accumulate spatio temporal similarity to achieve tracking.

Video surveillance is data heavy task, amount of video gathered is astronomical as video recording is done continuously with out break. Size of video data is also huge, as video is nothing but series of images transmitted at a given frame rate. With advent of video compression techniques, video storage problem has been solved and all the surveillance videos are stored in compressed format. In present day most popular and adopted compression standard is H.264 proposed by joint video team of ITU-T and MPEG [70], which achieves a compression of almost 99 percent. Further there are specialized compression standards especially being developed for video surveillance, as H.264 is the more general compression standard. Compression standard dedicated for video surveillance would be highly useful as it contains features like Region of interest etc. As discussed in

section 4.1Most of the current computer vision algorithms which are highly accurate operate on raw pixel data. Hence video has to be decoded from its compressed format, in order to apply the algorithms. Video decoding and storing raw data is computation and bandwidth intensive process. In the context of present problem, we require a very fast real time event detection algorithm for dynamic camera reconfiguration. Moreover even if enough computational power and bandwidth are provided, such an implementation would require a heavy hardware. Forcing the implementation to be highly centralized and leaving it vulnerable to breakdowns and failures. So in order propose a distributive and low bandwidth algorithm, we propose to operate entirely in compressed domain. Such an implementation would reduce the overhead of video decoding and also reduces the bandwidth requirement by large amount.

Any computer vision algorithm relies on feature extraction for performing a relevant task. Features like HOG,HOF,STIP etc are descriptors which describe the image/video in compact possible form. Feature extraction is an expensive operation computationally and require high computational resources. Video encoder theoretically speaking is a best feature extractor, as the compressed stream can be used to reconstruct the video almost identically. In compressed video, motion information is already embedded in the motion vectors, residual and macroblock (MB) types (among other parameters) can be used to accomplish various computer vision tasks.

4.2 Motion descriptors

In order to measure and monitor the movement of the objects in the camera view, we propose a descriptor based on the disorder, or entropy, of the motion vectors of the video. The standard for video coding H.264, as most of its predecessors, achieves compression through a block-based algorithm,

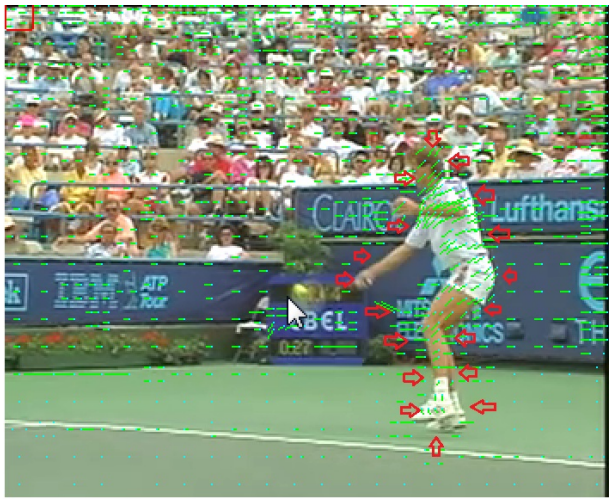


Figure 4.1: Motion vectors extracted from a frame of the standard video sequence Stefan. The red arrows highlight the regions in which the motion field exhibits strong disorder.

where blocks have variable size from 4×4 to 16×16 pixels [70]. Motion vectors are calculated for individual blocks in order to remove the temporal redundancy of the video. The distribution of the motion vectors throughout the frames gives us a very accurate insight about the analytics of the video. The distribution of these motion vectors tends to exhibit more disorder whenever there is any moving object in the video frame; this is not noticeable, for example, in presence of uniform global motion associated with camera movement, in case the camera is static with no moving object in the frame. An example is shown in Figure 4.1, and represents a frame in a video sequence in presence of a moving camera; motion vectors are overlaid on the picture. As can be seen, the motion vectors show a coherent behavior along most of the video frame, as it is expected in case of moving camera. However, the motion vectors distribution at the edges of the moving object tend to have higher disorder. We propose to utilize this aspect in order to measure the amount of information in the video frame.

4.2.1 Motion entropy measure

As mentioned in the previous section, we choose to operate in compression domain to achieve real time operational capabilities. Motion vectors are chosen as the main features for analysis, as they are immune to changes in bitrate and quantization parameters (QP) of the encoded H.264 video stream. The disorder in the motion field represents the information content in the video. In H.264, standard motion vectors are computed at 4×4 and the block size is based on the observed variance. Each motion vector consists of two components representing distances in pixels along X and Y direction from the best match found in the reference frame. In this context we represent the pixel difference along X and Y as $MV_x(i,j)$ and $MV_y(i,j)$, respectively, where i and j represent the location of a 4×4 block in the video frame. After reading the motion vectors from the H.264 stream, we group $MV_x(i,j)$ and $MV_y(i,j)$ into a 8×8 matrix, therefore each of these blocks represents the motion vectors of a region corresponding to an area of 32×32 pixels. On these super-blocks the 8×8 DCT transform is performed according to Eqs. (4.1) and (4.2). After the transform, each block describes the motion pattern of the 32×32 pixel region in X and Y directions, respectively, which becomes our motion descriptor. In the

$$MD_x^{(c,d)}(a,b) = [\frac{1}{4}\sum_{a=0}^{7}\sum_{b=0}^{7} MV_x[(c-1)*8+a,(d-1)+b]*\cos\frac{(2a+1)*\pi}{16}*\cos\frac{(2b+1)*\pi}{16}] \tag{4.1}$$

$$MD_y^{(c,d)}(a,b) = [\frac{1}{4}\sum_{a=0}^{7}\sum_{b=0}^{7} MV_y[(c-1)*8+a,(d-1)+b]*\cos\frac{(2a+1)*\pi}{16}*\cos\frac{(2b+1)*\pi}{16}] \tag{4.2}$$

equation (c,d) represent the block location of 32×32 in the frame, (a,b) represent the location of the 4×4 block within the 32×32 block.

The choice for a block size of 32×32 pixels is made to ensure minimum variability of motion vectors which occurs in the case of 16×16 mode in H.264 bit stream. The obtained result is a 2D DCT transform of 8×8 blocks of motion vectors. Inferring from the properties of the DCT transform we can notice that DC values $MD_x^{(c,d)}(0,0)$, $MD_y^{(c,d)}(0,0)$ represent the localized global motion and AC coefficients represent the variation in motion vectors. The frequency of variation increases as we move towards the bottom-right corner. We propose to accumulate the AC coefficients to arrive at a measure of motion disorder. However, higher frequencies represent more disorder in comparison to the lower ones, hence the accumulation has to be done in a weighted manner. This is exactly the opposite of what happens in image and video compressions, where lower frequencies are usually more important. Therefore, we calculate the entropy values along X and Y as E_X and E_Y from the equations Eq. (4.3) and Eq. 4.4, respectively. The unified entropy measure is given by Eq. (4.5)

$$E_X(c,d) = \sum_{a=0}^{7}\sum_{b=0}^{7} MD_x^{(c,d)}(a,b) * [2^{a-8} + 2^{b-8}] \tag{4.3}$$

$$E_Y(c,d) = \sum_{a=0}^{7}\sum_{b=0}^{7} MD_y^{(c,d)}(a,b) * [2^{a-8} + 2^{b-8}] \tag{4.4}$$

$$E_U(c,d) = \sqrt{E_X^2 + E_Y^2} \tag{4.5}$$

The aggregated entropy gives us a generalized measure of information present in the video frame. This measure is independent of the number of objects and patterns of motion in the scene. A frame level aggregated metric is defined in (4.6)also defined which gives overall information in the frame.

$$G_{XY} = \sum_{c=0}^{\frac{Width}{32}} \sum_{d=0}^{\frac{Height}{32}} [E_X(c,d) + E_Y(c,d)] \tag{4.6}$$

4.3 Object detection and segmentation

4.3.1 Algorithm

Moving object that is to be segmented is characterized by a high spatial motion entropy and high correlation in terms of temporal measure of order. Conversely all the areas belonging to the background will be characterized by low correlation and low spatial motion entropy. The combination can be expressed as a function for each of the 4×4 block in current frame, for both the components along X and Y given by Eqs. 4.7 and 4.8.

$$M_X(m,n) = E_X\left(\frac{m}{8}, \frac{n}{8}\right) * O_X(m,n) \tag{4.7}$$

$$M_Y(m,n) = E_Y\left(\frac{m}{8}, \frac{n}{8}\right) * O_Y(m,n) \tag{4.8}$$

Similarly, the probability of a given block belonging to the background region in a current frame, is modeled through two independent Gaussian distributions centered around zero and given by Eqs. 4.9 and 4.10.

$$G_X(m,n) = \frac{1}{\sqrt{2\pi}\sigma_X} * exp\left[-\frac{M_X(m,n)^2}{\sigma_X^2}\right] \tag{4.9}$$

$$G_Y(m,n) = \frac{1}{\sqrt{2\pi}\sigma_Y} * exp\left[-\frac{M_Y(m,n)^2}{\sigma_Y^2}\right] \tag{4.10}$$

σ_X and σ_Y are the variances of the Gaussians, which have to be determined experimentally. The probability of the block belonging to the moving objects can be defined in Eqs. (4.11) and (4.12)

$$P\left(Obj \mid O_X\right) = 1 - G_X(m,n) \tag{4.11}$$

$$P\left(Obj \mid O_Y\right) = 1 - G_Y(m,n) \tag{4.12}$$

Finally the probability of the block belonging to moving objects both with respect to X and Y is given by Eq.(4.13) as both of them are mutually exclusive subsets.

$$P\left(Obj \mid O_X, O_Y\right) = P\left(Obj \mid O_X\right) * P\left(Obj \mid O_Y\right) \tag{4.13}$$

In the segmentation map, each 4×4 block is represented by the pixel intensity proportional to its probability. In this way blocks belonging to the background are not marked and the segmentation is achieved.

4.3.2 Tuning the variance

As described in the previous section, variance of X and Y is highly dependent on the characteristics of the video, requiring to find a suitable value for it. Theoretically the variance of the two Gaussians represents the degree of freedom, for variation of O_X and O_Y. High value of variance will result in higher precision of segmentation and low values will ensure high recall with relatively low precision. In our case we set the value of precision to maximize the F-measure which is combination of both precision and recall. Further, the variance of both the components is highly localized in context. Hence tuning of variance is carried out by expressing it as function of respective mean. Objective of tuning is to maximize the F-measure.

$$\sigma_X^2 = \left[mean[O_X] * M\right]^2 \tag{4.14}$$

$$\sigma_Y^2 = \Big[mean[O_Y] * M\Big]^2 \tag{4.15}$$

In order to demonstrate the effect of variance on the F-measure, in this section we plot the variation of precision, recall and F-measure for the sequence *Stefan*, for different values of variance in Fig.4.2. Variance is expressed as mean square multiplied by a scalar factor.

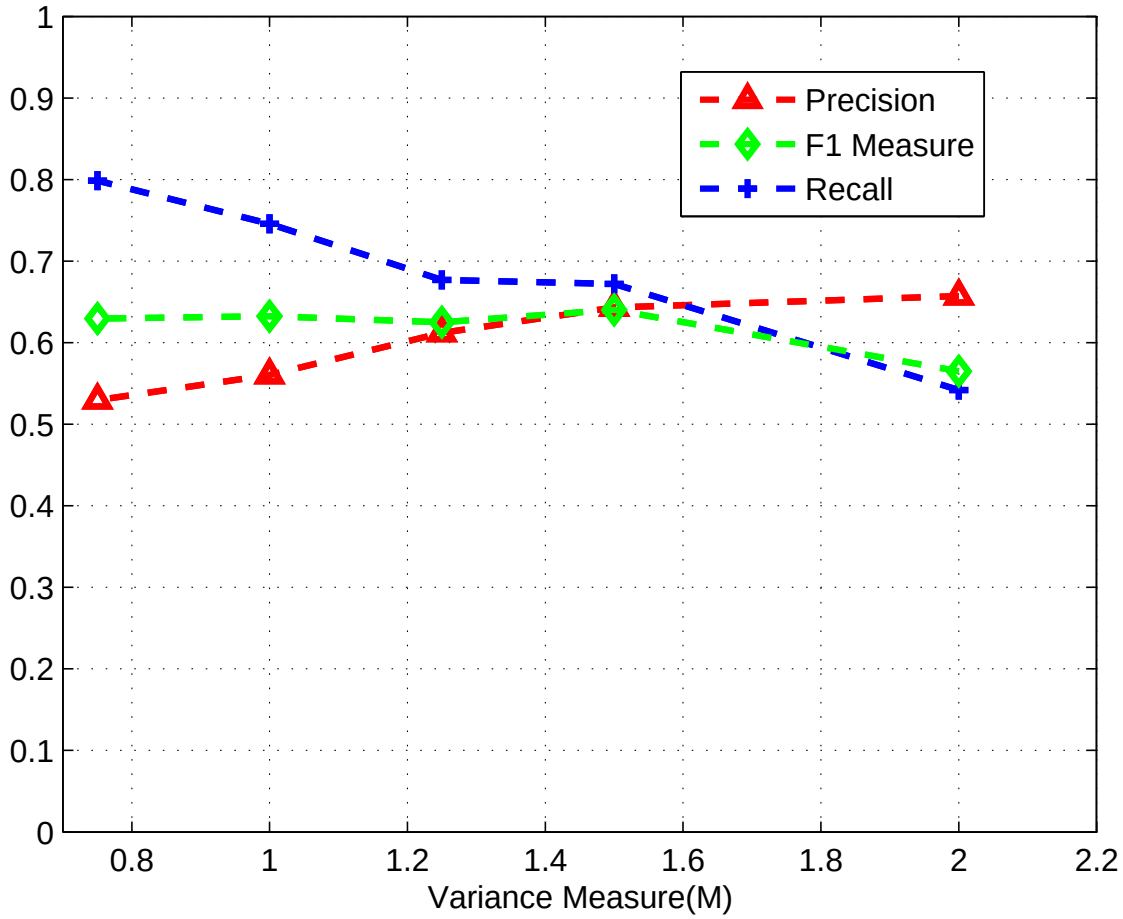


Figure 4.2: X axis represents variation of tuning factor M, mean of O_X and O_Y is multiplied by M and then squared to get σ_X^2 and σ_Y^2 respectively

As we can see from the plot as the variance increases the precision also

increases, while recall decreases. However, the F-measure value is more or less stable within certain range. This plot has been carried out for the first 90 frames of the sequence Stefan, which is rather challenging due to high camera movement, pan and tilt.

4.4 Fall detection

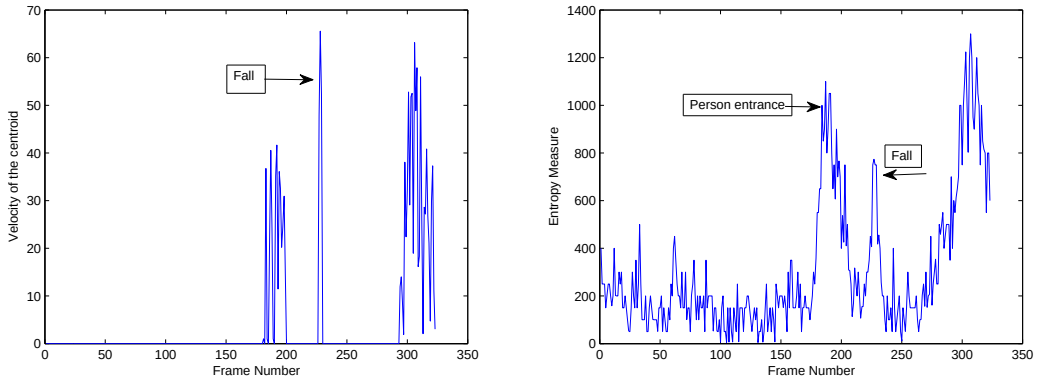
Motion entropy measure is a proposed in section 4.2.1 is a highly flexible and generic metric for video event description. With small modifications it can be easily adapted for variety of applications. In this section we present its utility in fall detection. Fall detection is one of the selected basis for camera reconfiguration in the current study. Reconfiguration is aimed at providing the better view of fallen person. Such a system would greatly improve the accuracy and timely response in emergency health care. This is especially relevant in developed world which increasingly houses growing percentage of aged population.

4.4.1 Proposed method

In the section 4.2.1 we defined motion descriptors and in previous section their usage in detection of moving objects and segmentation. We approach fall detection in similar manner. Fall detection can be described as a sudden event which causes rapid variation of video features in temporal direction. In line with that observation, the unified entropy measure defined also has large value and also high variation across the frames during the occurrence of the fall.

In order to identify potential candidate frames for the occurrence of fall, we discard the frames which have lower entropy. Lower entropy frames typically do not consist of any movement of objects, hence the likelihood of occurrence of the fall in these frames is almost negligible. After selecting

the frames with higher entropy with a cut off, which is specific to camera orientation and illumination conditions, we further analyse these frames for the detection of the fall. Similarly Figure 5.6 (a) shows the movement of centroid of the person per frame. As we can see during the fall velocity of centroid dramatically increases and then becomes zero.



(a) (b)

Figure 4.3: (a) Velocity of the centroid of the person per frame. (b) Variation of entropy and the events as marked using ground truth.

Another aspect of fall is a sudden change in location of the centroid over a very short number of frames. Since, whenever there is an occurrence of a fall of a standing person, the orientation of the person changes dramatically, we propose to utilize this aspect as well.

In order to further refine the accuracy of the prediction we apply another observation which is very characteristic of the fall. Whenever the fall occurs the person who has fallen down will become almost motion less, there by decreasing the entropy measure defined in earlier section. Figure 4.3 (b) illustrates the variation of entropy. We can notice that any sudden event results in spurt of entropy. However in case of fall there is dramatic decrease in entropy before a spike and later a fall.

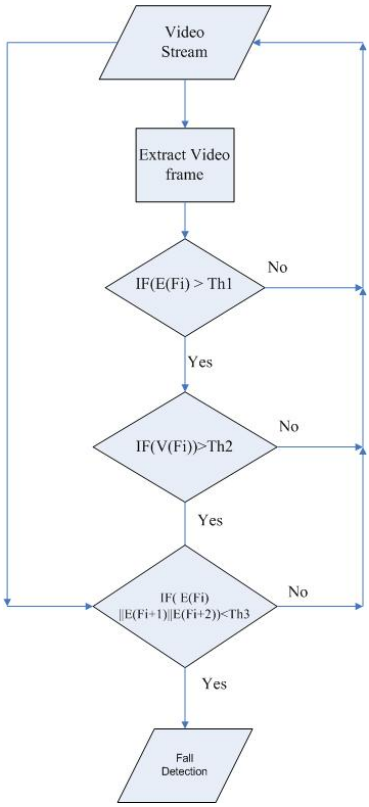


Figure 4.4: Flow chart of the proposed algorithm.

4.4.2 Algorithm

Let the frame at the time instant i for a given video stream from a surveillance camera be F_i, then entropy measure of that frame is given by $E(F_i)$ and the centroid of the segmented person as a pixel location in a video is given by $C(F_i)$ and the distance travelled by the centroid or velocity per frame is given by:

$$V(F_i) = Euclidean(C(F_i), C(F_{i-1})) \tag{4.16}$$

After defining these terms the proposed algorithm is given flowchart shown in Figure 4.4. As mentioned in earlier section, we check for the high variance condition of entropy and also velocity of moving object. After that we check for sudden drop of entropy in neighbourhood to check for the fall. Thresholds $Th1$, $Th2$ and $Th3$ are dependent on mean and variance of en-

tropy and velocity. Main purpose of these thresholds is to detect the peaks that occur for the entropy measure and also to detect the peak changes in velocity of the centroid. Ideally any values which fall above mean of the entropy and velocity should be considered, however background noise also contributes significantly to motion entropy and is largely dependent on deployed environment and is best learned in a given camera scenario.

4.5 Results

4.5.1 Video segmentation

In this section we present the results of video segmentation algorithm proposed in section 4.3. The algorithm proposed uses motion descriptors which form the basis for reconfiguration. Results reinforce the effectiveness of the motion descriptors in computer vision tasks.

In order to demonstrate the capability of the algorithm in segmenting moving objects, we report here the qualitative evaluation on four standard video sequences used in video compression, which exhibit different types of motions in Figs.4.6and 4.7. All the sequences are encoded using the JM 18.1 encoder using following configuration: intra period is set to 30, motion search range is $[-32, 32]$, 3 reference frames are used, and full block search is used for motion estimation. All sequences have a resolution of 352×288. We have also considered sequences from the iLids dataset [28], which represent a standard surveillance scenario. These videos instead have a resolution of 720×576. Samples of segmentation of iLids sequences are shown in Fig.4.8

The scalar factor M is kept unaltered for all sequences and set to 0.25, except for *Stefan* and *Mobile*. For *Stefan*, owing to high camera motion we set $M = 1.5$, while for the *Mobile* sequence, which shows a highly dispersive motion with low values, the best performance was obtained by setting

Table 4.1: Comparison of the results obtained with the proposed method against the reference approach.

Sequence		Precision	Recall	F1
Mobile	Proposed	0.88	0.65	0.77
	[43]	0.96	0.60	0.74
Stefan	Proposed	0.65	0.67	0.65
	[43]	0.39	0.97	0.49
Table Tennis	Proposed	0.74	0.77	0.76
	[43]	0.91	0.67	0.75

$M = 0.15$. However, considering that the sequences exhibit completely different scenarios, an ad-hoc tuning is necessary. In order to evaluate the obtained results quantitatively we have compared the algorithm with a recent video segmentation algorithm in H.264 domain [43] and the obtained precision, recall and F-measure are reported in Table 1 in the columns 3,4,5 respectively. As can be noticed, the algorithm outperforms the reference method in video segmentation in the compressed domain. Figure 4.5 shows frame to frame comparision of F1-measure between the proposed and reference methods for the *stefan* sequence. As we can see, proposed method performs better than the reference methods in terms of variance. However there is a dip in F1 measure from frame 15-20 as there is very little or zero motion during that period.

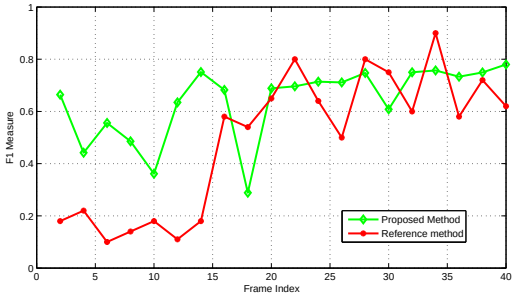


Figure 4.5: frame by frame comparison of proposed algorithm with reference method [43]
.

Computational complexity The proposed algorithm operates completely in the compressed domain, exclusively relying on motion vectors. Major computations involve 8×8 DCT transforms spatially and 8 point DCT transforms temporally. Let the height and width of the frame be W and H, then the total number of 32×32 pixel blocks becomes, considering both directions X and Y, $2 * \frac{W}{32} * \frac{H}{32}$. If the fast DCT proposed by Chen et al. [10] is used, the total number of computations for spatial motion descriptors SMD is given in Eq. (4.17)

$$C = 2 * \frac{W}{32} * \frac{H}{32} * 64 * log_{10}8 \tag{4.17}$$

Similarly, the computational complexity for temporal motion descriptors (TMD) is given by Eq. (4.18)

$$C_{TMD} = 2 * \frac{W}{4} * \frac{H}{4} * 8 * log_{10}8 \tag{4.18}$$

The method is implemented in C on a standard desktop computer with Intel processor clocking at 3.0 GHz. The time taken for processing one frame, is about 15 milliseconds on average for a CIF sequence, comparing to the algorithm proposed in [43], which takes about 125 milliseconds per frame. Another important fact is integration of the proposed method into H.264 encoder is straight forward, hence can be deployed directly in the surveillance camera hardware.

4.5.2 Fall detection

This section presents the result of fall detection in terms of accuracy and comparison with state of the art. Subsequently we also present the results of reconfiguration achieved using PTZ cameras.

Figure 4.6: Results obtained for a set of standard benchmarking sequences 1.

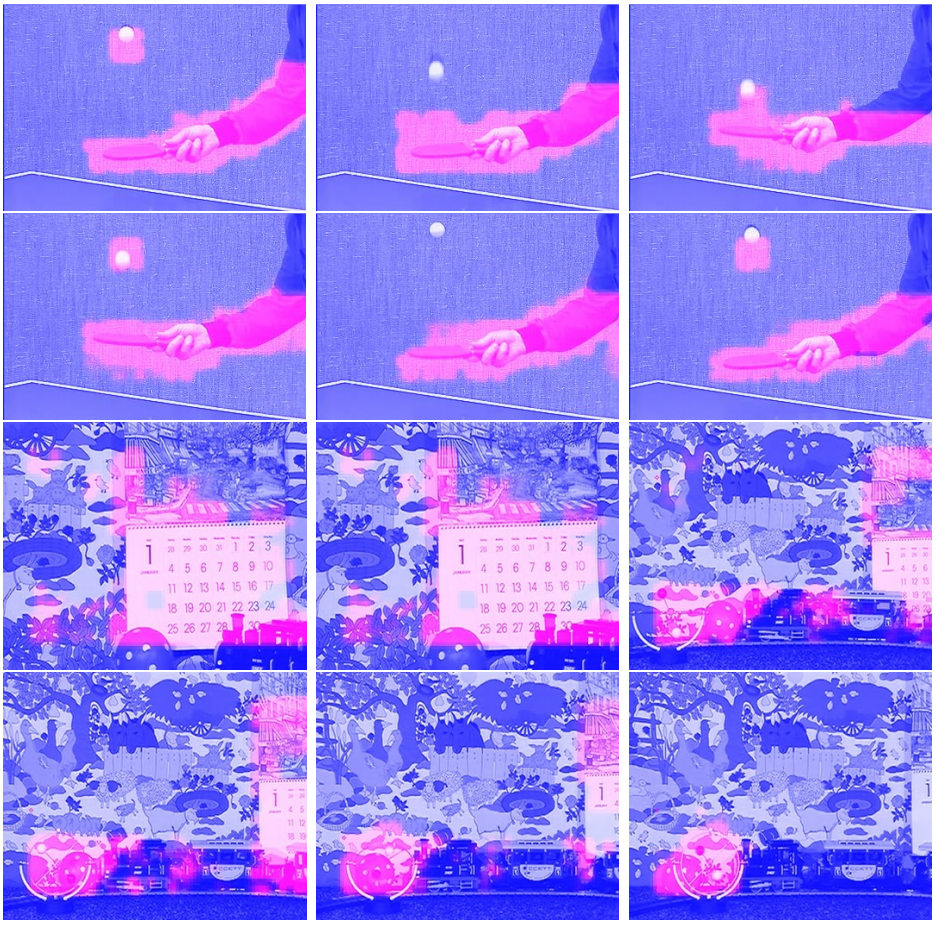


Figure 4.7: Results obtained for a set of standard benchmarking sequences 2.

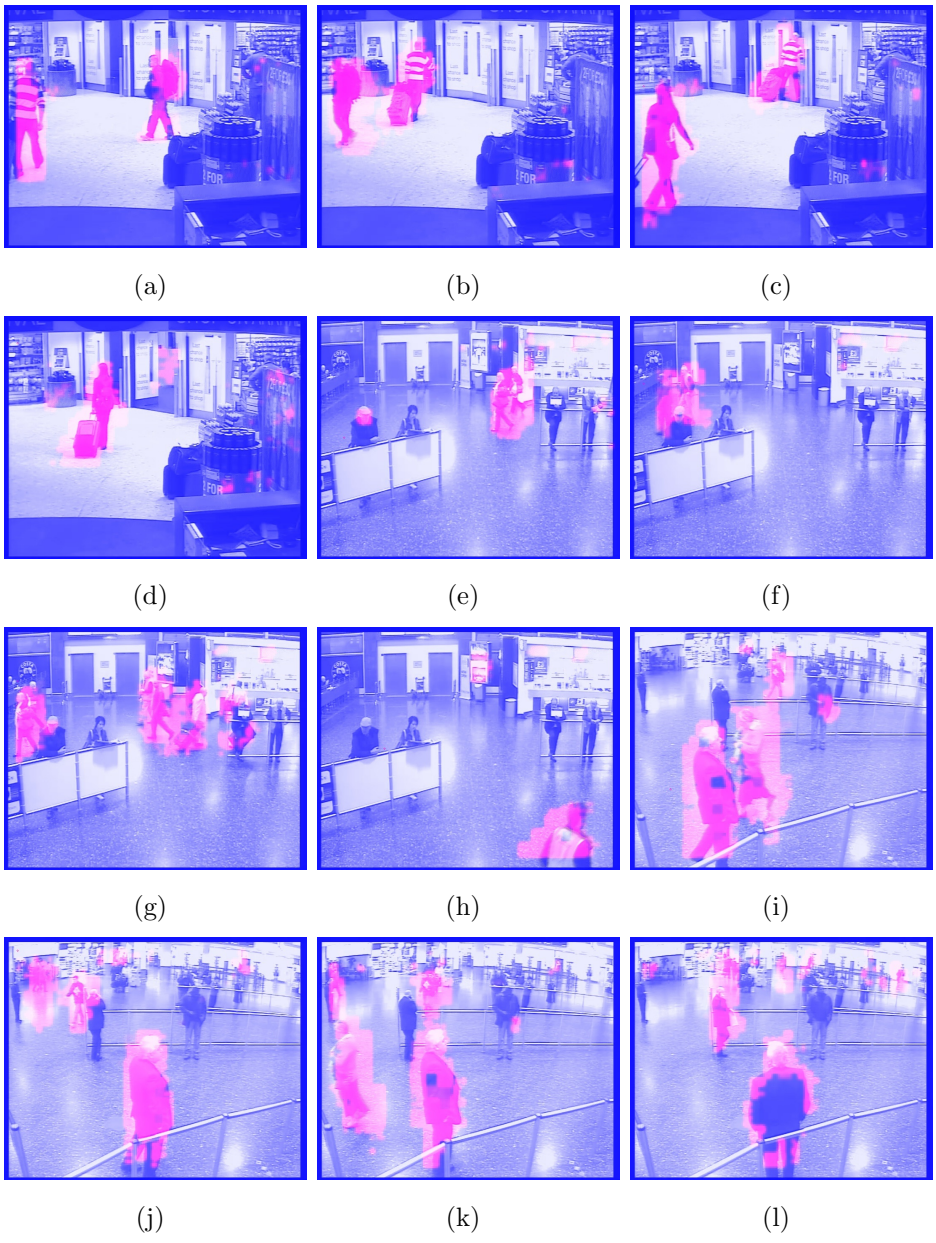


Figure 4.8: Samples taken from the i-Lids dataset.

Detection

Since the algorithm operates in the compressed domain, we had to convert all the videos in the dataset [2] into the H.264 format using the JM H.264 reference encoder [26], at the frame rate of 25 frames per second. The thresholds necessary for a proper operation of the algorithm are learned for each camera and are maintained constant for that particular camera for all scenarios. Fall is defined as an event lasting 5-10 seconds, starting from the momentary stop by the subject just before the fall and ending with a motion less layover of the subject. The total number of correct fall detections, as compared to the ground truth, are deemed as true positives (TP), while false detections are termed as false positives (FP). Finally, true falls which have been skipped by the detector are termed as false negatives (FN). The results obtained for the video dataset are given in Table 4.2 in terms of Precision, Recall and F-Measure. A comparison with respect to the state of the art techniques is provided in 4.3. As can be seen, the fall detection algorithm performs reasonably well especially given the fact that it operates in real time. The algorithm fails to detect the falls, when the subject is very far away from the camera and subsequently the motion entropy generated by the subject is very low. In such scenario noise becomes dominant thereby causing false detections. Another scenario where the algorithm fails is in case of actions, which correspond to bending down on the floor etc. However, since we also took into consideration the momentary fall entropy, just after the fall most of such false detections have been resolved.

Table 4.2: Performance of the algorithm on the dataset [2].

Precision	Recall	F-Measure
0.89	0.86	0.88

Table 4.3: Comparison to the state of the art approaches described in [23].

	Our method	K-NN	C4.5	SVM	Bayes	Feng et. all
Sensitivity	0.86	0.75	0.85	0.95	0.80	0.98

Comparison

Our algorithm completely operates in the compressed domain. Hence it has the advantage of being very light in terms of computational and memory requirements. Nevertheless it compares very well with the other pixel domain state of the art fall detection methods as we can see from the table 4.3. Our method also provides a significant improvement with respect to other compressed domain methods like [36]. Most of these methods rely on the segmentation of moving object and the trajectory of its centroid, and also include other features like velocity of centroid. Present algorithm also uses these aspects, but it turns out to be more robust as it also exploits the motion disorder as one of the factors to determine fall detection. Furthermore, the compressed domain method presented in [36] uses AC and DC coefficients along with motion vectors to achieve object segmentation, which are heavily dependent on the quantization parameter used for encoding the video bit stream. The proposed method, instead is entirely based on motion vectors, which are independent with respect to changes in QP. In terms of complexity our solution offers the lowest complexity of all compressed domain methods as it operates at the level of 32×32 blocks, and the number of operations required for processing one frame are 5.2K, 16K, 48K, 106K computations for CIF, VGA, HD, full HD resolutions, respectively.

Chapter 5

Reconfiguration

In this chapter we discuss the types of reconfiguration and the proposed approach to achieve them. Reconfiguration is broadly divided into two categories, static and dynamic. Static reconfiguration is triggered one time event like camera failures and environment change. Dynamic reconfiguration on the other hand takes place continuously and automatically with respect to the video events that take place in the environment.

5.1 State of the art

Research in camera reconfiguration is in a nascent stage, Micheloni et al. summarized the current state of the research in [44]. In general, camera reconfiguration is performed with respect to a specific task. One of the earliest works to consider PTZ cameras is [49]. In this work PTZ cameras were specifically used for tracking. In another instance Quaritsch et al. [54] adopt reconfiguration to achieve better tracking over multiple cameras. Scotti et al. [61] utilize PTZ cameras along with omnidirectional cameras in order to achieve tracking of objects at higher resolution. Another work combining omnidirectional and PTZ cameras for tracking is [9]. Here the authors approach the problem analyzing the spatial correlation, in order to map the targets across two types of cameras. Similarly, Picarelli et al. [52]

apply reconfiguration to avoid occlusions, which may occur in presence of changes in the environment. Karuppiah et al. [29] propose a smart camera reconfiguration algorithm exploiting the a priori knowledge of floor plans, and drive the reconfiguration process based on the changes that take place over time. Another work, which specifically deals with the reconfiguration of PTZ cameras is presented in [51], but in this case the model of the camera is fixed and does not well apply to the case of PTZ cameras. One of the few works to approach the problem of PTZ camera reconfiguration in a general sense rather than in an application specific scenario, is [64]. The authors propose a decentralized algorithm for reconfiguration based on game theory. Another similar work is presented in [15].

We have observed that a common deficiency of all the above algorithms is that they often do not consider zoom as a reconfigurable parameter. It is also worth noting that another important limitation in the state of the art is in the fact that most algorithms address the issue of reconfiguration as a separate problem from change detection. In fact, ideally, a reconfiguration algorithm should also include a methodology for change detection to trigger the reconfiguration when needed. This is one of the major aspects that we include in the current study, by proposing a low-complexity reconfiguration algorithm, which also includes the capability of detecting the changes in the environment. Another very important fact, which is often overlooked in smart camera systems, is the overall complexity of the system.

5.2 Static reconfiguration

In surveillance scenarios where the video feed is continuous with cameras continually relaying it. There is high chance of malfunction in such scenario. In fact it is regular feature where in one of the cameras in environment malfunctions and leads to surveillance blind spot. Such situations

may lead to serious consequences in terms evidence collection crime prevention. There also a high chance of perpetrator disabling the camera facing the crime scene. In this context reconfiguration of the rest of cameras to cover the blind spot is highly desirable and effective temporary measure. Similarly there may be additional scenarios where in the change environment may lead to occlusion of one of the deployed cameras. Camera reconfiguration also comes in handy in such situations. Although the cameras are fixed, advent of PTZ cameras has opened up has made reconfiguration a necessary part of a smart and realistic camera network.

Since the reconfiguration is static, the previous camera models defined in chapter 2 can be used to carry out simulation in virtual domain. Best possible configuration for the changed circumstances is achieved by carrying out simulation in virtual domain. Particle swarm optimization is used to arrive at a best solution with the changes in environment or cameras being replicated in the virtual domain. However only changeable parameters in this simulation are *pan*,*tilt* and *zoom*, while the positions (x, y, z) of the cameras are fixed. Hence static reconfiguration is an extension of camera planning problem with reduced parameter space. It is achieved by optimizing the camera parameters while keeping (x, y, z) constant. Validation of the proposed models with respect to static reconfiguration has been presented in chapter 2.

5.3 Dynamic reconfiguration

Dynamic reconfiguration involves continuously and collectively changing camera parameters of all the cameras in network, in order to get a best possible view of all the moving targets in the environment. Such an arrangement would greatly increase the effectiveness of the surveillance systems. It will also lead to cost reduction by reducing the required number of

sensors for a given environment. Better view of the moving targets would improve the accuracy and effectiveness of the computer vision algorithms like face detection, tracking etc. However another important factor that has to be taken into account is the overall coverage of the environment. While performing these changes of camera parameters, the overall minimum coverage should be maintained.

5.3.1 Area metric

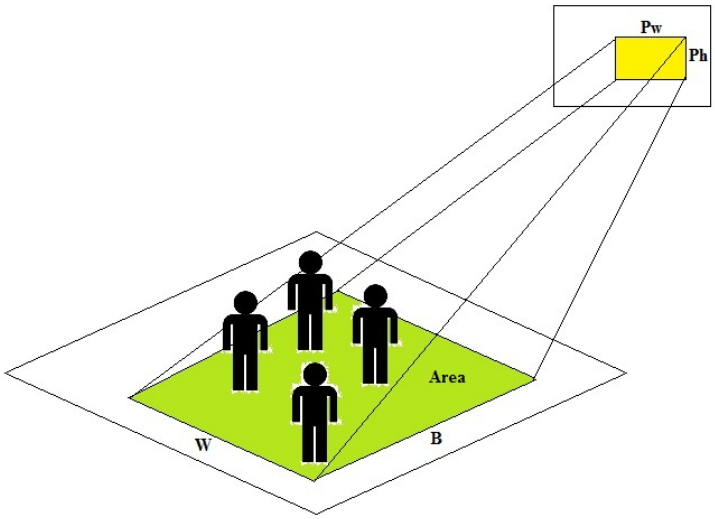


Figure 5.1: Segmented set of objects projected onto the environment using the camera model and current parameters.

As mentioned section 4.2.1 motion entropy metric defined in (4.6) forms the basis for reconfiguration. However the proposed metric does not quantify information in the context of whole object i.e metric does not care whether the whole object is visible or not. It simply gives the extent of information in the video frame, such information may not be useful. Moreover, if the target is close to the camera, it is likely that it will cover a large portion of the image plane, thus decreasing the automatic observabil-

ity of the event, mainly due to over segmentation, and making the image less suitable for feature extraction and analysis. Furthermore, the entropy metrics obtained for such situations are considerably high, leading to improper handling of camera reconfiguration. In order avoid such situations we propose to combine the motion entropy metric with the information about the area occupied by the moving object (or set of objects), in order to obtain a more balanced metric as a basis for reconfiguration. The objects observed by a camera are defined as the largest bounding box in the video frame that include all the detected and segmented objects using the method proposed in the previous section. This rectangle is then projected onto the real environment using the camera model and its current parameters. After the area has been obtained, it is normalized and combined with the entropy metric, which is used for camera reconfiguration in target mode (see Eq. (5.1)).

$$T_A = W * B \tag{5.1}$$

W and B in the equation are calculated as shown in Figure. 5.1

5.3.2 Camera network architecture

In order to efficiently handle and monitor the given surveillance scenario, we propose to operate the cameras in two modes, namely *target* and *global* mode. During operation, cameras in the network switch between the two modes based on the information coverage metrics. The detailed description of each mode is given in following subsection.

Camera modes

Global Mode The primary function of the camera in this mode is to ensure maximum coverage and visibility on the scene. All the cameras in this

mode are utilized to maintain at least the minimum amount of coverage as specified by the user. The total number of cameras in this mode can never be zero. These cameras also perform the role of scouting for any moving object detection. A given camera can switch from global mode to target mode only if certain conditions specified by the algorithm are met. One of the conditions to be met is that the coverage provided by the residual cameras in the network is greater than or equal to global coverage requirements.

Target Mode In this mode the primary function of the camera is to extract as much information as possible of to the object or set of objects, to which it has been assigned. While the objects are moving,target mode camera changes its PTZ parameters to attain the best possible view of the targets. The camera will switch back to global mode once the target moves out of range of the camera field of view; at this point the global camera in the network, which has the highest information metric for that target will then switch to *target mode* to continue monitoring the moving object.

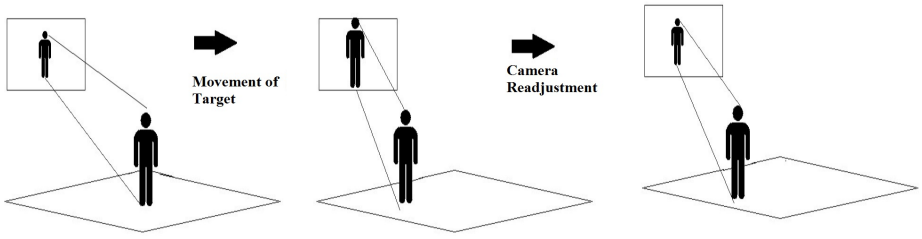


Figure 5.2: Target mode operation of the camera.

Monitoring is achieved by continuously adjusting the pan, tilt, and zoom of the camera. This process continues until the camera is assigned either to another object or switches back to global mode. The principle criterion for reconfiguration is to maintain the midpoint of the defined rectangle along the camera axis (through pan and tilt). Another objective is to ensure

that the height of the rectangle occupies around 60-80 percent of the video frame height (through zooming).

Since the rectangle may consist of a set of objects potentially showing rapid transitions in size, the reconfiguration is carried out according to the average width height of the rectangle over a time window. This ensures that the camera does not track random motion in short bursts. Figure. 5.2

5.3.3 Camera network and operation

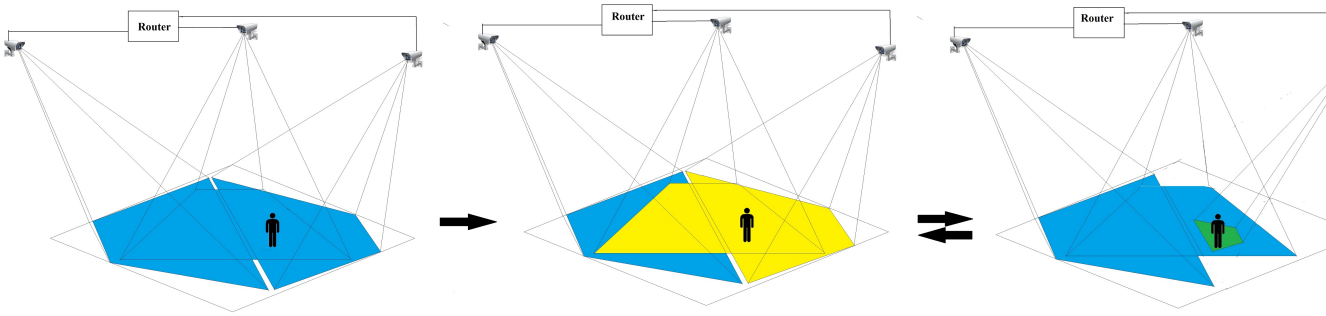


Figure 5.3: Various stages in transition of cameras from global to target mode.

Let us assume there are N cameras in the camera network that is to be deployed in the given environment. Each camera acts as an independent node in a mesh network and can communicate with any other camera using a predefined protocol. Initially the best configurations for various combinations of cameras varying from K to N are calculated according to the positioning algorithm proposed in [57]. The total number of configurations is given by:

$$N_c = {}^{N}C_K + {}^{N}C_{K+1} + {}^{N}C_{K+2} \ldots\ldots\ldots\ldots + {}^{N}C_N \tag{5.2}$$

where K is the minimum number of cameras, which are to be maintained in global mode, either owing to visual coverage requirements or

user specification. All these configurations are stored in each of the camera node. When the system is turned on all the cameras initially set into global mode. Now, let there an be a moving object i detected by a camera; its physical world location is given by

$$O_{X,Y,Z} = P(x, y) * T(\theta, \phi, f) \tag{5.3}$$

where (X, Y, Z) represents the physical world location of object (x, y) is the pixel location as seen by the camera. The transformation $T(\theta, \phi, f)$ is based on pan, tilt, and zoom of the camera at that instant and the pinhole camera model presented in [57]. If there are more objects in view, the largest rectangle which encompasses all the objects is taken. From the spread of objects in the environment, the area metric in Eq. 5.1 is calculated and combined with the unified entropy as defined by Eq. (4.6) according to:

$$M_i = 1 - \exp[-T_A^i * G_{XY}] \tag{5.4}$$

Successively, the camera transmits the object location and the combined metric to all other cameras. In order to remove noise and also due to the moving nature of the objects, location and entropy are calculated as a moving average over a period of time. The transmission interval is application-dependent. Each camera then compares the combined metric M_i of the other cameras observing the same object or set of objects. The camera with the highest entropy will switch to *target mode* and will be assigned to that particular target. The switching only happens if the total number of global mode cameras in the network is greater than K. After switching, the remaining cameras in *global mode* switch to the corresponding configuration referring to that particular combination. An example is shown in Figure. 5.3. As can be seen from the figure, all three cameras are initially in global mode (represented in blue); once the objects appear

in the scene, cameras are classified into groups based on the visibility of the objects. In the second stage, the combined measure defined earlier is compared across two cameras with respect to object visibility. This transition stage is represented in yellow. Finally, in the third stage, cameras with high metric are assigned to respective targets, while the others revert to global mode. The transition stage is repeated at equal time intervals in order to verify the assignments. The technical aspects of this transition are shown in Algorithm. 3. While assigning the targets algorithm assumes equal importance for all the targets, irrespective of their location unless specified by the user. The Algorithm is repeated at the end of the time interval as set by the user.

5.4 Validation

In this section set up for the reconfiguration and subsequent reconfiguration results are presented. In principle reconfiguration is based on the target mode described in section 5.3.2.

5.4.1 Fall detection

Setup

Reconfiguration of the camera is triggered by the fall detection algorithm mentioned in the section 4.4. The basis for reconfiguration is the fallen person. The main aim of the reconfiguration of the camera is to get the best possible view of the subject. In order to do so, we adjust the camera parameters in such a manner that the person to be observed falls at the centre of the image plane. Further precaution is also taken so that the person does not fill the entire image plane of the camera. This can be achieved in most PTZ cameras by specifying the particular area in a video frame, by using the available CGI (Commond Gateway Interface) commands. In this

scenario, the segmented object is selected as the area of interest. Cameras automatically adjust alignment at the midpoint of the area specified.

After the reconfiguration is complete in order to make the person fully visible, the new parameters of the camera are set using the subject segmentation information. This helps understanding the reason of fall and further monitoring of the person after the fall, that is especially relevant for elderly people living alone and monitored for their care.

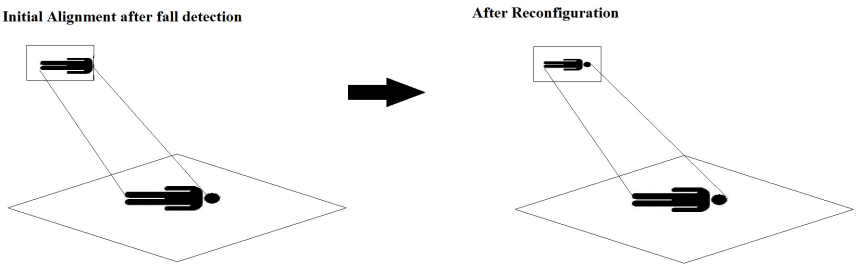


Figure 5.4: Reconfiguration of the camera carried out to guarantee full visibility of the subject of interest.

To show the reconfiguration capability, we deployed a set up in a real environment and observe its performance during the occurrence of fall. To this extent we used two cameras "Sony SNC-EP521 indoor", day/night, with PTZ. These IP cameras are equipped with a 36x optical zoom allowing operators to cover large, open areas and zoom in for detailed close-up shots. Panning can span from 0 to 340 degrees, with max 105 degrees tilt, and their configuration can change using built in network commands. The cameras have been installed in our Department facility, and falling events have been recorded thanks to the collaboration of volunteers.

Result

the video stream obtained from the camera has a resolution of 720×576 pixels and a frame rate of 25 frames per second. The H.264 bit stream obtained from the camera is encoded in the baseline profile. In order to access the Network Abstraction Layer (NAL) packets from the camera we have used the functions available in the *ffmpeg* library [66]. Fall detection and moving object segmentation are implemented using the motion vectors extracted from the H.264 (JM 18.6 version) decoder [26]. In order to control the camera automatically the *curl* library functions [65] are adopted. The whole set up is implemented on an Intel i5 processor, 3.10 GHz.
Fall detection and subsequent reconfiguration is shown in Figure 5.5. As we can see from the images, fall of the person occurs towards the end of the image in one of the frames. However, camera instantly reconfigures to bring back the view of the fallen person. This shows that the algorithm works in real time and is robust enough to work in tricky illumination conditions.

5.4.2 Entropy based reconfiguration

Final section deals with the generic reconfiguration algorithm proposed. Algorithm is proposed based on motion entropy. Ultimate aim of this reconfiguration algorithm as mentioned in chapter 5 is to concentrate all the video sensors on the area which has highest information. However there is a restriction that the overall, coverage of the environment should be maintained.

Implementation and testing scenario

In order to test the algorithm, we have deployed a camera network composed of two PTZ cameras. The cameras we have selected are Sony EP521,

because of their wide optical zoom (36x). Cameras have been deployed in the university wired network and accessed via IP address. The video stream obtained from the camera has a resolution of 720×576 pixels and a frame rate of 25 frames per second. The H.264 bit stream obtained from the camera is encoded in the baseline profile. In order to access the NAL packets from the camera we have used the functions available in the *ffmpeg* library [66]. Motion field entropy calculation and object segmentation are accomplished using the motion vectors extracted from the H.264 (JM 18.6 version) [26] decoder. In order to control the camera automatically the *curl* library functions [65] are adopted. The whole set up is implemented on an Intel i5 processor, 3.10 GHz. In terms of complexity algorithm requires 5.2K, 16K, 48K, 106K computations per frame for CIF, VGA, HD, full HD resolutions, respectively, which is negligible when compared against the video encoder complexity. Hence the proposed algorithm can be seamlessly deployed in a video encoder embedded in the camera. Cameras are deployed in a long corridor of about 15m in length and 3m wide in the university building. Cameras are deployed in the best positions in accordance with the algorithm proposed in [57] and this constitutes global mode configuration of the camera. Evaluation is performed by observing the change in configuration of the cameras with respect to movement of the people in the corridor. Observed changes in configuration are compared with expected behaviour as defined by the proposed algorithm. In this particular deployment, minimum number of global cameras is fixed at one and target camera tracks the objects in such a manner that they occupy 70 percent of the video frame height.

Evaluation of entropy metric

In order to evaluate the metric based on the motion field disorder, we plot the variation of the entropy for both cameras, in presence of motion

of people across the deployed corridor from one end to another end (see Figure. 5.6). As can be seen, the entropy metric oscillates in the range 20-40 with a random nature, whenever there is no motion. However, as a person moves by the camera, a noticeable increase in entropy is visible. For example, as the person moves by camera 1 (marked in Red) there is sudden increase in entropy and camera 1 switches to target mode. After the target has moved away from camera, entropy gradually reduces, and switches back to global mode. We can notice a similar behaviour from camera 2 (in blue), which takes over for camera one, when the target comes closer, thus switching to *target mode*.

5.4.3 Algorithm evaluation

In order to evaluate the algorithm we observe the camera behaviour in light of a predefined movement of a moving object. As a part of this the person walks from one end of the corridor to the other and then back to the initial starting point. The path followed by people and the point at which the reconfigurations happen are shown in Figure. 5.8

We observe the state transitions during these process. Initially, in Figure. 5.7 the global mode of both cameras is presented. The series of images where camera 1 transits from *global* mode and then follows the moving object in *target* mode, are shown in Figure. 5.5 A the target moves out of range of the camera, it reconfigures itself back to global mode. Similarly also camera 2 follows the same routine as the target approaches it and moves out of range (Figure. 5.10). On the whole, the setup performs satisfactorily with respect to reconfiguration. The only limitation we have experienced is in the robustness to rapid changes in the reconfiguration, since the total time required for repositioning the sensors is about 3s due to the network delay issues. This could be overcome by deploying the algorithm in the H.264 encoder embedded in the camera.

input : Object locations O_L across Camera network
input : Combined Camera Measures for all cameras C_{mode}
input : Minimum Global Cameras GC_{min}
output: Camera modes C_{mode}

CM_C ; % Combined Camera Metric using Area T_A and E_U
O_L ; % Object Locations using pinhole Camera Model O_L
N_{obj} ; % Number of object sets across camera network N_{obj}
$CS_{obj} = \phi$; % Set of Cameras per object
C_{mode} = Global $\forall$ $N_{cameras}$; % All cameras are initially in global mode
T_C =0 ; % Cameras in Target Mode
for $i \leftarrow 1$ **to** N_{obj} **do**
  for $j \leftarrow 1$ **to** $N_{cameras}$ **do**
    if `Visibility`$(i, j) == 1$ **then**
      $CS_{obj}(i) = CS_{obj}(i, :) \bigcup$ j ;
    end
  end
end
for $i \leftarrow 1$ **to** N_{obj} **do**
  `DescendingSort`$(CS_{obj}(i, :), CM_C)$;
end
for $i \leftarrow 1$ **to** N_{obj} **do**
  while $N_{cameras} - T_C >= GC_{min}$ **do**
    C_{mode} $(CS_{obj}(i, 1))$= Target;
    T_C =T_C +1;
  end
end

Return C_{mode};

Algorithm 3: Stage transition of cameras from global to target and vice versa.

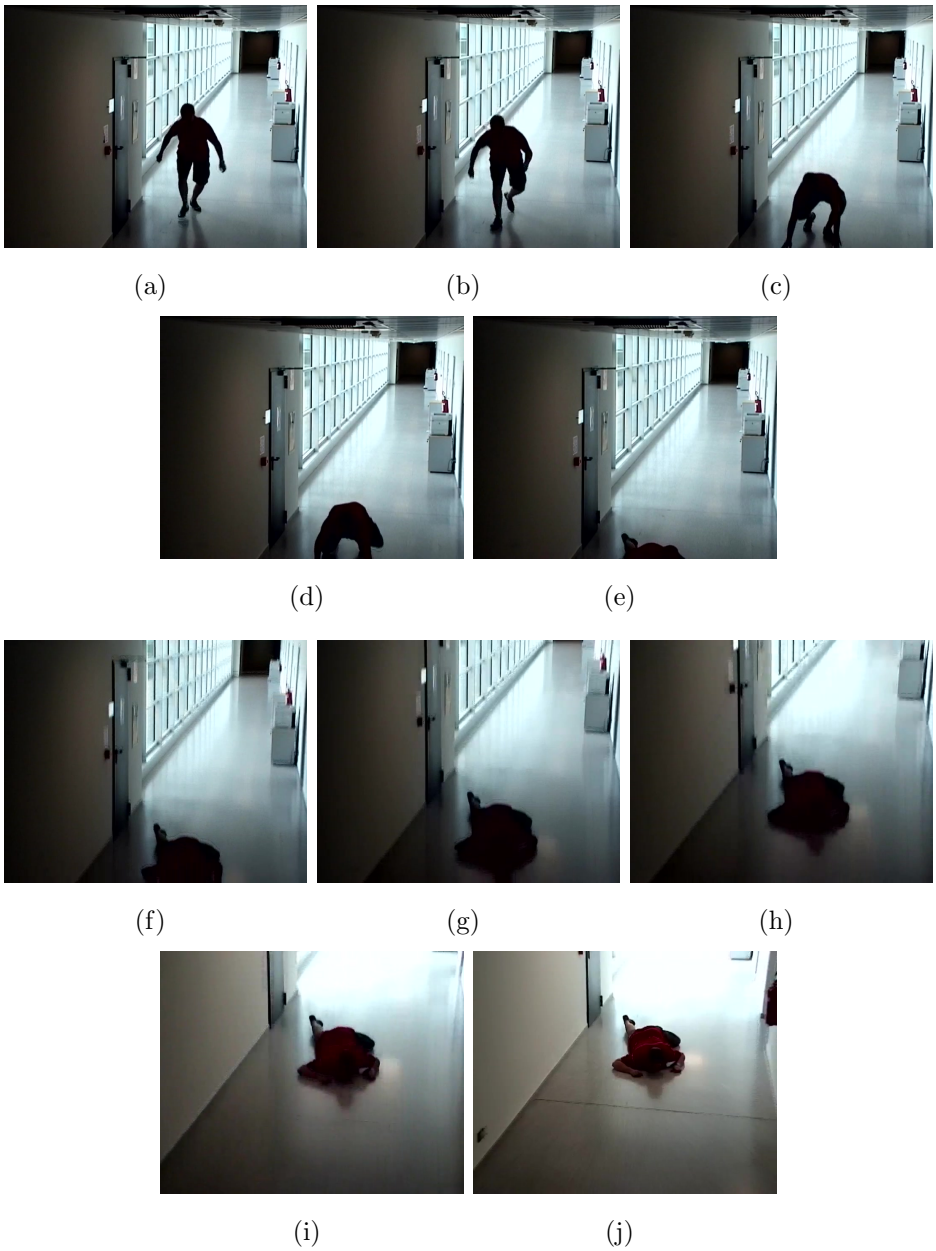


Figure 5.5: Fall detection and subsequent reconfiguration of the camera for better view.

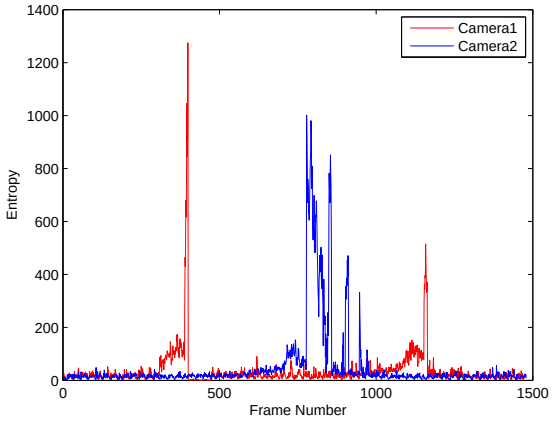


Figure 5.6: Variation of entropy metric with movement of people.

Figure 5.7: Global mode configurations of Camera 1 and Camera 2.

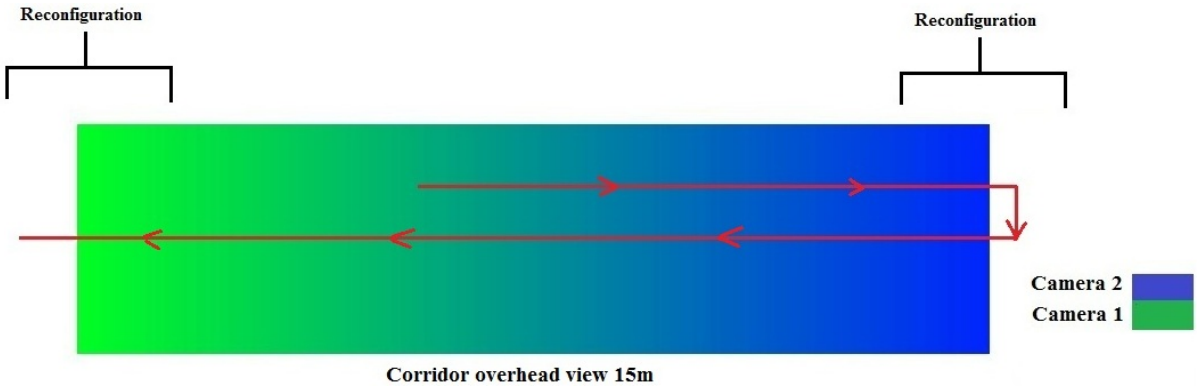


Figure 5.8: Path followed by people with respect images shown in Figures. 5.9 and 5.10.

Figure 5.9: Camera 1 switches to target mode; as the target moves out of range, it transits back to global mode.

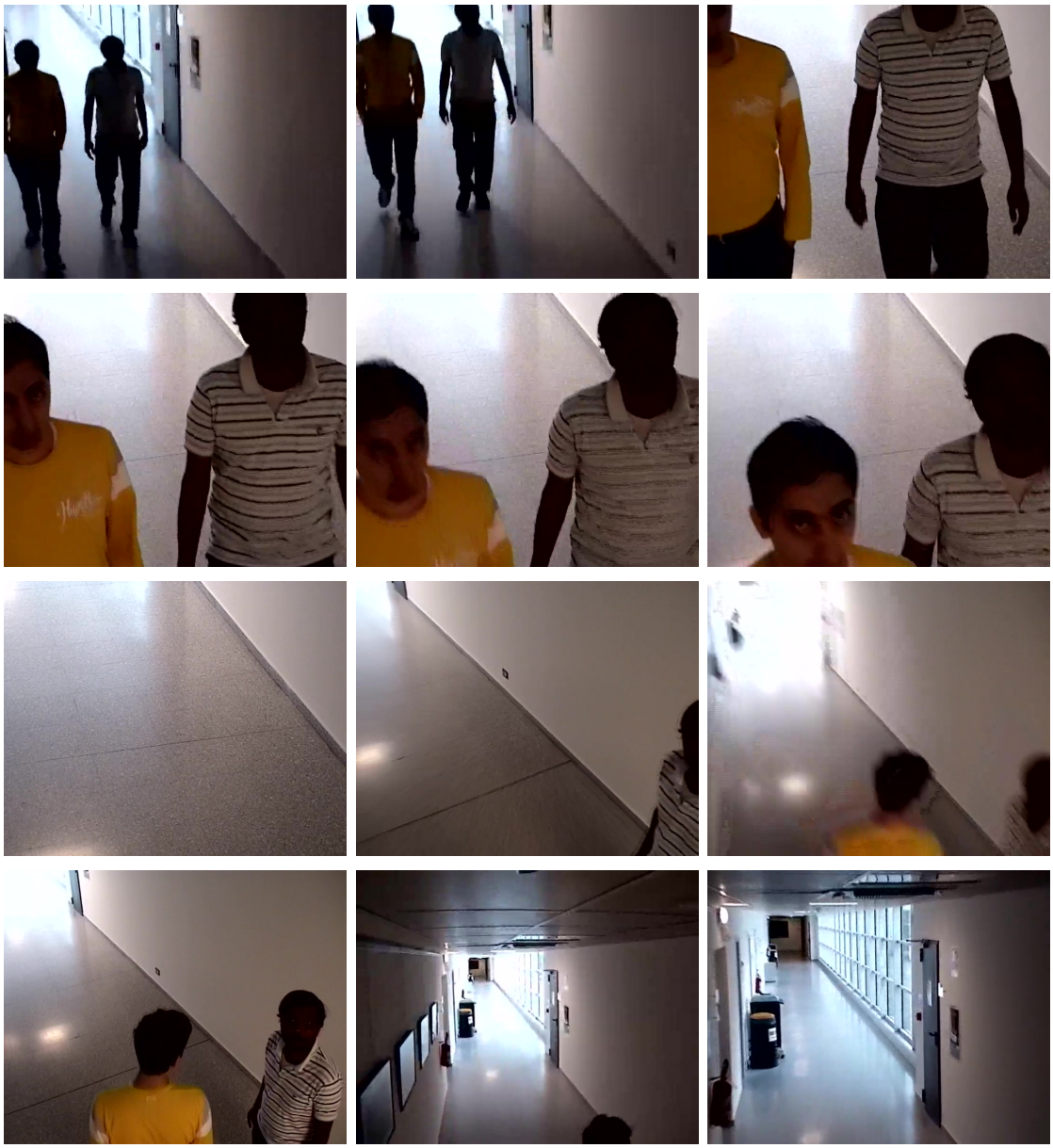


Figure 5.10: Camera 2 switches to target mode; as the target moves out of range, it transits back to global mode.

Chapter 6

Conclusion

Video surveillance is increasingly being used as a means to deter and identify crime. It also plays a role of documenting and recording events, which could be very important in certain scenarios. This is especially true for urban environments. Given the complexity of urban infrastructure, resulting occlusions and many other factors like lighting etc. Camera placement and configuration essentially determines the efficiency of the video surveillance system. In constantly changing urban scenario objects of interest move around, camera system does not remain optimal if it fails to adapt to the situation.

In light of above mentioned requirements, present day camera systems are in need of augmentation by a framework which configures the camera network, continually monitors the environment for change and reconfigures the network to suit the changing environment. Aim of this doctoral study is to propose such a framework with minimum possible computational complexity. Proposed method for camera planning and static reconfiguration are based on state of the art camera and environment models. Dynamic reconfiguration algorithm utilizes video features in compressed video bit stream for event detection, based on which real time reconfiguration is achieved.

Chapter 2 presents the camera and environment models used in camera planning along with the extensive discussion about the state of the art in this particular field. Camera model is based on many visual quality metrics like pixel density, perspective distortion and visual information. Planning is achieved by simulating the models and optimizing the camera parameters in virtual domain. Static reconfiguration which is based on one time events is also demonstrated and tested in the chapter. Finally in a real scenario comparison of the algorithm based system and traditional system is presented.

Chapter 3 Describes the light planning problem and its relevancy to camera planning. Present state of the art of light planning is also discussed. State of the art light planning algorithm with a focus on visual entropy is proposed and its validation is achieved by simulating the environment,camera and light models in virtual environment.

Chapter 4 is mainly about compressed domain video processing which forms the basis of camera reconfiguration in our work. Chapter starts with a brief discussion about compressed domain video processing for various computer vision tasks. A quantitative information metric for visual information called motion entropy is introduced. Motion entropy is based on disorder of motion field extracted from compressed video bit stream. Validity of this metric is demonstrated by utilizing it for two applications, namely video segmentation and fall detection. Both the results are compared with state of the art in their respective domains. Segmentation is further used as one of the basis for reconfiguration.

In chapter 5 we start off with state of the art of reconfiguration in multi camera networks. We then propose a distributive, low complexity, scalable dynamic reconfiguration algorithm based on motion field entropy and moving object localization. Algorithm is generic in nature and is not biased towards any particular task and is also of very low complexity.

Since the operation is in compressed domain, process of video decoding and feature extraction is eliminated. Proposed algorithm is also fully scalable in terms of number of cameras. Algorithm is tested by implementing a network of two PTZ cameras. In order to validate the system, an expected camera behaviour is charted out based on pre determined motion of the objects. Expected camera behaviour is compared with actual behaviour during the experiment for validation.

Applications Proposed framework has extensive applications in many areas. Efficiency of the video surveillance would be increased manifolds by augmenting the existing system with present framework. Almost all the computer vision methods will increase in efficiency by using the proposed framework. This is possible since the proposed system will continually reconfigure itself to produce a best possible view of a moving object in terms of visual information. Framework also addresses the common video surveillance problems like illumination changes, sensor malfunction and scaling of the system. Dynamic reconfiguration is especially useful in object tracking and face detection.

Future work In the proposed framework, dynamic reconfiguration has been only validated for the network of two cameras. We would like to extend it to the large network of the cameras to further validate the scalability and distributiveness. Motion entropy metric proposed in the study is a highly generic metric for video information. Hence there is a high possibility of training the dynamic reconfiguration algorithm for many of the specific computer vision tasks. We would also like to introduce a neural network based on line self learning capability for the system, making it more efficient and responsive to the given situation.

Chapter 7

Publications

1. **Konda Krishna Reddy** and Nicola Conci. Camera positioning for global and local coverage optimization. In *Distributed Smart Cameras(ICDSC), 2012 Sixth International Conference* IEEE, 2012.

2. **Konda Krishna Reddy** and Nicola Conci. Optimal configuration of ptz camera networks based on visual quality assessment and coverage maximization. In *Distributed Smart Cameras (ICDSC), 2013 Seventh International Conference* IEEE, 2013.

3. **Konda Krishna Reddy** and Nicola Conci. Real-time reconfiguration of ptz camera networks using motion field entropy and visual coverage. In *International Conference on Distributed Smart Cameras(ICDSC), 2013 Eighth International Conference* ACM, 2014.

4. **Konda Krishna Reddy** and Nicola Conci. Illumination modeling and optimization for indoor video surveillance. In *SPIE Electronic Imaging* International Society for Optics and Photonics, 2014.

5. **Konda Krishna Reddy** and Nicola Conci. Global and local coverage maximization in multi-camera networks by stochastic optimization.*Infocommunications Journal* 2013.

6. **Konda Krishna Reddy**, Andrea Rosani, Nicola Conci, and Francesco G.B Denatale. Smart camera reconfiguration in assisted home environments for elderly care. In *ECCV workshop proceedings. Springer*, 2014.

7. **Konda Krishna Reddy**, Nicola Conci and Francesco G.B Denatale.Global coverage maximization in PTZ-camera networks based on visual quality assessment. *IEEE sensors* 2014 (submitted)

8. **Konda Krishna Reddy** and Nicola Conci. Real-time moving object detection and segmentation in H.264 video streams *ICME 2015* (submitted)

Bibliography

[1] Typical lumen outputs and energy costs for outdoor lighting, 2011. U.S. Naval Observatory.

[2] Edouard Auvinet, Caroline Rougier, Jean Meunier, Alain St-Arnaud, and Jacqueline Rousseau. Multiple cameras fall dataset. *DIRO-Université de Montréal, Tech. Rep*, 1350, 2010.

[3] R Venkatesh Babu, KR Ramakrishnan, and SH Srinivasan. Video object segmentation: a compressed domain approach. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(4):462–474, 2004.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[5] Robert Bodor, Andrew Drenner, Paul Schrater, and Nikolaos Papanikolopoulos. Optimal camera placement for automated surveillance tasks. *Journal of Intelligent & Robotic Systems*, 50(3):257–295, 2007.

[6] Hermann Borotschnig, Lucas Paletta, Manfred Prantl, and Axel Pinz. Appearance-based active object recognition. *Image and Vision Computing*, 18(9):715–727, 2000.

[7] Svante Carlsson and Håkan Jonsson. Computing a shortest watchman path in a simple polygon in polynomial-time. In *Algorithms and Data Structures*, pages 122–134. Springer, 1995.

[8] Rizwan Chaudhry, Avinash Ravichandran, Gregory Hager, and René Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1932–1939. IEEE, 2009.

[9] Chung-Hao Chen, Yi Yao, David Page, Besma Abidi, Andreas Koschan, and Mongi Abidi. Heterogeneous fusion of omnidirectional and ptz cameras for multiple object tracking. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(8):1052–1063, 2008.

[10] Wen-Hsiung Chen, C Smith, and S Fralick. A fast computational algorithm for the discrete cosine transform. *Communications, IEEE Transactions on*, 25(9):1004–1009, 1977.

[11] Chien-Fu Cheng and Kuo-Tang Tsai. Distributed barrier coverage in wireless visual sensor networks with-qom. *Sensors Journal, IEEE*, 12(6):1726–1735, 2012.

[12] Vasek Chvatal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41, 1975.

[13] CK Cowan, Aviv Bergman, and D. Nitzan. Automatic placement of vision sensors. In *1990 NSF Manufacturing System Research Conference*, pages 389–395, 1990.

[14] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005.*

CVPR 2005. IEEE Computer Society Conference on, volume 1, pages 886–893. IEEE, 2005.

[15] Chong Ding, Bi Song, Akshay Morye, Jay A Farrell, and Amit K Roy-Chowdhury. Collaborative sensing in a distributed ptz camera network. *Image Processing, IEEE Transactions on*, 21(7):3282–3295, 2012.

[16] Russ C Eberhart and James Kennedy. Particle swarm optimization. In *IEEE Int. Neural. Networks Conf.*, volume 4, pages 1942–1948, 1995.

[17] Russell C Eberhart and Yuhui Shi. Evolving artificial neural networks. In *1998 Int. Conf. Neural. Networks and Brain*, 1998.

[18] Russell C Eberhart and Yuhui Shi. Particle swarm optimization: developments, applications and resources. In *Congress on Evolutionary Computation*, pages 81–86, 2001.

[19] Marc M Ellenrieder, Christian Wohler, and Pablo d'Angelo. Reflectivity function based illumination and sensor planning for industrial inspection. In *Optical Metrology*, pages 89–98. International Society for Optics and Photonics, 2005.

[20] CEN EN. 12464–1. light and lighting–lighting of work places–indoor work places. *Comité Européen de Normalisation CEN, Brussels, Belgium*, 2002.

[21] Ugur Murat Erdem and Stan Sclaroff. Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints. In *OMNIVIS Workshop*, 2004.

[22] W Fei and S Zhu. Mean shift clustering-based moving object segmentation in the h. 264 compressed domain. *Image Processing, IET*, 4(1):11–18, 2010.

[23] Weiguo Feng, Rui Liu, and Ming Zhu. Fall detection for elderly person care in a vision-based home surveillance environment using a monocular camera. *Signal, Image and Video Processing*, pages 1–10, 2014.

[24] E. Goldberg, D. *Genetic algorithms in search, optimization, and machine learning.* Addison-wesley, 1989.

[25] Samer Hanoun, Asim Bhatti, Doug Creighton, Saeid Nahavandi, Phillip Crothers, and Celeste Gloria Esparza. Target coverage in camera networks for manufacturing workplaces. *Journal of Intelligent Manufacturing*, pages 1–15, 2014.

[26] HHI. H.264 reference decoder from heinrich hertz institute, January 2014. http://iphome.hhi.de/suehring/tml/.

[27] Zujun Hou and Wei-Yun Yau. Visible entropy: A measure for image visibility. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 4448–4451. IEEE, 2010.

[28] i LIDS Team. Imagery library for intelligent detection systems (i-lids); a standard for testing video based detection systems. In *Carnahan Conferences Security Technology, Proceedings 2006 40th Annual IEEE International*, pages 75–80, Oct 2006.

[29] Deepak Karuppiah, Roderic Grupen, Allen Hanson, and Edward Riseman. Smart resource reconfiguration by exploiting dynamics in perceptual tasks. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1513–1519. IEEE, 2005.

[30] Spyros Kavadias, Bart Dierickx, Danny Scheffer, Andre Alaerts, Dirk Uwaerts, and Jan Bogaerts. A logarithmic response cmos image sen-

sor with on-chip calibration. *Solid-State Circuits, IEEE Journal of*, 35(8):1146–1152, 2000.

[31] Sayed Hossein Khatoonabadi and Ivan V Bajic. Video object tracking in the compressed domain using spatio-temporal markov random fields. *Image Processing, IEEE Transactions on*, 22(1):300–313, 2013.

[32] Changick Kim and Jenq-Neng Hwang. Fast and automatic video object segmentation and tracking for content-based applications. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(2):122–129, 2002.

[33] Munchurl Kim, Jae Gark Choi, Daehee Kim, Hyung Lee, Myoung Ho Lee, Chieteuk Ahn, and Yo-Sung Ho. A vop generation tool: automatic segmentation of moving objects in image sequences based on spatio-temporal information. *Circuits and Systems for Video Technology, IEEE Transactions on*, 9(8):1216–1226, 1999.

[34] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[35] David Legland. geom3d matlab library for 3d geometry, November 2012.

[36] Chia-Wen Lin and Zhi-Hong Ling. Automatic fall incident detection in compressed video for intelligent homecare. In *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, pages 1172–1177. IEEE, 2007.

[37] Junbin Liu, S Sridharan, C Fookes, and T. Wark. Optimal camera planning under versatile user constraints in multi-camera image processing systems. *Image Processing, IEEE Transactions on*, 23(1):171–184, Jan 2014.

[38] Zhi Liu, Yu Lu, and Zhaoyang Zhang. Real-time spatiotemporal segmentation of video objects in the h. 264 compressed domain. *Journal of visual communication and image representation*, 18(3):275–290, 2007.

[39] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[40] Aaron Mavrinac and Xiang Chen. Modeling coverage in camera networks: A survey. *International Journal of Computer Vision*, 101(1):205–226, 2013.

[41] Aaron Mavrinac, Xiang Chen, and Yonghong Tan. Coverage quality and smoothness criteria for online view selection in a multi-camera network. *ACM Transactions on Sensor Networks (TOSN)*, 10(2):33, 2014.

[42] Thomas Meier and King Ngi Ngan. Automatic segmentation of moving objects for video object plane generation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):525–538, 1998.

[43] Yue Meng, Ivan V Bajic, and PS Saeedi. Moving region segmentation from compressed video using global motion estimation and markov random fields. *Multimedia, IEEE Transactions on*, 13(3):421–431, 2011.

[44] Christian Micheloni, Bernhard Rinner, and Gian Luca Foresti. Video analysis in pan-tilt-zoom camera networks. *Signal Processing Magazine, IEEE*, 27(5):78–90, 2010.

[45] Anurag Mittal and Larry S Davis. A general method for sensor planning in multi-sensor systems: Extension to random occlusion. *International Journal of Computer Vision*, 76(1):31–52, 2008.

[46] Yacine Morsly, Nabil Aouf, Mohand Said Djouadi, and Mark Richardson. Particle swarm optimization inspired probability algorithm for optimal camera network placement. *Sensors Journal, IEEE*, 12(5):1402–1412, 2012.

[47] Vikram P Munishwar and Nael B Abu-Ghazaleh. Scalable target coverage in smart camera networks. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, pages 206–213. ACM, 2010.

[48] Hiroshi Murase and Shree K. Nayar. Illumination planning for object recognition using parametric eigenspaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(12):1219–1227, 1994.

[49] Don Murray and Anup Basu. Motion tracking with an active camera. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(5):449–459, 1994.

[50] Victoria Australia Persistence of Vision Pty. Ltd., Williamstown. Persistence of vision (tm) raytracer, 2004.

[51] Claudio Piciarelli, Christian Micheloni, and Gian Luca Foresti. Ptz camera network reconfiguration. In *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pages 1–7. IEEE, 2009.

[52] Claudio Piciarelli, Christian Micheloni, and Gian Luca Foresti. Occlusion-aware multiple camera reconfiguration. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, pages 88–94. ACM, 2010.

[53] Fatih Porikli, Faisal Bashir, and Huifang Sun. Compressed domain video object segmentation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(1):2–14, 2010.

[54] Markus Quaritsch, Markus Kreuzthaler, Bernhard Rinner, Horst Bischof, and Bernhard Strobl. Autonomous multicamera tracking on embedded smart cameras. *EURASIP Journal on Embedded Systems*, 2007(1):35–35, 2007.

[55] Konda Krishna Reddy and Nicola Conci. Camera positioning for global and local coverage optimization. In *Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on*, pages 1–6. IEEE, 2012.

[56] Konda Krishna Reddy and Nicola Conci. Global and local coverage maximization in multi-camera networks by stochastic optimization. *Infocommunications Journal*, 5(1):1–8, 2013.

[57] Konda Krishna Reddy and Nicola Conci. Optimal configuration of ptz camera networks based on visual quality assessment and coverage maximization. In *Distributed Smart Cameras (ICDSC), 2013 Seventh International Conference on*. IEEE, 2013.

[58] Jacob Robinson and Yahya Rahmat-Samii. Particle swarm optimization in electromagnetics. *IEEE Trans. on Antennas and Propagation*, 52:397–407, Feb 2004.

[59] Shigeyuki Sakane, Masaru Ish, and Masayoshi Kakikura. Occlusion avoidance of visual sensors based on a hand-eye action simulator system: Heaven. *Advanced robotics*, 2(2):149–165, 1987.

[60] Shigeyuki Sakane and Tomomasa Sato. Automatic planning of light source and camera placement for an active photometric stereo system.

In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1080–1087. IEEE, 1991.

[61] G Scotti, L Marcenaro, C Coelho, F Selvaggi, and CS Regazzoni. Dual camera intelligent sensor for high definition 360 degrees surveillance. *IEE Proceedings-Vision, Image and Signal Processing*, 152(2):250–257, 2005.

[62] Steven A Shafer. Automation and calibration for robot vision systems. Technical report, Technical Report CMU-CS-88-147, Carnegie Mellon University, 1988.

[63] Bi Song, Chong Ding, Ahmed T Kamal, Jay A Farrell, and Amit K Roy-Chowdhury. Distributed camera networks. *Signal Processing Magazine, IEEE*, 28(3):20–31, 2011.

[64] Bi Song, Cristian Soto, Amit K Roy-Chowdhury, and Jay A Farrell. Decentralized camera network control using game theory. In *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pages 1–8. IEEE, 2008.

[65] Open source multiple contributions. command line tool for transferring data with url syntax, March 2014. http://curl.haxx.se/.

[66] Open source multiple contributions. trans standard multimedia framework for media manipulation, March 2014. http://www.ffmpeg.org/.

[67] Orachat Sukmarg and Kamisetty R Rao. Fast object detection and segmentation in mpeg compressed domain. In *TENCON 2000. Proceedings*, volume 3, pages 364–368. IEEE, 2000.

[68] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[69] Brandon C Welsh and David P Farrington. Public area cctv and crime prevention: An updated systematic review and meta-analysis. *Justice Quarterly*, 26(4):716–745, 2009.

[70] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560–576, 2003.

[71] Yi-Chun Xu, Bangjun Lei, and Emile A Hendriks. Constrained particle swarm algorithms for optimizing coverage of large-scale camera networks with mobile nodes. *Soft Computing*, 17(6):1047–1057, 2013.

[72] David XD Yang, A El Gamal, Boyd Fowler, and Hui Tian. A 640× 512 cmos image sensor with ultrawide dynamic range floating-point pixel-level adc. *Solid-State Circuits, IEEE Journal of*, 34(12):1821–1834, 1999.

[73] Xu Yi-Chun, Lei Bangjun, Hendriks Emile A, et al. Camera network coverage improving by particle swarm optimization. *EURASIP Journal on Image and Video Processing*, 2011, 2010.

[74] Chun-Ping Yin, Yi-Feng Chen, Liao-Ni Wu, and Qi Lin. Optimization of camera calibration process based on pso algorithm. *Jidian Gongcheng/ Mechanical & Electrical Engineering Magazine*, 29(1):100–103, 2012.

[75] Xiao-Dong Yu, Ling-Yu Duan, and Qi Tian. Robust moving video object segmentation in the mpeg compressed domain. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 3, pages III–933. IEEE, 2003.

[76] Wei Zeng, Jun Du, Wen Gao, and Qingming Huang. Robust moving object segmentation on h. 264/avc compressed video using the block-based mrf model. *Real-Time Imaging*, 11(4):290–299, 2005.