8-2011

Collaborative Solutions to Visual Sensor Networks

Mahmut Karakaya
mkarakay@utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Mahmut Karakaya entitled "Collaborative Solutions to Visual Sensor Networks." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Engineering.

Hairong QI, Major Professor

We have read this dissertation and recommend its acceptance:

Qing Cao, Husheng Li, Hamparsum Bozdogan

Accepted for the Council:
Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Collaborative Solutions to Visual Sensor Networks

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Mahmut Karakaya

August 2011

This dissertation is dedicated to my loving wife Suveyda
and my beautiful daughter Zeynep Neda.

Acknowledgements

I am grateful to many people that made this work possible. First, I would like to thank my advisor, Dr. Hairong Qi. She always enlightened my path with her patient guidance, encouragement and careful assessment. Her broad knowledge in different areas inspired me and played an influential role in my research but more importantly, she has served not only as an advisor, but as a colleague and close friend. Also, I would like to thank Dr. Husheng Li, Dr. Qing Cao and Dr. Hamparsum Bozdogan. Their advice and counsel have been of equal importance. I greatly appreciate their time and input to this dissertation.

Within the AICIP Lab, I owe many thanks to my fellow graduate students. I enjoyed the many conversations and discussions that have had a tremendous impact on my research and myself as a person. Yang Bai, Sangwoo Moon, Dayu Yang, Wei Wang, Zhibo Wang, Jiajia Luo, Shuangjiang Li, Li He, Bryan Bodkin, thank you very much.

Last but not least, I would like to express my greatest gratitude to my parents and my sister and brother. I would not be able to achieve anything without their never-ending love and support. I feel very lucky and privileged to have them.

Abstract

Visual sensor networks (VSNs) merge computer vision, image processing and wireless sensor network disciplines to solve problems in multi-camera applications in large surveillance areas. Although potentially powerful, VSNs also present unique challenges that could hinder their practical deployment because of the unique camera features including the extremely higher data rate, the directional sensing characteristics, and the existence of visual occlusions.

In this dissertation, we first present a collaborative approach for target localization in VSNs. Traditionally; the problem is solved by localizing targets at the intersections of the back-projected 2D cones of each target. However, the existence of visual occlusions among targets would generate many false alarms. Instead of resolving the uncertainty about target existence at the intersections, we identify and study the non-occupied areas in 2D cones and generate the so-called certainty map of targets non-existence. We also propose distributed integration of local certainty maps by following a dynamic itinerary where the entire map is progressively clarified.

The accuracy of target localization is affected by the existence of faulty nodes in VSNs. Therefore, we present the design of a fault-tolerant localization algorithm that would not only accurately localize targets but also detect the faults in camera orientations, tolerate these errors and further correct them before they cascade. Based on the locations of detected targets in the fault-tolerated final certainty map, we construct a generative image model that estimates the camera orientations, detect inaccuracies and correct them.

In order to ensure the required visual coverage to accurately localize targets or tolerate the faulty nodes, we need to calculate the coverage before deploying sensors. Therefore, we derive the closed-form solution for the coverage estimation based on the "certainty-based detection" model that takes directional sensing of cameras and existence of visual occlusions into account.

The effectiveness of the proposed collaborative and fault-tolerant target localization algorithms in localization accuracy as well as fault detection and correction performance has been validated through the results obtained from both simulation and real experiments. In addition, conducted simulation shows extreme consistency with results from theoretical closed-form solution for visual coverage estimation, especially when considering the boundary effect.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Visual Sensor Networks

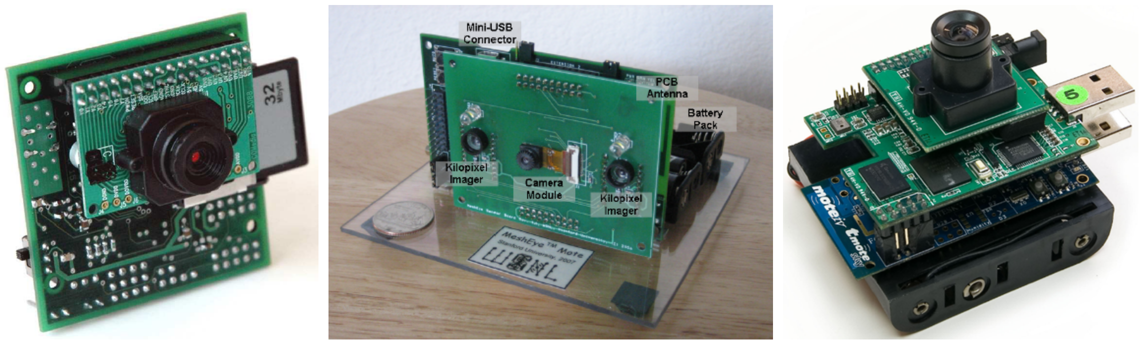
Vision is perhaps the most powerful of the human senses. Many multi-camera systems have been developed for different applications ranging from security monitoring to surveillance in the last few decades. In these applications, many high-resolution and expensive cameras are deployed into large buildings (i.e., malls and airports) and open areas (i.e., parking lots and public parks) to capture the events in the sensing field through a centralized architecture where all collected visual data sent to a central processing location via wires for storage or real-time analysis by a human operator. However, these systems are not scalable and subjective to decisions of human operators. In addition, it is unaffordable to deploy many cameras into large environments because of high cost of installation and system maintenance.

With recent developments in imaging, networking, embedded computing and circuit design technologies, it is not impossible any more to produce significantly small size and low cost visual sensor platforms with imaging, on-board processing and communication capabilities [Rinner and Wolf, 2008]. These technological improvements dramatically change the concept of cameras from being a black box that can only capture videos or images to being an intelligent device that not only takes pictures but also analyzes and reports the events in the scene. Deployment of a large number of such platforms to cover a wide surveillance area forms a so-called visual sensor network (VSN), that is capable of solving complex computer vision problems through distributed sensing and collaborative in-network processing. Therefore, VSNs have generated a new emerging interdisciplinary research field and got attractions from many diverse research disciplines including computer vision, image processing and wireless sensor networks.

Based on their potential capabilities, many researchers refer to the visual sensor network as the fundamental of the next generation of smart surveillance systems [Aghajan and Cavallaro, 2009]. Visual sensor networks are facilitated in many different multi-camera applications in diverse environments. Surveillance and security are the most obvious applications of visual sensor networks to cover the large environments. In addition to this, visual sensor networks have different application areas including smart buildings, medicine and entertainment. We summarize some of these applications below:

Surveillance: It is one of the primary applications of visual sensor networks where hundreds of cameras are deployed to monitor a large environment for a specific task such as automatic target detection and tracking in public places, traffic monitoring and flow control at intersections and parking lots. Exchange of visual information among the sensor nodes is required to achieve these specific purposes [Sankaranarayanan et al., 2008].

Security monitoring: In order to monitor the remote and secure areas against the intruders, an energy-efficient VSN is required to prolong the task for a long period of time. Therefore, sleep-wake scheduling is maintained to awake the visual sensors periodically in homogenous sensor networks. Moreover, in heterogenous sensor networks, different types of sensors that have lower power consumption than cameras might be deployed into sensing field to awake the cameras when an intruder is detected [Soro and Heinzelman, 2009].



(a) (b) (c)

Figure 1.1: Example visual sensor platforms (a) CMUcam3, [Rowe et al., 2007] (b) MeshEye, [Hengstler et al., 2007] and (c) CITRIC, [Chen et al., 2008]

Smart building: Instead of just monitoring the environment, VSNs is capable of generating automated responses to some events that require emergency actions to ensure the human safety such as patients in hospitals, elderly people in assisted living or physically disabled people in their homes, [Fleck and Strasser, 2008]. These systems analyze the captured visual data and provide relevant information about any unusual behavior to alert the emergency units for immediate actions.

Since there are many diverse application areas, many research groups in various institutions and companies have developed their own low-cost and easy-deployment visual sensor platforms based on their task-specific applications. Representative platforms in the recent literature include WiCa, Cyclops, MeshEye, CMUCam3, Citric and DSPcam (see Fig. 1.1).

One of the early designs of visual sensor platforms is the WiCa (Wireless Camera) node from NXP research and Philips research by Abbo and Kleihorst [2002] for a gesture recognition system. Cyclops is designed in UCLA by Rahimi et al. [2004] for object-tracking applications. In Stanford, Hengstler et al. [2007] developed MeshEye for low-power performance by using two low-resolution cameras to trigger a high resolution camera when target is detected. CMUCam3 designed in Carnegie Mellon by Rowe et al. [2007] is the third version of CMU cameras with limited processing

Table 1.1: Hardware specifications of example visual sensor platforms by [Abbo and Kleihorst, 2002], [Rahimi et al., 2004], [Hengstler et al., 2007], [Rowe et al., 2007], [Chen et al., 2008], and [Kandhalu et al., 2009], respectively.

Platform	Visual Sensor	Processor	Communication	Power
WiCa (2002)	CMOS, 640×480	NXP Xetal, 80MHz	Zigbee, 802.15.4	$\approx$ 1 W
Cyclops (2004)	CMOS, 352×288	ATmega128, 7.3MHz	Mica2, 802.15.4	$\approx$ 70 mW
MeshEye (2007)	CMOS, 640×480	AT91SAM7S, 55MHz	Zigbee, 802.15.4	$\approx$ 1 W
CMUCam3 (2007)	CMOS, 352×288	LPC2106, 60MHz	Firefly, 802.15.4	$\approx$ 500 mW
Citric (2008)	CMOS, 1280×1024	PXA270, 520MHz	Tmote, 802.15.4	$\approx$ 1 W
DSPcam (2009)	CMOS, 1280×1024	Blackfin, 133MHz	FireFly, 802.11g/b	$\approx$ 50 mW

power. As a result of joint work by UC Berkeley and Merced, CITRIC camera node is designed by Chen et al. [2008] and used for image compression and target tracking applications. The most recent visual sensor platform is DSPcam designed by Kandhalu et al. [2009]. The overview of the hardware specification of these platforms are summarized in Table 1.1.

1.2 Challenges in Visual Sensor Networks

Although many potential applications have been made possible using these powerful visual sensor platforms, VSNs also present unique challenges that could hinder their practical deployment compared to conventional 1-D omnidirectional scalar sensor networks (SSNs) (e.g., temperature sensors, microphones and geophones) because of unique features of cameras. These features include the extremely higher data rate, the directional sensing characteristics with limited field of view, and the existence of visual occlusion.

The main challenge in visual sensor networks is the huge data volume of visual sensors which generally requires high network bandwidth for data transmission, which cannot be adequately addressed because of the energy constraint and the usage of the low-bandwidth wireless communication. Among all the major processes taken place in a VSN, i.e., sensing, processing, and communication, communication consumes most of the energy. Usually, we can neglect the energy consumption on sensing and processing compared to that on communication. However, when processing images, some computationally expensive algorithms can easily consume as much energy as communication and become a problematic issue for practical deployment of VSNs. Therefore, we need to follow two basic guidelines when solving computer vision problems in a distributed environment. First, the sensor data, i.e., the image, should be pre-processed locally to reduce the amount of transmitted data volume [Kahn et al., 1999]. Second, simple but effective local processing algorithms should be developed to reduce the computational cost.

Another challenge in VSNs is the existence of "visual occlusions" among targets which cannot be avoided because of the directional sensing nature of visual sensors. Since the light emits directly and cannot pass through the non-transparent objects, a camera can visually capture a target only when the target stands in the field of view and there is no other occluding targets between the camera and the target. Usually, it is not possible to cover all targets in a crowded environment by using a single camera. Therefore, we need to deploy and use multiple visual sensor nodes to cover a large sensing field, taking the visual occlusion into account.

In addition to huge data volume and visual occlusion, since visual sensors have limited field of views, and limited computational capacity, visual information obtained by each sensor node is neither sufficient nor accurate. Therefore, collaboration in visual sensor networks is essential not only to compensate for the processing, sensing, energy, and bandwidth limitations of each sensor node but also to improve the accuracy and robustness of the network.

1.3 Motivation and Contributions

In order to address challenges in VSNs listed above, many researchers from various institutions have proposed different approaches and hundreds of journal and conference papers have been published in recent years. However, there are still many partially solved or unsolved issues in visual sensor networks that need to be investigated in detail.

In this dissertation, we first present an *energy-efficient* and *light-weight* approach to localize targets in a crowded environment using a visual sensor network through distributed sensing and collaborative in-network processing by taking the directional sensing and visual occlusion issues in visual sensors into account. Since the presence of faulty sensor nodes in VSNs affects the accuracy of target localization, we design a fault-tolerant target localization algorithm that would not only accurately localize targets but also detect the faults in camera orientation, tolerate these errors and further correct them before they cascade.

In addition to target localization and fault tolerance in VSNs, we also consider a relevant challenging topics in visual sensor networks, namely visual coverage estimation. In order to ensure the required coverage in a sensing field to accurately localize targets or tolerate the faulty nodes, we derive the closed-form solution to estimate the visual coverage probability based on the "certainty-based target detection" model that takes directional sensing of cameras and existence of visual occlusions into account. We briefly introduce each of these topics and emphasize our research contributions as follows.

The Certainty Map Model: One of the most significant contributions of this dissertation work is the development of certainty map method for collaborative solutions to visual sensor networks. Traditionally, the intersections of the back-projected 2D cones of each target is used to solve the problems in a VSN (e.g., target localization and visual coverage estimation). However, the existence of visual occlusion among targets would generate many false alarms. Instead of resolving the uncertainty about target existence at the intersections, we identify and study the non-occupied areas in the cone and generate the so-called *certainty map* of non-existence of targets.

Dynamic Itinerary for Target Localization: The second major contribution is the development of dynamic itinerary for progressive certainty map integration in target localization algorithms. In order to localize target in crowded environments, we focus on the design of a light-weight, energy-efficient, and robust solution where not only each camera node transmits a very limited amount of data but that a limited number of camera nodes is involved. We propose a dynamic itinerary for certainty map integration where the entire map is progressively clarified from sensor to sensor starting the integration with the sensor that has the greatest contribution information to the current certainty map. When the confidence of the certainty map is satisfied, targets are localized at the remaining unresolved regions in the certainty map.

Fault Tolerance, Detection and Correction: The third contribution is the design of the fault tolerance, detection and correction algorithm in target localization. Fault tolerance in VSNs is more challenging than in conventional scalar sensor networks (SSNs) because of the directional sensing nature of cameras and the existence of visual occlusion. We focus on the design of a collaborative target localization algorithm in VSNs that would not only accurately localize targets but also detect the faults in camera orientation, tolerate these errors and further correct them before they cascade. Based on the locations of detected targets in the final certainty map, we construct a generative image model in each camera that estimates the camera orientation, detect inaccuracies in camera orientations and correct them before the fault in the system cascades and reaches a point where the performance of the algorithm dramatically drops.

Visual Coverage Estimation: The fourth contribution is a first attempt toward a closed-form solution for the visual coverage estimation problem in the presence of occlusions to guarantee the required coverage in a sensing field. By adapting the certainty-based target detection model in coverage estimation in a randomly deployed VSN, we derive a closed-form solution for the estimation of the visual coverage estimation. Therefore, the sensor related parameters (e.g., sensor density, sensing range, etc.) can be decided before deployment in order to have proper visual coverage in the sensing field. In addition, since the visual coverage probability in a crowded environment depends not only on the sensor density and deployment but also on the target density and distribution, our proposed closed-form solution considers both the directional sensing nature of cameras and the visual occlusions among targets and provides more accurate and realistic coverage estimation in a crowded VSN.

1.4 Dissertation Organization

The dissertation is organized as follows: In Chapter 2, we study target localization using a progressive certainty map in visual sensor networks and its distributed version for certainty map integration. Then, Chapter 3 focuses on the design of a collaborative target localization algorithm in VSNs that would not only accurately localize targets but also detect the faults in camera orientation, tolerate these errors and further correct them before they cascade. In Chapter 4, we represent a first attempt toward a closed-form solution for the visual coverage estimation problem in the presence of visual occlusions among crowded targets in a visual sensor network. Finally, we conclude this dissertation with a summary of accomplished and directions for future research in Chapter 5.

Chapter 2

Collaborative Target Localization in Visual Sensor Networks

2.1 Introduction

In this chapter, we study a traditional computer vision problem, target localization, using visual sensor networks. The target localization problem in VSNs faces two major challenges [Qian and Qi, 2008]. First of all, because of the crowded targets, "visual occlusions" among targets cannot be avoided. Secondly, since visual sensors have limited field of views, and limited computational capacity, visual information obtained by each sensor node is neither sufficient nor accurate. To localize targets in a crowded environment with the existence of visual occlusion and partial or inaccurate information is the major challenge of this work. In this chapter, we present a progressive solution to localize crowded targets by performing simple but effective image processing algorithms on each sensor node, transmitting very limited amount of processed data among only a limited number of nodes, and using an efficient data fusion algorithm to obtain the final result.

The work in this chapter was first published in [Karakaya and Qi, 2011b] and [Karakaya and Qi, 2009].

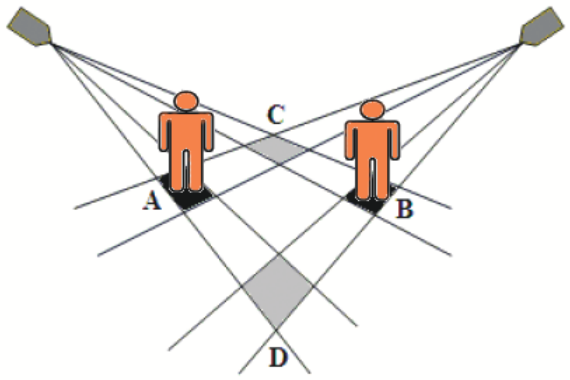


Figure 2.1: Four intersections are created from back projection, where intersections A and B are locations of two real targets but empty intersections C and D are also generated due to occlusion.

In traditional target localization algorithms, intersections of the back-projected 2D cones of the targets are calculated to localize all the individual targets. Occupied areas in 2D visual cones correspond to the possible occupancy information generated by planar projection of the 3D cones onto a plane parallel to the ground, also referred to as the *existence information* in this dissertation. If the cones from different sensors intersect at the same point, it can be considered there is at least one target in that intersection. However, in crowded environments, many “empty” intersections that are not actually occupied by any targets are created because of occlusion, as shown in Fig. 2.1. Therefore, the existence information in the corresponding intersections is not certain. To remove the uncertainty about the target existence at the intersections and to detect the real locations of the targets have been very challenging problems in computer vision.

To solve this problem, we present a technique to localize crowded targets by a collaborative effort from a group of sensor nodes. Instead of resolving the uncertainty about the target existence, we identify and study the non-occupied areas in 2D visual cones, also referred to as the *non-existence information* in this dissertation. Since it is certain that there is no target in the corresponding region, we refer to the map generated from this process as the *certainty map* [Karakaya and Qi, 2009].

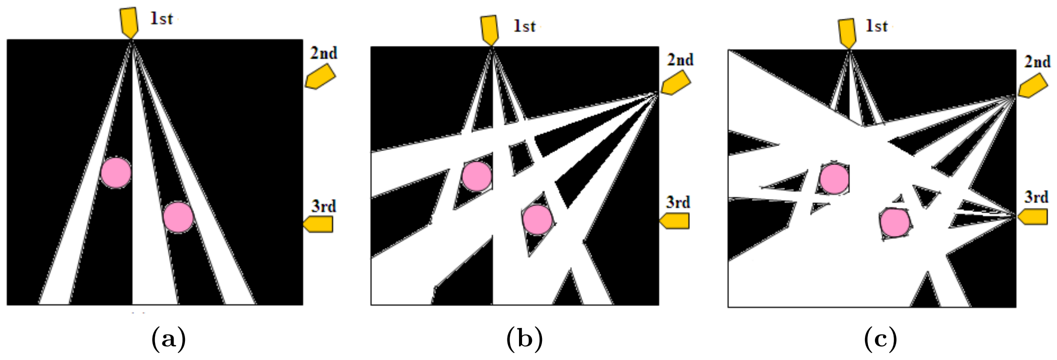


Figure 2.2: (a) The certainty map generated by the 1^{st} sensor node (b) Fusion of certainty maps generated by the 1^{st} and the 2^{nd} sensor nodes (c) Fusion of certainty maps generated by the 1^{st}, 2^{nd}, and 3^{rd} sensor nodes.

In our system, each camera extracts objects of interest from the background, computes the 2D cones of objects, determines the non-occupied areas by targets and combines them with other sensor nodes to jointly assess the situation in a more accurate way. Certainty map is used as the information exchange unit between sensor nodes when fusing the non-occupied regions to yield a globally consistent belief of non-occupancy, as shown in Fig. 2.2. The final certainty map, which keeps the real occupied regions, is used to estimate the target locations. The advantage of our system is its efficiency in both computation and bandwidth usage.

The chapter is organized as follows: Section 2.2 describes the background and related works in the visual sensor network. Section 2.3 introduces our proposed progressive target localization method. In Section 2.5, collaborative processing and itinerary selection techniques are described. Section 2.6 compares the certainty map with non-existence information and occupancy map with existence information. Comprehensive analytical study on communication and energy efficiency is performed in Section 2.7. Experimental results are presented in Section 2.8. Finally, we conclude in Section 2.9.

2.2 Background and Related Works

In the literature, there exist many works related to larger-scale surveillance and monitoring of activities using different types of sensing modalities such as range sensors and visual sensors. In robotics, range sensors like laser scanners and sonar sensors are widely used to obtain distance information between objects and sensor nodes for mapping and localization applications [Kleeman and Kuc, 1995; Tards et al., 2002]. Although sonar sensors have the advantage of fast response and cheaper cost compared to laser sensor, mapping and localization applications with range-only measurements are challenging issues due to the partial observability and large amount of reflectance outliers especially when objects are small or with wide angle [Tsalatsanis et al., 2006]. Because of these limitations, visual sensors that have similar cost as sonar sensors become a viable choice to localize objects within the field of view. Moreover, visual sensors may provide additional useful information when integrating with other image processing related applications (such as target recognition).

Single camera approaches [Han et al., 2004; Haritaoglu et al., 1998; Isard and MacCormick, 2001] are easy to deploy but they could not be applied efficiently in complex environments such as crowded and occluded target scenarios because they could not provide 3D information by using a single camera view. Therefore, several multi-camera systems are researched and proposed to detect and track multiple objects and compute their accurate 3D locations in a complex environment. Applications like target detection [Qian and Qi, 2008], tracking [Krumm et al., 2000] and counting [Yang et al., 2003] based on wireless networks have been previously investigated. In [Krumm et al., 2000], and [Sogo et al., 2000], stereo techniques are used to collect the information from different cameras to localize people. However, these methods are computationally complex as they use color, shape, or texture based methods [Collins, 1996; Haritaoglu et al., 1999; Mittal and Davis, 2003] to segment, detect, and track multiple targets in a scene by matching each pair of views between camera nodes, and matching is computationally expensive in crowded environments because of the large number of possibilities.

In recent multi-camera systems, occupancy map is a popularly used technique to localize objects by using foreground images captured by overhead cameras [Halvorson and Parr, 2007] or using visual hull procedure and motion information of the objects [Yang et al., 2003]. However, these methods either require too high computational complexity to be deployed in VSNs or a central processing center in which information from every camera in the network is collected to achieve specific tasks. The probabilistic version of occupancy map (POM) uses the visual hull idea with color and motion models to estimate the probability of target occupancy by comparing the foreground image with its estimated trajectory in each camera, iteratively [Fleuret et al., 2008]. However, this approach requires many iterations to localize individual targets in uncrowded scene and additional color and motion models which are computationally expensive. Therefore, they may be useful for a small local group of sensors and targets but may not be applicable in a large and crowded VSN environment. Overhead cameras mounted on the ceiling have also been used for target localization. However, the deployment is not ad hoc and is limited to only indoor environment.

Shape from silhouette (SfS) techniques [Laurentini, 1994] resolve the volumetric description of a target by combining the silhouette cones in 3D which can be used to localize targets by projecting the resultant silhouette to 2D plane surface. However, it is more expensive to compute than back-projected 2D cone methods which can be described as a subset of the SfS techniques with similar detection performance because planar projection of 3D cone preserves the most useful information of moving targets along a plane which is perpendicular to the projection plane in target localization applications [Yang et al., 2003].

2.3 Target Localization with Certainty Map

In this section, we present the centralized version of the proposed method to localize crowded targets within the sensing region of a VSN. Its progressive or distributed implementation will be discussed in Section 2.5. In the proposed method, each sensor node captures a snapshot of the scene, processes it locally to compute the local certainty map and sends the result to the processing center to collaboratively reach a decision about target locations in the scene.

Because of the sheer amount of data generated at each camera node, local processing is needed to provide the necessary information with a much smaller data volume. We need to keep in mind that local processing cannot be computationally expensive; otherwise, it will consume as much energy as communication. Due to its algorithmic simplicity, background subtraction is adopted for object segmentation. By using the background subtracted image, 2D visual cones can be generated and the non-occupied areas in the field of view (FOV) of each camera node can be derived. If an area within the FOV of a sensor node is not occupied or occluded by any object, it is declared as a non-occupied area. On the other hand, the occupied areas are the ones where it is possible that there exist targets. The uncertainty is due to either occlusion or outside of the FOV of the camera. We use the so-called *certainty map* to record the non-occupied areas.

In a certainty map, the environment is divided into uniformly sampled grids where each grid point represents that the corresponding ground space in the surveillance area is *certain* about target non-existence (labeled with one or white) or *uncertain* about target existence (labeled with zero or black). Note that since the certainty map is essentially a binary image consisting of zeros and ones, it can be further compressed using existing coding techniques to decrease the data size to save the communication cost.

Fig. 2.3 illustrates the steps in constructing the certainty map at a node. For object segmentation, background subtraction is utilized because of its algorithmic

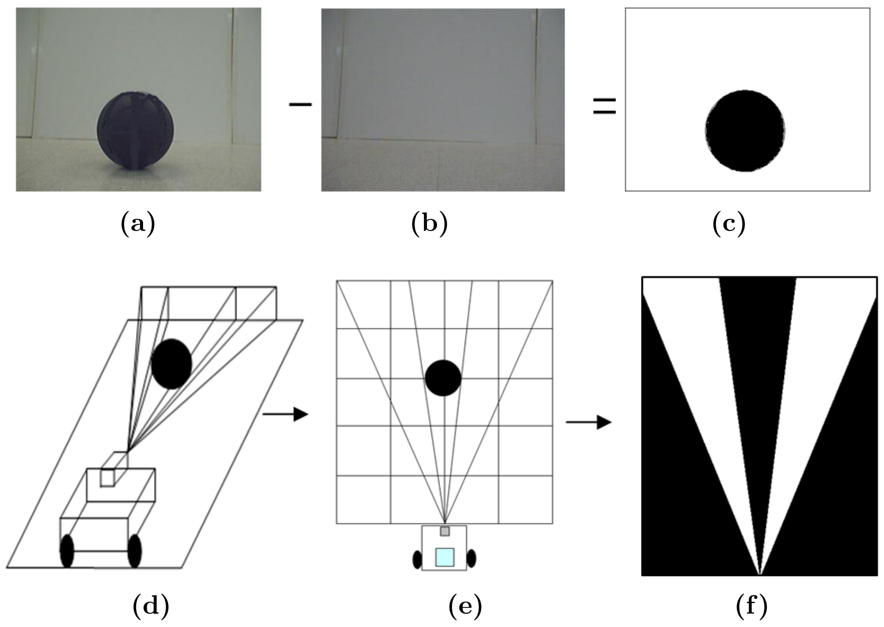


Figure 2.3: Illustration of local processing and construction of the certainty map at a visual sensor node. (a) The original captured image by sensor node, (b) The background image, (c) The foreground image, (d) 3D cones of an object (e) Projection of 3D cones onto a plane parallel to the ground, and (f) Constructed local certainty map.

simplicity. For the corresponding scenario in Fig. 2.3, we obtain the foreground image in Fig. 2.3c by subtracting the background image in Fig. 2.3b from the original image in Fig. 2.3a. Each object sweeps a cone in 3D space as shown in Fig. 2.3d. To find 2D visual cones of the object, these 3D cones are projected onto a plane parallel to the ground as shown in Fig. 2.3e. The non-occupied (white) areas in the 2D visual cones are thus determined to construct the local certainty map as shown in Fig. 2.3f.

In this work, it is assumed that each sensor node, s_i, knows its coordinates in the 2D global coordinate system as (x_{s_i}, y_{s_i}). Let $\mathbf{v}_{s_i}$ denote the vector that describes the non-occupied areas within the FOV of the sensor node, s_i. The vector is composed of a series of $(\varphi_{i,j}, \psi_{i,j})$ pairs, where $\varphi_{i,j}$ and $\psi_{i,j}$ record the starting and ending angles, respectively, of the j^{th} non-occupied area in the corresponding planar projection of 3D cones onto the 2D ground space, $j = 1, \ldots, B_i$, and B_i is the total number of non-occupied areas of the image taken by node s_i. We can then describe the certainty map through a much condensed vector representation,

$$\mathbf{v}_{s_i} = [x_{s_i}, y_{s_i}, \varphi_{i,1}, \psi_{i,1}, \ldots, \varphi_{i,B_i}, \psi_{i,B_i}] \tag{2.1}$$

The conversion between $\mathbf{v}_{s_i}$ and the certainty map can be done through a mapping function, $f(\mathbf{v}_{s_i})$, whose value at coordinate (x, y) is,

$$f_{x,y}(\mathbf{v}_{s_i}) = \begin{cases} 1, & \text{if } \varphi_{i,j} \le \arctan \frac{x - x_{s_i}}{y - y_{s_i}} \le \psi_{i,j},\ j = 1, \ldots, B_i \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

Let $S = \{s_1, s_2, \ldots, s_N\}$ denote the set of sensor nodes in the network, we then have

$$U(S) = \bigcup_{i=1}^{N} f(\mathbf{v}_{s_i}) \tag{2.3}$$

where *U(S)* denotes the union formed by all the local certainty maps in *S*. Targets are located in the *complement* of *U(S)*, that is, the unresolved regions (labeled in black). Note that the sensor node communicates only the vector representation of the certainty map, $\mathbf{v}_{s_i}$, to the processing center to fuse its local certainty map with the global certainty map.

Assume that information obtained from each sensor node is accurate or trustworthy (relaxation to this assumption will be discussed in Chapter 3), since the size of the total uncertain region in the global certainty map monotonically decreases, the convergence of the global certainty map is guaranteed. The idea is that the uncertain regions will be shrinking as local certainty maps are fused. If the non-existence of target for certain region is declared by one sensor node, the corresponding region is globally announced as non-occupied and cleared from the certainty map. If the entire surveillance area is covered by sensor nodes, then the only uncertain region left would be the location of targets.

By taking this inverse approach to traditional target localization problem, instead of having to resolve the uncertainty of target existence, we remove the uncertainty of target non-existence which is a more computationally efficient solution. Its progressive implementation which further reduces communication overhead will be discussed in Section 2.5.

2.4 Target Counting Algorithm

As further application, we can count targets in the final certainty map after localizing them in the sensing region. The final certainty map consists of sets of small regions with potential locations of targets, referred to as the phantoms. Phantoms are the remaining areas in the certainty map that could not be cleared by any sensor nodes. In literature, there are different approaches to count the objects in each phantom. In [Yang et al., 2003], the area of each phantom is divided by object size regardless of object shape. However, because of the occlusion of objects in crowded environments, there will be some residual areas in the phantom that cannot be clarified by any cameras and make the size of the phantoms bigger than the object size. Therefore, it is necessary to consider the shape of the object and residual areas when to compute the number of objects in each phantom. If there is no occlusion and there is infinite number of cameras, the size of the phantom converges to the actual size of the object. However, in crowded targets, it is not possible to prevent occlusion and the residual areas in the phantom. In Fig. 2.4, the possible residual areas are shown for cylindrical objects. The planer projections of the cylindrical objects are discs in the ground space. The smallest residual areas occurs when the objects touch each other assuming there is no overlapping between the objects.

In Fig. 2.4(a), the residual area around the object converges to zero if there is infinite number of cameras. In Fig. 2.4(b) and 2.4(c), the smallest residual areas around the objects, shown for two and three objects cases, are the areas between the objects.

These residual areas, R_i, cannot be prevented so they have to be taken into consideration in the calculation of minimum size of the total area for different number

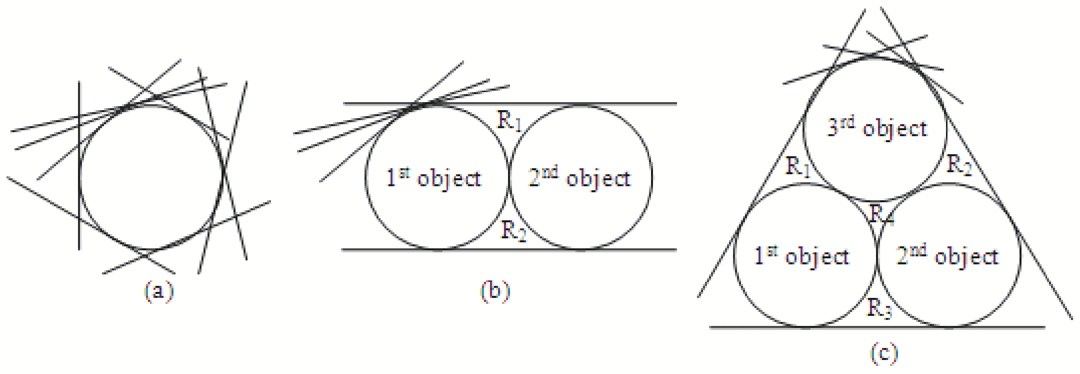


Figure 2.4: Three different scenarios of residual areas for cylindrical objects.

of object cases in a phantom. They are calculated by adding the corresponding minimum residual areas to total area of objects, such as two and three object cases shown in Fig. 2.4. The number of objects can be found by comparison between the area of each phantom and the pre-calculated minimum size of the total area for different number of object cases.

2.5 Collaborative Processing and Itinerary Selection

In order to fuse or integrate the certainty map, two mechanisms can be adopted, centralized or distributed. In the centralized approach, each camera node sends its local certainty map to a processing center for information fusion. In the distributed approach, the certainty map is propagated through the network. At each camera node, the certainty map is refined by integrating with local certainty map to add more certainty to the previous map. In other words, the local information helps clarify uncertain areas as this fusion process prolongs. Hence, we also refer to the distributed approach as *progressive integration* and use them interchangeably in this dissertation.

If there is no energy or bandwidth limitation, the information from all the available cameras can be used to compute the best possible certainty map since the residual areas in the certainty map monotonically decreases as the number of cameras increases. However, the energy resource is very limited in VSNs and increasing the number of used camera nodes decreases the network lifetime. In the next section, we will compare the performance between centralized and progressive integration of the certainty map through experiments. Here, we first tackle a critical problem in realizing the progressive integration, i.e., how to determine the itinerary along which the certainty map propagates. This problem is two-fold: the itinerary of propagation and the stop condition of propagation.

2.5.1 Itinerary Selection

The visual data provided by geographically close camera nodes might be highly correlated. If removal of a sensor node from the itinerary does not shrink the uncertainty region of the certainty map too much, the information provided by that camera node can be deemed redundant. In order to save energy, transmission of the certainty map between these sensor nodes must be avoided. To find the minimum subset of camera nodes to construct the accurate certainty map is the main goal of itinerary selection.

In literature, different approaches for sensor selection have been proposed for different scenarios. In [Yang et al., 2004], clustering the parallel and perpendicular cameras to the vectors connecting the objects is a good approach for two-object case. However, how to find the vectors connecting the objects without prior knowledge of object locations or how to update the vectors for moving objects is another problem.

In this dissertation, we study three different itinerary selection criteria, including fixed, random and dynamic itinerary, to fuse the certainty map.

Fixed Itinerary

To overcome the problems identified above, the cameras can be clustered by using their orientation angles regardless of the target location. The best choice for the next

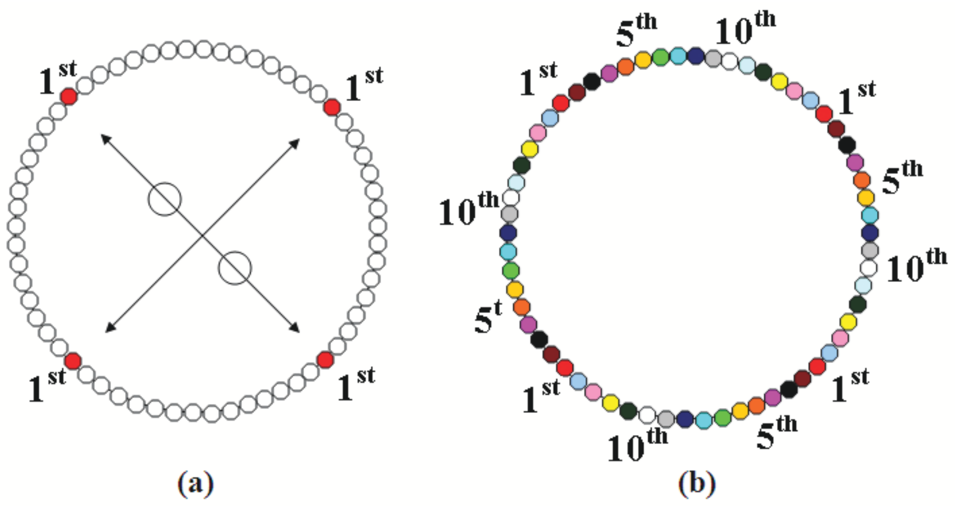


Figure 2.5: Best camera selection with clustering (a) by using the parallel and perpendicular cameras to the vectors connecting the objects and (b) best complimentary FOV.

camera is to select the camera which has the best complimentary field of view of the previous camera as shown in Fig. 2.5. The 90 degree difference in the camera orientation with the previous camera gives the best complimentary FOV. Therefore, to determine the fixed itinerary, cameras which have the 90 degree difference in the camera orientation with others are clustered into small subsets.

Random Itinerary

Instead of using the fixed itinerary, we also implement the itinerary in which cameras are randomly selected to integrate the certainty map.

Dynamic Itinerary

Neither fixed nor random itinerary determines the route of certainty map migration based on the *content* of the map. Therefore, it is not guaranteed that the next integration would clear the most uncertain region leading to the shortest route. To overcome this problem, we propose a dynamic itinerary selection method, where in each iteration sensor nodes compute their local certainty maps, calculate the size of

Algorithm 1: Dynamic Itinerary

```
Data: Certainty map, $CM = \emptyset$, is fully occupied; $S = \{s_1, s_2, \ldots, s_N\}$ is the
set of sensor nodes. while $S \neq \emptyset$ do
    for each $s_i$ in S do
        Compute clarification amount, $|f(\mathbf{v}_{s_i})|$, by using the current $CM$;
        Broadcast $|f(\mathbf{v}_{s_i})|$ to the network;
        Pause for a short waiting time to receive the $|f(\mathbf{v}_{s_i})|$ from other nodes;
        if $S \neq \emptyset$ and $|f(\mathbf{v}_{s_i})| < \delta$ then
            $S = S - s_i$ to remove redundant sensors;
        end
        if $|f(\mathbf{v}_{s_i})|$ is the largest then
            $S = S - s_i$ to remove the winning sensor;
        end
    end
    if $s_j = \arg\max_{s_i \in S} |f(\mathbf{v}_{s_i})|$ then
        Update $CM$ by $s_j$, the $j^{th}$ sensor node;
        Broadcast updated $CM$ to the network;
    end
end
```

area that can be cleared from the current certainty map and broadcast it through the network. The sensor node which has the largest clarification area gets the priority over others to integrate its local certainty map with current certainty map and broadcasts the updated certainty map to the network to allow other sensor nodes to recalculate the amount of additional clearance on the certainty map. This procedure repeats until the confidence test (to be discussed in Section 2.5.2) is passed.

This progressive integration method is described in Algorithm 1, where S denotes the set of sensor nodes and $|f(\mathbf{v}_{s_i})|$ denotes the total area that can be cleared from the current certainty map by sensor node s_i. δ is the threshold to determine if the sensor node holds adequate additional clarification information to remain in set S. If $|f(\mathbf{v}_{s_i})| < \delta$, the sensor node, s_i, is removed from S. Note that set S is maintained at *each* sensor node and will remain consistent across the entire network at each iteration. Assume the sensor nodes within the network are synchronized. After each node broadcasting $|f(\mathbf{v}_{s_i})|$ to the network, each node is able to sort the received

$|f(\mathbf{v}_{s_i})|$'s from other nodes and remove the nodes from S with either the highest clarification area or the clarification area below the threshold, δ. Since δ is fixed across the entire network, the content of S should always be the same within each iteration although it is distributively maintained.

2.5.2 Confidence Test

No matter which itinerary approach is adopted, a common question each faces is when to stop the integration process. Because of energy constraints, we have to stop transferring the certainty map in the VSN when certain criteria can be satisfied.

In the fixed or random itinerary, the certainty map is transmitted to the next sensor node without prior knowledge about how much it can clarify from the certainty map. If the sensor node does not contribute to certainty more than a pre-defined threshold, it raises a stopping flag. If the number of consecutive stopping flags reaches a certain percentage of the total number of sensor nodes, transmission is stopped and the final certainty map is declared. This method has an apparent drawback. If the certainty map travels among sensor nodes with redundant information at some portion of the itinerary, the procedure will be stopped even though other sensors can still contribute.

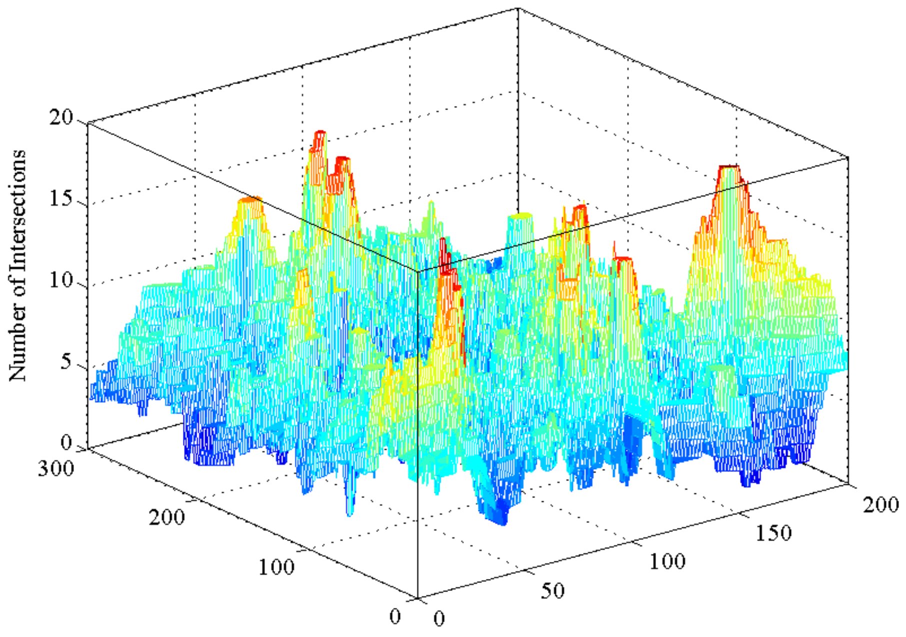
In dynamic itinerary, it is not necessary to count the number of flags and the stopping criterion is naturally realized with the design of Algorithm 1. When the winning sensor with the largest clarification amount presents a $|f(\mathbf{v}_{s_i})| < \delta$, that means none of the remaining sensor nodes would have adequate additional clearance area on the certainty map as the size of clearance area monotonically decreases and that the integration process can be stopped immediately. This is also equivalent to saying that the iterations should be terminated when $S = \emptyset$.

2.6 Comparison between Certainty Map and Occupancy Map

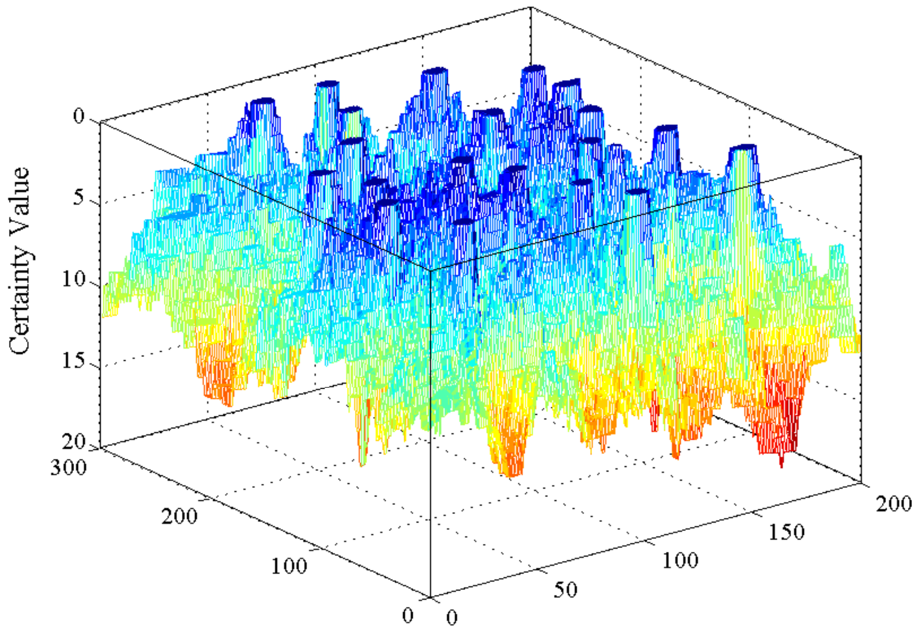
To better understand the difference between the certainty map-based approach that integrates target non-existence information and the occupancy map-based approach that integrates target existence information for target localization purpose, we perform the following simulation to visually demonstrate the advantage of the certainty map-based approach. Details regarding simulation setup are provided in Sec. 2.8.1. In the simulation, 20 targets and 150 visual sensor nodes are randomly deployed in the sensing field. Only centralized processing is adopted where each sensor node generates the corresponding foreground image of targets and computes the 2D visual cones of the non-occupied or occupied areas using the planar projection of 3D cones. Then, the results are sent to the processing center to arrive at the final decision about the target locations by using either the certainty map or occupancy map approaches. Before applying the threshold to localize targets, the final integrated/fused versions of the occupancy and certainty maps are shown in Fig. 2.6 where the left figure is the result from the occupancy map, and the right is from the certainty map. The z-axis of the 3-D occupancy map represents the number of sensors that suspects target existence at the corresponding spatial location and the z-axis of the 3-D certainty map represents the number of sensors that is certain about target non-existence at the corresponding spatial location.

We observe that both methods show local maxima at the target locations. However, the value of local maxima in the occupancy map differ from target to target depending on the density of sensor coverage at that location; on the contrary, the local maxima in the certainty map show the same height. To localize the targets from the occupancy map by selecting an appropriate threshold is a challenging problem because of the various pick values present.

As shown in Fig. 2.7(b-c), if the threshold value is chosen as 70% or 65% of the pick value in the occupancy map, 9-10 targets can then be localized without



(a)



(b)

Figure 2.6: The final versions of (a) the occupancy map and (b) certainty map generated at the processing center before applying the threshold to localize targets.

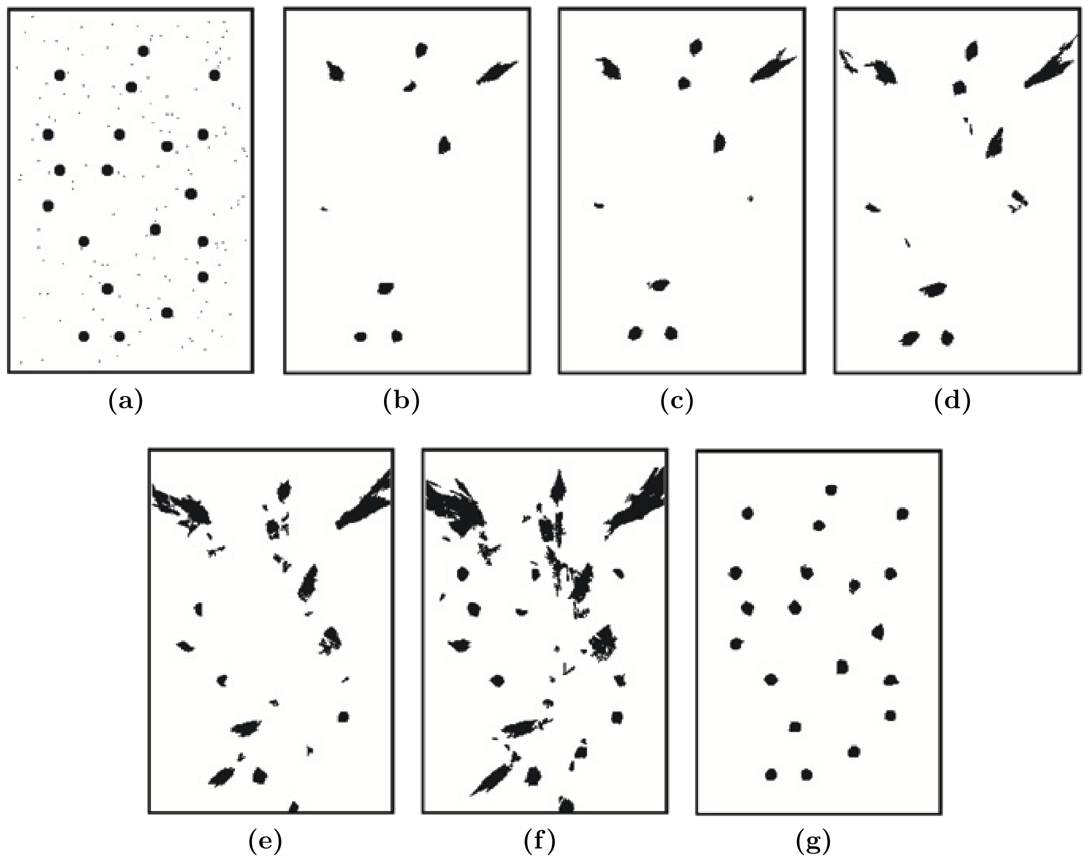


Figure 2.7: (a) Simulation setup with 20 targets and 150 sensor nodes, (b-f) Final version of the occupancy map using threshold values 70%, 65%, 60%, 55%, and 50%, of the pick value in the occupancy map, respectively and (g) Final version of the certainty map using threshold value 0.

any false alarms but half of the targets would be missed. To detect the missing targets, lower threshold values are demanded which introduce many false alarms, as shown in Fig. 2.7(d-f). Moreover, the areas of some detected target regions are smaller or larger than the target size because of higher or lower threshold values selected, that would affect the exact localization of the detected targets. To solve this problem and to choose the threshold value automatically for each individual target, an additional post-processing algorithm, such as searching for the local maxima of the occupancy map, is required. However, this additional step would introduce additional computational burden on the target localization algorithm. Instead of searching to find accurate threshold to convert the gray-scale map to binary map, in certainty map, threshold selection is trivial because the regions, which cannot be cleared by any sensor, i.e., a threshold of 1, give target locations, as shown in Fig. 2.7g.

To summarize, the certainty map approach presents better performance in target localization by showing fewer false alarms without the post-processing step of searching for the accurate threshold value for each individual target. In addition to the performance superiority, the certainty map approach can be carried out in either the centralized or the distributed fashion to fuse the information from each sensor node. By using the dynamic itinerary scheme in distributed processing, the certainty map can be progressively clarified until the confidence level is met by involving limited number of sensor nodes. However, the occupancy map approach can be implemented only in the centralized processing fashion because the nature of intersection requires to fuse all the information from every sensor node.

2.7 Analytical Study on Communication and Energy Efficiency

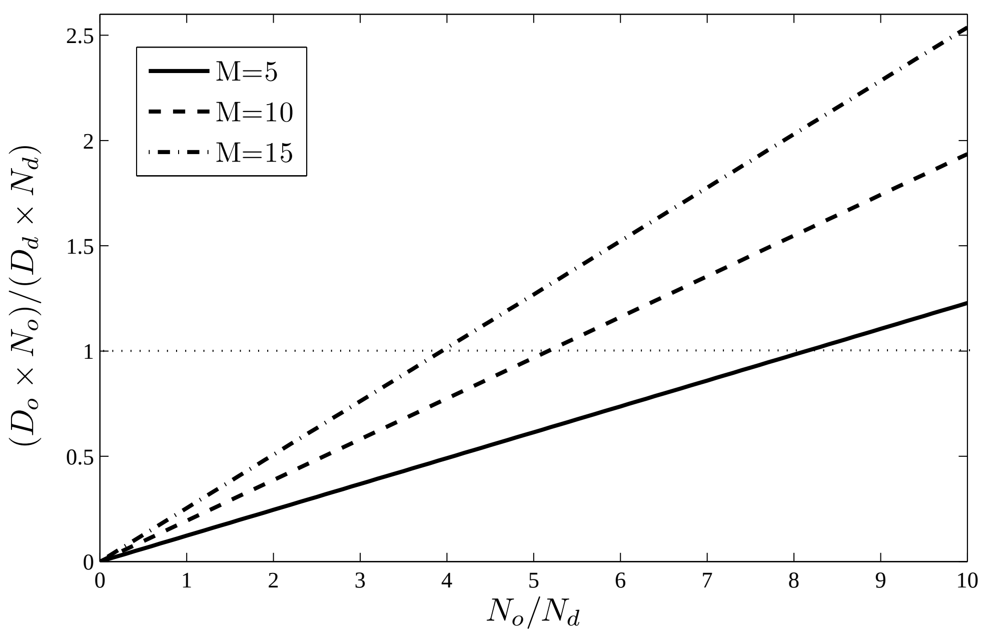
The advantage of using dynamic itinerary in progressive processing is that less number of nodes might be involved in the integration process and that the cleared areas from the certainty map monotonically decreases between integrations. However, dynamic itinerary also introduces overhead by having to communicate both $|f(\mathbf{v}_{s_i})|$ and $\mathbf{v}_{s_i}$ in several integrations, while in centralized processing only $\mathbf{v}_{s_i}$ is communicated in one integration. The key is how many sensors are involved. Therefore, a comprehensive analytical study is necessary to evaluate communication efficiency between different integration schemes, i.e., centralized integration, progressive integration with fixed, random, and dynamic itinerary, in reaching a final certainty map. We first perform analytical study on the number of bits transmitted within the network which is a good indicator of energy consumption and bandwidth usage. Then, we calculate the total energy consumption during the data transmission through different integration schemes. We use a subscript “d” to indicate *dynamic* itinerary and “o” for *other* integration schemes.

2.7.1 Analytical Study on the Number of Bits Transmitted

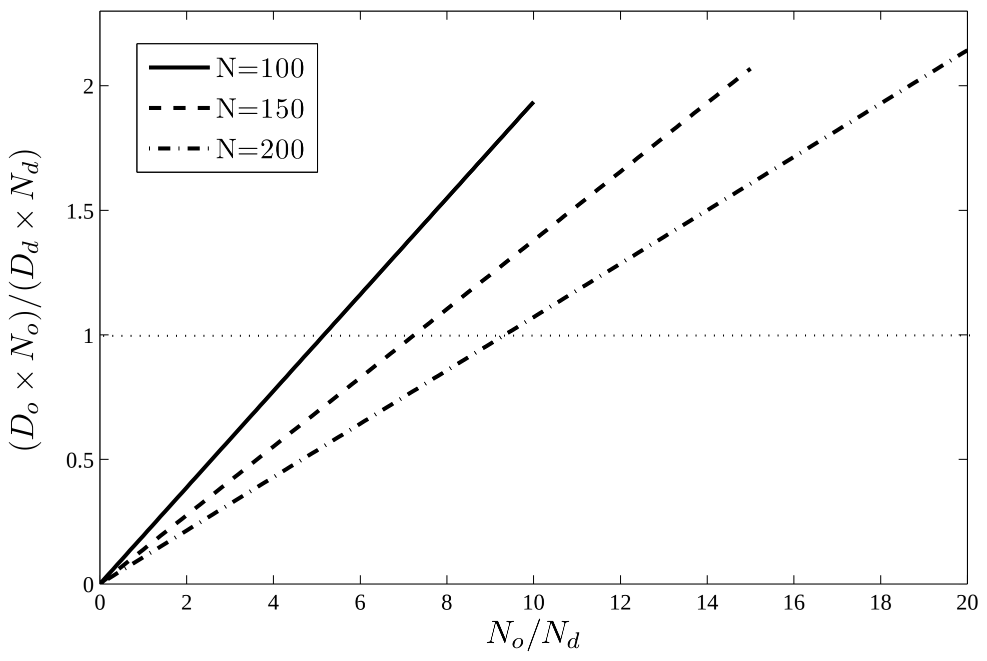
Assume there are N sensors in the network, among which N_d sensors are involved in the target localization process if using dynamic itinerary, and N_o sensors involved if using fixed or random itinerary (note that when centralized integration is adopted, N_o=N). Assume there are M targets within the surveillance area of the N sensors, the worst case scenario for data transmission is that there are $M+1$ non-occupied regions in local certainty map which can be expressed as a data vector according to the notations defined in Section 2.3, $\mathbf{v}_{s_i} = [x_i, y_i, \varphi_{i,1}, \psi_{i,1}, \ldots, , \varphi_{i,M+1}, \psi_{i,M+1}]$. The size of the data vector is $D_o = 32 \times (2(M+1)+2) = 32 \times (4+2M)$ bits per transmission, assuming each element in the vector is represented as a 32-bit floating point. In dynamic itinerary, besides the transmission of $\mathbf{v}_{s_i}$, the total size of area that can be cleared from the current certainty map by sensor node $s_i \in S$, $|f(\mathbf{v}_{s_i})|$, is also required to be propagated which is an extra 32-bit floating point for each sensor in each iteration. Then, the total size of the data vector in dynamic itinerary is $D_d = 32 \times (4+2M) + 32 \times N$ bits per integration. Therefore, the total amount of bits transmitted in dynamic itinerary, $D_d \times N_d$, is less than that in other integration schemes, $D_o \times N_o$, only if the following inequation holds:

$$\frac{32 \times (4+2M+N) \times N_d}{32 \times (4+2M) \times N_o} = (1 + \frac{N}{4+2M}) \times \frac{N_d}{N_o} < 1 \tag{2.4}$$

Fig. 2.8a and Fig. 2.8b illustrate the relationship between the ratio of transmitted data, $\frac{D_o \times N_o}{D_d \times N_d}$, and the ratio of involved sensor nodes, N_o/N_d. We observe that when N_o/N_d is greater than certain number, dynamic itinerary would require less amount of data transmission indicating less energy consumption and less bandwidth occupation than other integration schemes. We also observe that the dynamic itinerary becomes more advantageous if the number of targets, M, is increased with a fixed N, as shown



(a)



(b)

Figure 2.8: Comparison of the total amount of data transmitted in VSN by using dynamic itinerary and other integration schemes for (a) different number of targets, M where $N = 100$ and (b) different total number of deployed sensor nodes, N where $M = 10$. Note that $N_d = 10$.

in Fig. 2.8a, or the number of deployed sensor nodes, N, is decreased with a fixed M, as shown in Fig. 2.8b.

2.7.2 Efficiency on Energy Consumption

The energy consumption of a wireless network interface in broadcast and unicast transmission can be represented by a summation of linear equations for sending, receiving and discarding data [Feeney and Nilsson, 2001]. In this dissertation, we evaluate and compare the total energy consumption, $\mathbb{E}$ during the transmission of certainty maps in dynamic itinerary and other integration schemes by using the linear energy consumption approach which can be described as follows:

$$\mathbb{E} = c \times D + \xi \tag{2.5}$$

where c is the energy cost per byte, D is the data size and ξ is the fixed cost for different communication modes. We assume that each sensor node is in the communication range of other nodes to avoid the hidden terminal problem and the energy consumption due to the bit error, package and channel loss is ignored to simplify the problem.

In broadcast transmission, a sensor node sends data and all sensor nodes within the transmission range receive it. However, in unicast transmission where point-to-point communication is used, only one sensor node sends data and only the destination node receives it while non-destination sensor nodes discard the data. Therefore, the certainty map can be transmitted within the network through broadcast transmission if using dynamic itinerary and unicast transmission if using other integration schemes. Note that when computing the energy consumption in unicast traffic, we consider not only the sending and receiving data, but also the discarding data at the non-destination sensor nodes because sensor node consumes energy until determining it is non-destination. In addition, the total energy consumption of sending, receiving and discarding data is proportional to total number of sensor nodes which send, receive

Table 2.1: Linear Model Energy Consumption Parameters for IEEE 802.11 11Mbps Wireless Network Card, [Feeney and Nilsson, 2001].

		$Energy cost per bit (c)$	$Fixed cost (\xi)$
		$\mu W \cdot sec/byte$	$\mu W \cdot sec$
	send	2.1	272
broadcast	*receive*	0.26	50
	send	0.48	431
unicast	*receive*	0.12	316
	discard	0.11	66

and discard the data. Let $\mathbb{E}_d^{send}$ and $\mathbb{E}_d^{recv}$ denote the energy consumption for each sensor node when sending and receiving D_d amount of data in each itinerary. The total amount of energy consumption in dynamic itinerary is then

$$\mathbb{E}_d = N_d \times [\mathbb{E}_d^{send} + (N-1) \times \mathbb{E}_d^{recv}] \tag{2.6}$$

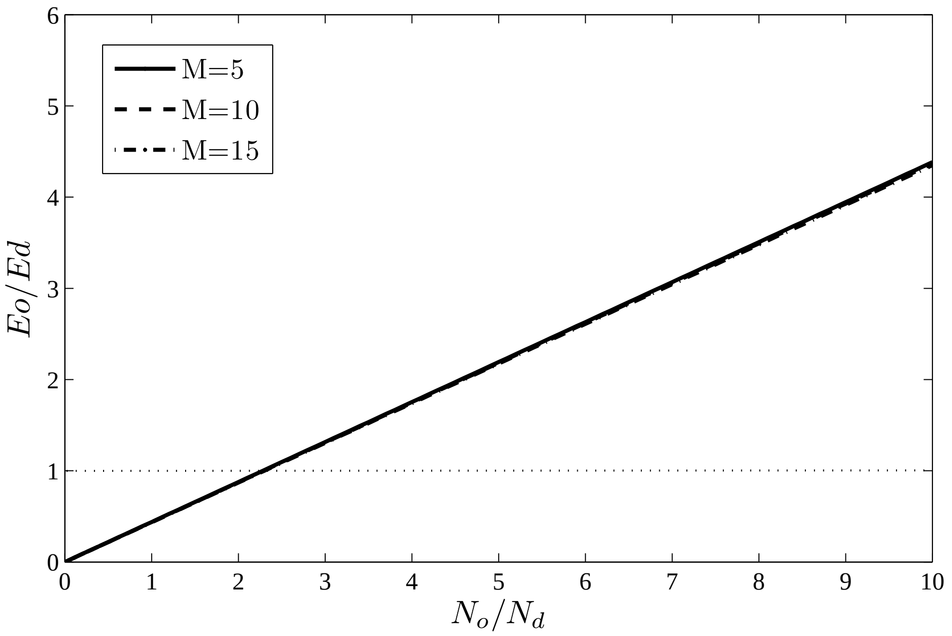
where N_d is the number of sensors involved in dynamic iteration and in each iteration only one sensor sends data but $(N-1)$ sensors receive data.

Let $\mathbb{E}_o^{send}$, $\mathbb{E}_o^{recv}$ and $\mathbb{E}_o^{disc}$ denote the energy consumption for each sensor node when sending, receiving, and discarding D_o amount of data. The total amount of energy consumed in other integration schemes is then

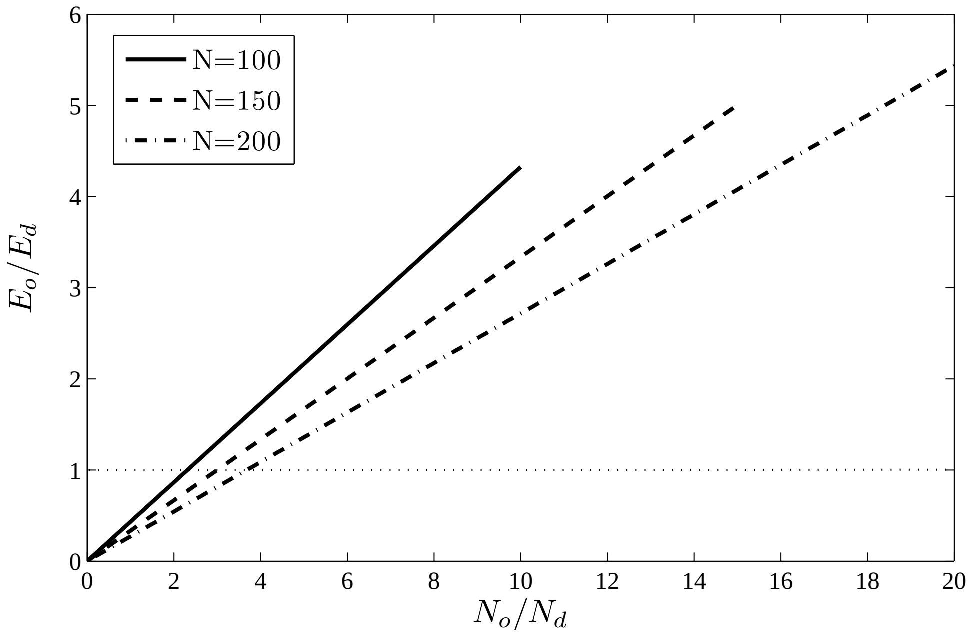
$$\mathbb{E}_o = N_o \times [\mathbb{E}_o^{send} + \mathbb{E}_o^{recv} + (N-2) \times \mathbb{E}_o^{disc}] \tag{2.7}$$

where in each iteration, only one sensor sends data, one sensor receives data, but $(N-2)$ sensors discard the data.

Table 2.1 shows the linear energy consumption parameters, c and ξ to send/ receive/ discard a byte in broadcast and unicast traffic of an IEEE 802.11 11Mbps wireless network card [Feeney and Nilsson, 2001]. By using these parameters, a more quantitative representation of the energy consumption in dynamic and other integration schemes can be derived. Fig. 2.9a-2.9b show the relationship between $\mathbb{E}_o/\mathbb{E}_d$ and N_o/N_d with respect to different numbers of deployed targets, M and



(a)



(b)

Figure 2.9: Comparison of the total amount of energy consumed in VSN by using dynamic itinerary and other integration schemes for (a) different number of targets, M where $N = 100$ and (b) different total number of deployed sensor nodes, N where $M = 10$. Note that $N_d = 10$.

sensor node, N. We observe that $\mathbb{E}_o/\mathbb{E}_d$ is greater than 1, that is, $\mathbb{E}_o > \mathbb{E}_d$, when N_o/N_d is greater than certain value. In addition, we observe similar trend as in Fig. 2.8a- 2.8b, where the increment in the total number of targets, M and the decrement in the total number of deployed sensor nodes, N make the usage of dynamic itinerary more advantageous. However, the effect of M is far less compared to the effect of N. This is especially true when $N \gg M$.

2.8 Experiments and Results

In this section, we evaluate the performance of the target localization algorithm with both simulated and real experiments for various integration scenarios, including centralized integration and progressive integration with random and dynamic itineraries. The evaluation is conducted from four perspectives, that is, effect of the node density, effect of the target density, effect of the number of consecutive stopping flags as stopping criterion, and effect of the voting threshold. We use two metrics to present the results, the number of detected targets as compared to the true number of targets and the number of involved sensor nodes. We use the following default values for the two parameters introduced in the algorithm:

- δ, the threshold to determine if a sensor node holds adequate additional clarification information, is set to be 10% of the object size. When below this threshold, a stopping flag is raised.
- The default number of consecutive stopping flags is set to be 10% of the total number of nodes in the network. If the number of consecutive flags is reached, then the itinerary is stopped and a final certainty map is generated. This parameter is used in fixed and random itineraries.

2.8.1 Experiments using Simulation

In our simulation, square-shaped targets of uniform size are deployed on a 2D sensing field, and infinitely small-size sensor nodes with uniform FOV and focal length are directed horizontally on the sensing field. The location of each sensor and target is randomly generated in the simulation by assuming there is no overlapping between the targets and sensors. For the deployment of large-scale sensor networks, random deployment has been a viable choice for its easiness in operation and ability to rapidly form the coverage network. Random deployment is especially advantageous when sensors need to be placed in harsh or hostile environment [Hynes et al., 2004]. In each set of experiments presented below, targets and sensors are deployed randomly for four times and the results are averaged. To insert more randomness to random itinerary, 100 different random itineraries are generated in each deployment scenario and the results are averaged as well.

In all the simulations, we assume each node is accurately calibrated, synchronized with each other, captures and extracts the targets from their backgrounds. Also, each node is able to find its location and orientation by using a positioning system (such as GPS) and a digital compass, respectively. Following is the setup of some typical parameters: The 2D sensing field is 30m $\times$ 20m large. The size of each target is 0.5m $\times$ 0.5m. The uniform sensing range of nodes is 20m in length and $45°$ in angle. The orientation of each sensor node is a floating point number randomly generated in $[0°, 360°]$. Each node is able to communicate with each other. Fig. 2.10 illustrates a sample random deployment of 20 targets, represented as squares, and 200 cameras, represented as points.

After each deployment of targets and cameras, the simulation software generates the corresponding foreground image of targets in the FOV of each sensor node. Each sensor node then computes the 2D visual cones of the non-occupied areas using the planar projection of 3D cones and generates local certainty map as described in Sec. 2.3. By using the different integration processes, the certainty map is

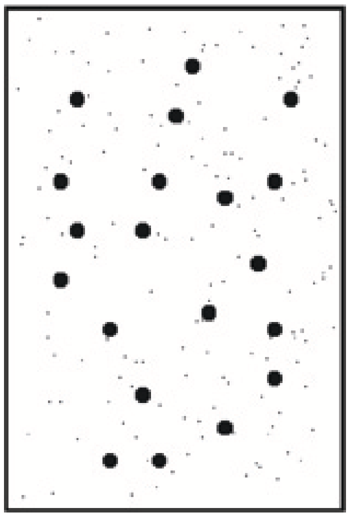


Figure 2.10: Simulation setup with 20 targets and 200 sensor nodes.

progressively clarified until the confidence level is met and targets are located at the remaining uncertain regions in the map.

We conduct two sets of simulated experiments to study the effect of sensor density and effect of target density on the performance of target localization with certainty map.

Effect of Node Density

In this set of experiments, centralized and progressive processing with random and dynamic itineraries are tested to show the performance of the algorithm by deploying different numbers of sensor nodes versus a fixed number of targets in the 2D sensing field. In the simulation, 10 targets and different numbers of sensor nodes are randomly deployed. The final version of the certainty map derived using different integration approaches is shown in Fig. 2.11 where the left column is the results from centralized processing, the middle column is from dynamic itinerary, and the right column is from random itinerary.

Fig. 2.12 illustrates the total number of detected targets versus different numbers of deployed sensor nodes as a performance metric. We observe that dynamic itinerary and centralized processing show almost identical performance in target detection and

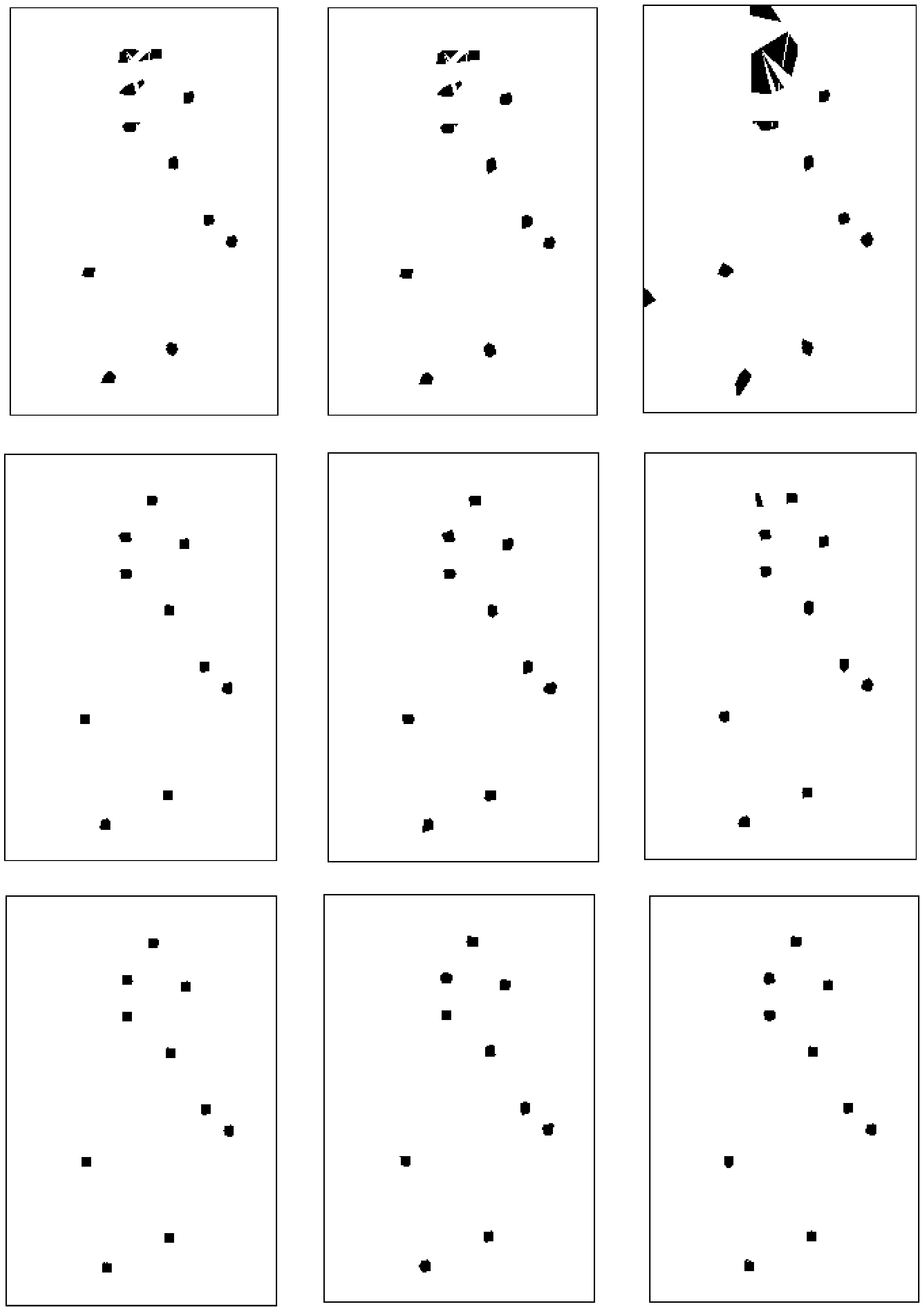


Figure 2.11: The final version of certainty map when detecting 10 deployed targets using centralized processing (left column), dynamic itinerary (middle column), random itinerary (right column) with 100 (top row), 200 (middle row), 300 (bottom row) cameras, respectively.

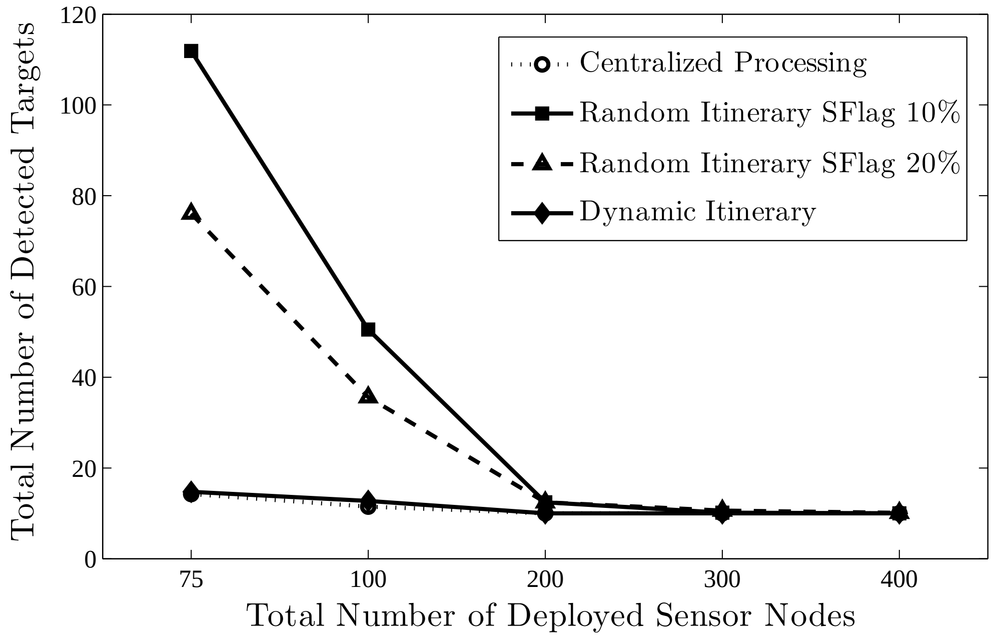


Figure 2.12: Total number of detected targets in localization for different number of deployed nodes. Note that the true number of targets is 10.

better performance than random itinerary with less false alarms, when total number of deployed cameras is less than 300. To have better performance in random itinerary, an increase in total number of deployed cameras is required. However, this increment also implies more financial cost. Another way to have more accurate results in random itinerary is to increase the number of consecutive stopping flags such that more sensor nodes are involved in the integration process. As shown in Fig. 2.12, to set the number of consecutive stopping flags to 20% of total number of sensor nodes has less false alarms than to set it to 10%.

The other important performance metric in VSN is the total number of involved sensor nodes in the integration process. When detection accuracy is not affected much, having less number of nodes involved in the integration process saves both energy and communication bandwidth. In Fig. 2.13, the total number of involved sensor nodes is shown for different integration mechanisms.

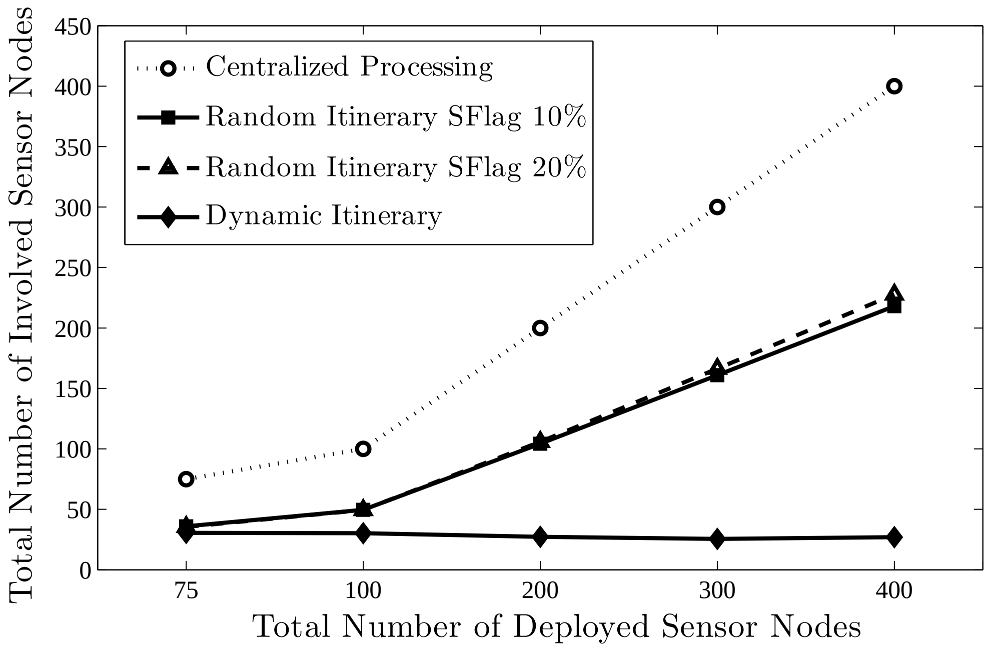


Figure 2.13: Total number of involved sensor nodes in localization for different number of deployed nodes. Note that the true number of targets is 10.

We observe that to have the same accuracy in target detection, dynamic itinerary uses limited number of cameras in the integration process which is less than that used in other integration schemes. For example, when total number of deployed cameras is 300, although all integration processes can result in accurate detection and localization of targets, the total number of involved cameras, as shown in Fig. 2.13, is much less using dynamic itinerary compared to the other schemes. In general, an increase in the total number of deployed cameras is needed to yield more accurate results. However, increment in total number of deployed cameras slightly decreases the number of used cameras in the dynamic itinerary because it is more probable to find nodes with bigger clarification regions in denser node deployments.

Effect of Target Density

In this set of experiments, the effect of deploying different numbers of targets versus fix number of cameras is studied. We randomly deploy 200 cameras and different

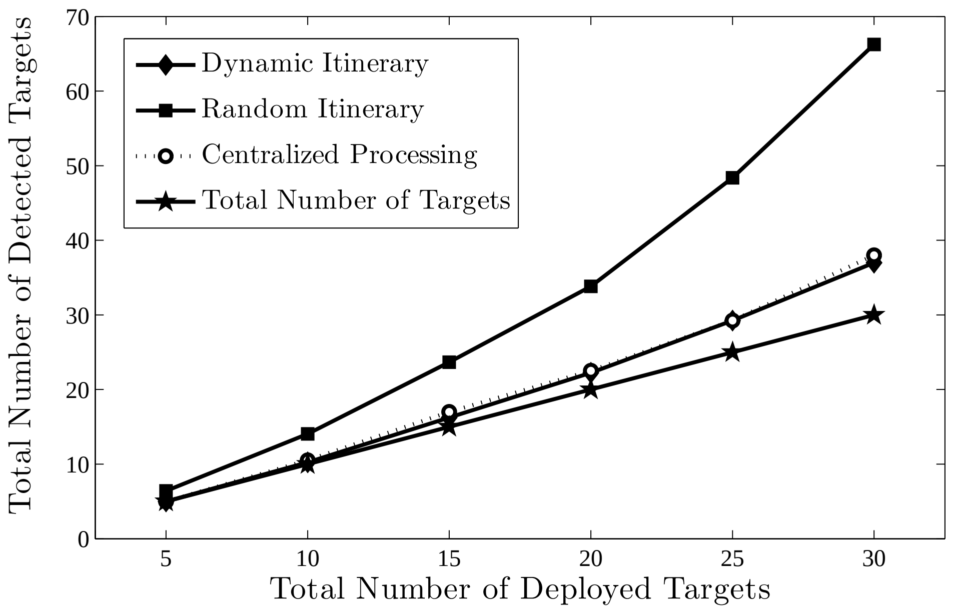


Figure 2.14: Total number of detected targets in localization for different numbers of deployed targets. Note that the number of deployed cameras is fixed at 200.

numbers of targets. Fig. 2.14 shows the number of detected targets for different numbers of deployed targets using different integration schemes.

We observe again that the target detection performance of dynamic itinerary and centralized processing is almost identical and better than random itinerary by showing less false alarms. While the density of targets increases, the false alarms of all integration schemes increase. However, the increment in false alarms in random itinerary is faster than others.

The total number of involved sensor nodes in centralized processing, random and dynamic itineraries is shown in Fig. 2.15. We observe that centralized processing uses all information available from sensor nodes so the number of used cameras is the same. An increment in total number of deployed targets increases the number of used cameras in dynamic itinerary because crowded targets reduce the size of the clearance area of each node. To cover the entire 2D sensing field, dynamic itinerary requires more cameras involved in the integration process. On the other hand, the number of used cameras in random itinerary slightly decreases because the decrease

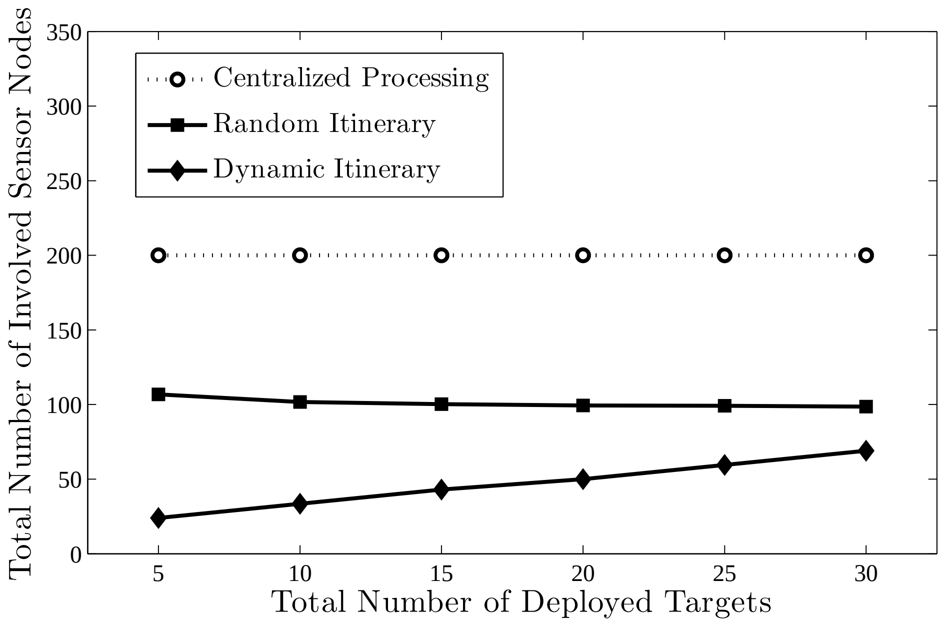


Figure 2.15: Total number of involved sensor nodes in localization for different numbers of deployed targets. Note that the number of deployed cameras is fixed at 200.

in the size of the clearance area of sensor nodes actually increases the number of sensor nodes with inadequate clearance information and makes it more possible to raise consecutive stopping flags that would stop the integration process before the certainty map could cover the entire sensing field.

2.8.2 Experiments using Real Data

Besides simulation, we also conduct three sets of experiments using real data captured from a visual sensor network composed of a number of mobile sensor platforms (MSPs). Each MSP is equipped with a 1GHz Mini-ITX motherboard which provides onboard processing capability to be able to apply image processing algorithms on captured images. Each platform is mounted with a Logitech QuickCam 4000 Pro webcam with a resolution of 640×480 pixels. To facilitate communication between the nodes and a workstation serving as the processing center, a wireless network is set up using the 802.11b wireless card. To supply the required energy, each MSP

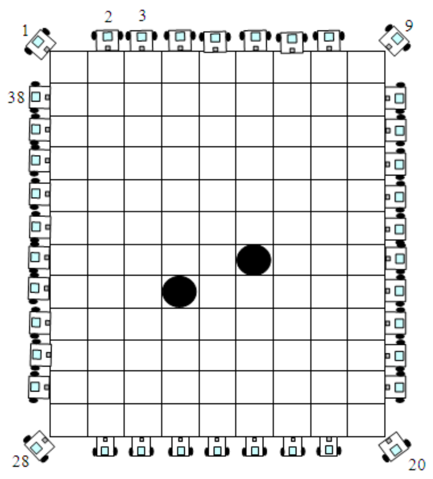


Figure 2.16: Experimental setup with 2 objects and 38 cameras.

is powered by a small 12V DC power supply. The operating system is Linux and C/C++ programming language can be used for application development.

Simple Two Target Localization

In our first experimental setup shown in Fig. 2.16, two static objects are located in a 9 by 12 feet square area surrounded by 38 mobile sensor platforms (MSPs) with onboard processing, wireless communication and imaging capabilities. The objects are 1 foot in height and cameras are located at 6 inches in height. Four of the MSPs are located at the corners of the experimental area and oriented toward the center of the area. The rest of the MSPs are located 1 foot apart and oriented to the room with perpendicular angle with the sides of the area. In this experimental setup, each foot square area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution.

Images are captured by each MSP with different field of views as shown in Fig. 2.17(top). Background subtraction is first performed to obtain the foreground objects shown in Fig. 2.17(middle). 2D visual cones of the non-occupied areas are computed by using the planer projection, as shown in Fig. 2.17(bottom).

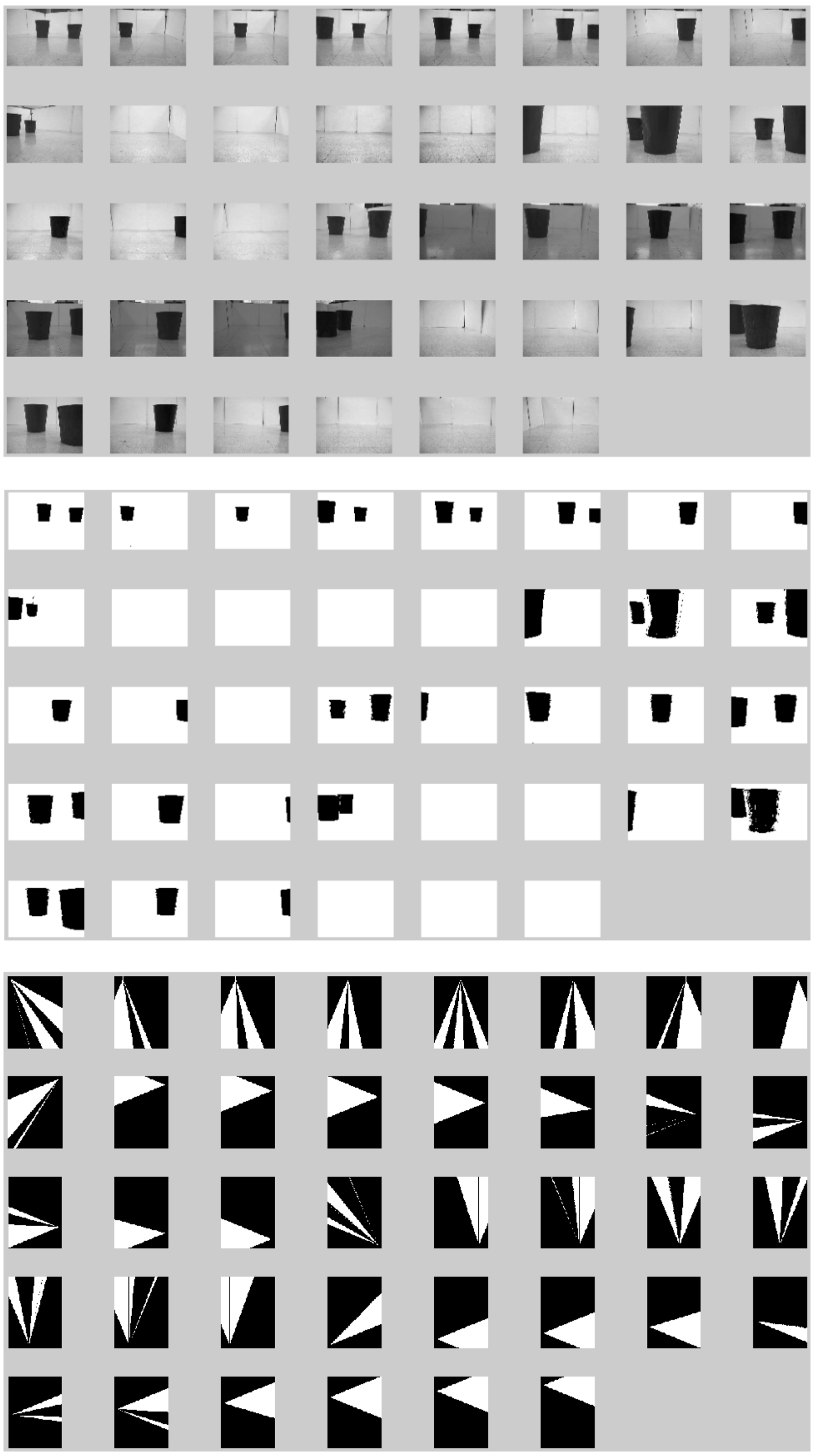


Figure 2.17: (top) Images captured by cameras 1 to 38. (middle) Foreground images from cameras 1 to 38. (bottom) Non-occluded 2D visual cones of each camera.

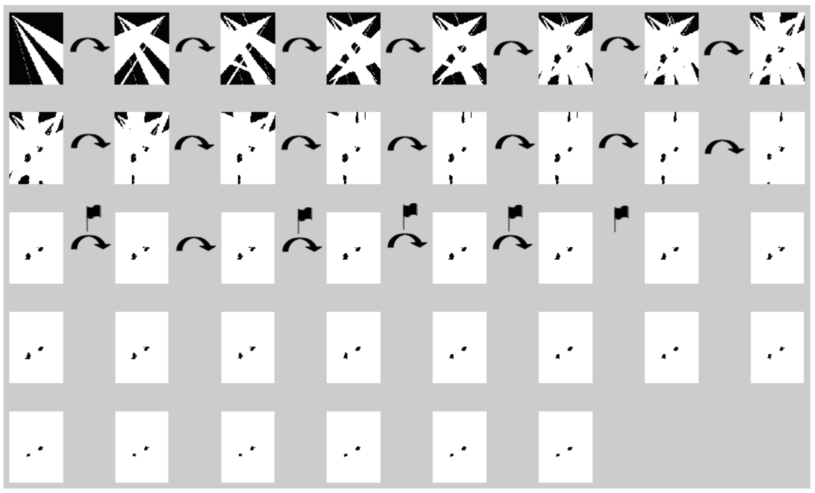
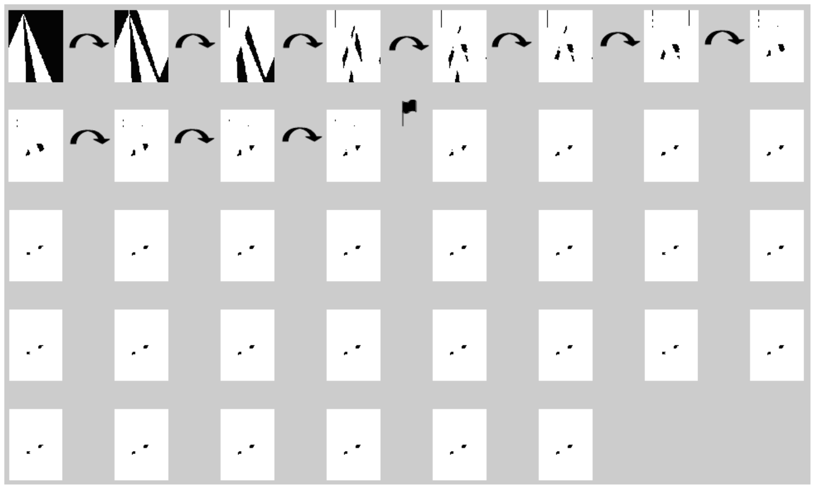
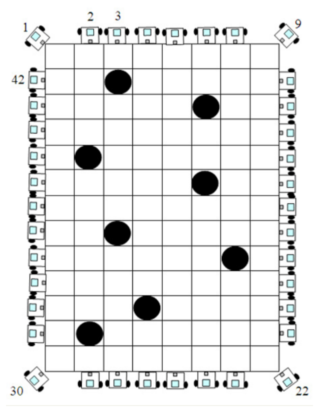
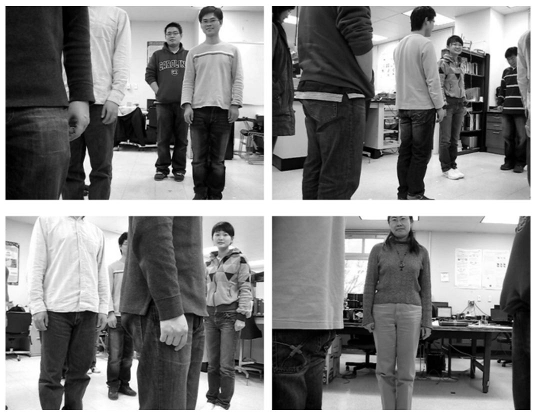



Figure 2.18: (top) The clarification of the certainty map using the fix itinerary. (bottom) The clarification of the certainty map by using the dynamic itinerary.

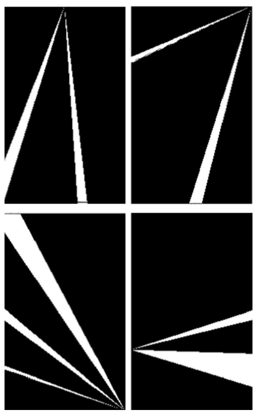
Fix and dynamic itineraries are tested to show the effect of the itinerary selection on the performance of the algorithm. Fig. 2.18(top) shows the intermediate certainty maps that is progressively improved from node to node following a fixed itinerary, discussed in Sec. 2.5.1. After transmitting the certainty map to the 19^{th} MSP, the additional clearance from the certainty map is less than the threshold, δ, therefore, the corresponding MSP raises a stopping flag to be carried to others about its inadequate contribution to the certainty map. When there have been four consecutive flags raised for robustness purpose, the itinerary would stop in order to save energy. In the fix itinerary, the itinerary stopped at the 22^{nd} MSP.



(a)



(b)



(c)

Figure 2.19: (a) Experimental setup with 8 people and 42 cameras, (b) Images captured by $5^{th}, 9^{th}, 22^{nd}, 34^{th}$ cameras, (c) Corresponding non-occupied 2D visual cones of the cameras in (b).

For dynamic itinerary, since the amount of cleared areas monotonically decreases from sensor to sensor, we do not have to wait until getting four consecutive flags to stop the integration. The itinerary is stopped as soon as the size of the additional clearance region is less than the threshold, δ. In Fig. 2.18(bottom), the progress of the certainty map in dynamic itinerary is shown. The itinerary is stopped at the 12^{th} sensor node. We observe that the dynamic itinerary scheme generates the best route to clarify the certainty map by using the least amount of sensor nodes.

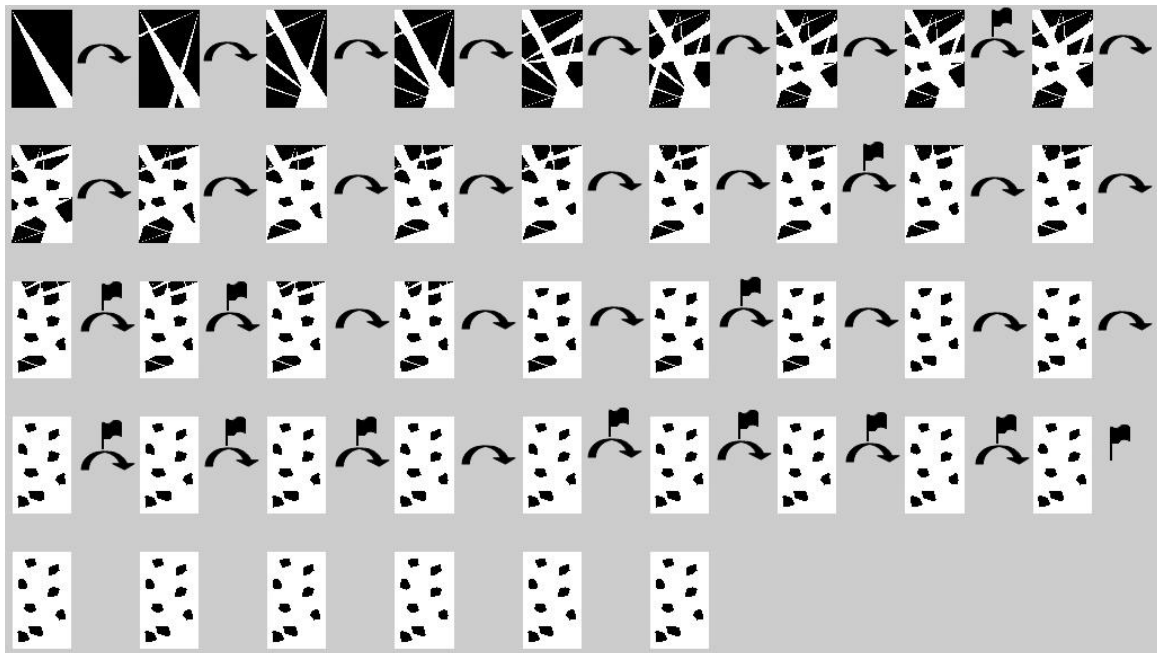
Multiple People Localization in Crowded Scene

In the second experimental setup shown in Fig. 2.19a, eight people stand in an 8 by 13 feet square area surrounded by 42 mobile sensor platforms (MSPs). People, as targets, are 5 to 6 feet in height and cameras are located at 3 feet in height. Four of the MSPs are located at the corners of the experimental area and oriented toward the center of the area. The rest of the MSPs are located 1 foot apart and oriented to the room with perpendicular angle with the sides of the area. For the 8 by 13 feet square surveillance area, 42 MSPs are deployed. In this experimental setup, each foot square area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution. Noted that although 42 MSPs are deployed to form the VSN, not all of them are used to localize the 8 targets. The effect of the number of sensors deployed on the localization performance has been thoroughly evaluated through simulation (See Sec. 2.8.1). In this experiment, our main purpose is to show the certainty map-based algorithm works in a real-world setup. By deploying a sensor network with denser sensor nodes than needed, the factor related to sensing would not be an issue, so that we can focus on localization performance.

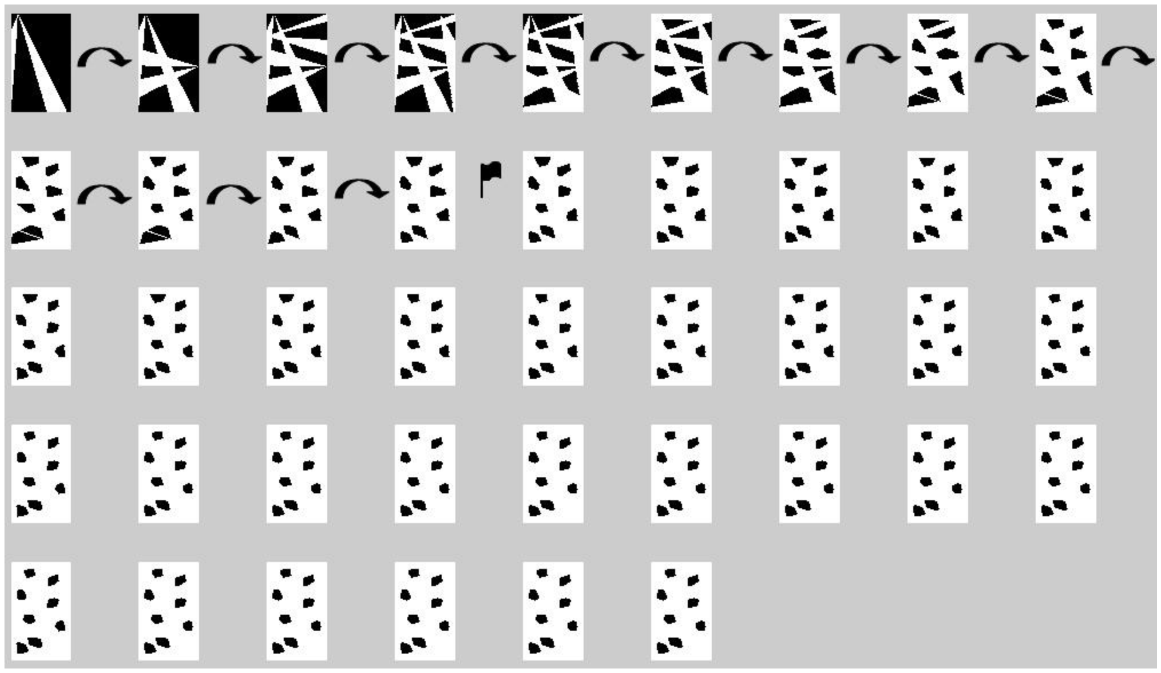
We first evaluate the performance of the proposed algorithm on its capability in localizing crowded targets, e.g., to identify eight targets within an 8×13 square feet area. Images are captured by each MSP with different field of views as shown in Fig. 2.19b and background subtraction is first performed to obtain the foreground objects. The 2D visual cones of the non-occupied areas are computed by using the planar projection, as shown in Fig. 2.19c.

Fig. 2.20a shows the intermediate certainty maps that are progressively improved from node to node following a fixed itinerary, discussed in Sec. 2.5.1. After transmitting the certainty map to the 32^{nd} MSP, the additional clearance from the certainty map is less than the threshold, δ (30% of the size of target), so the corresponding MSP raises a stopping flag to be carried to others about its inadequate contribution to the certainty map. When there have been five consecutive flags (10% of the total number of cameras deployed) raised for robustness purpose, the itinerary would stop in order to save energy. In the fixed itinerary, the itinerary stopped at the 36^{th} MSP.

For dynamic itinerary, since the amount of cleared areas monotonically decreases from sensor to sensor, we do not have to wait until getting five consecutive flags to stop the integration. The itinerary is stopped as soon as the size of the additional clearance region is less than the threshold, δ. In Fig. 2.20b, the progress of the certainty map in dynamic itinerary is shown. The itinerary is stopped at the 13^{th}



(a)



(b)

Figure 2.20: The clarification of the certainty map using (a) the fixed itinerary and (b) the dynamic itinerary.

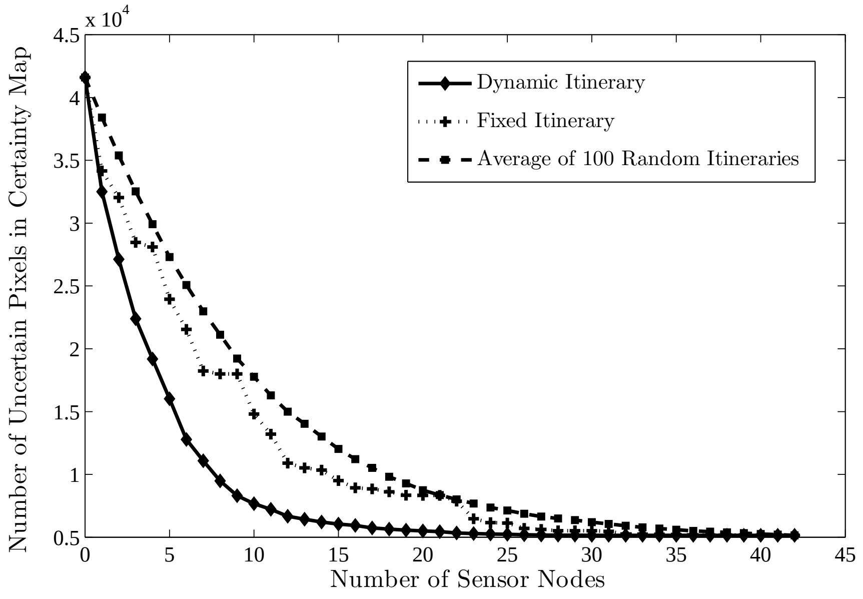


Figure 2.21: Total number of uncertain pixels in certainty map for different itinerary selection.

sensor node. We observe that the dynamic itinerary scheme generates the best route to clarify the certainty map by using the least amount of sensor nodes.

As we observe from Figs. 2.20a and 2.20b, both integration schemes are able to successfully localize the target. Therefore, instead of using the detected number of targets as a performance metric, we study the number of uncertain pixels in the certainty map as a metric that shows how accurate the localization algorithm performs. The results are displayed in Fig. 2.21. We see that the dynamic itinerary shows the best performance by clarifying the certainty map at the fastest rate. In addition, fixed selection of the itinerary clarifies the certainty map faster than random itinerary and gives better result than random itinerary in general.

Multiple People Counting in a Video Sequence

In the third experimental setup shown in Fig. 2.22, different numbers of people enter a 22 by 36 feet square sensing field surrounded by 24 cameras. People, as targets, are 5 to 6 feet in height and cameras are located at 4 feet in height. Four cameras

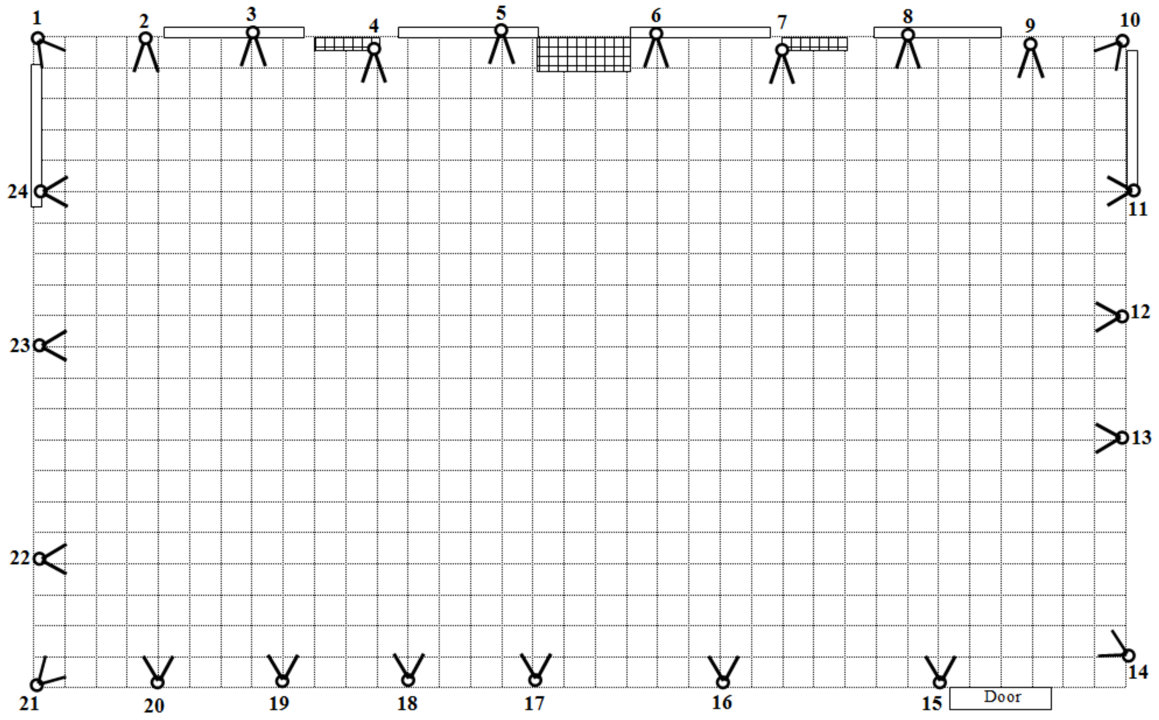
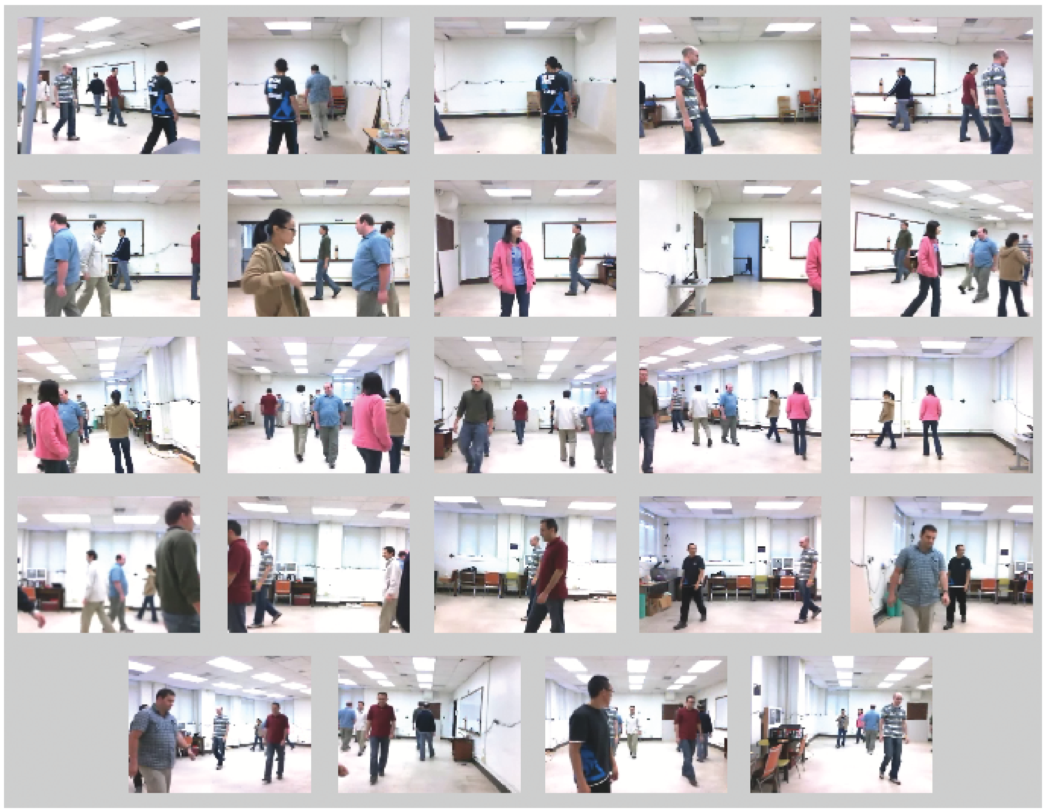


Figure 2.22: Total number of uncertain pixels in certainty map for different itinerary selection.

are located at the corners of the experimental area and oriented toward the center of the area. The rest of the cameras are located 3 to 7 feet apart from each other and oriented to the room with perpendicular angle to the sides of the area. In this experimental setup, each foot square area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution. In this experiment, our main purpose is to show the certainty map-based algorithm works in a real-world application, i.e., target counting in a video sequence.

We first evaluate the performance of the proposed algorithm through centralized and progressive integration on its capability in counting different numbers of targets in a video sequence where targets enter the sensing area and exit from the sensing area. Images are captured by each camera with different field of views as shown in Fig. 2.23a and background subtraction is first performed to obtain the foreground objects. The 2D visual cones of the non-occupied areas are computed by using the


(a)

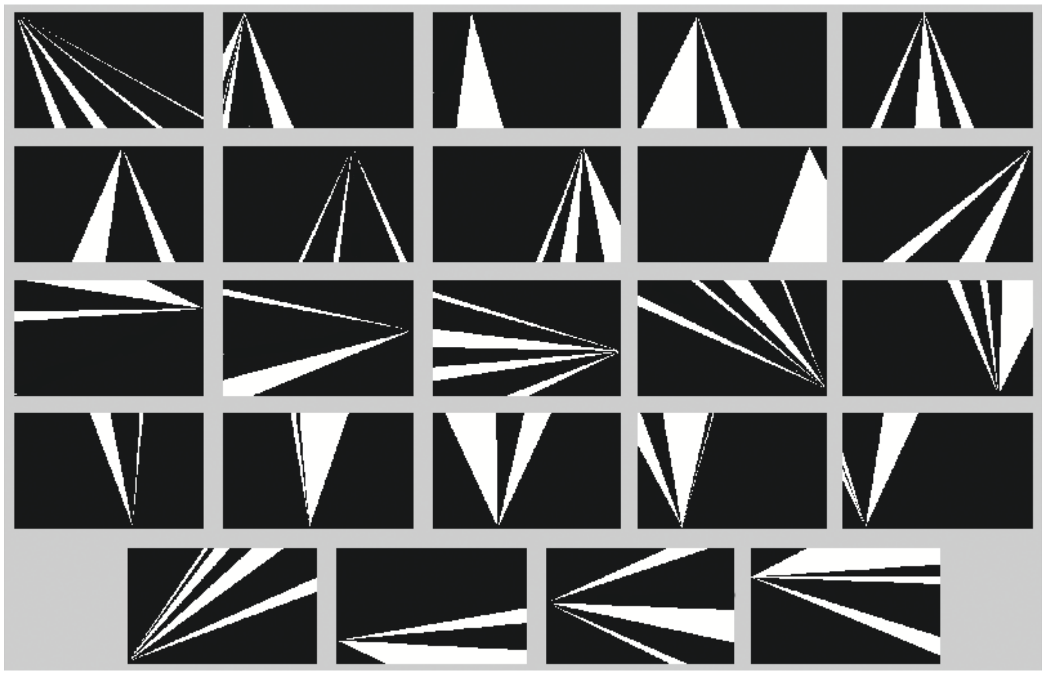

(b)

Figure 2.23: (a) Images captured by each camera at 1000th frame in the video sequence and (b) Corresponding non-occupied areas of the images in (a).

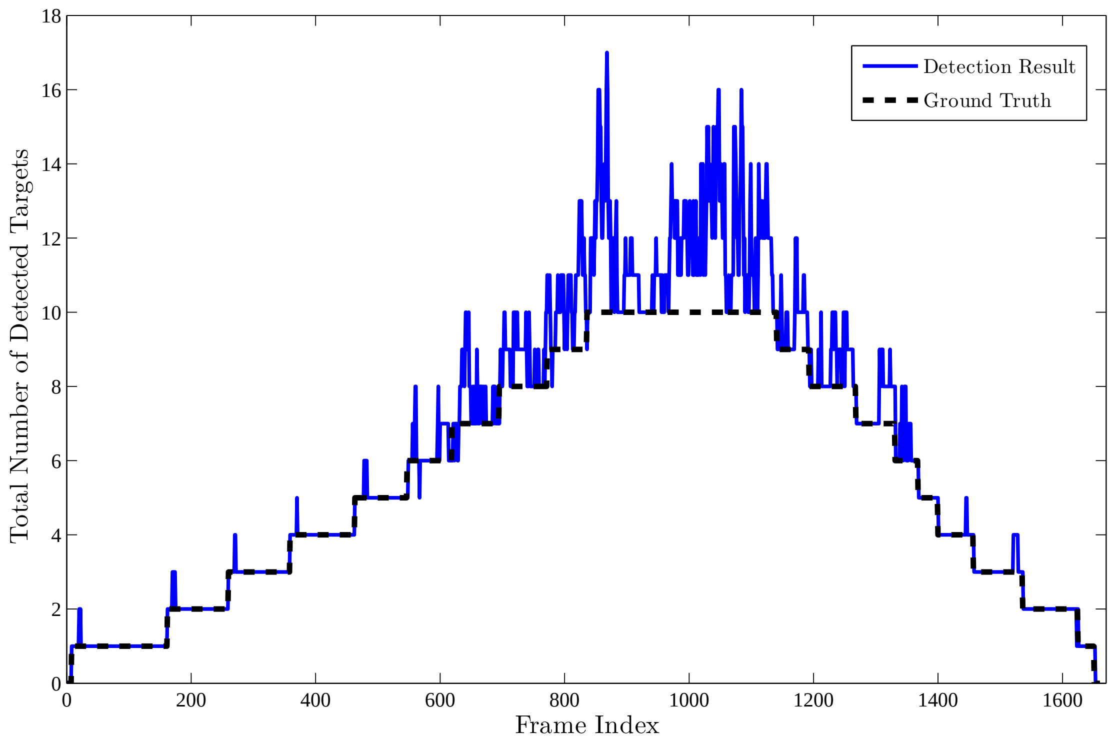


Figure 2.24: Total number of detected targets in the final certainty map for centralized integration and its ground truth values.

planar projection and local certainty map for each corresponding image computed, as shown in Fig. 2.23b.

The results of target detection and counting algorithm for each frame in the video sequence through the centralized integration is displayed in Fig. 2.24. We observe that if the total number of targets in the sensing area is less than certain value (i.e., six in this data), the performance of centralized integration is good and could detect all the targets in the sensing field. However, when the density of targets exceeds the certain value, the performance of the algorithm begins to reduce due to the visual occlusion among crowded targets. We observe that there exists some false alarms at frames from 600^{th} to 1350^{th} where there are more than six targets in the sensing field. The main reason for false alarms is that the density of cameras does not provide accurate coverage in the sensing field. In addition, the false alarms are at the frame

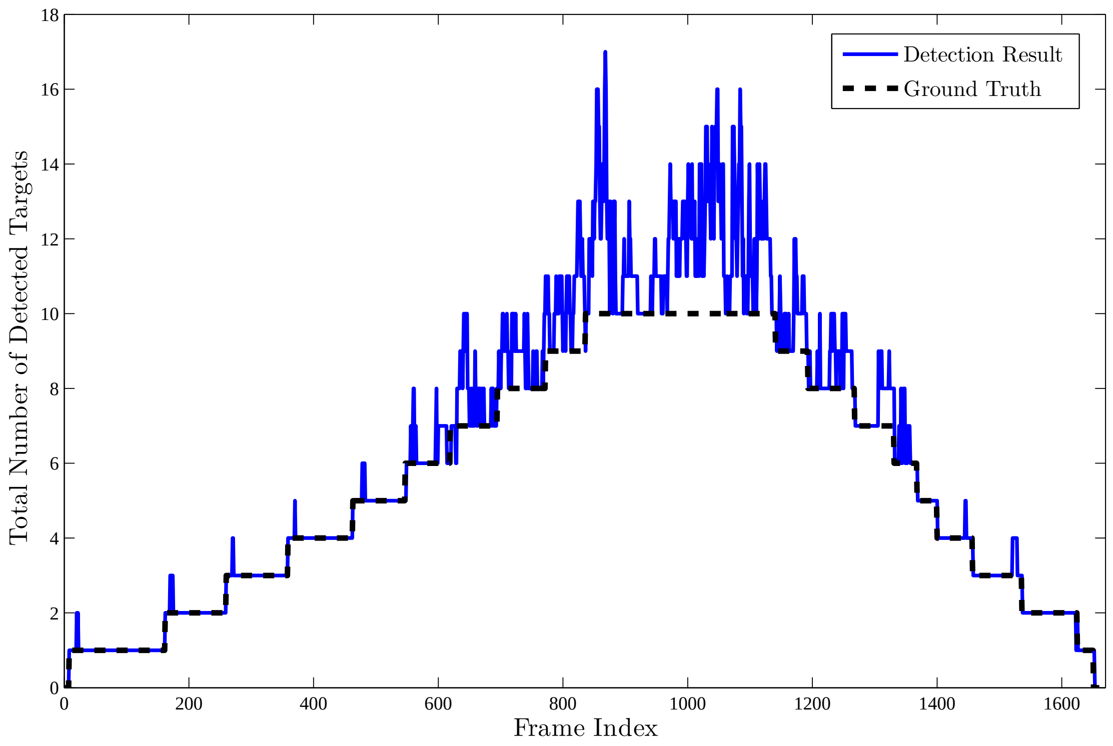


Figure 2.25: Total number of detected targets in the final certainty map and its ground truth by using progressive integration through dynamic itinerary .

where a new target enters the sensing field. This is caused by the change at the background close to the door area.

The same video sequence is also processed by using the progressive integration by clarifying certainty maps through a dynamic itinerary. Total number of detected targets in the final certainty map through dynamic itinerary and its ground truth values are shown in Fig. 2.25. We observe that the dynamic itinerary shows similar performance as the centralized integration but involving less number of visual sensor nodes as shown in Fig. 2.26. We also observe that in the progressive integration through dynamic itinerary, the total number of involved sensor nodes depends on the density of targets in the sensing field. When targets are not too dense, less number of sensor nodes are involved into progressive integration. An increment in the target

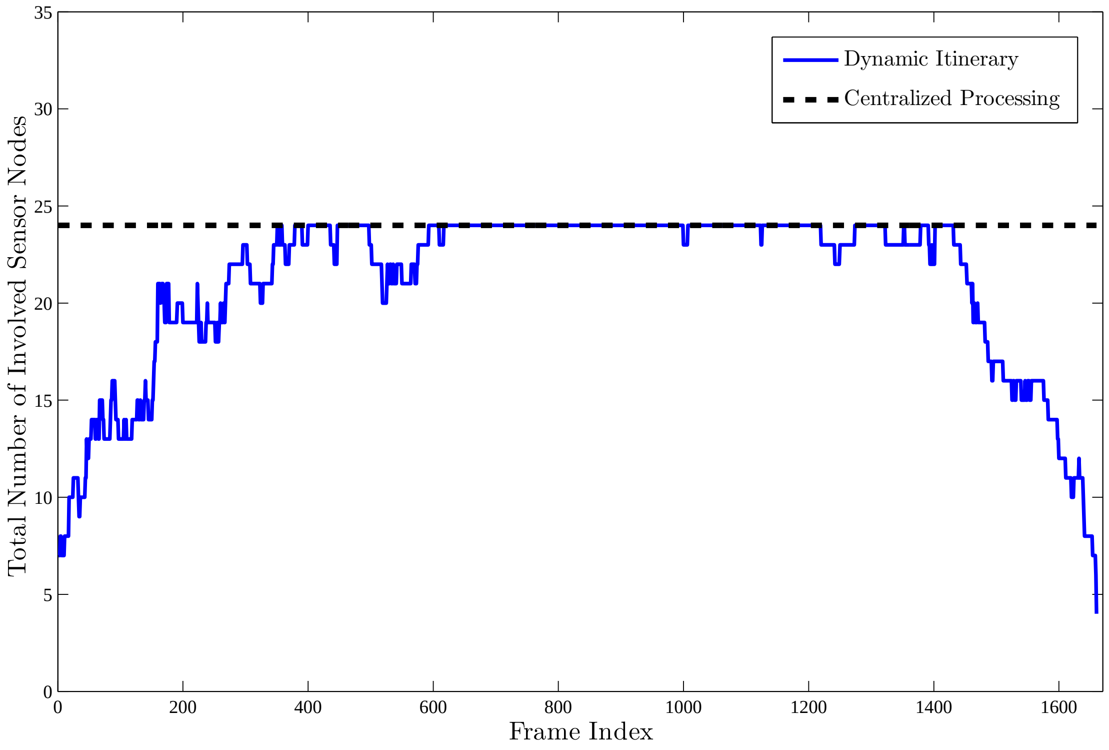


Figure 2.26: Total number of involved sensor node during dynamic itinerary and centralized integration.

density increases the total number of involved sensor node automatically. Therefore, dynamic itinerary provides an adaptive integration scheme.

The false alarms in too dense environments can be eliminated by using the information from previous frames. By assuming that targets cannot move faster than a specific speed, we can estimate a boundary for each target to be in the next frame. If a target appears outside its boundary, it is a false alarm due to the visual occlusion and can be eliminated. The results of target detection and counting algorithm for consecutive frames in the video sequence through the centralized integration by using the information from previous frames is displayed in Fig. 2.27. We observe that by using the time information in the video sequence helps to eliminate most of the false alarms in too dense environments compared to centralized integration without time information shown in Fig. 2.24.

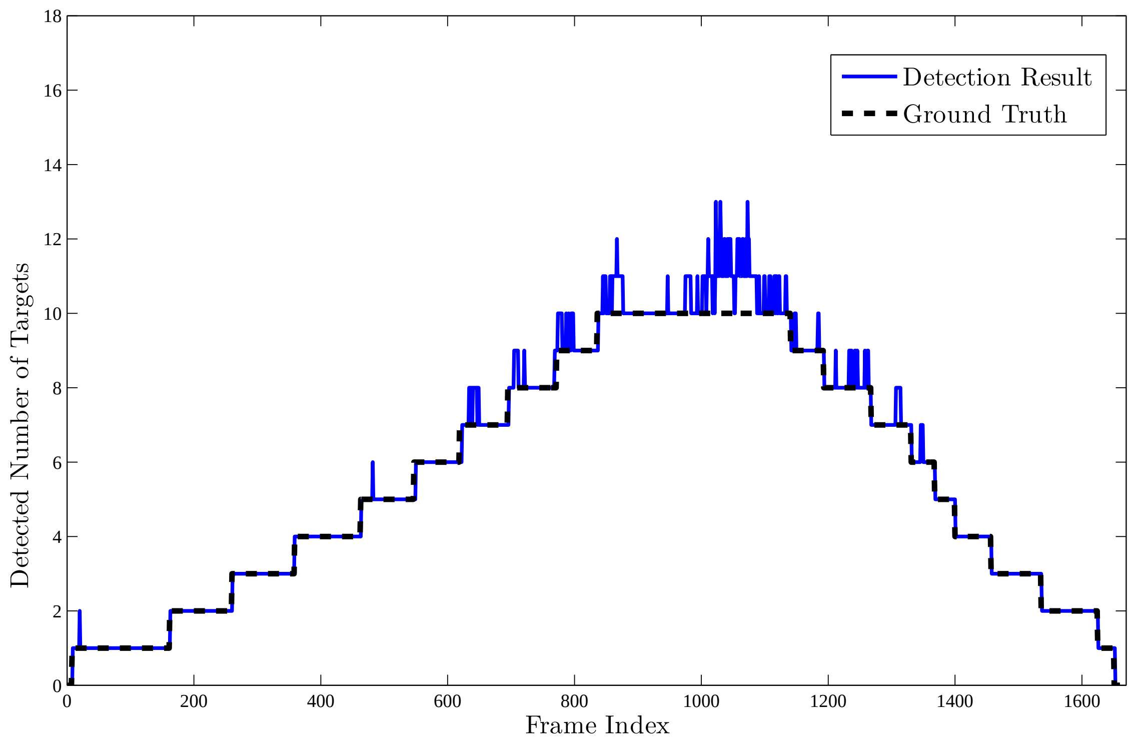


Figure 2.27: Total number of detected targets in the final certainty map for centralized integration by using time information and its ground truth values.

However, we also observe that exceeding the certain value of the target density, time information is also not adequate to eliminate some false alarms and the performance of the algorithm begins to reduce due to the too dense visual occlusion among crowded targets.

2.9 Summary

In this chapter, we presented an algorithm that can reliably detect the position of crowded targets in a distributed fashion in wireless VSNs under certain energy and bandwidth constraints. To achieve our goal, we designed a light-weight, energy-efficient, and robust solution where not only each camera node transmits a very limited amount of data but that a limited number of camera nodes is used to locate the targets. We identified and studied the non-occupied areas in the back-projected 2D cones, generated the certainty map of the non-existence of targets and defined a dynamic itinerary for certainty map integration where the entire map is progressively clarified from sensor to sensor until the confidence of the certainty map is satisfied. From both analytical study and experiments with simulated and real data, the results of the proposed progressive method showed effectiveness in detection accuracy as well as energy and bandwidth efficiency.

Chapter 3

Fault Tolerance, Detection and Correction in VSN

3.1 Introduction

Although the VSN is potentially powerful, its practical deployment could be hindered because of the limited capability of each sensor node (i.e., low processing speed and scarce power supply) and the low communication bandwidth among sensor nodes. In addition, although sensor nodes are usually deployed with initial rough calibration, the calibration of some nodes may not be accurate or may change during the network's lifetime because of external effects such as wind, seismic events, and precipitation [Clouqueur et al., 2004]. These sensor nodes can provide "faulty" information that would affect the accuracy of decision making. Therefore, the design of a practical solution for various VSN applications, e.g., calibration, clustering, target localization, etc., requires collaboration between sensor nodes not only to compensate for the limitations of each sensor node but also to improve the accuracy and robustness of the sensor network.

The work in this chapter was first published in Karakaya and Qi [2010].

In this chapter, we focus on the problem of target localization in VSNs with the existence of visual occlusion, partial information and a number of faulty nodes. We assume that the VSN is calibrated already after deployment using algorithms like [Devarajan and Radke, 2007]. However, because of some external effects or initial calibration inaccuracies, camera orientation would include certain degree of error. We design and develop a light-weight collaborative processing algorithm for target localization in crowds that would detect, tolerate these errors and further correct them before they cascade.

The chapter is organized as follows: Section 3.2 describes the background and related works in the fault tolerance study in sensor networks. In Section 3.3, information fusion and the fault tolerance technique in a centralized implementation are described. Section 3.4 describes the distributed implementation of fault-tolerant collaborative localization. Experimental results are presented in Section 3.5. Finally, we summarize the work in Chapter 3.6.

3.2 Background and Related Works

There exist many works related to robust surveillance and monitoring using sensor networks. Applications like target detection and localization [Clouqueur et al., 2004; Karakaya and Qi, 2009], target tracking [Krumm et al., 2000] and target counting [Yang et al., 2003], have been previously investigated.

In [Clouqueur et al., 2004], target detection with faulty sensors is based on taking the "mean" of local decisions obtained by each scalar sensor with extreme values dropped and results filtered by certain threshold. In [Ding et al., 2007], an exploratory work is introduced toward fault-tolerant target localization in SSNs by utilizing "median" to filter out extreme values and combining estimations of target locations from multiple epochs/iterations. In [Zou and Chakrabarty, 2004], an energy-aware target localization method is proposed for cluster-based SSNs by collecting event notification from sensors within the cluster and then executing a probabilistic localization algorithm to determine candidate nodes to be queried for target information.

Fault tolerance in VSNs is more challenging than in SSNs because of the unique features of cameras, including the existence of visual occlusion and the directional sensing characteristics with limited field of view. Accurate and periodic camera calibration has been considered one of the effective approaches to correct “faults”.

Camera calibration is an essential prerequisite and demanding task in VSNs. One of the widely used techniques to calibrate the cameras is to deploy additional markers to the environment, such as a bright red LED or a set of special patterns like checker boards, and to estimate the camera parameters by using the epipolar geometry between cameras. In [Poelman and Kanade, 1997] and [Devarajan and Radke, 2007], the extracted feature points from the captured images are used to estimate the camera parameters by utilizing matrix factorization and iterative belief propagation. However, these methods either require additional tools to deploy or the computational complexity of the feature extraction algorithm is too high to be deployed in real applications.

3.3 Fault-Tolerant Collaborative Localization

In Chapter 2, we assumed that the information obtained from each visual sensor node is accurate and trustworthy. However, in real world applications, this is seldom true. Sensor faults due to initial calibration error or external effects frequently occur that would deviate the initial calibration results. In this section, we first model the error incurred in camera orientations. Then, we present, voting, an ad-hoc algorithm for fault tolerant design of the centralized and distributed integration of local certainty maps and discuss the necessity of the normalized voting scheme. To prevent the system from providing faulty localization results, we propose two further steps of fault tolerance where faulty nodes are detected and errors in camera orientation corrected before they cascade.

3.3.1 Fault Model in Visual Sensors

We only study errors occurred in camera orientations. However, the proposed schemes can be generalized to handle inaccuracies in other camera parameters, e.g., camera location. The error in orientation of the faulty nodes can be modeled by a combination of two types of errors. First of all, the Gaussian noise models the initial calibration inaccuracy. Secondly, the Byzantine fault [Clouqueur et al., 2004] models the error generated by external effects where orientation becomes arbitrary and can be any value in $[0°, 360°]$. The Byzantine fault originates from Byzantine generals' problem [Lamport et al., 1982], an agreement problem to reach a unanimous decision in the presence of the traitors whether to attack enemy or to withdraw. Therefore, the camera orientation can be expressed as,

$$\theta_{s_i} = \theta^*_{s_i} + N_{s_i}(0, \sigma) + \delta_{s_i} \tag{3.1}$$

where θ_{s_i} and $\theta^*_{s_i}$ are the actual (or inaccurate) and ground truth (or calibrated) orientations of the i^{th} sensor node, s_i, respectively. $N_{s_i}(0, \sigma)$ is the Gaussian noise with zero-mean and standard deviation σ, and δ_{s_i} denotes the Byzantine fault in orientation.

3.3.2 Voting

Single camera node, which gives inaccurate information about the location of the targets, negatively impacts the performance and sometimes causes failure of the target localization algorithm. To obtain more accurate and robust results, certain degree of redundancy is necessary to tolerate the inaccurate information and failure of some visual sensor nodes.

Voting is one of the most commonly used multiple sensor fusion techniques to integrate individual sensor results [Klein, 1993]. In [Karakaya and Qi, 2009], we proposed to utilize the voting approach to target detection and counting to tolerate

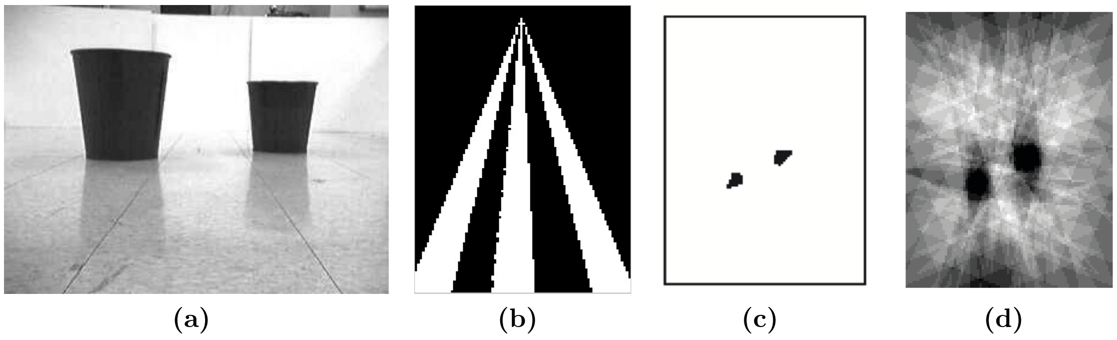


Figure 3.1: (a) Image captured by a camera and (b) its local certainty map. Data fusion by using (c) Binary certainty map and (d) Gray-scale certainty map.

potentially inaccurate information from visual sensor nodes where each sensor node has equal importance to contribute to the voting result. In the voting approach, if a visual sensor node declares the target non-existence at the specific location of the certainty map, the certainty value of that location is increased. Therefore, instead of a binary certainty map with 1 indicating 100% certainty of non-existence of targets, as shown in Fig. 3.1c, the voting approach generates a gray-scale certainty map, shown in Fig. 3.1d with non-zero regions indicating certain degree of the non-existence of targets. The higher the number of votes, the more certain it is.

A threshold value needs to be specified in the end to convert the gray-scale certainty map to binary for decision making purpose. To choose 1 as the threshold value means that there is no tolerance for failure of any sensor node. If one of the sensor nodes claims the non-existence of any object at any location in the certainty map, the algorithm believes it and clears the corresponding region from the certainty map as in the case of the binary certainty map. To be more robust to sensor failures, the threshold value can be chosen greater than one. For example, if the threshold value is selected as two, to clear a specific area from the certainty map, at least two sensors must declare the clearness of that region. Higher threshold value requires more sensor nodes to reach a consensus. Fig. 3.2(a-e) shows the final versions of gray-scale certainty map using the voting approach with different threshold values 2, 3, 4, 5, and 6, respectively.

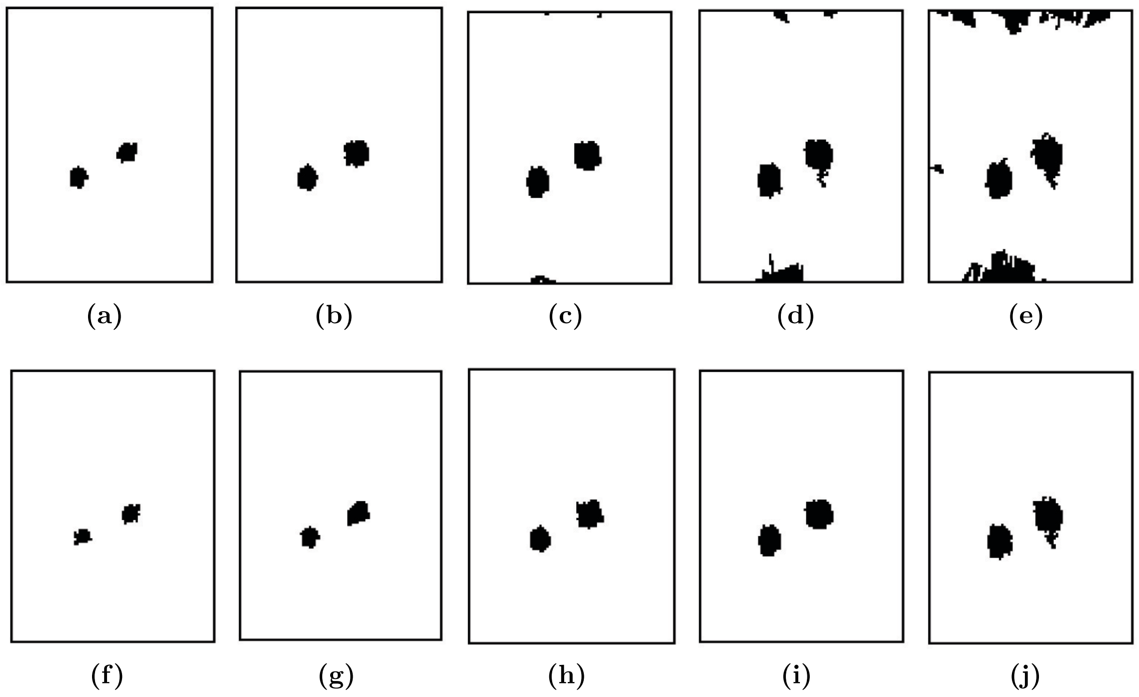


Figure 3.2: Final version of the certainty map. For (a) to (e), using voting with the threshold 2, 3, 4, 5, and 6, respectively. For (g) to (j), using normalized voting with the threshold 0.95, 0.9, 0.85, 0.8, and 0.75, respectively.

3.3.3 Normalized Voting

In the above mentioned voting scheme, we did not consider the coverage issue. That is, for an area that is densely covered, the number of votes would tend to be higher than areas with sparse coverage. Hence, a constant threshold would not reflect fairness of the decision. To this end, normalization should be applied such that coverage density is not an issue.

Here we propose the normalized voting algorithm for certainty map integration where the voting value of each grid pixel of the certainty map is normalized by its sensor coverage. At coordinate (x, y), suppose the grid pixel is covered by $C_{x,y}$ number of sensor nodes, then its normalized voting algorithm value, $V_{x,y}(S)$, is,

$$V_{x,y}(S) = \frac{1}{C_{x,y}} \sum_{i=1}^{N} f_{x,y}(\mathbf{v}_{s_i}) \tag{3.2}$$

where $f_{x,y}(\mathbf{v}_{s_i})$ denotes the local certainty map value and $S = \{s_1, s_2, \ldots, s_N\}$ is the set of sensor nodes in the network.

Fig. 3.2(f-j) shows the final version of the certainty map using normalized voting method with the different threshold values. We observe that the boundary effect, caused by less coverage at the boundary, in the voting approach, shown in Fig. 3.2(c-e), is suppressed to a great extent by using the normalized voting algorithm.

3.3.4 Fault Detection and Correction of Camera Orientation Errors

We assume the initial errors in camera orientation estimation can be tolerated by the proposed normalized voting algorithm. Therefore, the target locations generated by a collaborative effort of visual sensor nodes is trustworthy. However, if the initial errors do not get to be corrected in a timely fashion, together with potential Byzantine faults, errors can cascade that would eventually affect the accuracy of the location result. In Sec. 3.3.1, faults in camera orientations are modeled in two types, namely, Gaussian noise and Byzantine fault. Gaussian noise is due to the inaccurate camera orientation due to environmental noise or calibration error. The Byzantine fault model assumes that camera orientation might be an arbitrary value so provided information from that sensor node is arbitrary as well.

An algorithm is defined as *t-resilient* if it can continue to operate correctly until t out of $3t + 1$ processes fail [Clouqueur et al., 2004]. Similarly, the proposed normalized voting method is t-resilient if and only if it can correctly localize the targets in the certainty map when the ratio of the number of faulty nodes, $t_{x,y}$, to the sensor coverage, $C_{x,y}$, at a specific pixel location, (x, y), is less than one-third, $t_{x,y}/C_{x,y} < 1/3$.

To detect faulty nodes with inaccuracies in camera orientation estimation, we take into account both the actual image captured by each node and the final certainty map generated from the collaborative processing. We first utilize a so-called "generative

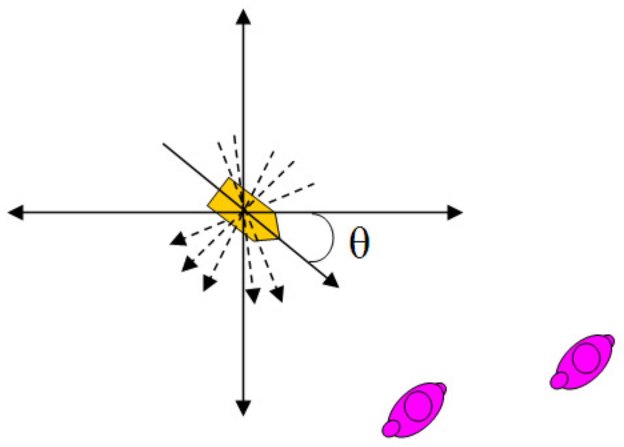


Figure 3.3: Camera orientation model.

image model" to estimate the ideal foreground image of each camera based on the target locations in the final certainty map and the camera's orientation and location. We then propose the fault detection and correction model to identify the faulty nodes and correct them from orientation inaccuracies.

Generative Image Model

The generative image model was proposed in [Fleuret et al., 2008] to generate the ideal background subtracted images if targets and camera locations are known. Let R_{s_i} denote the actual 2D foreground image and E_{s_i} denote the synthetic 2D image generated by the sensor node, s_i, based on the final certainty map, where each target is represented as a cylindrical object. The synthetic foreground images, $e_{s_i}(\theta)$, are generated based on all possible camera orientations in $[0°, 360°]$ with a selected `step_size` such that 360/`step_size` many candidate synthetic images are generated as illustrated in Fig 3.4.

For each sensor node, s_i, we calculate the normalized pseudo-distance, Ψ, to measure distance between actual 2D foreground image, R_{s_i} and the synthetic 2D

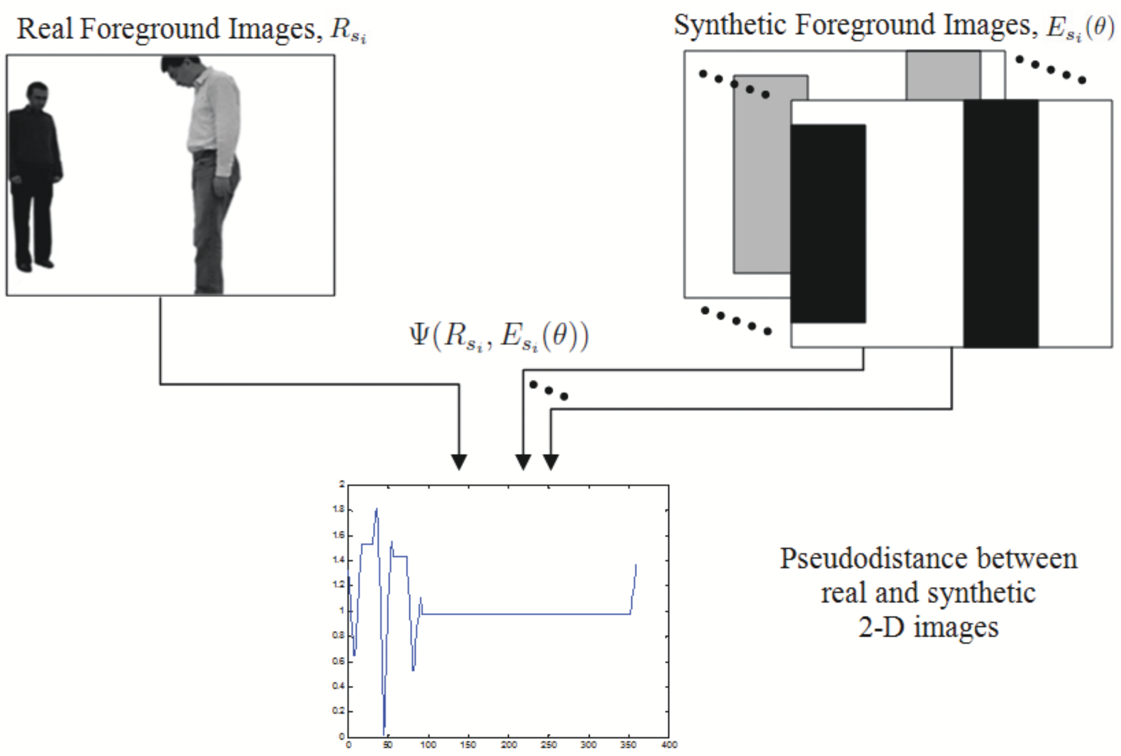

Real Foreground Images, R_{s_i}
Synthetic Foreground Images, $E_{s_i}(\theta)$
$\Psi(R_{s_i}, E_{s_i}(\theta))$
Pseudodistance between real and synthetic 2-D images

Figure 3.4: Illustration of the calculation of the pseudo-distance between two 2D images and estimation of camera orientation.

image generated by the sensor node, E_{s_i} [Fleuret et al., 2008],

$$\Psi(R_{s_i}, E_{s_i}(\theta)) = \frac{|R_{s_i} \otimes (1 - E_{s_i}(\theta)) + (1 - R_{s_i}) \otimes E_{s_i}(\theta)|}{|R_{s_i}|} \tag{3.3}$$

where $\otimes$ denotes the element-wise product between two images and $|R_{s_i}|$ is the sum of its pixel values for any binary image R_{s_i}.

The planer projection of 3D visual cones preserves the most useful information of moving targets along a plane which is perpendicular to the projection plane in target localization applications [Yang et al., 2003]. Therefore, instead of using 2D images (E_{s_i} and R_{s_i}) to estimate the orientation of each sensor node, we might use 1D scanline images (e_{s_i} and r_{s_i}), that are generated by summing the rows of the foreground image, to measure the distance between a synthetic foreground image, E_{s_i}, and the actual foreground image, R_{s_i}.

For each sensor node, s_i, we calculate the normalized pseudo-distance, Ψ, to measure distance between two 1D scanline images, r_{s_i} and $e_{s_i}(\theta)$,

$$\Psi(r_{s_i}, e_{s_i}(\theta)) = \frac{|r_{s_i} \otimes (1 - e_{s_i}(\theta)) + (1 - r_{s_i}) \otimes e_{s_i}(\theta)|}{|r_{s_i}|} \tag{3.4}$$

where $\otimes$ denotes the element-wise product between two images and $|r_{s_i}|$ is the sum of its pixel values for any gray-scale image r_{s_i}. Therefore, we might decrease the computational complexity by using 1D scanline images (e_{s_i} and r_{s_i}) to measure the distance between a synthetic foreground image, E_{s_i}, and the actual foreground image, R_{s_i}.

Fault Correction Model

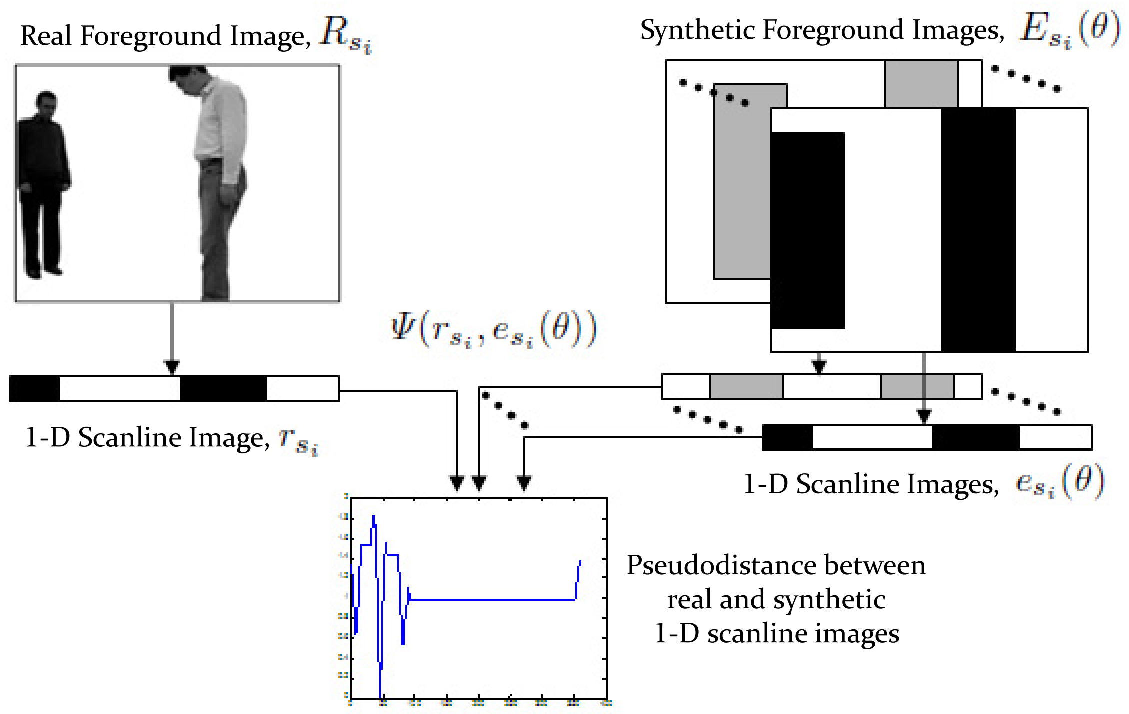
The expected orientation of each camera, $\theta^e_{s_i}$, is the one which minimizes the pseudo-distance,

$$\theta^e_{s_i} = \arg\min_{\theta} \; \Psi(R_{s_i}, E_{s_i}(\theta)) \tag{3.5}$$

where the fault in camera orientation is detected if there is a difference between expected camera orientation, $\theta^e_{s_i}$ and actual camera orientation, θ_{s_i}. We then update the actual camera orientation,

$$\theta_{s_i} = \theta^e_{s_i}. \tag{3.6}$$

Fig. 3.5 illustrates the calculation of the pseudo-distance between the actual and the synthetic foreground images. The simple scenario for fault-tolerant target localization is shown in Fig. 3.4 with two targets. In Fig. 3.5, the actual and synthetic foreground images, (R_{s_i}, E_{s_i}) and their 1-D scanline images, r_{s_i} and e_{s_i} are shown respectively. Pseudo-distances of real and synthetic 1-D scanline images for different θ values are shown at the bottom of Fig. 3.5. The expected orientation is the orientation value, θ which shows the minimum pseudo-distance.



Real Foreground Image, R_{s_i}
Synthetic Foreground Images, $E_{s_i}(\theta)$
$\Psi(r_{s_i}, e_{s_i}(\theta))$
1-D Scanline Image, r_{s_i}
1-D Scanline Images, $e_{s_i}(\theta)$
Pseudodistance between real and synthetic 1-D scanline images

Figure 3.5: Illustration of the calculation of the pseudo-distance between two 1D scanline images and estimation of camera orientation.

3.4 Distributed Implementation of Fault-Tolerant Collaborative Localization

In Section 3.3, we focused on the design of the centralized fault tolerance, detection and correction algorithm for target localization in VSNs with the existence of visual occlusion, partial information and a number of faulty nodes. In centralized implementation, we collect the available information from every sensor node to tolerate the error in sensors, detect the faulty sensor nodes and correct them before they cascade where redundant information is required to tolerate the faulty nodes. However, in the centralized implementation, more than required amount of redundant sensor nodes send their information to the sink which consumes large amount of energy.

In this section, we implement the distributed fault tolerance, detection and correction algorithm in VSNs. In our distributed implementation, we first focus on finding a tradeoff between energy conservation and required redundant information based on the coverage estimation probability to be described in Chapter 4 and the probability that an occupied grid point is determined as non-occupied by faulty sensor nodes. Then, we tolerate the fault in the network by choosing an automatic threshold based on this tradeoff between energy conservation and required redundant information and localize the existing targets in the sensing field. Based on the detected target locations, we generate the synthetic foreground images for each sensor node and detect the faulty nodes by comparing the actual and synthetic foreground. Finally, we correct these detected faulty nodes by using the generative image model.

3.4.1 Distributed Fault Tolerance in VSN

In order to conserve energy in an energy starving sensor network, not only each visual sensor node transmits a very limited amount of data but that a limited number of sensor nodes is involved in the decision making procedure to localize targets. In Chapter 2, we presented different collaborative processing methods for certainty map integration in detail which involves limited number of sensor nodes. However, in the sensing field where faulty nodes are likely to exist, we cannot trust a single sensor which might be a faulty node and declare the non-existence of a target within a region. Therefore, adequate amount of redundant information is required to tolerate the fault in the sensor network.

In Section 3.3, we proposed to use voting algorithms to tolerate the error of faulty nodes in the sensor network which requires redundant information to declare target non-existence in a region. However, the amount of required redundant information is unknown. It has to be decided accurately before the certainty map integration based on the estimated number of faulty nodes in the sensor network in order not only to reach a trustworthy final certainty map but also to save energy by involving a limited number of sensor nodes.

Automatic Voting Threshold Selection

In order to find a tradeoff between energy conservation and required redundant information, we first estimate the probability that a specific grid point of the sensing field is covered by exactly w many faulty sensor nodes, $P_f(w)$. We assume that the infinitesimal visual sensor nodes with uniform FOV and sensing radius are randomly deployed within a very large two-dimensional sensing field, R. Since each region in the sensing field has equal importance based on the probability of target existence, all sensor nodes are uniformly and independently distributed into the sensing field. Based on this deployment strategy, the locations of visual sensor nodes can be modeled by a two-dimensional stationary Poisson point process with sensor density λ_s [Wang et al., 2010]. It is also assumed that orientations of visual sensors are uniformly distributed over $[0°, 360°)$. Let ρ and θ denote, respectively, the sensing radius and angle of view of a sensor node. The detailed discussion about uniform random sensor deployment will be presented in Chapter 4.

A sensor node covers a specific grid point $(x, y) \in R$, of the sensing field if the node is located in a circular area A with radius ρ centered at the corresponding grid point and is oriented towards the center of the circle which is illustrated in Fig. 3.6. In the rest of the dissertation, the circular area A is referred to as the "detectability area". Therefore, probability that exactly w many faulty sensor nodes cover a specific grid point is

$$P_f(w) = \sum_{j=k}^{\infty} \sum_{i=w}^{j} \mathcal{P}(j; \lambda_s \times A) \mathcal{C}_i^j (p)^i (1-p)^{j-i} \mathcal{C}_w^i (p_f)^w (1-p_f)^{i-w} \tag{3.7}$$

where $\mathcal{P}(j; \lambda_s \times A)$ denotes the probability that a detectability area A contains exactly j sensor nodes from a Poisson point process with sensor density λ_s, i.e., $\mathcal{P}(j; \lambda_s \times A) =$

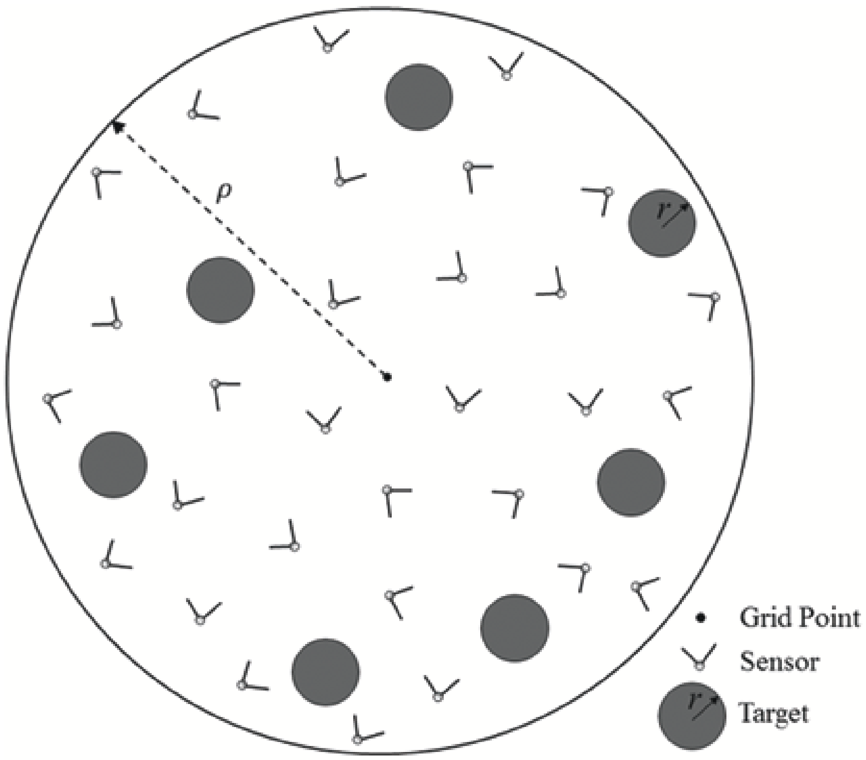


Figure 3.6: Illustration of the sensor deployment in a detectability area, A.

$e^{-\lambda_s \times A}(\lambda_s \times A)^j/j!$ where $A = \pi\rho^2$. Also, p denotes the probability of the sensor node facing towards the center of detectability area, A, i.e., $p = \theta/(2\pi)$, p_f denotes the probability of the sensor node being faulty, and $\mathcal{C}_i^j$ denotes the number of combinations of i-node subset from a j-node set. Eq. 3.7 can be further derived as,

$$\begin{aligned}
P_f(w) &= \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^w p_f^w (1-p)^{j-w} \sum_{i=w}^{j} \mathcal{C}_i^j \mathcal{C}_w^i \left(\frac{p(1-p_f)}{1-p}\right)^{i-w} \\
&= \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^w p_f^w (1-p)^{j-w} \sum_{z=0}^{j-w} \mathcal{C}_{w+z}^j \mathcal{C}_w^{w+z} \left(\frac{p(1-p_f)}{1-p}\right)^{z} \\
&\overset{(a)}{=} \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^w p_f^w (1-p)^{j-w} \sum_{z=0}^{j-w} \mathcal{C}_w^j \mathcal{C}_z^{j-w} \left(\frac{p(1-p_f)}{1-p}\right)^{z} \\
&= \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^w p_f^w (1-p)^{j-w} \mathcal{C}_w^j \sum_{z=0}^{j-w} \mathcal{C}_z^{j-w} \left(\frac{p(1-p_f)}{1-p}\right)^{z}
\end{aligned}$$

$$\begin{aligned}
&\overset{(b)}{=} \sum_{j=w}^{\infty} \mathcal{P}(j;\lambda_s \times A) p^w p_f^w (1-p)^{j-w} \mathcal{C}_w^j \left(\frac{p(1-p_f)}{1-p} + 1 \right)^{j-w} \\
&= \sum_{j=w}^{\infty} e^{-\lambda_s \times A} (\lambda_s \times A)^j \frac{1}{j!} p^w p_f^w (1-p)^{j-w} \frac{j!}{w!(j-w)!} \left(\frac{1-pp_f}{1-p} \right)^{j-w} \\
&= \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s App_f)^w \sum_{j=w}^{\infty} \frac{\Big((\lambda_s \times A)(1-pp_f)\Big)^{j-w}}{(j-w)!} \\
&= \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s App_f)^w \sum_{n=0}^{\infty} \frac{\Big((\lambda_s \times A)(1-pp_f)\Big)^{n}}{(n)!} \\
&\overset{(c)}{=} \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s App_f)^w e^{(\lambda_s \times A)(1-pp_f)} \\
&= \frac{1}{w!} (\lambda_s App_f)^w e^{-\lambda_s App_f} \\
&= \mathcal{P}(w; \lambda_s \times A \times p \times p_f) \\
&= \mathcal{P}(w; \lambda_f \times A \times p) \qquad (3.8)
\end{aligned}$$

where (a) follows the combination properties, $\mathcal{C}_{w+z}^j . \mathcal{C}_w^{w+z} = \mathcal{C}_w^j . \mathcal{C}_z^{j-w}$, (b) follows the binomial coefficient property, $(x+y)^n = \sum_{z=0}^n \mathcal{C}_z^n x^{n-z} y^z$ where $z = i - w$, and (c) follows property of power series, $\sum_{n=0}^\infty \frac{x^n}{(n)!} = e^x$. λ_f denotes the density of faulty sensor nodes i.e., $\lambda_f = \lambda_s \times p_f = \frac{N_s}{A} \times p_f = \frac{N_f}{A}$ where N_s and N_f denote the total number of deployed sensor nodes and expected number of faulty nodes in the sensing field, respectively.

From the derivation result in Eq. 3.8, we observe that the probability that exactly w many faulty sensor nodes cover a specific grid point, $P_f(w)$, follows the Poisson point process with density $\lambda_s \times p \times p_f$ in the detectability area A. In order to accurately tolerate the faulty nodes in a sensor network, the redundant information at a specific grid point of the sensing field should be more than the number of faulty nodes. As described in Section 3.3.2, the amount of the redundant information is controlled by the selection of the voting threshold. Therefore, the selected voting threshold should ensure the probability of fault that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes is smaller than a tolerance value ε_1. This probability of fault follows,

$$\begin{aligned} P_f(w \geq W) &< \varepsilon_1 \\ \sum_{w=W}^{\infty} P_f(w) &< \varepsilon_1 \\ \sum_{w=W}^{\infty} \mathcal{P}(w; \lambda_f \times A \times p) &< \varepsilon_1 \\ 1 - \sum_{w=0}^{W-1} \mathcal{P}(w; \lambda_f \times A \times p) &< \varepsilon_1 \\ 1 - F_{\mathcal{P}}(W-1; \lambda_f \times A \times p) &< \varepsilon_1 \end{aligned} \tag{3.9}$$

where $F_{\mathcal{P}}(W-1; \lambda_f \times A \times p)$ is the cumulative probability distribution (cdf) of Poisson distribution with parameter $\lambda_f \times A \times p$. Thus, Eq. 3.9 introduces the lower bound for the voting threshold selection.

In addition, the selected voting threshold should also ensure the visual K-coverage probability that each grid point is covered by at least K sensor nodes is higher than a certain probability in order to accurately tolerate the faulty sensor nodes in the sensing field. In other words, the probability that each point is covered by less than K sensor nodes is smaller than a tolerance value ε_2. Therefore, the K-Coverage constraint for voting threshold selection is

$$\begin{aligned} P(k \geq K) &> 1 - \varepsilon_2 \\ \sum_{k=K}^{\infty} P(k) &> 1 - \varepsilon_2 \\ 1 - \sum_{k=0}^{K-1} P(k) &> 1 - \varepsilon_2 \\ \sum_{k=0}^{K-1} P(k) &< \varepsilon_2 \\ F_P(K-1) &< \varepsilon_2 \end{aligned} \tag{3.10}$$

where $F_P(K-1)$ is the cumulative probability distribution (cdf) of visual coverage probability, $P(k)$ which will be discussed in Chapter 4. Thus, Eq. 3.10 introduces the upper bound for the voting threshold selection.

The optimization problem of voting threshold selection which ensures the fault-tolerant target localization can be expressed as,

$$V_{thr} = \underset{\substack{W \\ s.t. F_{\mathcal{P}}(W-1)<\varepsilon_2}}{\arg\min} \left|1 - F_{\mathcal{P}}(W-1; \lambda_f \times A \times p) - \varepsilon_1\right| \tag{3.11}$$

where V_{thr} is the selected voting threshold. Therefore, the solution for optimization problem is that voting threshold is the smallest positive root V_{thr} of Eq. 3.11. However, there is no explicit solution for Eq. 3.11. V_{thr} can be found by using the exhaustive search method.

Distributed Certainty Map Integration

To conserve energy in a sensor network, one of the basic guidelines is the involvement of limited number of sensor nodes into the decision making procedure to localize targets. In Section 2.5, we presented different itinerary selection algorithms for certainty map integration in detail. Dynamic itinerary is the one that guarantees that a minimum number of sensor nodes is involved into certainty map integration. Since visual sensor nodes in the sensing field might be faulty, dynamic itinerary requires redundant information to declare the non-existence of a target within a region. By selecting the voting threshold automatically, we guarantee that there is adequate amount of redundant information in dynamic itinerary to tolerate the fault in the sensor network. The initially occupied certainty map is progressively clarified by migrating through the dynamic itinerary from sensor node to sensor node. Finally, targets are localized at the uncleared areas and target locations are broadcasted to each sensor node to detect the fault in camera orientation.

3.4.2 Distributed Fault Detection and Correction in VSN

In order to detect faulty nodes with inaccuracies in camera orientation estimation, we compare the actual foreground image captured by each node and synthetic foreground image generated by generative image model based on the location of targets in the final certainty map and the camera's orientation and coordinates.

For each sensor node, s_i, we calculate the normalized pseudo-distance, Ψ, to measure distance between actual 2D foreground image, R_{s_i} and the synthetic 2D image generated by the sensor node, E_{s_i} Fleuret et al. [2008]. If the pseudo-distance, $\Psi(R_{s_i}, E_{s_i}(\theta))$, between actual image and synthetic image is larger than defined threshold, the sensor is detected as faulty.

Since the existence of faulty nodes might affect the localization accuracy, we localize targets again by excluding the faulty nodes from dynamic itinerary for certainty map integration. In order to correct the orientation of detected faulty nodes, we calculate the pseudo-distance for each possible orientation as described in generative image model and update the actual camera orientation, θ_{s_i} with the expected camera orientation $\theta^e_{s_i}$ which minimizes the pseudo-distance (see details in Section 3.3.4).

3.5 Experiments and Results

In this section, we evaluate the performance of the centralized fault-tolerant target localization algorithm using both simulated and real experiments with different amount of Gaussian noise and Byzantine faults added to camera orientations. Also, distributed implementation of fault-tolerant target localization algorithm is evaluated by using a simulated experiment with different amount of Byzantine faults added to camera orientations.

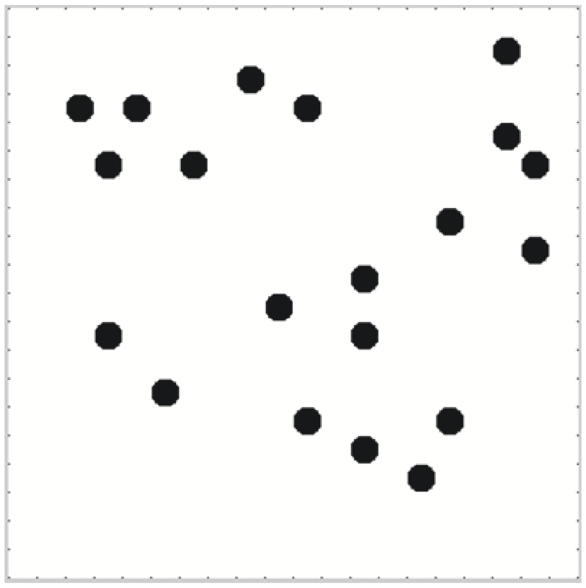


Figure 3.7: Simulation setup with 20 targets and 80 sensor nodes.

3.5.1 Experiments using Simulation

In our simulation, round-shaped targets of uniform size are deployed on a 2D sensing field, and infinitely small-size sensor nodes with uniform FOV and focal length are located at the sides of the sensing field and directed horizontally facing the sensing field. The orientation of each sensor node is a floating point number randomly generated in $[0°, 360°]$. The location of targets are randomly generated assuming there is no overlap between the targets and sensors. In all the simulations, we assume each node is accurately calibrated and synchronized with each other after initial deployment. Also, each node is able to find its location by using a positioning system, such as GPS.

Following is the setup of some typical parameters: The 2D sensing field is 20m $\times$ 20m large. The size of each target is 0.5m $\times$ 0.5m. The uniform sensing range of sensor nodes is 20m in length and 45° in angle. Each node is in the communication range of other nodes and is able to communicate with each other. Fig. 3.7 illustrates a sample random deployment of 20 targets, represented as discs, and 80 cameras, represented as points.

After each deployment of targets and cameras, the simulation software generates the corresponding foreground image of targets in the FOV of each sensor node. Each sensor node then computes the 2D visual cones of the non-occupied areas using the planer projection and generates local certainty map as described in Sec. 2.3. By using the centralized integration, the certainty map is progressively clarified and targets are located at the remaining uncertain regions in the map. To detect and correct the faulty nodes, target locations in the final certainty map are broadcasted to each node. Each node first estimates the camera orientation by using the generative image model and then updates its orientation if it is not accurate.

We conduct three sets of simulation experiments to study the effect of voting threshold, sensor node density and target density on the performance of the target localization algorithm. In each set of experiments, different amount of Byzantine faults and Gaussian noise with zero mean and various standard deviation values are generated and added to orientations of sensor nodes randomly for ten times and the results are averaged.

Effect of the Voting Threshold

In this set of experiments, we show how the usage of the voting mechanism can help reduce the effect of faulty sensor nodes, providing robust performance based on inaccurate sensor inputs. We deploy 10 targets and 80 sensors in the 2D sensing field with known sensor locations and orientations.

Fig. 3.8 shows the total number of detected targets by using different voting values with Byzantine faults added. We observe that if the threshold value is set to 1, there is no tolerance for any sensor error. Therefore, lower voting thresholds might be selected for fault tolerance purpose. As shown in Fig. 3.8, the algorithm successfully tolerates the faulty nodes and accurately detects all targets until more than 16, i.e., 20%, nodes experience Byzantine fault.

We also observe that to improve performance, a lower voting threshold value is required. However, we cannot select the voting threshold value too low otherwise false

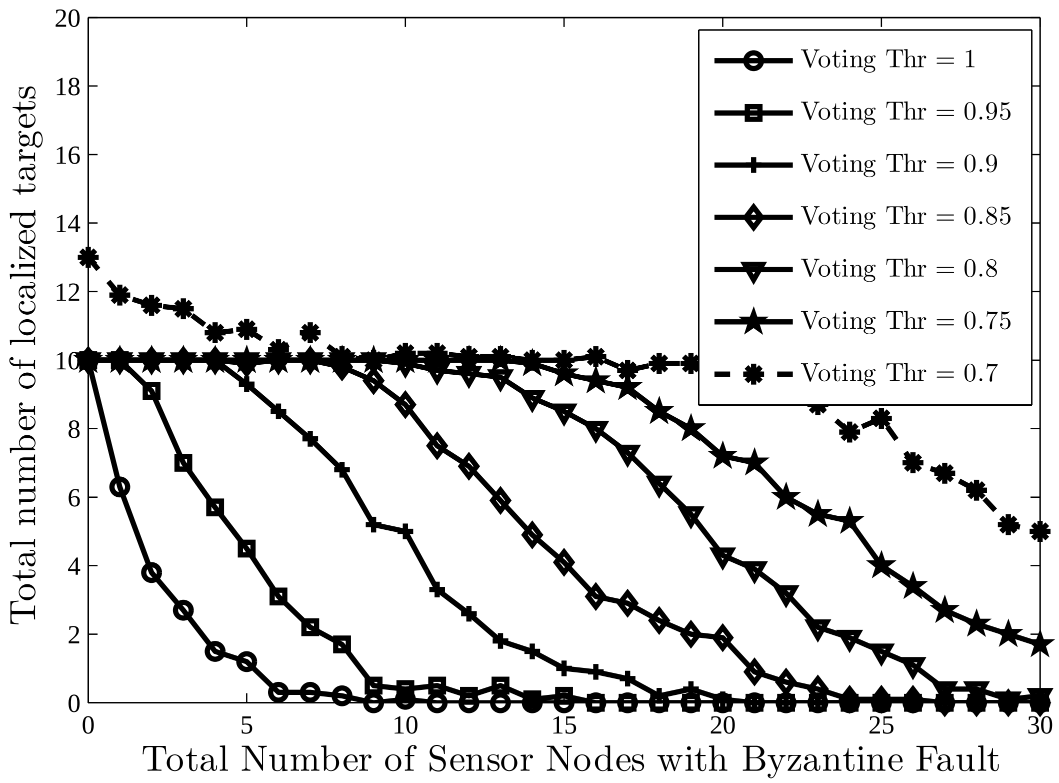
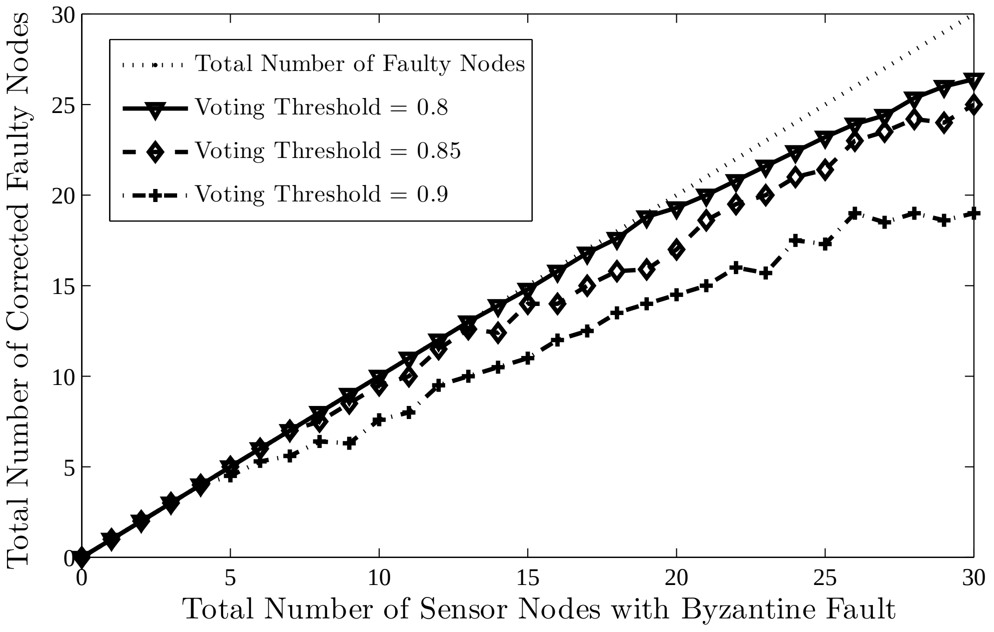


Figure 3.8: Total number of localized targets for different voting threshold values for Byzantine fault in different number nodes. Note that the number of deployed targets is fixed at 10.

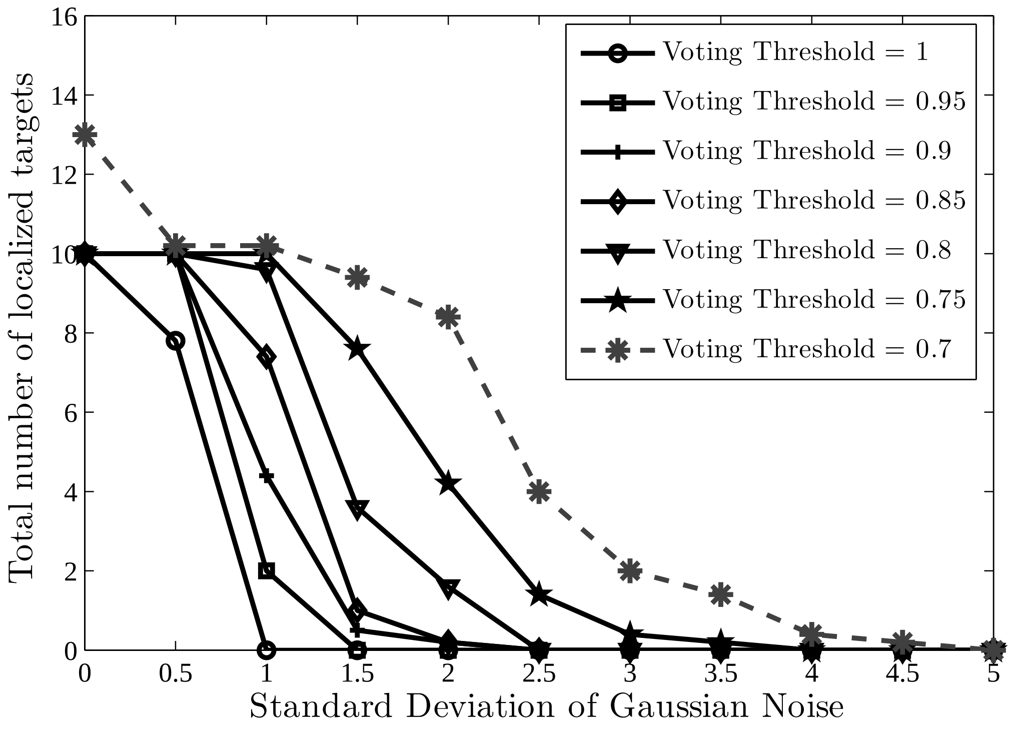
alarms would have been generated as shown in Fig. 3.8 for voting threshold values as low as 0.7.

To evaluate the fault detection and correction performance of the proposed method, we use the number of corrected nodes as the performance metric when Byzantine fault is added, as shown in Fig. 3.9a. We again observe that to add more noise reduces the fault detection and the correction performance of the algorithm, demanding a lower voting threshold value to avoid the performance drop.

Fig. 3.9b shows the total number of detected targets by using different voting values with Gaussian noise added. We observe that if the threshold value is set to 1, there is no tolerance for any sensor error. Therefore, lower voting thresholds might be selected for fault tolerance purpose. As shown in Fig. 3.9b, the algorithm



(a)



(b)

Figure 3.9: (a) Number of corrected faulty nodes for Byzantine fault and (b) Total number of localized targets for different voting threshold values for Gaussian noise with different standard deviation. Note that the number of targets is fixed at 10.

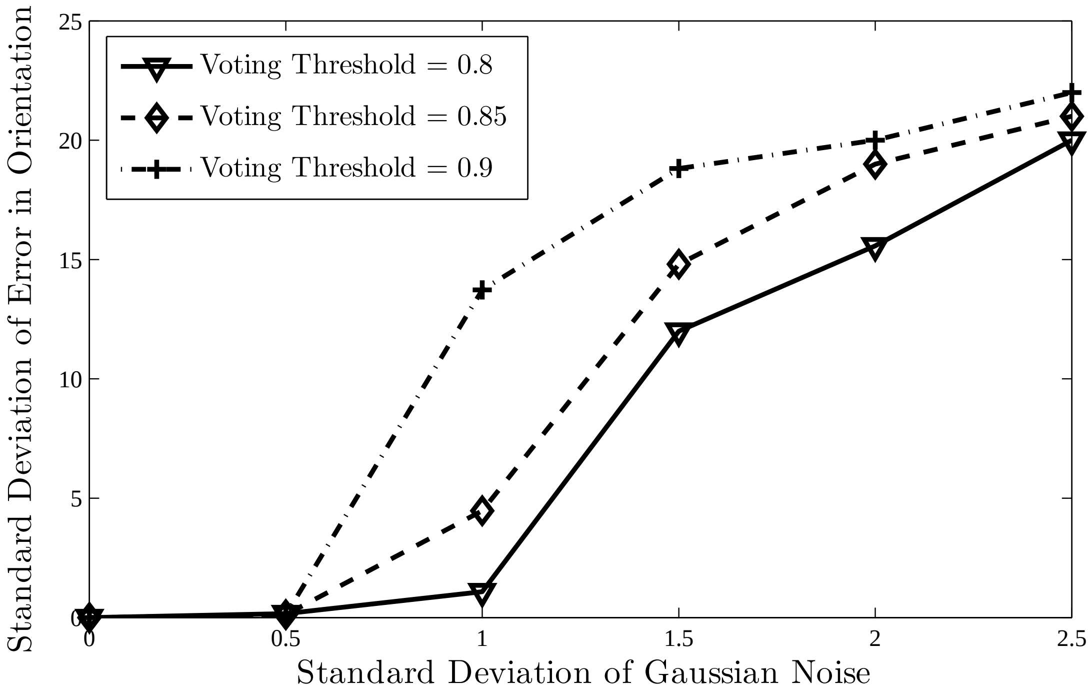


Figure 3.10: Resultant error in camera orientation for Gaussian noise. Note that the number of deployed targets is fixed at 10.

successfully tolerates the faulty nodes and accurately detects all targets until the standard deviation of Gaussian noise is 1.2.

We also observe that to improve performance, a lower voting threshold value is required. However, we cannot select the voting threshold value too low otherwise false alarms would have been generated as shown in Fig. 3.9b for voting threshold values as low as 0.7.

To evaluate the fault detection and correction performance of the proposed method, we use the standard deviation in the resulting camera orientation when Gaussian noise is added, as shown in Fig. 3.10. We again observe that to add more noise reduces the fault detection and the correction performance of the algorithm, demanding a lower voting threshold value to avoid the performance drop.

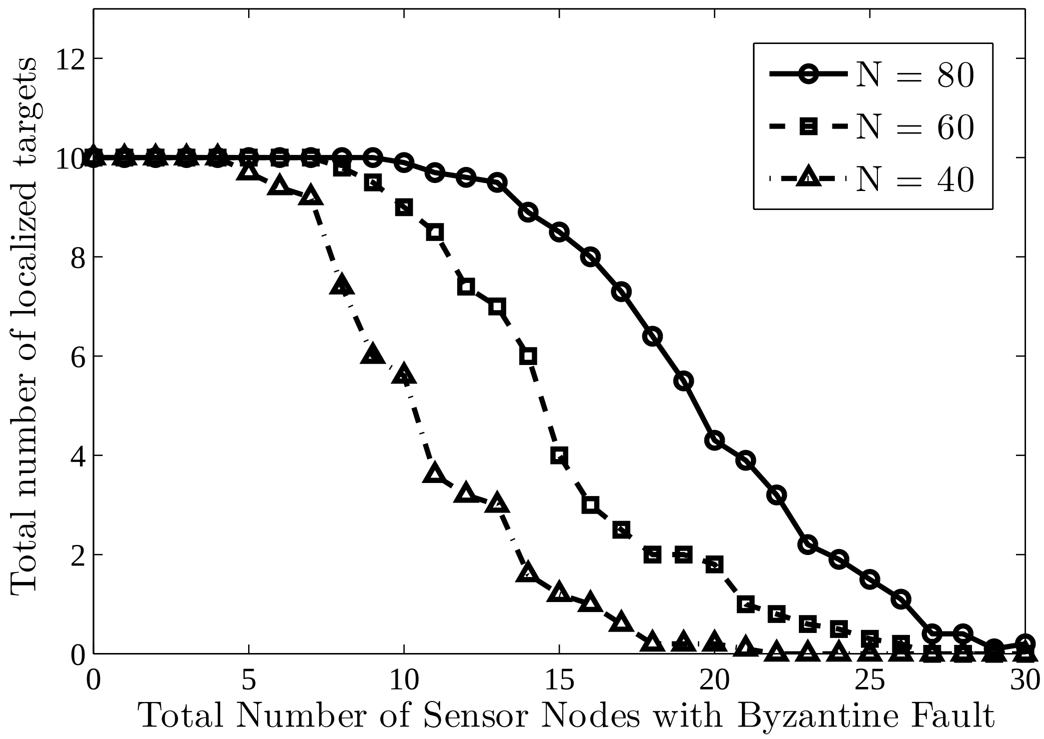


Figure 3.11: Total number of localized targets for different number of deployed nodes, N for different number of Byzantine faulty node. Note that the number of deployed targets is fixed at 10.

Effect of Node Density

In this set of experiments, the fault-tolerant target localization algorithm is tested to show its performance against different numbers of sensor nodes versus a fixed number of targets. In the simulation, 10 targets and different numbers of sensor nodes are deployed.

Fig. 3.11 illustrates the total number of detected targets for different numbers of deployed sensor nodes as a performance metric where Byzantine fault is added to the camera orientations. We observe again that the increased level of camera orientations inaccuracies reduces the target localization performance of the algorithm. To deploy more nodes in the sensing field makes the target localization algorithm more robust

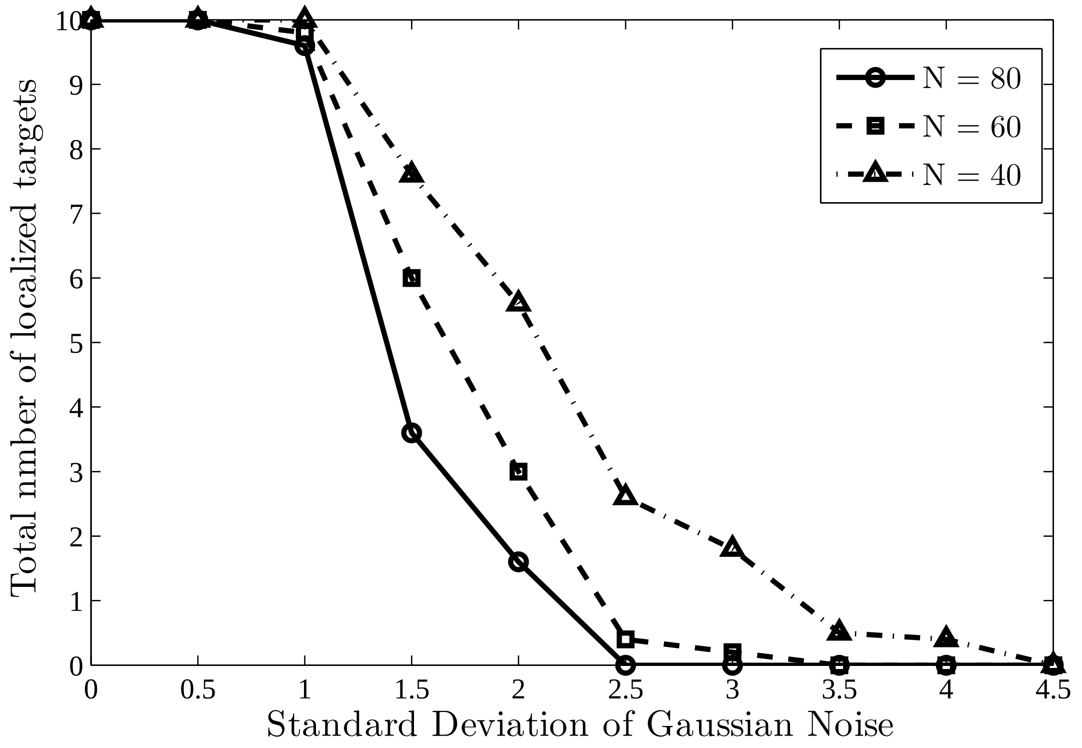
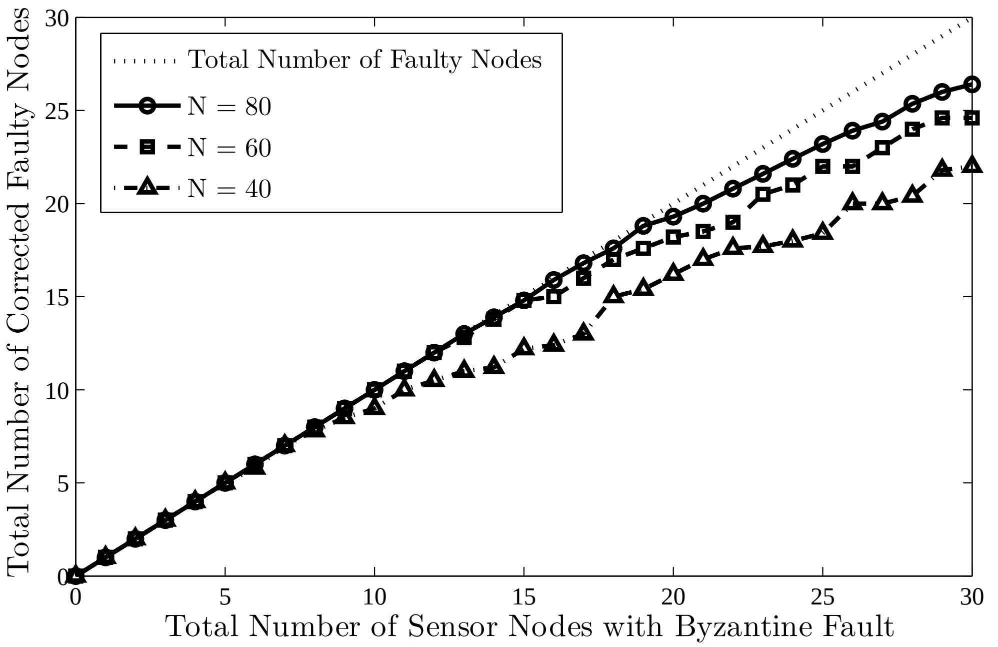


Figure 3.12: Total number of localized targets for different number of deployed nodes, N for Gaussian noise with different standard deviation. Note that the number of deployed targets is fixed at 10.

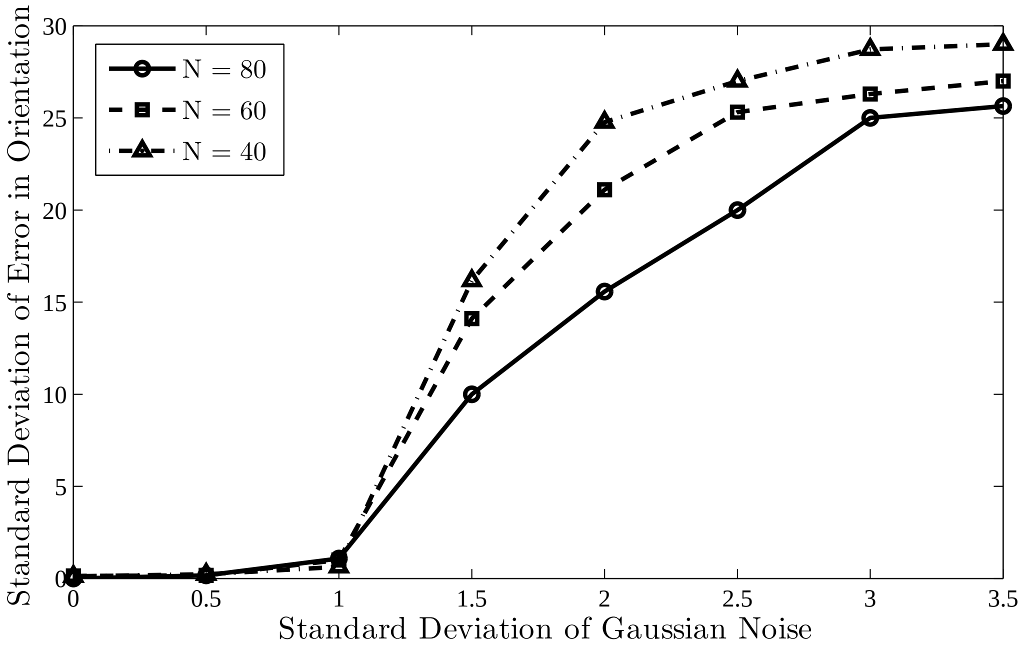
against Byzantine fault because it is more probable to tolerate the Byzantine fault in dense sensor deployment.

Fig. 3.12 illustrates the total number of detected targets for different numbers of deployed sensor nodes as a performance metric where Gaussian noise is added to the camera orientations. We observe again that the increased level of camera orientations inaccuracies reduces the target localization performance of the algorithm. To deploy more nodes in the sensing field makes the target localization algorithm more fragile to Gaussian noise because it is more probable to miss a target because Gaussian noise is added to all camera orientations and to have accurately calibrated VSN is less probable.

The effect of the sensor node density on the resultant error of fault correction algorithm and the number of corrected nodes are shown in Fig. 3.13a and Fig. 3.13b



(a)



(b)

Figure 3.13: (a) Number of corrected faulty nodes for Byzantine fault and (b) Resultant error in camera orientation for Gaussian noise. Note that the number of deployed targets is fixed at 10.

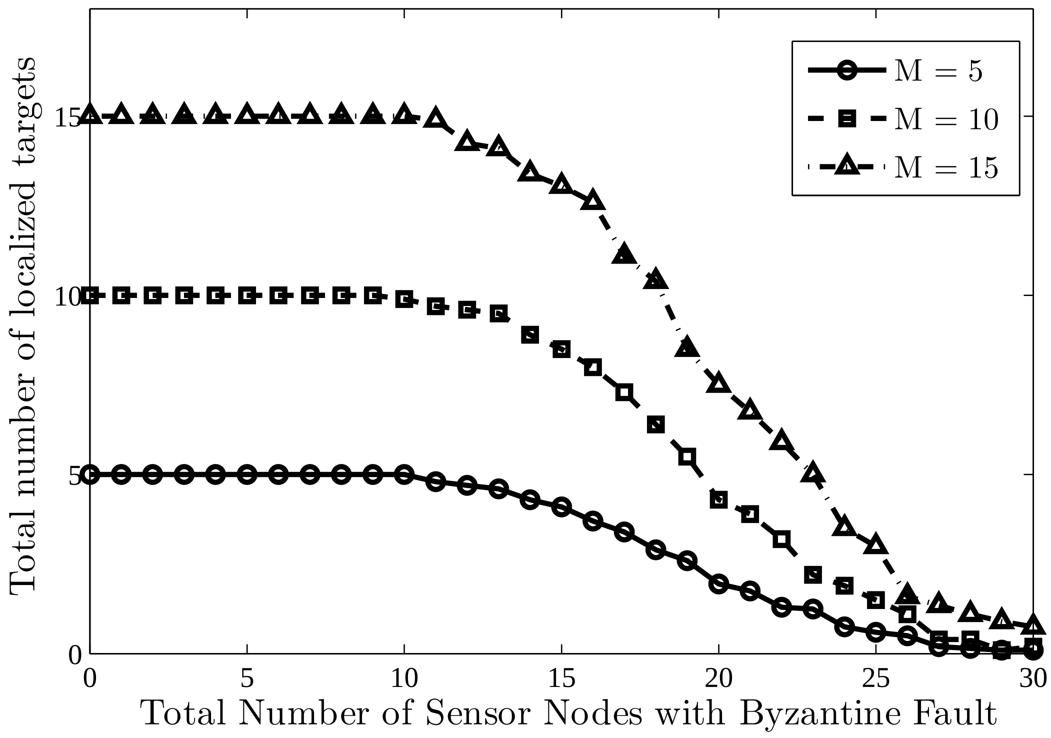


Figure 3.14: Total number of localized targets for different numbers of deployed targets, M for different number of Byzantine faulty node. Note that the number of deployed nodes is fixed at 80.

for various amount of Byzantine fault and Gaussian noise, respectively. We observe that to have better performance on fault detection and correction, the deployment of more sensor nodes in the sensing field is needed.

Effect of Target Density

In this set of experiments, the effect of deploying different numbers of targets, M versus fix number of cameras, N is studied with the existence of faulty nodes because of the Gaussian noise and Byzantine fault in their camera orientations. We deploy 80 cameras and different numbers of targets in the sensing field.

Fig. 3.14 shows the number of detected targets for different amount of Byzantine fault by deploying different numbers of targets. We observe that the performance

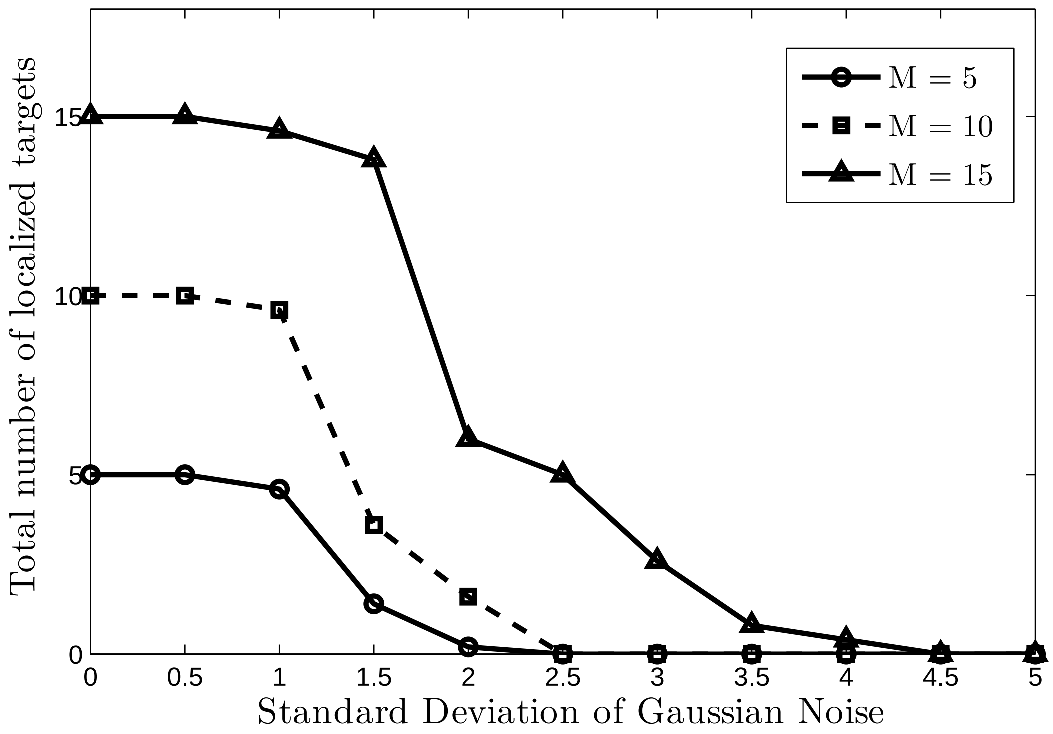
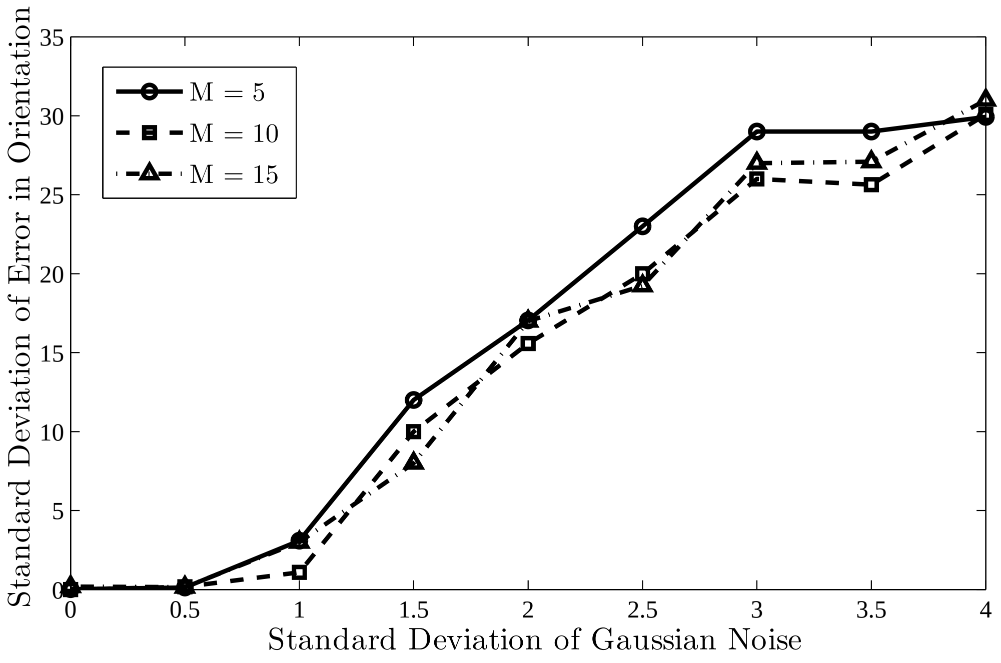


Figure 3.15: Total number of localized targets for different numbers of deployed targets, M for Gaussian noise with different standard deviation. Note that the number of deployed nodes is fixed at 80.

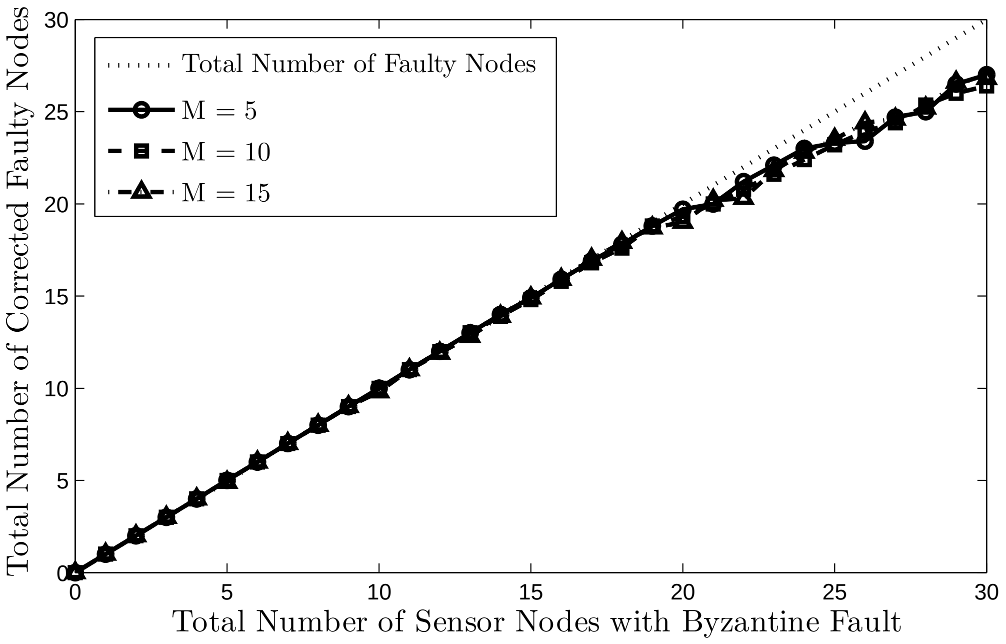
of target detection algorithms is almost identical for different numbers of deployed targets. An increment in total number of deployed targets does not affect the performance of target localization much compared to the effect of the number of deployed sensor nodes.

Fig. 3.15 shows the number of detected targets for different amount of Gaussian noise by deploying different numbers of targets. We observe that the performance of target detection algorithms is almost identical for different numbers of deployed targets. An increment in total number of deployed targets does not affect the performance of target localization much compared to the effect of the number of deployed sensor nodes.

In Fig. 3.16(a-b), the effect of the target density on the fault detection and correction model is shown for various amount of Gaussian noise and Byzantine fault,

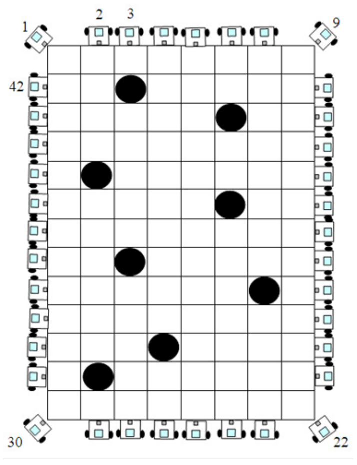


(a)

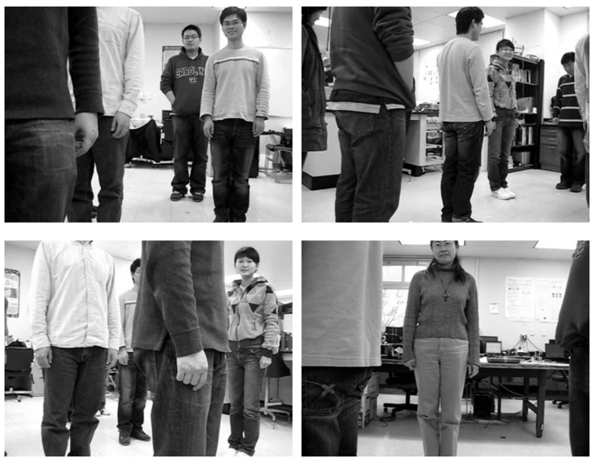


(b)

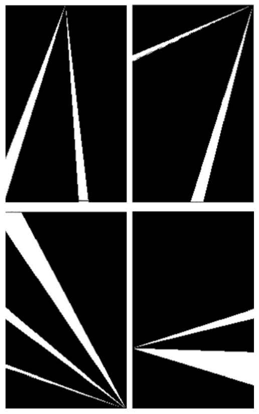
Figure 3.16: (a) Resultant error in camera orientation for Gaussian noise and (b) Number of corrected faulty nodes for Byzantine fault. Note that the number of deployed nodes is fixed at 80.



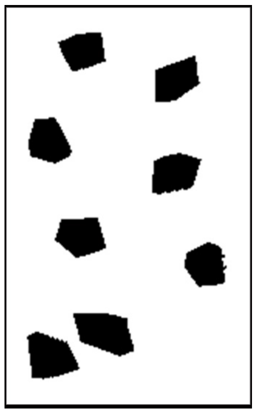
(a)



(b)



(c)



(d)

Figure 3.17: (a) Experimental setup with 8 people and 42 cameras, (b) Images captured by $5^{th}, 9^{th}, 22^{nd}, 34^{th}$ cameras, (c) Corresponding non-occupied 2D visual cones of the images in Fig. 3.17b and (d) the final certainty map.

respectively. We observe that the effect of the number of deployed targets, M on the resultant error of fault correction algorithm for Gaussian noise and the number of corrected sensor nodes in Byzantine fault is far less than the effect of the number of deployed sensor nodes, N. This is especially true when $M \ll N$.

3.5.2 Experiments using Real Data

Besides simulation, we also conduct a set of real experiments, shown in Fig. 3.17a, where an 8 by 13 square feet area is surrounded by 42 mobile sensor platforms (MSPs) with onboard processing, wireless communication and imaging capabilities. Four of the MSPs are located at the corners of the experimental area and oriented toward the center of the area at 3 feet in height. The rest of the MSPs are located 1 foot apart from each other and oriented in perpendicular angle with the sides of the room at 3 feet in height. In this experimental setup, each square foot area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution.

We first evaluate the performance of the proposed algorithm on its capability in localizing targets in a crowded scene, e.g., to identify eight targets within an 8×13 square feet area as shown in Fig. 3.17a. Images are captured by each MSP, as shown in Fig. 3.17b. Fig. 3.17c illustrates the local certainty maps generated at the $5^{th}, 9^{th}, 22^{nd}$, and 34^{th} MSPs. After integration of local certainty maps, targets are localized in the final certainty map as shown in Fig. 3.17d.

To study the effect of voting threshold on target localization accuracy, we add Byzantine faults to orientations of 6 sensor nodes to evaluate the localization accuracy of four people, whose true locations are shown in Fig. 3.18a. In Fig. 3.18(b-f), the final version of the certainty maps are presented using different threshold values as 1, 0.95, 0.9, 0.8, and 0.7, respectively. In Fig. 3.18b, the threshold value is 1 so there is no tolerance for any sensor failure. If one of the MSPs claims non-existence of the target, it is 100% accepted. As a result, the algorithm failed to localize one of the four targets in the scene. However, this missing target can be identified if using a lower voting threshold value, as shown in Fig. 3.18(c-e), where the threshold is set to 0.95, 0.9 and 0.85, respectively, indicating that to clear the specific area from the certainty map at least 5%, 10% or 15% of the MSPs must agree that region should be cleared. Nevertheless, the voting threshold value cannot be selected too low as some of the non-occupied areas would then start to be mislabeled as target, as shown in Fig. 3.18f.

Fig. 3.18(g-l) shows the robustness of the proposed target localization algorithm against the Byzantine fault. We set the normalized voting threshold value to 0.85 and

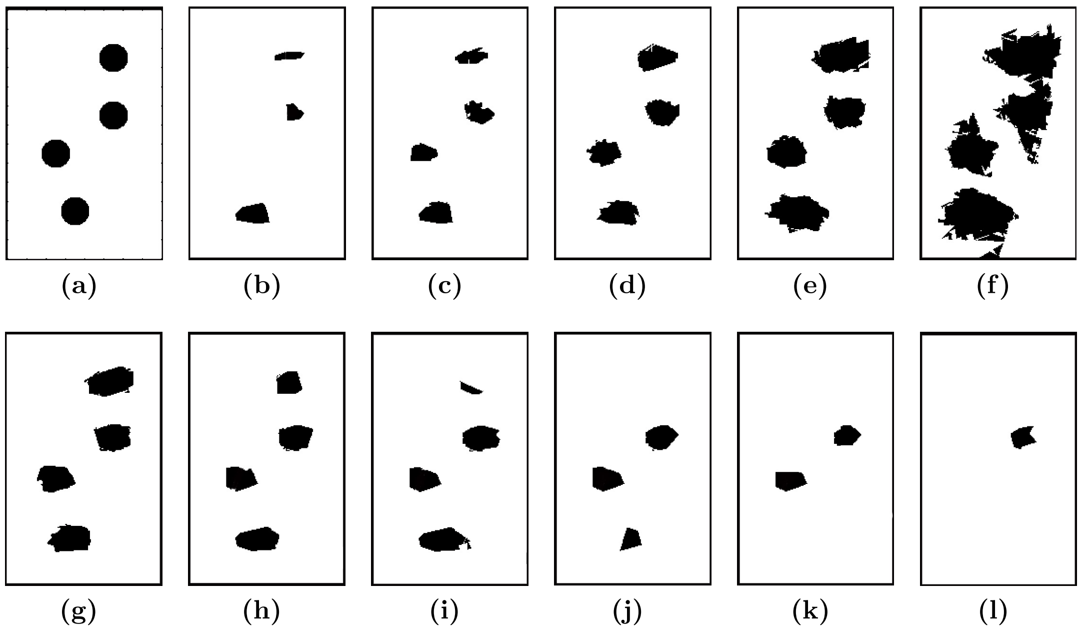


Figure 3.18: (a) True Locations of 4 people, (b) to (f) Final version of the certainty map using different voting threshold values 1, 0.95, 0.9, 0.85, 0.8, 0.75, and 0.7, respectively. (g) to (l) Final version of the certainty map for different number of node with Byzantine fault as 4, 8, 9, 10, 12, 14, and 18 nodes, respectively.

add Byzantine fault to different numbers of sensor nodes, as 4, 8, 10, 12, 14, and 18 nodes, respectively. We observe that the proposed method is able to tolerate faulty nodes inputs before its total number reaches 10 (25% of the total number of deployed sensor nodes).

To further study the effect of voting threshold on target localization accuracy, we add zero-mean Gaussian noise with different standard deviation values to all the camera orientations. In Fig. 3.19, the total number of uncertain pixels in the final certainty map is shown for different standard deviation values and for different voting threshold values. Suppose each person should occupy 500 pixels in the final certainty map, then we set the lower bound and upper bound of number of uncertain pixels for one person as (375, 875), i.e., if the size of one segment is less than 75% of the target size, we deem it as noise and if the segment size is larger than 1.75 times the target size, we deem it as containing two targets. Therefore, the lower and upper bounds of

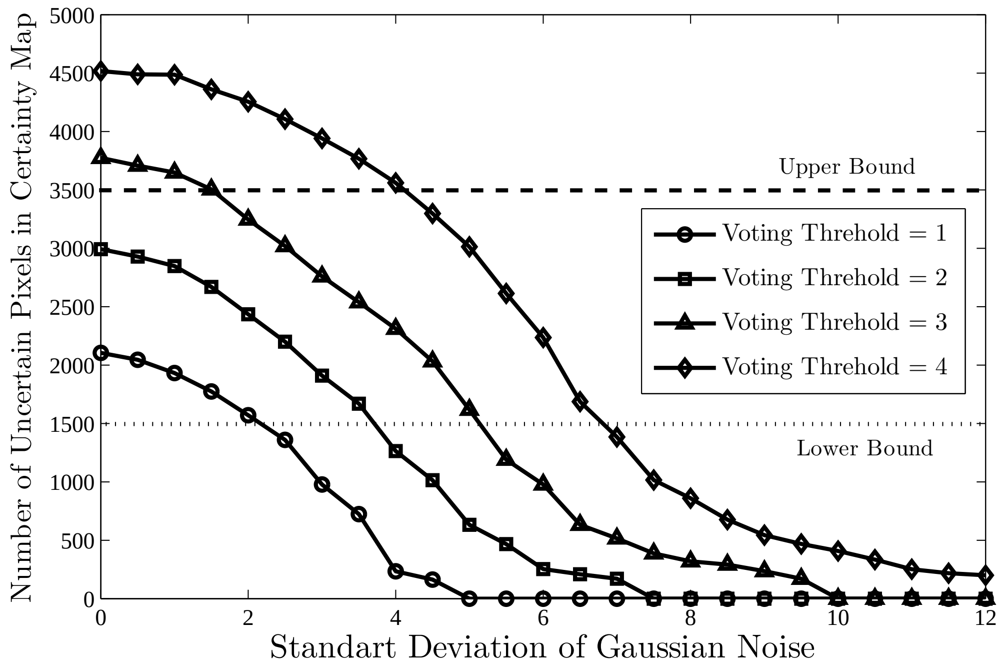


Figure 3.19: Total area in the final certainty map for Gaussian noise with zero mean and different standard deviation values added to orientation of the visual sensors for different voting threshold values.

the number of uncertain pixels in the certainty map for detecting 4 people is (1500, 3500) pixels, as shown in Fig. 3.19.

We observe in Fig. 3.19 that if the standard deviation of Gaussian noise is $\sigma = 3$, the performance of the system with the voting threshold of both two and three are more appropriate as the profiles stand right between the lower and upper bounds.

To add more Gaussian noise by increasing the standard deviation value reduces the system performance, demanding a higher voting threshold value to avoid the performance drop. However, we cannot select the voting threshold value too high otherwise false alarms would have been generated.

3.5.3 Experiments for Distributed Implementation of Fault Tolerant Collaborative Localization

In our simulation, round-shaped targets of uniform size are deployed on a 2D sensing field, and infinitely small-size sensor nodes with uniform FOV and focal length are randomly deployed into the sensing field and directed horizontally facing the sensing field. The locations of each sensor node and target are randomly generated assuming there is no overlap between the targets and sensors. The orientation of each sensor node is a floating point number randomly generated in $[0°, 360°]$. In all the simulations, we assume each node is accurately calibrated and synchronized with each other after initial deployment. Also, each node is able to find its location by using a positioning system, such as GPS.

Following is the setup of some typical parameters: The 2D sensing field is 50m $\times$ 50m large. The size of each target is 0.5m $\times$ 0.5m. The uniform sensing range of sensor nodes is 20m in length and 45° in angle. Each node is in the communication range of other nodes and is able to communicate with each other. Fig. 3.20 illustrates a sample random deployment of 50 targets, represented as discs, and 500 cameras, represented as points.

After each deployment of targets and cameras, the simulation software generates the corresponding foreground image of targets in the FOV of each sensor node. Each sensor node then computes the 2D visual cones of the non-occupied areas using the planer projection and generates local certainty map as described in Sec. 2.3. By using the distributed integration through a dynamic itinerary, the certainty map is progressively clarified where the voting threshold is automatically selected. Targets are located at the remaining uncertain regions in the map. In order to detect and correct the faulty nodes, target locations in the final certainty map are broadcasted to each node. Then, each node first generates its synthetic image by using the generative image model and compares its actual image with the synthetic image. If the pseudo-distance between the actual image and synthetic image is not smaller

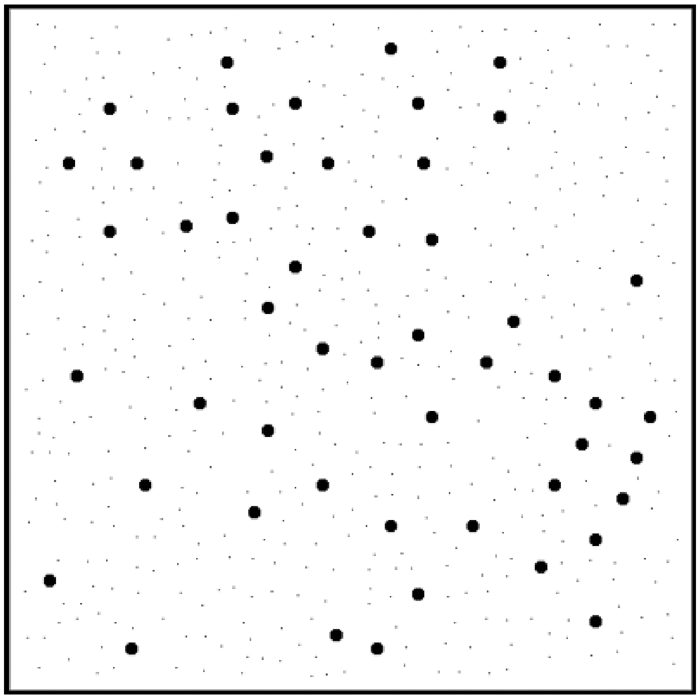
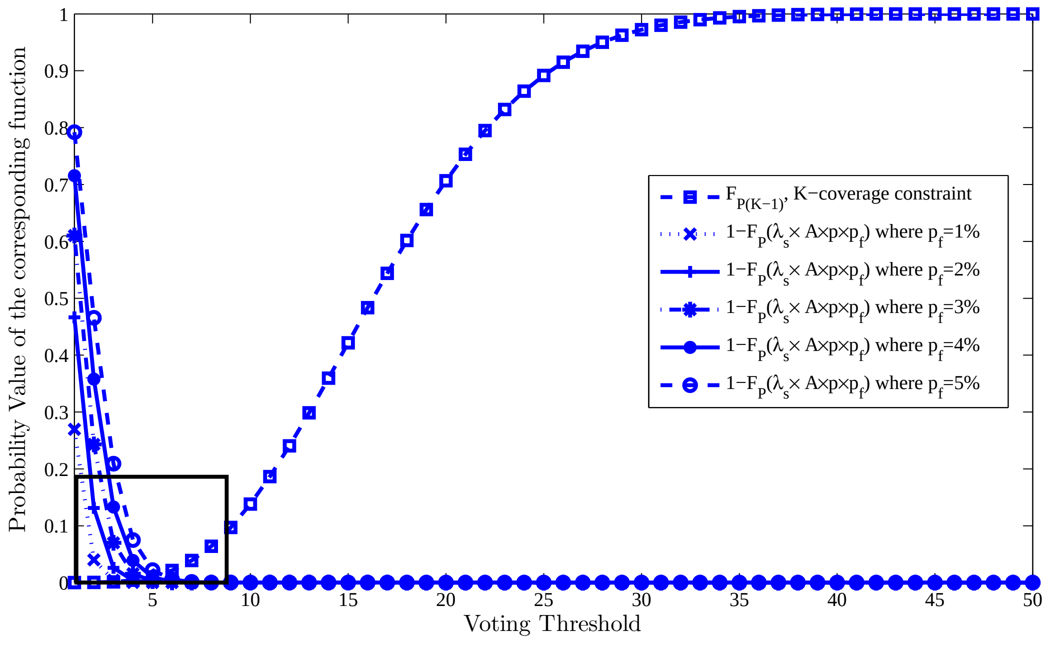


Figure 3.20: Simulation setup with randomly deployed 50 targets and 500 sensor nodes.

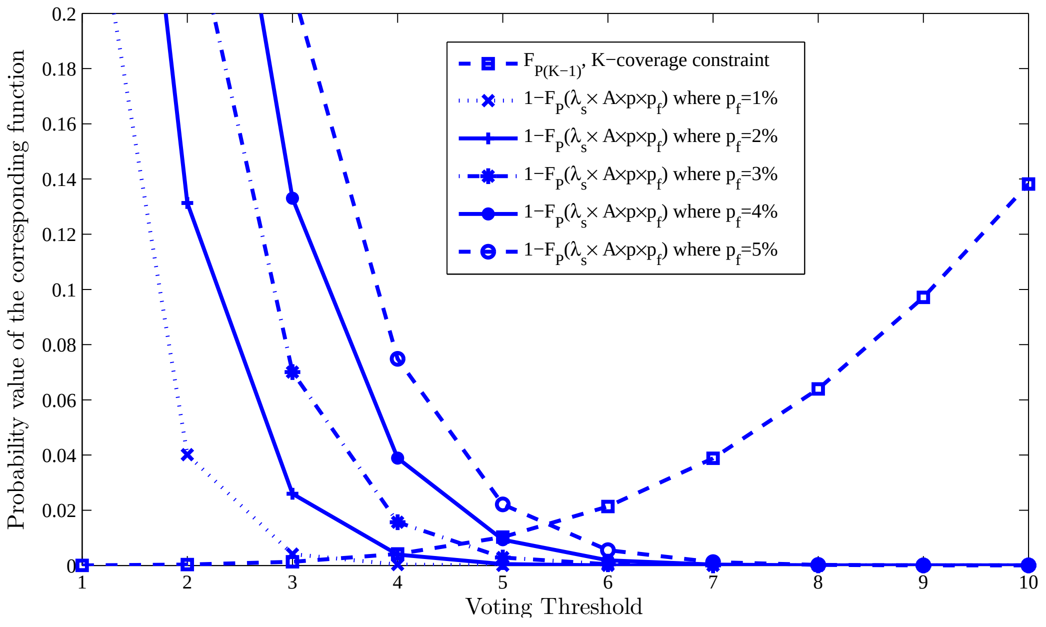
than the defined threshold, the sensor nodes determines itself as faulty and estimates its camera orientation by using the generative image model and then updates its orientation if it is not accurate.

We conduct five simulation experiments to study the effect of probability that sensor node is faulty, p_f, on the performance of the fault detection and fault correction algorithm. In each set of experiments, different amount of Byzantine faults are generated and added to orientations of sensor nodes randomly for ten times.

First, the threshold value should be selected for distributed integration through dynamic itinerary by using the automatic threshold selection method described in Section 3.4.1. Fig. 3.21 illustrates the probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes, $1 - F_{\mathcal{P}}(W-1; \lambda_f \times A \times p)$ and the probability that each grid point is covered by less than K sensor nodes, $F_{P(K-1)}$, that ensures the visual K-coverage probability.



(a)



(b)

Figure 3.21: (a) Probability that a specific grid point is covered by at least W-many faulty sensor nodes, $1-F_{\mathcal{P}}(W-1;\lambda_f \times A \times p)$ and the probability that each grid point is covered by less than K sensor nodes, $F_{P(K-1)}$ for automatic threshold selection and (b) zoom in of rectangle area in (a).

We observe that to decrease the probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes, $1 - F_{\mathcal{P}}(W-1; \lambda_f \times A \times p)$, a higher voting threshold value is required. In addition, to increase the probability that sensor node is faulty, p_f, increases probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes and requires higher voting threshold value. However, the voting threshold value cannot be selected too high otherwise it is not ensured to have accurate K-coverage in the sensor network for fault tolerance. In order to select the threshold value automatically based on the solution of Eq. 3.11, we set the tolerance values as $\varepsilon_1 = 0.005$ and $\varepsilon_2 = 0.0025$, threshold value for minimum pseudo-distance to 0.7 and select the probability that sensor node is faulty, p_f from 0.01 to 0.05.

In the first experiment, we set the probability that a sensor node is faulty to 0.01, i.e., $p_f = 0.01$. Since 500 sensor nodes are deployed into the sensing field, there are 5 faulty sensor nodes in average, i.e., $N_f = 5$. The voting threshold value is automatically selected as 3, i.e., $V_{thr} = 3$. Table 3.1 shows the confusion matrix of fault detection results and their related calculations (i.e., accuracy, sensitivity, specificity, and precision) and fault correction results. The confusion matrix consists of four result cells that report true positive (i.e., faulty nodes detected as faulty), false positive (i.e., non-faulty nodes detected as faulty), true negative (i.e., non-faulty nodes detected as non-faulty), and false negative (i.e., faulty nodes detected as non-faulty). Accuracy is the ratio of the true results (both true positives and true negatives) to the total number of all results. Precision measures the proportion of the true positives against all the positive results (both true positives and false positives). Sensitivity and specificity are defined, respectively, as the proportion of true positives which are correctly detected faulty nodes as faulty and the proportion of true negatives which are correctly detected non-faulty nodes as non-faulty.

We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 78% of the faulty nodes are corrected by fault correction algorithm and 18% of the faulty nodes are classified as symmetric sensor

Table 3.1: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.01$ and $V_{thr} = 3$.

		Predicted	
		Faulty	Not Faulty
Actual	Faulty	49	1
	Not Faulty	3	4946

Accuracy	0.9990
Sensitivity	0.9800
Specificity	0.9996
Precision	0.9412

Total Number of Corrected Faulty Nodes	39
Total Number of Symmetric Faulty Nodes	9
Total Number of Faulty Nodes	50
Total Number of Falsified Non-Faulty Node	2
Total Number of Corrected Non-Faulty Node	1
Total Number of Non-Faulty Node Detected Faulty	3

Faulty Nodes	
Corrected	0.780
Symmetric	0.180
Total	0.960
Non-Faulty Nodes	
Falsified	0.00040

node. A sensor node is classified as a symmetric sensor node if the fault correction algorithm gives several possible orientation for the corresponding sensor node. This might be happen if there is no target in the field of view of the sensor node at that specific time frame and the actual captured image of the sensor is empty. Symmetric sensor nodes can be detected as faulty but they might not be corrected because there are more than one possible orientation angle that shows the minimum pseudo-distance. Whenever a symmetric node covers a target, the symmetry in their field of view will be eliminated and its orientation will be corrected.

In addition, we observe that the fault detection algorithm misclassifies three non-faulty sensor node as faulty node because the residual areas around the targets might cause the localized target to slightly shift due to the digitization of sensing area as grids. One out of three misclassified sensor nodes is reoriented to its actual orientation. However, the orientation of two misclassified sensor node is falsified which makes the falsification ratio as 0.00040.

Table 3.2: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.02$ and $V_{thr} = 4$.

		Predicted	
		Faulty	Not Faulty
Actual	Faulty	99	1
	Not Faulty	19	4881

Accuracy	0.9958
Sensitivity	0.9900
Specificity	0.9961
Precision	0.8376

Total Number of Corrected Faulty Nodes	71
Total Number of Symmetric Faulty Nodes	24
Total Number of Faulty Nodes	100
Total Number of Falsified Non-Faulty Node	16
Total Number of Corrected Non-Faulty Node	3
Total Number of Non-Faulty Node Detected Faulty	19

Faulty Nodes	
Corrected	0.710
Symmetric	0.240
Total	0.950
Non-Faulty Nodes	
Falsified	0.00320

In the second experiment, we set the probability that a sensor node is faulty to 0.02, i.e., $p_f = 0.02$. Since 500 sensor nodes are deployed into the sensing field, there are 10 faulty sensor nodes in average i.e., $N_f = 10$. The voting threshold value is automatically selected as 4, i.e., $V_{thr} = 4$. Table 3.2 shows the confusion matrix and related calculations of fault detection algorithm and fault correction results.

We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 71% of the faulty nodes are corrected by fault correction algorithm and 24% of the faulty nodes are classified as symmetric sensor node. The overall falsification ratio is 0.00320.

In the third experiment, we set the probability that a sensor node is faulty to 0.03, i.e., $p_f = 0.03$. Since 500 sensor nodes are deployed into the sensing field, there are fifteen faulty sensor nodes in average i.e., $N_f = 15$. The voting threshold value is automatically selected as 4, i.e., $V_{thr} = 4$. Table 3.3 shows the confusion matrix and related calculations of fault detection algorithm and fault correction results.

Table 3.3: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.03$ and $V_{thr} = 4$.

		Predicted	
		Faulty	Not Faulty
Actual	Faulty	148	2
	Not Faulty	25	4825

Accuracy	0.9946
Sensitivity	0.9867
Specificity	0.9948
Precision	0.8555

Total Number of Corrected Faulty Nodes	122
Total Number of Symmetric Faulty Nodes	21
Total Number of Faulty Nodes	150
Total Number of Falsified Non-Faulty Node	17
Total Number of Corrected Non-Faulty Node	8
Total Number of Non-Faulty Node Detected Faulty	25

Faulty Nodes	
Corrected	0.813
Symmetric	0.140
Total	0.953
Non-Faulty Nodes	
Falsified	0.00340

We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 81.3% of the faulty nodes are corrected by fault correction algorithm and 14% of the faulty nodes are classified as symmetric sensor node. The overall falsification ratio is 0.00340.

In the fourth experiment, we set the probability that a sensor node is faulty to 0.04, i.e., $p_f = 0.04$. Since 500 sensor nodes are deployed into the sensing field, there are twenty faulty sensor nodes in average i.e., $N_f = 20$. The voting threshold value is automatically selected as 5, i.e., $V_{thr} = 5$. Table 3.4 shows the confusion matrix and related calculations of fault detection algorithm and fault correction results.

We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 72.5% of the faulty nodes are corrected by fault correction algorithm and 21% of the faulty nodes are classified as symmetric sensor node. The overall falsification ratio is 0.00720.

Table 3.4: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.04$ and $V_{thr} = 5$.

		Predicted	
		Faulty	Not Faulty
Actual	Faulty	197	3
	Not Faulty	45	4755

Accuracy	0.9904
Sensitivity	0.9850
Specificity	0.9906
Precision	0.8140

Total Number of Corrected Faulty Nodes	145
Total Number of Symmetric Faulty Nodes	42
Total Number of Faulty Nodes	200
Total Number of Falsified Non-Faulty Node	36
Total Number of Corrected Non-Faulty Node	9
Total Number of Non-Faulty Node Detected Faulty	45

Faulty Nodes	
Corrected	0.725
Symmetric	0.210
Total	0.935
Non-Faulty Nodes	
Falsified	0.00720

In the fifth experiment, we set the probability that a sensor node is faulty to 0.05, i.e., $p_f = 0.05$. Since 500 sensor nodes are deployed into the sensing field, there are twentyfive faulty sensor nodes in average i.e., $N_f = 25$. The voting threshold value is automatically selected as 6, i.e., $V_{thr} = 6$. Table 3.5 shows the confusion matrix and related calculations of fault detection algorithm and fault correction results.

We observe that the fault detection algorithm shows a high accuracy, sensitivity, and specificity rates. However precision rate decreased to 64.5%. Also, 75.2% of the faulty nodes are corrected by fault correction algorithm and 16.8% of the faulty nodes are classified as symmetric sensor node. The overall falsification ratio increase to 0.02320.

As an overall comment on these five experiments, we observe that to increase the probability that a sensor is faulty, p_f, decreases the accuracy, sensitivity, specificity and precision rates of fault detection algorithm. Especially, the decrement on the precision rate is obvious. In addition, the performance of the fault correction decreases

Table 3.5: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.05$ and $V_{thr} = 6$.

		Predicted	
		Faulty	Not Faulty
Actual	Faulty	239	11
	Not Faulty	131	4619

Accuracy	0.9716
Sensitivity	0.9560
Specificity	0.9724
Precision	0.6459

Total Number of Corrected Faulty Nodes	188
Total Number of Symmetric Faulty Nodes	42
Total Number of Faulty Nodes	250
Total Number of Falsified Non-Faulty Node	116
Total Number of Corrected Non-Faulty Node	15
Total Number of Not Faulty Node Detected Faulty	131

Faulty Nodes	
Corrected	0.752
Symmetric	0.168
Total	0.920
Not Faulty Nodes	
Falsified	0.02320

as the probability that a sensor is faulty, p_f, increases. The falsification ratio is increased at higher p_f values. The main reason of the decrement on the precision rate and increment on the falsification ratio is the inadequate visual coverage probability. At the beginning, we set the tolerance value $\varepsilon_2 = 0.005$ for the K-Coverage constraint. However, the sensor network allows the voting threshold to be selected at most four which satisfies the tolerance value ε_2. In order to select the required voting threshold for fault tolerance algorithm with higher p_f, we relax the K-coverage constraint by changing ε_2 from 0.005 to 0.025 which means that on average there is 2.5% chance that a grid point in the sensing field is covered by less than the selected threshold value. Therefore, it might be possible that some non-occupied regions cannot be removed and appear as targets. Thus, fault detection algorithm detects non-faulty nodes as faulty.

3.6 Summary

In this chapter, we presented a centralized and distributed collaborative target localization algorithm that can reliably detect the position of crowded targets with the existence of a number of faulty sensor nodes, detect the faulty nodes and correct them. To achieve our goal, targets are localized based on centralized and distributed camera nodes integrating the local certainty maps with a fault-tolerant fusion algorithm. Camera orientations are estimated by a generative image model in each camera to detect inaccuracy in camera orientations and correct faulty nodes. From both simulation and experimental results, we showed that the proposed fault-tolerant method is effectiveness in providing high localization accuracy as well as satisfactory fault detection and correction performance.

Chapter 4

Coverage Estimation in VSN

4.1 Introduction

In this chapter, coverage estimation, one of the fundamental problems in sensor networks is described. In many multi-camera applications, expensive and high-resolution cameras are usually deployed into large buildings and open areas to capture the events in a controlled sensing field. In general, the position and orientation of cameras are predetermined and well-ordered to optimize the placement of cameras. However, in a hostile and dangerous environment (e.g., battlefield), it is not possible or feasible to deploy the cameras with accurate position and orientation. Therefore, camera nodes might be randomly deployed into the sensing field from a moving platform (e.g., airplane or vehicle) in order to monitor the environment [Hynes et al., 2004].

Due to random deployment of sensor nodes, their positions may not be predetermined. Additionally, due to the large amount of sensors deployed, it is impractical to manipulate sensor locations after deployment in order to reach a desired coverage [Akyildiz et al., 2002]. Therefore, to have proper sensor coverage in the sensing field,

The work in this chapter was first published in Karakaya and Qi [2011a].

some sensor related parameters, such as sensor density, sensing range, etc., should be decided based on the estimated sensor coverage probability *before* deployment.

Traditionally, coverage probability has been evaluated based on the total number of deployed omnidirectional sensor nodes that captures an arbitrary target within their circular sensing range. If every target in the sensing field is captured by at least K sensor nodes, it is called a K-covered sensor network [Li and Kao, 2010]. However, coverage estimation in VSNs is more challenging than in conventional scalar sensor networks (SSNs) because of unique challenges of cameras [Karakaya and Qi, 2011b]. Therefore, visual coverage estimation in a crowded environment depends not only on the sensor density and deployment but also on the target density and distribution.

In this chapter, we focus on the formulation of a closed-form solution for the visual K-coverage estimation in VSNs with the presence of visual occlusion among crowded targets where a large number of visual sensor nodes has already been deployed. Having a closed-form solution for the coverage estimation problem facilitates many application deployments in VSNs. For example, efficient deployment of the sensor nodes can be achieved with minimum sensor density. Additionally, effective algorithms can be designed to yield optimal sensor sleep scheduling for energy saving purpose [Cai et al., 2009].

To formulate the visual coverage probability in the crowded environment, we first need to investigate into the target detection algorithm. Traditionally, targets are detected based on the identification of intersections of the back-projected 2D cones of the targets. However, the existence of visual occlusion among targets would generate many empty intersections (false alarms) which makes the derivation of a closed-form solution for visual coverage estimation extremely difficult. In this paper, instead of resolving the uncertainty about target existence at the intersections, we model the target detection algorithm based on distributed camera nodes integrating the target *non-existence* information within the camera's field of view at each sensor node. According to this target detection model, we then construct a closed-form solution to estimate the visual coverage probability that deals with the directional sensing nature of cameras and the visual occlusions among crowded targets. Based on the closed-form solution of the coverage estimation, we further propose an estimate for the minimum sensor density that suffices to ensure a visual K-coverage in a crowded sensing field.

The main contributions of this chapter are two-fold:

1. We derive a closed-form solution for visual coverage estimation in a randomly deployed VSN by adapting the *non-existence* information based target detection model into formulation. Therefore, the sensor related parameters (e.g., sensor density, sensing range, etc.) can be decided before deployment in order to have proper visual coverage in the sensing field. This facilitates many application deployments such as efficient sensor deployment and sensor scheduling in VSNs.

2. In a crowded environment, the visual coverage probability depends not only on the sensor density and deployment but also on the target density and distribution. Our closed-form solution considers both the directional sensing nature of cameras and the visual occlusions among targets to estimate the visual coverage in VSNs. Thus, we have more accurate and more realistic visual coverage estimation in a crowded VSN.

The remainder of this chapter is organized as follows: Section 4.2 briefly describes the background and related works. Section 4.3 presents the target detection model. In Section 4.4, we provide the closed-form solution for visual coverage when occlusion is not taken into consideration. And in Section 4.5, we consider the more complex problem with occlusion taken into account. To show how the proposed target detection model enables the closed-form solution for visual coverage estimation, we present a detailed comparison between the proposed and traditional target detection models for visual coverage estimation in Section 4.6. Section 4.7 investigates into the complicated boundary effect for more accurate visual coverage estimation. Based on the closed-form solution of visual coverage estimation, Section 4.8 formulates the minimum sensor density estimation problem as an application example. Section 4.9

presents the experimental results to validate the theoretical derivation of visual coverage estimation and to show the effects of various parameters on the minimum sensor density. Discussion on heterogeneous sensor deployment and target existence in VSNs and their effect on the coverage estimation is presented in 4.10.

4.2 Background and Related Works

In literature, there exist many works related to coverage estimation in scalar sensor networks (SSNs) where the sensing devices are normally 1-D omnidirectional (e.g., acoustic or seismic sensor). In order to effectively cover the given sensing region, various criteria have been considered, including quality of surveillance [Gui and Mohapatra, 2004], maximal or minimal exposure of a path [Veltri et al., 2003], area coverage [Ahmed et al., 2005], etc. In SSNs, the area coverage of a sensor node is modeled by a simple omnidirectional sensing model as a circular disk whose radius, ρ, is the sensing range of the sensor node [Huang and Tseng, 2003].

In [Meguerdichian et al., 2001], the coverage problem in SSNs was defined from several point of views including deterministic, statistical, worst and best case to determine the lower and higher observability in sensing field by combining computational geometry and graph theoretic techniques. Xing et al. [2005] presented a design of coverage configuration protocol that can dynamically configure a network to achieve guaranteed degrees of coverage and connectivity and provide a geometric analysis of the relationship between coverage and connectivity. Kumar et al. [2005] introduced the K-barrier coverage for a belt-shape region and established the optimal deployment pattern to achieve it. In [Wan and Yi, 2006], the effect of the sensing radius or the total number of deployed sensor nodes on the probability of the K-coverage was studied for randomly deployed scalar sensor nodes and the boundary effect was taken into account. In [Brass, 2007], the coverage estimation problem was analyzed with the Boolean sensing model for either mobile or stationary sensors and targets, under random or optimal placement. Yen et al. [2006] proposed a mathematical expression to predict the coverage rate for an expected area in a wireless sensor network that can be K-covered to determine the related sensing parameters. Wang et al. [2007] considered the coverage problem from the perspective of target localization to estimate the minimum sensor density to keep the target localization error within an acceptable bound.

Different from the scalar sensors, the sensing region of a camera, also referred to as the *field of view* (FOV), is limited and directional which is less than 180° in general. Therefore, existing works related to SSNs cannot be directly applied to visual sensor networks. In [Ai and Abouzeid, 2006], the directional sensor coverage problem was investigated by utilizing linear programming to maximize the sensor coverage with minimum number of sensors. An energy-efficient target-oriented sleep scheduling algorithm was presented in [Cai et al., 2009] to extend the lifetime of directional sensor networks. Liu et al. [2008] proposed directional and effective sensing models to capture the frontal view of the human face for orientation detection. Meanwhile, other research efforts Isler and Bajcsy [2006]; Yang et al. [2004] applied directional coverage analysis to minimize the sensor density to reach the accurate estimation for target localization and occupancy reasoning, respectively.

Since the coverage issue in VSNs is also related to the orientations of cameras, the problem of selecting a minimum number of sensors has been investigated based on automatic control of visual sensors by reorienting the deployed cameras to provide best possible coverage on a given area or targets. Munishwar and Abu-Ghazaleh [2010] presented a novel centralized force-based approach to compute near optimal solutions using integer linear programming in a large-scale PTZ (pan, tilt, and zoom) camera network. Fusco and Gupta [2009] designed a simple greedy algorithm that delivers a solution for selecting and orienting visual sensors that K-covers at least half of the target points using at most $M \log(k|C|)$ sensors, where $|C|$ is the maximum number of target points covered by a sensor and M is the minimum number of sensors required to K-cover all the given points. For more detailed survey, readers may refer to [Guvensan and Yavuz, 2011] where the existing coverage optimization and enhancement solutions for directional sensor networks were classified into four categories as target-based coverage enhancement, area-based coverage enhancement, coverage enhancement with guaranteed connectivity, and network lifetime prolonging. Although all these works investigated into the directional sensing models, none of which considered the visual occlusion problem among crowded targets.

For a target, to stand within the FOV of a visual sensor may not mean being captured by the camera because there may be other targets standing between the target and the camera and visually occluding them. Lin et al. [2011] developed analytical expressions to derive expected coverage by a randomly deployed single camera in a sensing field that is occlusion free or with occlusion. Then, they extended this method to the expected joint coverage after deploying additional cameras into field iteratively. Qian and Qi [2008] derived several parameters such as minimum, maximum and expectation values for visual coverage estimation in the presence of visual occlusions for VSNs. However, neither approach derived a closed-form solution for visual K-coverage probability estimation due to the target detection model used.

4.3 Target Detection Model

In traditional target detection algorithms, the intersections of the back-projected 2D visual cones of the targets are calculated to localize all the individual targets (described in detail in Chapter 2). If the cones from different sensor nodes intersect at the same point, it can be considered there is at least one target in that intersection. Existing coverage estimation algorithms are based on the information about the target "existence" at the intersections. However, this information cannot be certain since in crowded environments, many "empty" intersections that are not actually occupied by any target are created because of the visual occlusion or ghost positioning. In addition to this, existing coverage estimation approaches do not take the partial appearance of targets in the FOV of sensor nodes into account. However, in a crowded scene, it might not be realistic to have a free sight for all targets in the sensing field because of

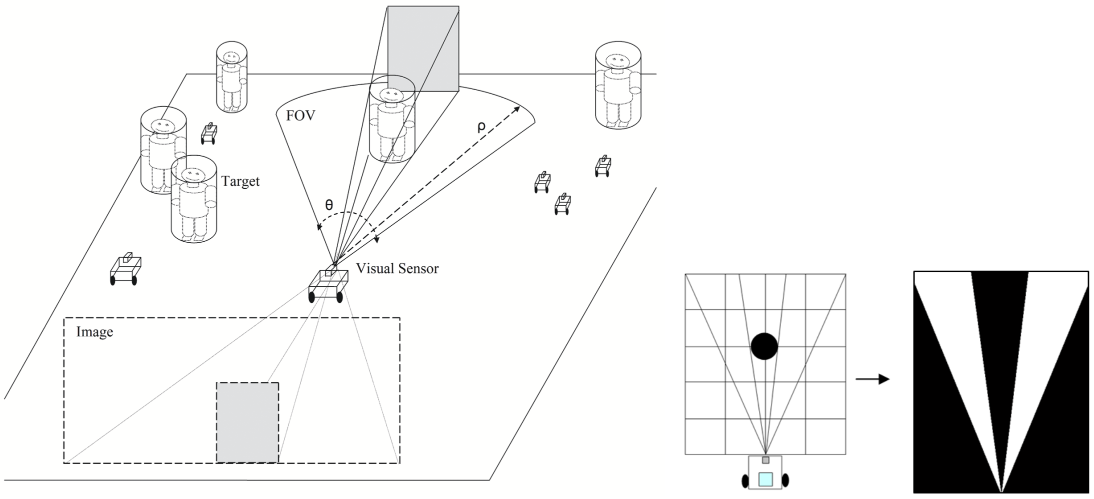


Figure 4.1: The Certainty-based Target Detection Model. (a) Visual hulls of targets in 3D (b) Projection of 3D cones onto the ground plane, and (c) Certain areas about target non-existence (labeled with white).

the visual occlusion among the crowd. Due to the uncertainty about real locations of the targets and partial appearance of targets, the derivation of a closed-form solution for the coverage estimation has been a very challenging problem. We refer to this traditional model as "uncertainty-based or occupancy-based target detection".

In this section, we briefly describe an inverse approach to traditional target detection problem proposed in Chapter 2. Instead of resolving the uncertainty about target existence, we identify and study the non-occupied areas in the visual cone. If an area within the FOV of a sensor node is not occupied or occluded by any object, it is *certain* about target non-existence (labeled with white) and declared as a non-occupied area. Otherwise, it is *uncertain* about target existence (labeled with black) in the corresponding region. The occupied areas are the ones where it is possible that there exist targets. The uncertainty is due to either occlusion or outside of the FOV of the camera. We refer to this model as the "certainty-based target detection".

The certainty-based target detection model is illustrated in Fig. 4.1. Each target is modeled by a uniform size cylindrical object in 3D where texture and shape signatures of the target are contained within the cylinder space around the axis as shown in

Fig. 4.1a. Yang et al. [2003] showed that it is reasonable to model the objects of similar heights and widths using cylinders for people and vehicle detection algorithms. After background subtraction, each target can be extracted from the scene, which sweeps a cone in 3D space as shown in Fig. 4.1a. To find visual cone of the target, these 3D cones are projected onto a plane parallel to the ground as seen in Fig. 4.1b. The non-occupied (labeled with white) areas in the visual cone are thus determined as shown in Fig. 4.1c. The detailed comparison between certainty-based and the traditional occupancy-based target detection models and their impact on the closed-form solution for visual coverage estimation will be presented in Section 4.6.

In this chapter, we assume that the infinitesimal visual sensor nodes with uniform FOV and sensing radius are randomly deployed within a very large two-dimensional sensing field, R. Since each region in the sensing field has equal importance based on the probability of target existence, all sensor nodes are uniformly and independently distributed into the sensing field. Based on this deployment strategy, the locations of visual sensor nodes can be modeled by a two-dimensional stationary Poisson point process with sensor density λ_s [Wang et al., 2010]. It is also assumed that orientations of visual sensors are uniformly distributed over $[0°, 360°)$. Let ρ and θ denote, respectively, the sensing radius and angle of view of a sensor node.

Let us model a target as a uniform disc on the 2D plane, R, with radius, r, when the cylindrical object is projected onto a plane parallel to the ground. In addition, there is no overlap between the targets and sensors in R. We further assume that the centers of all existing targets in the scene are uniformly distributed which means that the probability of any point in R to be occupied by a target is the same across the sensing field. Based on this random and uniform target distribution model, the probability that a number of target centers are located within a region, A, can be estimated by a two-dimensional stationary Poisson point process with a parameter $\lambda_t \times A$, where λ_t and A denote, respectively, the target density and the area of region A.

4.4 Visual Coverage without Visual Occlusions

If the radius of targets is infinitely small, i.e., $r \to 0$, we can ignore the visual occlusion. That is, in the "certainty-based target detection", all areas within the FOV of the sensor node would be marked as "white" (see Fig. 4.1c), for sensor coverage. In other words, a sensor node covers a specific grid point $(x, y) \in R$, of the sensing field and determines target non-existence, if the node is located in a circular area A with radius ρ centered at the corresponding grid point and is oriented towards the center of the circle. In the rest of the paper, the circular area A is referred to as the "detectability area". Therefore, the visual coverage probability, defined as the probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence is

$$P(k) = \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A)\mathcal{C}_k^j (p)^k (1-p)^{j-k} \tag{4.1}$$

where $\mathcal{P}(j; \lambda_s \times A)$ denotes the probability that a detectability area A contains exactly j sensor nodes from a Poisson point process with sensor density λ_s, i.e., $\mathcal{P}(j; \lambda_s \times A) = e^{-\lambda_s \times A}(\lambda_s \times A)^j / j!$ where $A = \pi\rho^2$. And, p denotes the probability of the sensor node facing towards the center of detectability area, A, i.e., $p = \theta/(2\pi)$ and $\mathcal{C}_k^j$ denotes the number of combinations of k-node subset from a j-node set. Eq. 4.1 can be further derived as,

$$\begin{aligned}
P(k) &= \sum_{j=k}^{\infty} \frac{e^{-\lambda_s \times A}(\lambda_s \times A)^j}{j!} \frac{j!}{k!(j-k)!}(p)^k (1-p)^{j-k} \\
&= \frac{1}{k!} e^{-\lambda_s \times A}(\lambda_s \times A \times p)^k \sum_{j=k}^{\infty} \frac{\Big((\lambda_s \times A)(1-p)\Big)^{j-k}}{(j-k)!} \\
&= \frac{1}{k!} e^{-\lambda_s \times A}(\lambda_s \times A \times p)^k e^{((\lambda_s \times A)(1-p))} \\
&= \frac{1}{k!} e^{-\lambda_s \times A \times p}(\lambda_s \times A \times p)^k \\
&= \mathcal{P}(k; \lambda_s \times A \times p)
\end{aligned} \tag{4.2}$$

From the derivation result in Eq. 4.2, we observe that the visual coverage probability without visual occlusions follows the Poisson point process with sensor density $\lambda_s \times \theta/(2\pi)$ in the detectability area A.

4.5 Visual Coverage with Visual Occlusions

In an environment with crowded targets, it is no longer appropriate to assume an infinitely small target and target radius r becomes a finite value, i.e., $r > 0$. Hence, visual occlusions should be taken into account. To cover a specific grid point of the sensing field and determine the target non-existence at that point, not only the corresponding grid point must be inside the FOV of the sensor node, the centers of all targets should also be outside of the occlusion zone between the corresponding grid point and the node, which is illustrated as the bold-boundary region in Fig. 4.2.

The shaded region in Fig. 4.2 is the area of the occlusion zone, denoted as A_o. The value of A_o depends on the target radius r and the distance l between the corresponding grid point and visual sensor node and can be expressed as $A_o = \pi r^2 + 2rl$. Let q denote the probability that there is no visual occlusion between the grid point and the sensor node. Since the probability that a number of target centers are located within a region, A_o, follows Poisson distribution, the probability of having no targets in the occlusion zone, q equals to $e^{-\lambda_t(\pi r^2+2rl)}$ which is a random value with respect to the randomness of the distance l between the grid point and the visual sensor node. Let Q denote the probability of covering a specific grid point of the sensing field and determining the target non-existence. Since the visual coverage depends on two independent factors, i.e., the grid point is within the FOV of the sensor and that there is no occlusion between the sensor and the grid point, Q can be expressed as

$$Q = p \times q = \frac{\theta}{2\pi} \times e^{-\lambda_t(\pi r^2+2rl)} \tag{4.3}$$

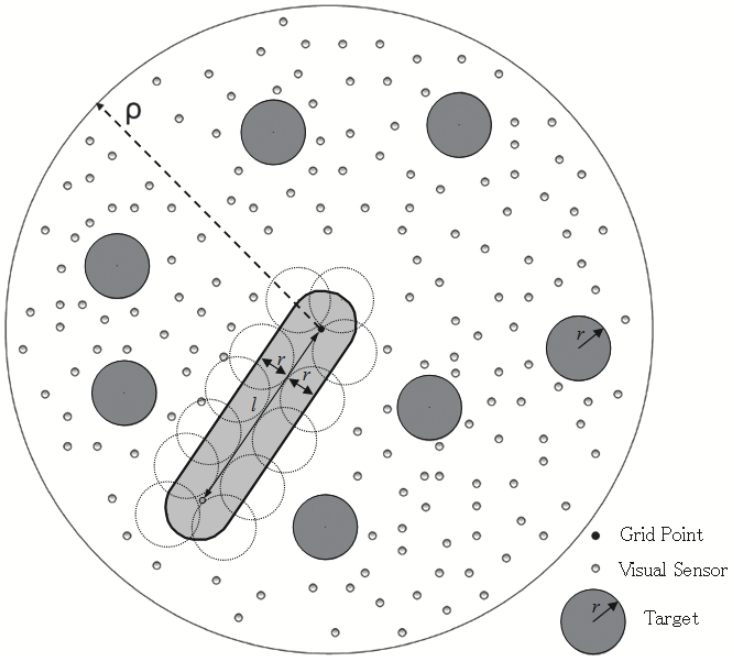


Figure 4.2: Occlusion zone model.

Thus, the visual coverage probability that exactly k nodes cover a specific grid point of the sensing field and determine the target non-existence, $P(k)$, is

$$P(k) = \int P(k, Q) f(Q) dQ \tag{4.4}$$

where $P(k, Q)$ is the probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence at that point with respect to Q, and $f(Q)$ is the probability density function (pdf) of Q with respect to distance l_i between the corresponding grid point and each node s_i in the circular detectability area A with radius ρ centered at the grid point, $i = 1 \ldots N_s$, and N_s is the number of visual sensor nodes within area A.

Since sensor nodes are uniformly distributed at random in the sensing field, the probability of sensor nodes appears at the same distance to the center of the circular detectability area A is proportional to the area of the region. Therefore, $f(l)$, the pdf of distance l_i between the corresponding grid point and each sensor node s_i, follows linear distribution from 0 to ρ

$$f(l) = \begin{cases} \frac{2}{\rho^2} l & \text{for } 0 \le l \le \rho \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

To calculate the pdf of function Q, $f(Q)$, we utilize the change of variable property (see detailed derivation in Appendix B). Since Q is a monotonically decreasing function of l, $f(Q)$ is

$$f(Q) = \begin{cases} \frac{2}{\rho^2} \times \frac{\ln\left(\frac{\frac{\theta}{2\pi} e^{-\lambda_t \pi r^2}}{Q}\right)}{2\lambda_t r} \times \frac{1}{2\lambda_t r Q} & \text{for } Q(l=\rho) \le Q \le Q(l=0) \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

$P(k, Q)$ can be derived as

$$\begin{aligned}
P(k,Q) &= \sum_{j=k}^{\infty} \sum_{i=k}^{j} \mathcal{P}(j; \lambda_s \times A) \mathcal{C}_i^j p^i (1-p)^{j-i} \mathcal{C}_k^i q^k (1-q)^{i-k} \\
&= \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^k q^k (1-p)^{j-k} \sum_{i=k}^{j} \mathcal{C}_i^j \mathcal{C}_k^i \left(\frac{p(1-q)}{1-p}\right)^{i-k} \\
&= \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^k q^k (1-p)^{j-k} \sum_{s=0}^{j-k} \mathcal{C}_{k+s}^j \mathcal{C}_k^{k+s} \left(\frac{p(1-q)}{1-p}\right)^{s} \\
&\overset{(a)}{=} \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^k q^k (1-p)^{j-k} \sum_{s=0}^{j-k} \mathcal{C}_k^j \mathcal{C}_s^{j-k} \left(\frac{p(1-q)}{1-p}\right)^{s} \\
&= \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^k q^k (1-p)^{j-k} \mathcal{C}_k^j \sum_{s=0}^{j-k} \mathcal{C}_s^{j-k} \left(\frac{p(1-q)}{1-p}\right)^{s} \\
&\overset{(b)}{=} \sum_{j=k}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^k q^k (1-p)^{j-k} \mathcal{C}_k^j \left(\frac{p(1-q)}{1-p} + 1\right)^{j-k} \\
&= \sum_{j=k}^{\infty} e^{-\lambda_s \times A} (\lambda_s \times A)^j \frac{1}{j!} p^k q^k (1-p)^{j-k} \frac{j!}{k!(j-k)!} \left(\frac{1-pq}{1-p}\right)^{j-k}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{k!} e^{-\lambda_s \times A} (\lambda_s Apq)^k \sum_{j=k}^{\infty} \Big[(\lambda_s \times A)(1-pq)\Big]^{j-k} \frac{1}{(j-k)!} \\
&= \frac{1}{k!} e^{-\lambda_s \times A} (\lambda_s Apq)^k e^{(\lambda_s \times A)(1-pq)} \\
&= \frac{1}{k!} (\lambda_s Apq)^k e^{-\lambda_s Apq} \\
&= \mathcal{P}(k; \lambda_s \times A \times Q) \qquad (4.7)
\end{aligned}
$$

where (a) follows the combination properties, $\mathcal{C}^j_{k+s}.\mathcal{C}^{k+s}_k = \mathcal{C}^j_k.\mathcal{C}^{j-k}_s$, and (b) follows the binomial coefficient property, $(x+y)^n = \sum_{s=0}^n \mathcal{C}^n_s x^{n-s} y^s$ where $s = i - k$. Also, $A = \pi\rho^2$, $p = \theta/(2\pi)$, $q = e^{-\lambda_t(\pi r^2 + 2rl)}$, and $Q = p \times q$.

From the derivation result in Eq. 4.7, we observe that the visual coverage probability with visual occlusions also follows the Poisson point process with the sensor density $\lambda_s \times \theta/(2\pi) \times e^{-\lambda_t(\pi r^2+2rl)}$ in area A. If $\lambda_t \to 0$ or $r \to 0$, then $Q \to p = \theta/(2\pi)$ which means visual occlusions among the targets can be ignored. Therefore, Eq. 4.7 converges to Eq. 4.2.

The derivation of the visual coverage probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence at that point can then be derived as

$$
\begin{aligned}
P(k) &= \int \mathcal{P}(k; \lambda_s \times A \times Q) f(Q) dQ \\
&= \int_{Q(l=\rho)}^{Q(l=0)} \frac{e^{-\lambda_s AQ}(\lambda_s AQ)^k}{k!} \times \frac{2}{\rho^2} \times \frac{\ln\left(\frac{\frac{\theta}{2\pi} e^{-\lambda_t \pi r^2}}{Q}\right)}{4\lambda_t^2 r^2 Q} dQ \qquad (4.8)
\end{aligned}
$$

Let $u = \lambda_s \pi \rho^2 Q$, $\lambda_c = \lambda_s \pi \rho^2 \frac{\theta}{2\pi} e^{-\lambda_t \pi r^2}$ and $\lambda_b = \lambda_c \times e^{-2\lambda_t r \rho}$

$$
P(k) = \frac{1}{2k!\rho^2 r^2 \lambda_t^2} \int_{u_{Q(l=\rho)}}^{u_{Q(l=0)}} u^k e^{-u} \times \frac{\ln\left(\frac{\frac{\theta}{2\pi} e^{-\lambda_t \pi r^2} \lambda_s \pi \rho^2}{u}\right)}{\frac{u}{\lambda_s \pi \rho^2}} \times \frac{du}{\lambda_s \pi \rho^2}
$$

$$\begin{aligned}
&= \frac{1}{2k!\rho^2 r^2 \lambda_t^2} \int_{\lambda_b}^{\lambda_c} u^{k-1} e^{-u} \ln(\frac{\lambda_c}{u}) du \\
&= \frac{1}{2k!\rho^2 r^2 \lambda_t^2} \left[\underbrace{\int_{\lambda_b}^{\lambda_c} u^{k-1} e^{-u} \ln \lambda_c du}_{P_1(k)} - \underbrace{\int_{\lambda_b}^{\lambda_c} u^{k-1} e^{-u} \ln u du}_{P_2(k)} \right] \qquad (4.9)
\end{aligned}$$

where

$$\begin{aligned}
P_1(k) &= \ln \lambda_c \int_{\lambda_b}^{\lambda_c} u^{k-1} e^{-u} du \\
&= \ln \lambda_c \left[\int_{\lambda_b}^{\infty} u^{k-1} e^{-u} du - \int_{\lambda_c}^{\infty} u^{k-1} e^{-u} du \right] \\
&= \ln \lambda_c \left[\Gamma(k, \lambda_b) - \Gamma(k, \lambda_c) \right] \qquad (4.10)
\end{aligned}$$

and

$$\begin{aligned}
P_2(k) &= \int_{\lambda_b}^{\lambda_c} u^{k-1} e^{-u} \ln u du \\
&= \int_{\lambda_b}^{\infty} u^{k-1} e^{-u} \ln u du - \int_{\lambda_c}^{\infty} u^{k-1} e^{-u} \ln u du \\
&= \frac{\partial}{\partial k} \Gamma(k, \lambda_b) - \frac{\partial}{\partial k} \Gamma(k, \lambda_c) \\
&\overset{(a)}{=} \left[\ln \lambda_b \Gamma(k, \lambda_b) + \lambda_b \mathrm{T}(3, k, \lambda_b) \right] - \left[\ln \lambda_c \Gamma(k, \lambda_c) + \lambda_c \mathrm{T}(3, k, \lambda_c) \right]
\end{aligned}$$

where $\Gamma(k, \lambda)$ is the upper incomplete gamma function and $\frac{\partial}{\partial k}\Gamma(k, \lambda)$ is its first derivative with respect to k [Abramowitz, 1970]. In this derivation, (a) follows $\frac{\partial}{\partial k}\Gamma(k, \lambda) = \ln \lambda \Gamma(k, \lambda) + \lambda \mathrm{T}(3, k, \lambda)$ where $\mathrm{T}(3, k, \lambda)$ is a special case of Meijer G-function [Geddes et al., 1990]. By using the simple recurrence formula of function $\mathrm{T}(3, k, \lambda)$, we can find its generalized recurrence formula as $\lambda \mathrm{T}(3, k, \lambda) = (k - 1)!\mathrm{E}_1(\lambda) + \sum_{i=1}^{k-1} \frac{k-1!}{i!} \Gamma(i, \lambda)$ where $\mathrm{E}_1(\lambda)$ is the exponential integral (see detailed derivation in Appendix B.3). These special cases of function $\mathrm{T}(3, k, \lambda)$ provide an extension of $P_2(k)$ as

$$P_2(k) = \left[\ln \lambda_b \Gamma(k, \lambda_b) + (k-1)!\mathrm{E}_1(\lambda_\mathrm{b}) + (k-1!)\sum_{i=1}^{k-1} \frac{\Gamma(i, \lambda_b)}{i!}\right] - \left[\ln \lambda_c \Gamma(k, \lambda_c) + (k-1)!\mathrm{E}_1(\lambda_\mathrm{c}) + (k-1)!\sum_{i=1}^{k-1} \frac{\Gamma(i, \lambda_c)}{i!}\right] \tag{4.11}$$

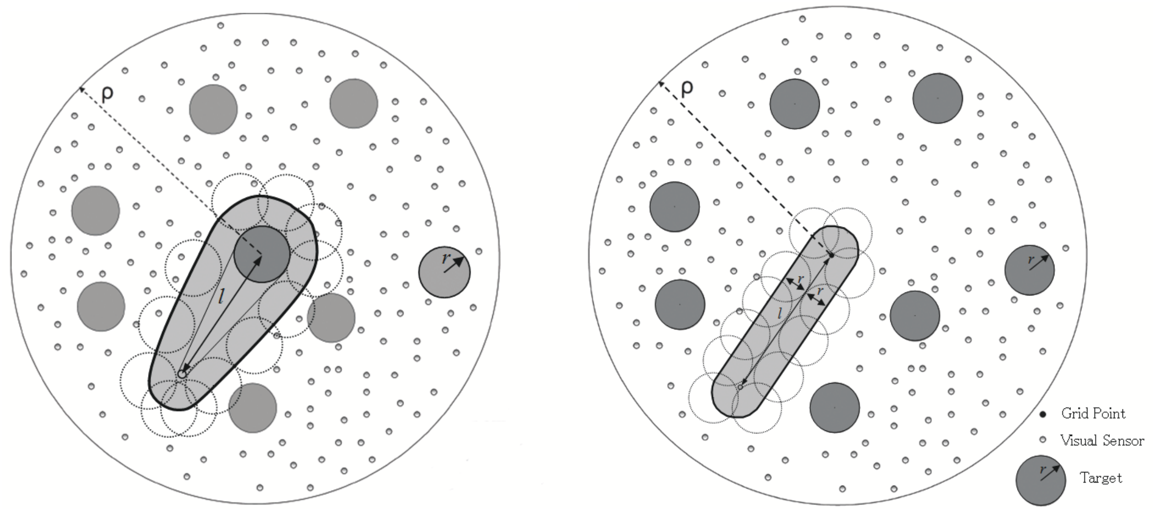
By substituting Eqs. 4.10 and 4.11 into Eq. 4.9, a closed-form solution of the visual coverage probability, $P(k)$ can be further derived as,

$$\begin{aligned} P_k &= \frac{1}{2k!\rho^2 r^2 \lambda_t^2}\Big[P_1(k) - P_2(k)\Big] \\ &= \frac{1}{2k!\rho^2 r^2 \lambda_t^2}\left[\ln\left(\frac{\lambda_c}{\lambda_b}\right)\Gamma(k, \lambda_b) - (k-1)!\Big(\mathrm{E}_1(\lambda_\mathrm{b}) - \mathrm{E}_1(\lambda_\mathrm{c})\Big) - \right. \\ &\qquad \left. (k-1)!\sum_{i=1}^{k-1} \frac{\Gamma(i, \lambda_b) - \Gamma(i, \lambda_c)}{i!}\right] \\ &\overset{(a)}{=} \frac{1}{2k\rho^2 r^2 \lambda_t^2}\left[\ln\left(\frac{\lambda_c}{\lambda_b}\right)F_{\mathcal{P}}(k-1, \lambda_b) - \Big(\mathrm{E}_1(\lambda_\mathrm{b}) - \mathrm{E}_1(\lambda_\mathrm{c})\Big) - \right. \\ &\qquad \left. \sum_{i=1}^{k-1} \frac{F_{\mathcal{P}}(i-1, \lambda_b) - F_{\mathcal{P}}(i-1, \lambda_c)}{i}\right] \end{aligned} \tag{4.12}$$

where (a) follows $\Gamma(a, x) = \int_x^\infty e^{-t}t^{a-1}dt$. When a is an integer, $\Gamma(n, x) = (n-1)!e^{-x}\sum_{j=0}^{n-1}\frac{x^j}{j!}$ [Press et al., 2007]. In this derivation, $F_{\mathcal{P}}(k; \lambda)$ is the cumulative probability distribution (cdf) of Poisson distribution with parameter λ.

4.6 Comparison between Occupancy-based and Certainty-based Visual Coverage

In this section, we explicitly discuss the advantages of adapting the certainty-based target detection approach that integrates target non-existence information versus the occupancy-based target detection approach that integrates target existence



(a) **(b)**

Figure 4.3: (a) Occupancy-based occlusion zone model and (b) Certainty-based occlusion zone model.

information in deriving the closed-form solution for visual coverage estimation in VSNs with visual occlusions.

In traditional occupancy-based target detection algorithms, since the intersections of the back-projected 2D visual cones of the targets are calculated to localize all the individual targets, occupancy maps hold the information about total number of sensor nodes detecting target existence at each intersection. Therefore, existing coverage estimation algorithms are based on the occupancy information about the target "existence" at the intersections and define the target detection only if the front arc of the disc bounded by two tangent viewing rays is completely visible to the sensor node, as shown in Fig. 4.3a. More specifically, in the occupancy-based model, to declare target existence at a specific intersection, not only the arc of the disc bounded by tangent rays must be inside the FOV of the sensor node, the centers of all other targets should also be outside of the occlusion zone between the corresponding target and the sensor node, which is illustrated as the bold-boundary region in Fig. 4.3a.

The shaded regions in Fig. 4.4 are the occupancy-based occlusion zones, denoted as A_o which are random values with respect to the randomness of the distance l

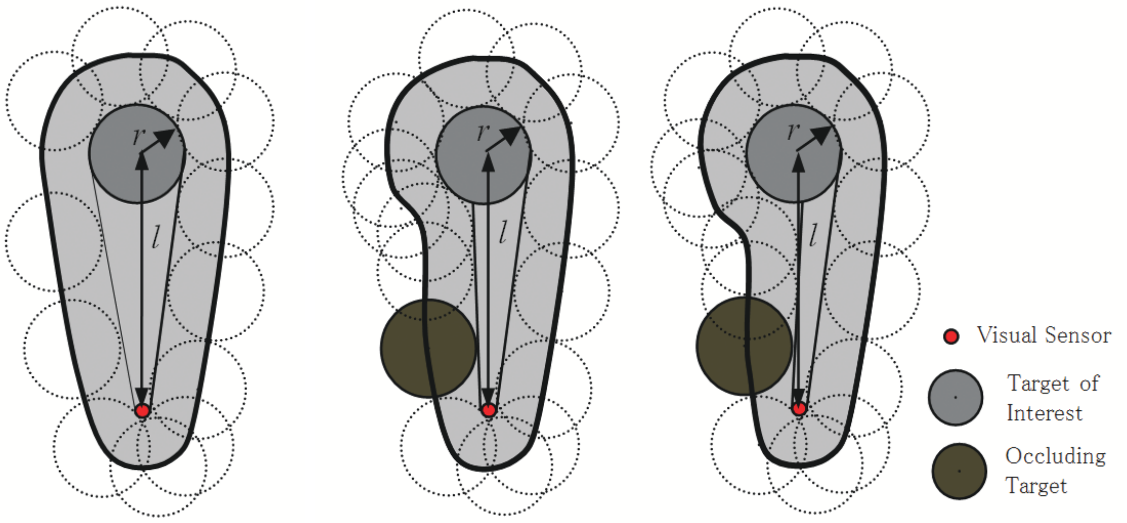


Figure 4.4: Occupancy-based occlusion zone if (a) there is free sight or (b) partial appearance of targets exists due to another occluding target.

between the target and the sensor node. Since the visual coverage depends on two independent factors, i.e., the target is within the FOV of the sensor and that there is no occlusion between the sensor and the target, the probability of covering a specific target, Q, can be

$$Q(l) = p \times q = \frac{\theta - 2\arcsin(r/l)}{2\pi} \times e^{-\lambda_t \times A_o} \tag{4.13}$$

where p denotes the probability of the sensor node facing towards the target, i.e., $p = \Big(\theta - 2\arcsin(r/l)\Big)/(2\pi)$ and q denotes the probability of having no targets in the occlusion zone i.e., $q = e^{-\lambda_t \times A_o}$.

However, Eq. 4.13 is valid if and only if free sight is available for all targets in the sensing field as illustrated as the bold-boundary regions in Fig. 4.4a. In a crowded scene, it is not appropriate to assume the existence of a free sight for all targets in the sensing field because of the visual occlusions among targets. Since some targets partially appear in the FOV of sensor nodes, the area of the occlusion zone, A_o becomes a random variable with respect to not only the distance l but also the location of other occluding targets as illustrated as the bold-boundary regions in Fig. 4.4b. Therefore, the probability density function (pdf) of Q, $f(Q)$ could not be calculated explicitly. As a result, due to the partial appearance of targets, the derivation of a closed-form solution for the occupancy-based coverage estimation has been a very challenging problem in VSNs.

To solve this problem, we adopt certainty-based target detection model in the derivation of visual coverage estimation in VSNs. Instead of resolving the uncertainty about target existence, we identify and study the non-occupied areas in the visual cone to detect targets. When the non-existence information coming from different sensor nodes is fused in a certainty map to remove the uncertainty, the only uncertain regions left would be the location of targets. In other words, certainty maps hold the information about total number of sensor nodes detecting target non-existence at any specific region. Therefore, in the certainty-based approach, to cover a specific grid point of the sensing field and determine the target non-existence at that point, not only the corresponding grid point must be inside the FOV of the sensor node, the centers of all targets should also be outside of the occlusion zone between the corresponding grid point and the node, which is illustrated as the bold-boundary region in Fig. 4.3b.

Unlike the occupancy-based occlusion zone model, the area of the certainty-based occlusion zone, A_o depends on only the distance l between the corresponding grid point and sensor node and can be expressed as $A_o = \pi r^2 + 2rl$. As described in Section 4.5, the calculation the pdf of function $Q(l)$, $f(Q)$, is utilized by the change of variable property on function $f(l)$, the pdf of distance l. Therefore, the certainty-based model enables the computation the pdf of function $Q(l)$, $f(Q)$. As a result, the derivation of the closed-form solution for visual coverage estimation in VSNs has been possible by adopting the certainty-based target detection model.

4.7 Boundary Effect on the Visual Coverage Estimation

In this section, we investigate the boundary effect on the visual coverage estimation. For a sensor node close to the boundary of the sensing field, part of the area within its FOV will fall outside of the sensing field R. Therefore, the visual sensor coverage probability at the boundary of the sensing field is less than that in central areas of the sensing field. This is commonly referred to as the *boundary effect* in sensor networks.

As shown in the derivation of Eq.4.12, the visual coverage probability $P(k)$ denotes the probability that a specific grid point within the sensing field R is covered by exactly k many visual sensor nodes out of j many nodes distributed in a circular detectability area A with radius ρ centered at the grid point. If the boundary effect is ignored, $A = \pi\rho^2$ holds for all grid points within the sensing field, R, so $P(k)$ is similar at all points in R as well. However, due to the boundary effect, the grid points close to the boundary have a partial circular detectability area $A(x, y)$, shown as gray regions in Fig.4.5 and $A(x, y) \leq \pi\rho^2$. Therefore, visual coverage probability $P(k)$ depends on the location in the sensing field R.

Yen et al. [2006] discussed region partitioning to estimate the boundary effect on the expected coverage in wireless sensor networks according to the locations of omnidirectional scalar sensors. Following the similar partitioning idea but with different partitioning approaches for visual sensor networks, we divide the sensing field R into three types of sub-regions according to the location of grid point (x,y). Let A^C, A^S, A^M represent the sub-regions where a grid point is located in the corner sub-regions, side sub-regions and middle sub-regions of the sensing field R, respectively.

As shown in Fig.4.5, the detectability area of each grid point in the middle sub-region A^M has circular shape because distances of a grid point to the two closest borders of the sensing field R is more than the sensing range of the sensor, ρ. Therefore, $A^M(x, y) = \pi\rho^2$, $\forall(x, y) \in A^M$. In the following subsections, we estimate

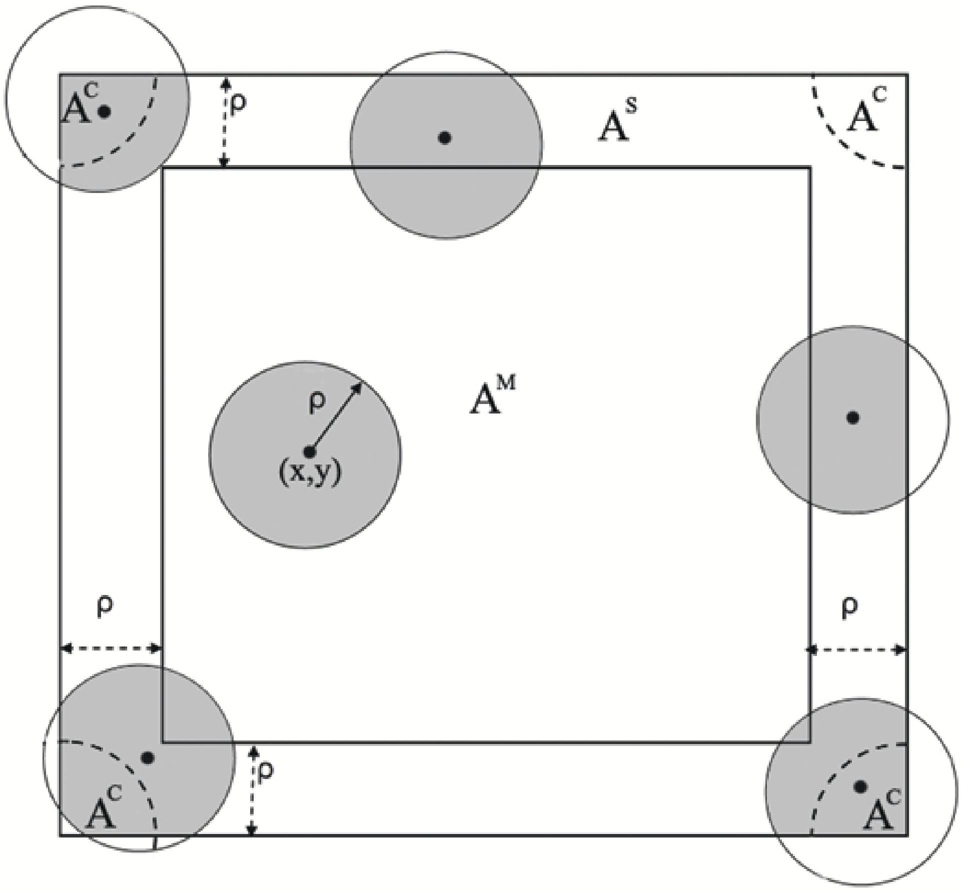


Figure 4.5: Partitioned boundary sub-regions of a rectangular sensing field.

the detectability area of the corner sub-regions A^C and side sub-regions A^S within the sensing field R.

4.7.1 Computing Detectability Area at the Corner Sub-region A^C

Fig.4.6 (top) illustrates the detectability area of a grid point in a corner sub-region A^C where the distance of a grid point to the closest corner is less than the sensing range of a visual sensor, ρ. Let u and v denote the minimum distances from a grid point in a corner sub-region A^C to two borders of the $M \times N$ rectangle sensing field $R^{M\times N}$, respectively, i.e., $u = \min(x, M - x)$, $v = \min(y, N - y)$ and $u^2 + v^2 \le \rho^2$.

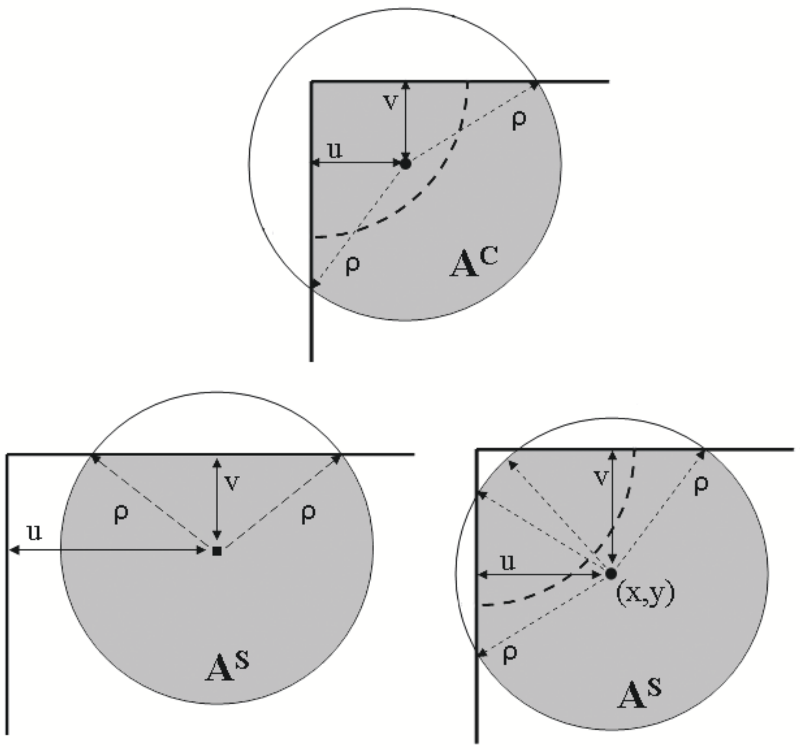


Figure 4.6: The detectability areas in (top) corner sub-region A^C, (bottom) side sub-region A^S.

By geometry, the detectability area of a grid point in a corner sub-region $A^C(x, y)$ is expressed as,

$$A^C(x,y) = u \times v + \frac{u\sqrt{\rho^2 - u^2}}{2} + \frac{v\sqrt{\rho^2 - v^2}}{2} + \Big(\frac{\frac{\pi}{2} + \arcsin(\frac{u}{\rho}) + \arcsin(\frac{v}{\rho})}{2\pi}\Big)\pi\rho^2 \quad (4.14)$$

Thus, the detectability area of a grid point in a corner sub-region A^C decreases as the point is located closer to the corner of the sensing field R. Based on the decrease in the detectability area, the visual coverage probability $P(k)$ decreases accordingly.

4.7.2 Computing Detectability Area at the Side Sub-region A^S

Fig.4.6 (bottom) illustrates two types of detectability areas of grid points in a side sub-region A^S where at least one of the distances between a grid point and the two closest borders of the sensing field R is less than the sensing range of a visual sensor, ρ and the distance of a grid point to the closest corner is larger than ρ. Let u and v, again, denote the minimum distances from a grid point in a side sub-region A^S to borders of the $M \times N$ rectangle sensing field $R^{M\times N}$, respectively, i.e., $u = \min(x, M-x)$, $v = \min(y, N-y)$, $u^2 + v^2 > \rho^2$ and $u \leq \rho$ or $v \leq \rho$. By geometry, three types of detectability area of a grid point in a side sub-region $A^S(x,y)$ can be expressed as,

$$A^S(x,y) = \begin{cases} u\sqrt{\rho^2-u^2} + v\sqrt{\rho^2-v^2} + \\ \left(\frac{2\pi - 2\arccos(\frac{u}{\rho}) - 2\arccos(\frac{v}{\rho})}{2\pi}\right)\pi\rho^2, & \text{if } u \leq \rho \text{ and } v \leq \rho \\ u\sqrt{\rho^2-u^2} + \left(\frac{2\pi - 2\arccos(\frac{u}{\rho})}{2\pi}\right)\pi\rho^2, & \text{if } u \leq \rho \text{ and } v > \rho \\ v\sqrt{\rho^2-v^2} + \left(\frac{2\pi - 2\arccos(\frac{v}{\rho})}{2\pi}\right)\pi\rho^2, & \text{if } u > \rho \text{ and } v \leq \rho \end{cases} \tag{4.15}$$

Thus, the detectability area of a grid point in a side sub-region A^S decreases as the point gets closer to the borders of the sensing field R. Based on the decrease in the detectability area, the visual coverage probability $P(k)$ decreases in a side sub-region A^S accordingly.

4.8 Minimum Sensor Density Estimation

In many visual sensor deployment applications, one of the major tasks is to find accurate estimation of the minimum sensor density to deploy into the sensing field which is sufficient to ensure the visual coverage probability that each point is covered by at least K sensor nodes is higher than a certain percentage. In other words, the probability that each point is covered by less than K sensor nodes is smaller than a tolerance value ε. The optimization problem of minimum node density to ensure visual K-coverage can be expressed as,

$$\hat{\lambda}_s = \underset{\substack{\lambda_s \\ s.j. P(k) \geq 0}}{\arg\min} \left| \sum_{k=0}^{K-1} P(k) - \varepsilon \right| \tag{4.16}$$

where $P(k)$ is parameterized by λ_s and other fixed parameters λ_t, r, ρ, and θ as shown in Eq. 4.12. Therefore, the solution for optimization problem is that minimum node density is the smallest positive root $\hat{\lambda}_s$ of the following equation

$$\sum_{k=0}^{K-1} \frac{1}{2k\rho^2 r^2 \lambda_t^2} \Bigg[\ln\left(\frac{\lambda_c}{\lambda_b}\right) F_p(k-1, \lambda_b) - \Big(\mathrm{E}_1(\lambda_\mathrm{b}) - \mathrm{E}_1(\lambda_\mathrm{c})\Big) - \sum_{i=1}^{k-1} \frac{F_p(i-1, \lambda_b) - F_p(i-1, \lambda_c)}{i} \Bigg] = \epsilon \tag{4.17}$$

where λ_c and λ_b are the Poisson distribution parameters, i.e., $\lambda_c = \lambda_s \pi \rho^2 \frac{\theta}{2\pi} e^{-\lambda_t \pi r^2}$ and $\lambda_b = \lambda_s \pi \rho^2 \frac{\theta}{2\pi} e^{-\lambda_t (\pi r^2 + 2\lambda_t r \rho)}$. There is no explicit solution for Eq. 4.17, so minimum sensor density, $\hat{\lambda}_s$ can be found by using the exhaustive search method.

4.9 Experiments and Results

In this section, we first present the comparison between the simulation results and theoretical values to validate the theoretical derivation of visual coverage probability. Then, the results of minimum sensor density $\hat{\lambda}_s$ are presented to show the effect of visual occlusions among crowded targets on the visual coverage probability that ensures the K-coverage in the sensing field.

In our simulations, circular targets with uniform size are deployed on a 2D sensing field, infinitely small-size sensor nodes with uniform FOV and focal length are located and directed horizontally facing the sensing field. The locations of each sensor node and target are randomly generated assuming there is no overlap between the targets and sensor nodes. The orientation of each node is a floating point number randomly generated in $[0°, 360°)$. In all the simulations, we assume each sensor node is able to

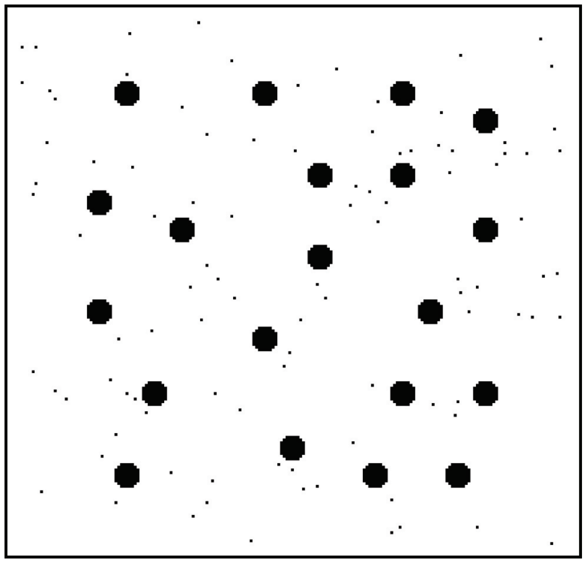


Figure 4.7: Simulation setup with 20 targets and 100 sensor nodes.

find its orientation and location by using a digital compass and positioning system, such as GPS.

Following is the setup of some typical parameters: The 2D sensing field is 40m $\times$ 40m large. The radius of each target, r is 0.5m. Each sensor node has a uniform FOV with $\rho = 10$m of sensing range and $\theta = 45°$ of angle of view. Each node is in the communication range of other nodes and is able to communicate with each other. Fig. 4.7 illustrates a sample random deployment of 20 targets, represented as discs, and 100 cameras, represented as points.

We conduct two sets of experiments to validate the theoretical derivation of visual coverage probability, where one set does not consider the boundary effect and the other one does. We also show the effect of parameter selection on the minimum sensor density, $\hat{\lambda}_s$. In each set of experiments, different amount of coverage requirements K ($K = 1$, 2, 3) are selected for ten times and the results are averaged.

4.9.1 Comparison between Theoretical Values and Experimental Results Without Boundary Effects

In this set of experiments, boundary effect is not considered. The effects of two groups of parameters are studied, including sensor node related parameters and target related parameters.

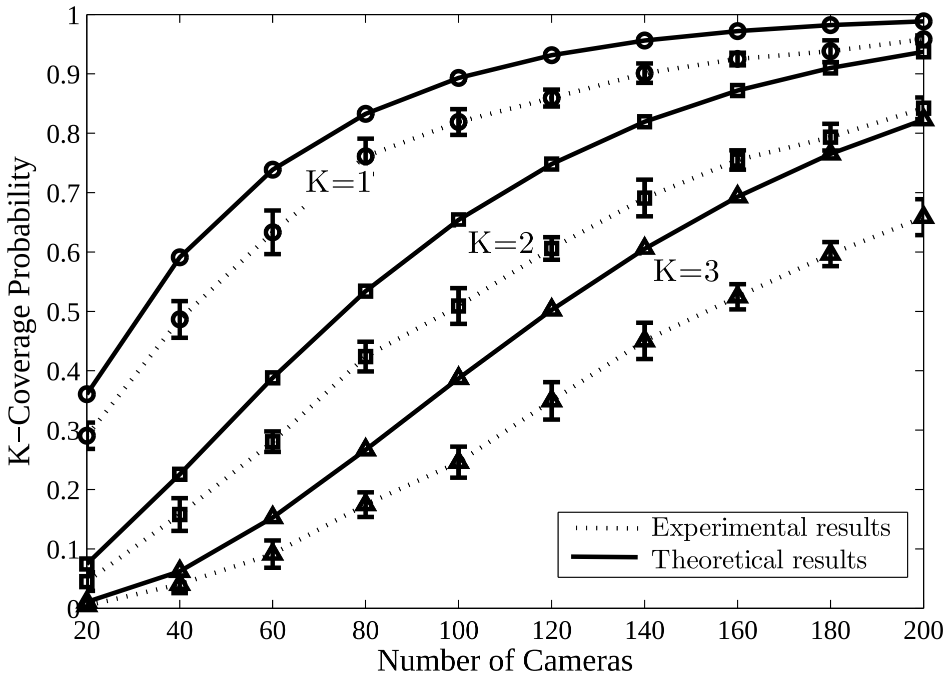
Effect of Sensor Node Related Parameters

In this experiment, we study the effect of the sensor node related parameters, sensor node density λ_s, sensing range ρ, and angle of view θ on the visual coverage probability for different visual K-coverage requirements.

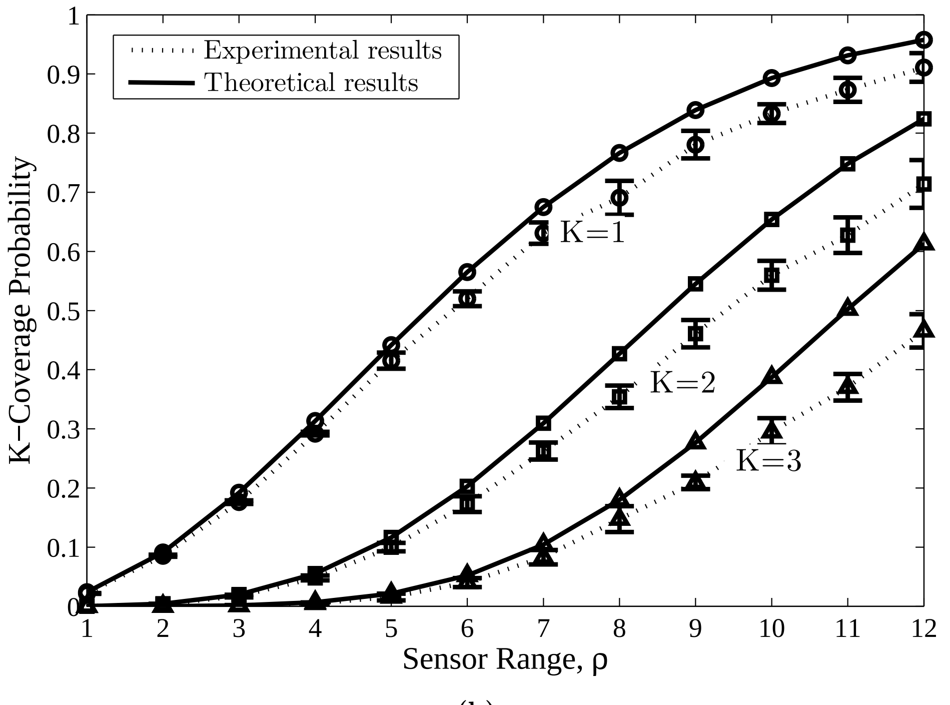
In the first simulation, 20 targets and different numbers of sensor nodes are randomly deployed into the sensing field. Fig. 4.8a shows the visual coverage probability for different K values corresponding to different numbers of sensor nodes, N_s. We observe that visual coverage probability decreases as K increases because of the more demanding coverage requirement. In addition, visual coverage probability increases as N_s increases due to more dense visual sensor nodes deployed.

Secondly, fixed number of sensor nodes and fixed number of targets are deployed into the sensing field where $N_s = 100$ and $N_t = 20$. However, in each deployment, we vary the value of the uniform sensing range ρ of every visual sensor node. Fig. 4.8b shows the visual coverage probability for different K values corresponding to different sensing range ρ. We observe that visual coverage probability increases as ρ increases or K decreases because of larger visual coverage of each sensor node with larger FOV and less demanding coverage requirements, respectively.

In the third simulation, we select different values for the angle of view θ of each sensor node to show its effect on the visual coverage probability where fixed number of targets and fixed number of sensor nodes with fixed sensing range ρ are deployed into the sensing field, i.e. $N_t = 20$, $N_s = 100$ and $\rho = 10$m. Fig. 4.9 shows the visual coverage probability for different K values corresponding to different angles of view



(a)



(b)

Figure 4.8: Comparison of theoretical values and simulation results corresponding to sensor node related parameters, (a) different numbers of sensor nodes N_s, (b) different sensing range ρ and (c) different angle of views θ.

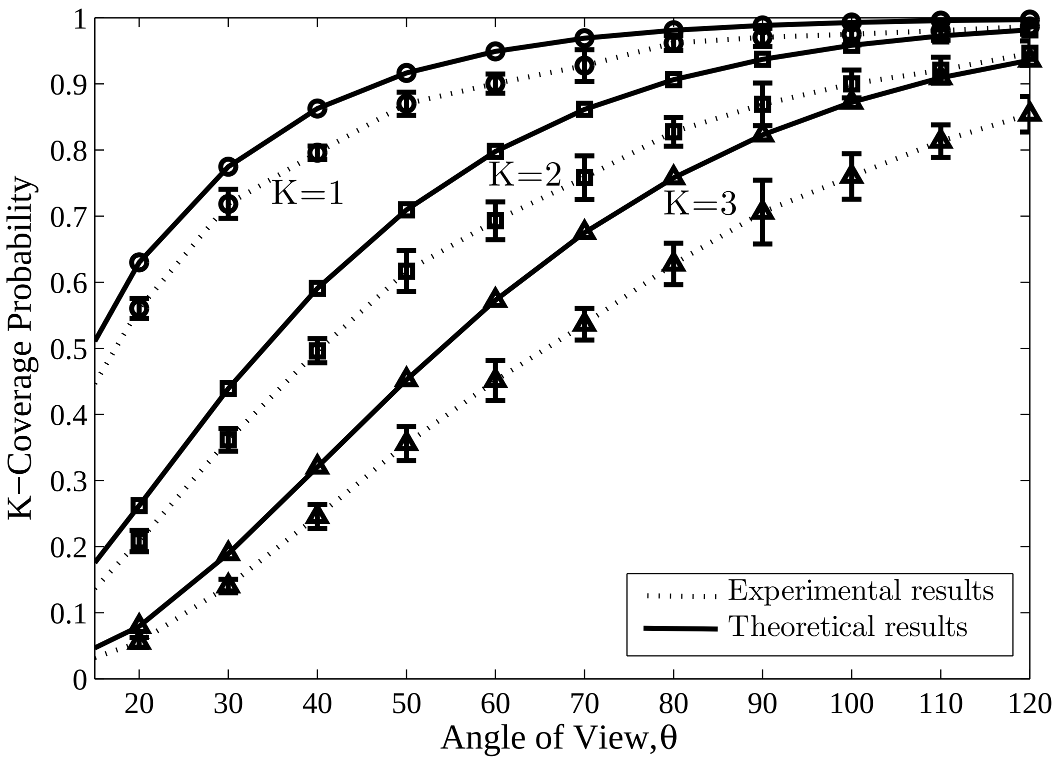


Figure 4.9: Comparison of theoretical values and simulation results corresponding to different angle of views θ.

θ. We observe that visual coverage probability increases as θ increases or K decreases because of, again, larger visual coverage of each sensor node with larger FOV and less demanding coverage requirements, respectively.

As shown in Fig. 4.8 and 4.9, the simulated experimental results for the sensor node related parameters, sensor node density λ_s, sensing range ρ, and angle of view θ are consistent with the theoretical values. However, because of the boundary effect, we observe that the visual K-coverage probability resulted from simulated experiments are slightly less than theoretical values. Moreover, the difference between theoretical values and experimental results increases as either N_s or ρ or θ increases because the boundary effect is more severe.

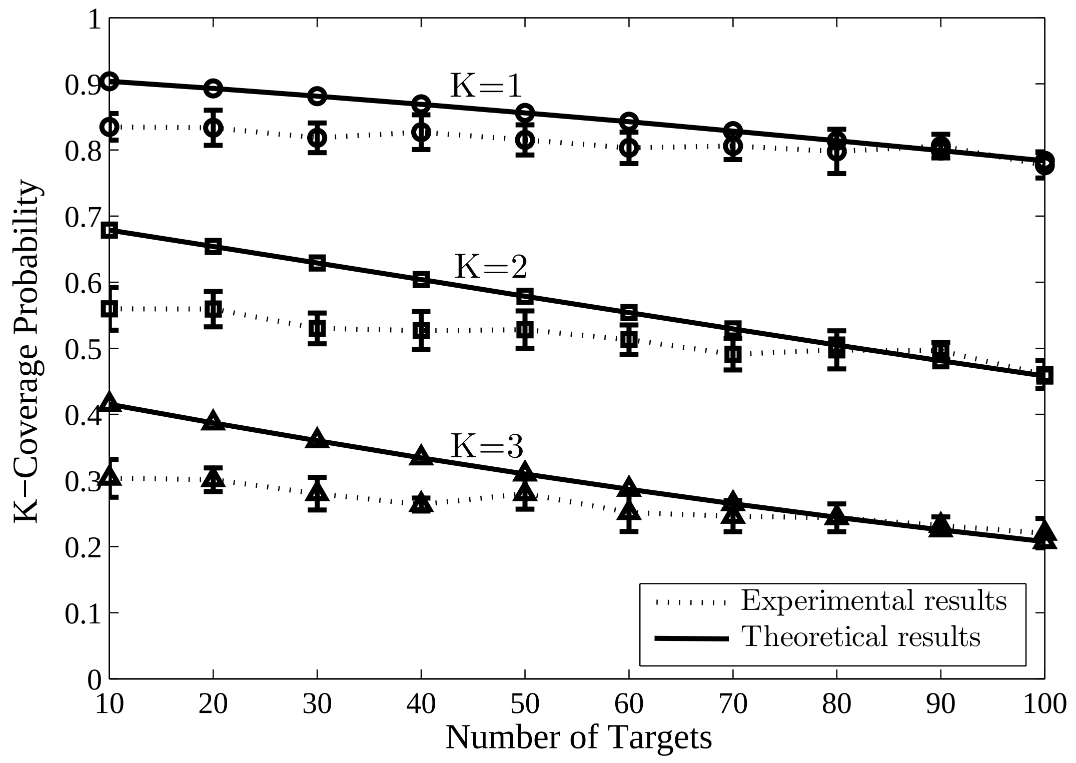


Figure 4.10: Comparison of theoretical values and simulation results corresponding to different number of targets N_t.

Effect of Target Related Parameters

In this experiment, we study the effect of the target related parameters, the number of deployed target N_t and target radius r on the visual coverage probability for different visual K-coverage.

First, different numbers of target, N_t, are deployed in the sensing field where the total number of cameras, N_s, is fixed at 100. Fig. 4.10 shows the visual coverage probability for different K values corresponding to different numbers of deployed targets, N_t. We observe that visual coverage probability decreases as either N_t or K increases because of presence of more visual occlusions among more dense targets and more demanding coverage requirements, respectively.

In the second simulation, fixed number of sensor nodes and fixed number of targets are deployed where $N_s = 100$ and $N_t = 20$. However, in each deployment

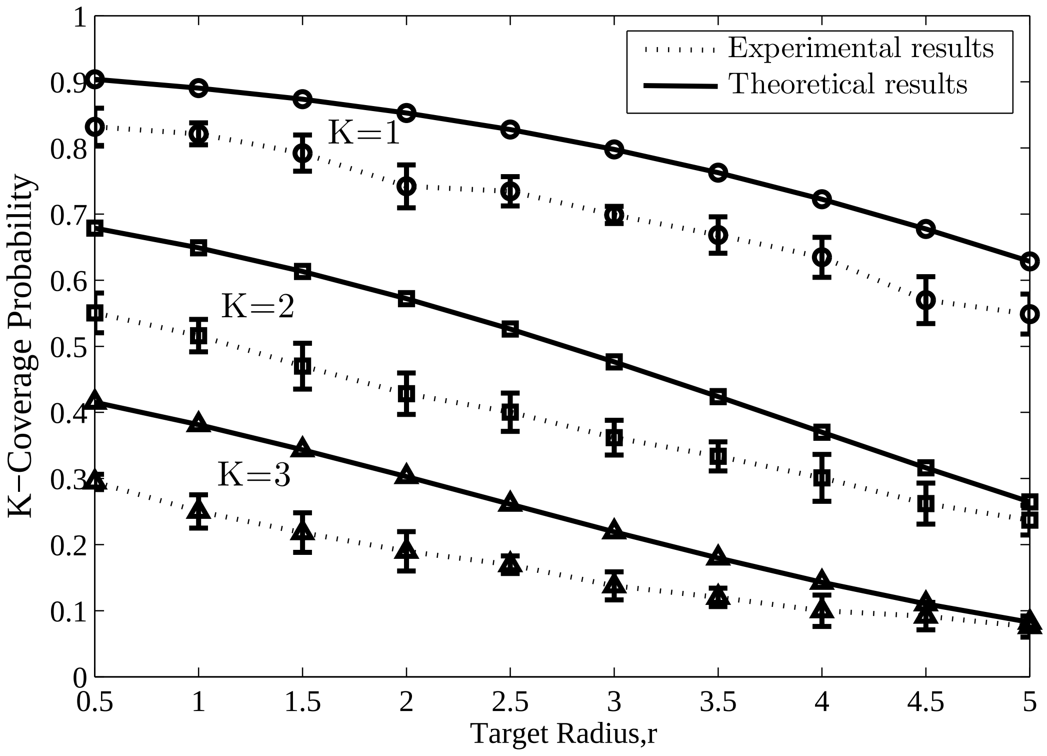


Figure 4.11: Comparison of theoretical values and simulation results corresponding to different target radius r.

uniform radius of each target is chosen with different values to show the effect of the target radius r. Fig. 4.11 shows the visual coverage probability for different K values corresponding to different target radius, r. We observe that visual coverage probability decreases as either r or K increases because of the presence of more visual occlusions among bigger targets and more demanding coverage requirements, respectively.

Results in Fig. 4.10 and 4.11 further validate the theoretical derivation of visual sensor coverage by showing the consistent theoretical values with simulated experimental results. However, again due to the boundary effect, the visual coverage probability from simulated experiments shows slight difference from theoretical values.

4.9.2 Boundary Effect on the Coverage Estimation Probability

In this experiment, we study the boundary effect on the visual coverage estimation probability corresponding to different number of sensor nodes N_s for different visual K-coverage requirements. Fig. 4.12a shows the visual K-coverage probability in 2D sensing region R corresponding to $N_t = 20$, $N_s = 100$, $\rho = 10$, $\theta = 45°$ and $K = 2$. We observe that the visual K-coverage probability decreases in the boundary region as close to the edge of the sensing region because of the boundary effect.

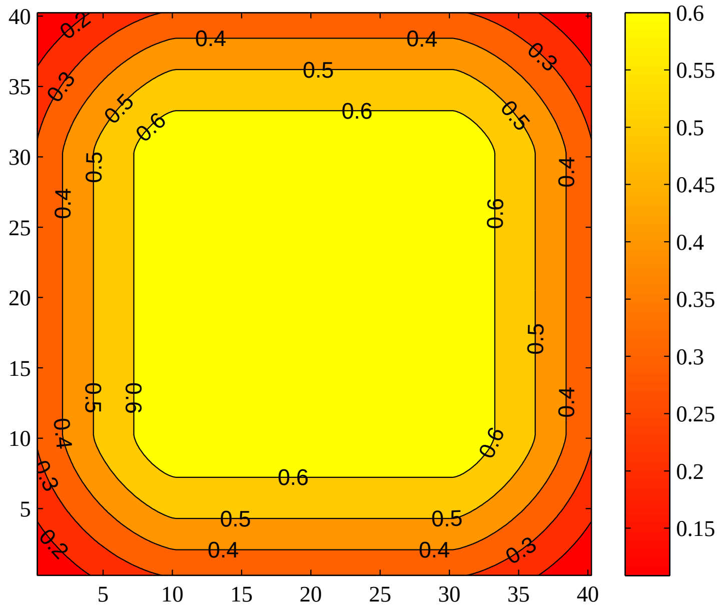
We randomly deploy 20 targets and different numbers of sensor nodes into the sensing field. Fig. 4.12b shows the visual coverage probability of simulated experiment, theoretical results with and without boundary effect for different K values. We observe that visual coverage probability decreases as K increases because of the more demanding coverage requirement and visual coverage probability increases as N_s increases due to more dense sensor nodes.

Results in Fig. 4.12b validate the proposed theoretical derivation of visual sensor coverage by showing exactly the same theoretical values when boundary effect is taken into account with simulated experimental results.

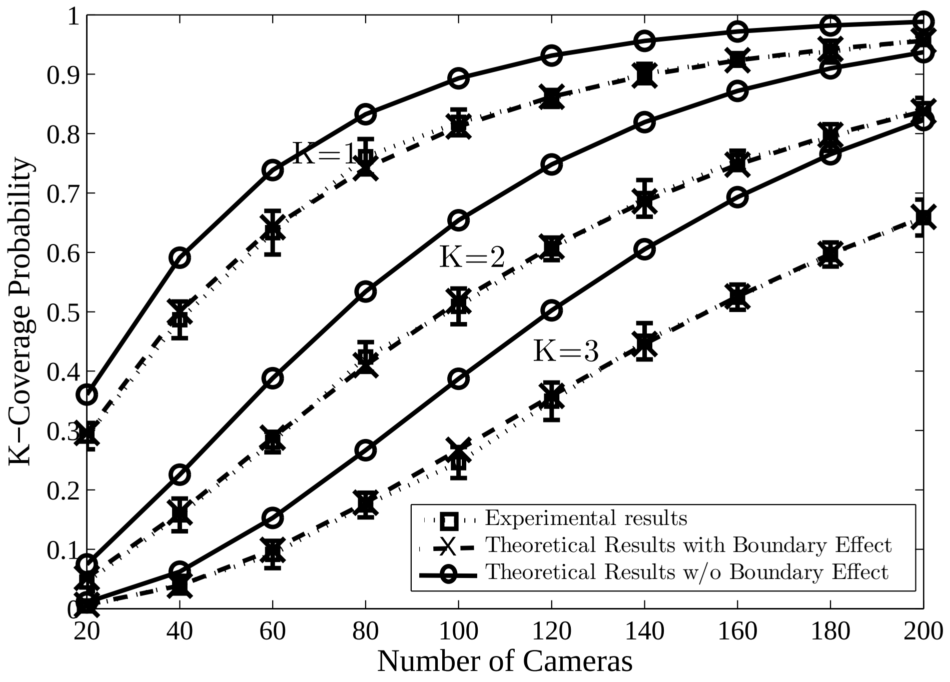
4.9.3 Minimum Sensor Density

In this set of simulation results, we compute the minimum sensor density, $\hat{\lambda}_s$, that ensures visual K-coverage. We study the effect of different parameters, i.e. target density λ_t, target radius r, sensing range ρ and angle of view θ. In each experiment, we change the value of one of these parameters and fix other parameters by setting $\lambda_t = 0.1$, $r = 0.5m$, $\rho = 10m$, $\theta = 45°$ and tolerance value $\epsilon = 0.05$.

First of all, we change the number of deployed target, N_t from 10 to 300. Fig. 4.13 shows the minimum sensor density $\hat{\lambda}_s$ corresponding to different number of deployed target, N_t under different K-coverage requirements. We observe that $\hat{\lambda}_s$ increases as N_t increases because of the presence of more visual occlusions among



(a)



(b)

Figure 4.12: (a) Visual coverage probability for sensing field R corresponding to $N_t = 20$, $N_s = 100$, $\rho = 10$, $\theta = 45°$, $K = 2$ (b) Comparison of theoretical values with and without boundary effect with simulation results corresponding to different number of sensor nodes N_s.

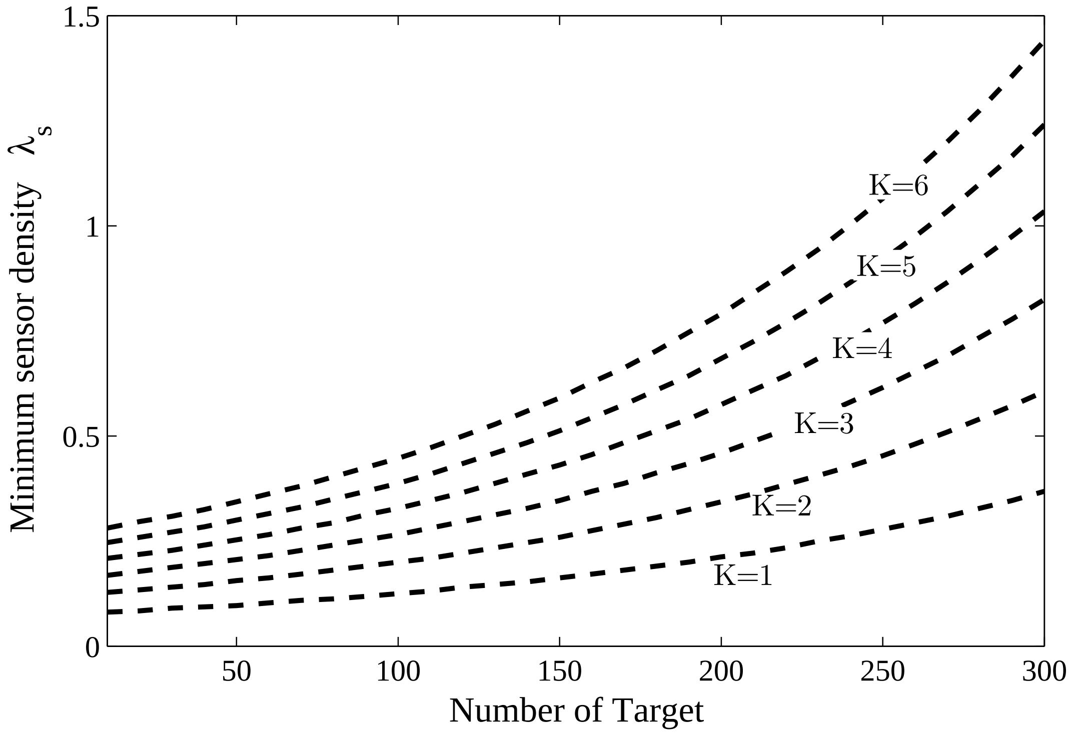


Figure 4.13: Minimum node density $\hat{\lambda}_s$ vs target density, λ_t.

more dense targets; $\hat{\lambda}_s$ also increases as K increases due to the more demanding coverage requirements. Moreover, we observe that to get double the K-coverage requires less than double increment in the sensor density because of the overlapping FOV of sensor nodes. However, to get more K-coverage in a crowded environment requires more proportional increment in the minimum sensor density $\hat{\lambda}_s$ than in a sparse target environment because of the more occlusion among crowded targets.

Secondly, to show the effect of the target radius r on the minimum sensor density, we change its value from 0.1m to 5m. Fig. 4.14 shows the minimum sensor density $\hat{\lambda}_s$ corresponding to different target radius r. We observe that $\hat{\lambda}_s$ increases as either λ_t or K increases because of the presence of more visual occlusions among targets of larger size and more demanding coverage requirements, respectively. Moreover, we observe that to update K-coverage requires less than K times increment in the sensor density. However, in an environment with large size targets, it requires more

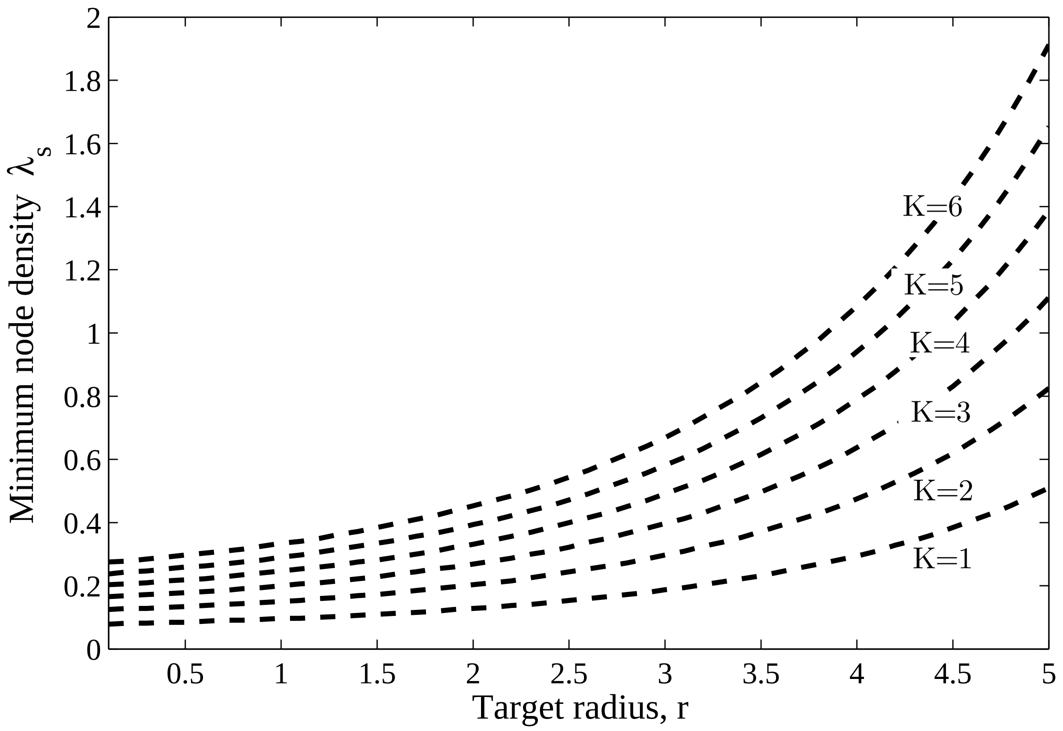
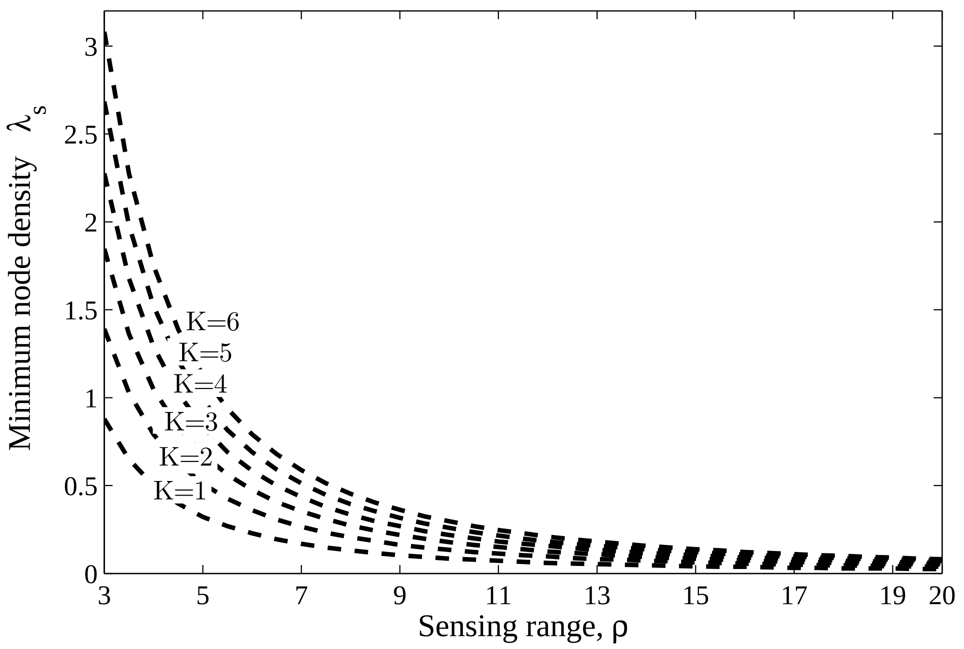


Figure 4.14: Minimum node density $\hat{\lambda}_s$ vs target radius, r.

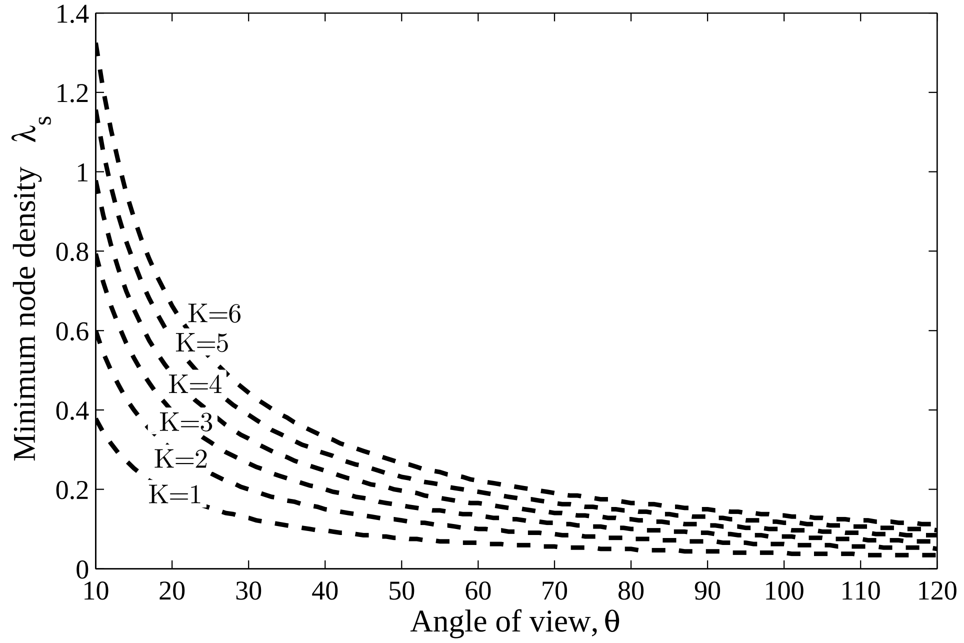
proportional increment in the minimum sensor density $\hat{\lambda}_s$ than in an environment with small-size targets because of the more visual occlusion among large targets.

Third, we change the sensing range ρ from 3m to 20m. Fig. 4.15a shows the minimum sensor density $\hat{\lambda}_s$ corresponding to different sensing range ρ under different K-coverage requirements. We observe that $\hat{\lambda}_s$ decreases as ρ increases because of the larger FOV of each sensor node; $\hat{\lambda}_s$ also increases as K increases due to the more demanding coverage requirements.

In the fourth simulation, to show the effect of angle of view θ on the minimum sensor density, we change its value from 10° to 120°). Fig. 4.15b shows the minimum sensor density $\hat{\lambda}_s$ corresponding to different angle of view θ. We observe that $\hat{\lambda}_s$ decreases as θ increases due to the larger FOV of each sensor node and $\hat{\lambda}_s$ increases as K increases because of more demanding coverage requirements.



(a)



(b)

Figure 4.15: Minimum node density $\hat{\lambda}_s$ vs (a) sensing range, ρ, (b) Angle of view, θ.

4.10 Discussion on Visual Coverage Estimation in Heterogeneous VSNs

In this chapter, it is assumed that homogeneous visual sensor nodes with the same sensing radius, ρ, and angle of view, θ, are deployed into the sensing field to detect the homogeneous targets with uniform target radius, r. In order to make the scenario more realistic where heterogeneous visual sensors and targets are likely to be deployed, we can relax these assumptions by considering the heterogeneous visual sensor deployment and heterogeneous target existence in the sensing field. In this section, we discuss the effect of heterogeneous sensor deployment and heterogeneous existence on the derivation of the closed-form solution for visual coverage estimation.

4.10.1 Effect of Heterogeneous Sensor Deployment on the Visual Coverage Estimation

In the heterogeneous visual sensor deployment, we deploy different types of visual sensor nodes into the sensing field with different sensor density, λ_s, sensing radius, ρ, and angle of view, θ. If n types of sensor nodes are deployed into the sensing field, a target can be covered by k many sensor nodes with any combinations of these n types of sensor nodes. Therefore, in the derivation of the closed-form solution for visual coverage estimation, we have to consider the different detection probability of each type of sensor node in their different size of detectability area, A and derive the closed-form solution based on their sensor related parameters (i.e., λ_s, ρ, and θ).

For simplicity, we consider that two types of sensor nodes are deployed in a heterogeneous VSN: Type I and Type II with sensor density, λ_{s_1} and λ_{s_2}, sensing radius, ρ_1 and ρ_2, and angle of view, θ_1 and θ_2, respectively. The probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence is

$$P(k) = \sum_{j=k}^{\infty} \sum_{i=0}^{j} \sum_{m=0}^{k} P_1(i, m; \lambda_{s_1}, \theta_1, \rho_1, \lambda_t, r) P_2(j-i, k-m; \lambda_{s_2}, \theta_2, \rho_2, \lambda_t, r) \quad (4.18)$$

where $P_1(i, m; \lambda_{s_1}, \theta_1, \rho_1)$ denotes the probability that a detectability area contains exactly i many Type I sensor nodes and m of them can cover the corresponding grid point based on its sensor related parameters (i.e., λ_{s_1}, ρ_1, and θ_1) and target related parameters (i.e., λ_t and r). And, similarly, $P_2(j-i, k-m; \lambda_{s_2}, \theta_2, \rho_2)$ denotes the probability that a detectability area contains exactly $j-i$ many Type II sensor nodes and $k-m$ of them can cover the corresponding grid point based on its sensor related parameters (i.e., λ_{s_2}, ρ_2, and θ_2) target related parameters (i.e., λ_t and r).

In the following subsections, we present the complete analysis of the heterogeneous visual sensor deployment in different cases of sensor related parameters (i.e., λ_s, ρ, and θ) of Type I and Type II sensor nodes. For simplicity, we first ignore the visual occlusion in VSN to derive the closed-form solution for visual coverage estimation. Then, we relax our assumption by considering visual occlusion and discuss the visual coverage estimation with visual occlusion in a heterogeneous VSN.

Heterogeneous Sensor Deployment without Visual Occlusions

As discussed in Section 4.4, if the radius of targets is infinitely small, i.e., $r \to 0$, we can ignore the visual occlusion. In this case, a sensor node covers a specific grid point $(x, y) \in R$, of the sensing field and determines target non-existence, if the node is located in a circular area A with radius ρ centered at the corresponding grid point and is oriented towards the center of the circle. The probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence is

$$P(k) = \sum_{j=k}^{\infty}\sum_{i=0}^{j}\sum_{m=0}^{k}\Big(\mathcal{P}(i;\lambda_{s_1}\times A_1)\mathcal{C}_m^i(p_1)^m(1-p_1)^{i-m}\times \\ \mathcal{P}(j-i;\lambda_{s_2}\times A_2)\mathcal{C}_{k-m}^{j-i}(p_2)^{k-m}(1-p_2)^{j-i-(k-m)}\Big) \tag{4.19}$$

where $\mathcal{P}(i;\lambda_s\times A)$ denotes the probability that a detectability area A contains exactly i sensor nodes from a Poisson point process with sensor density λ_s where $A=\pi\rho^2$. And, p denotes the probability of the sensor node facing towards the center of detectability area, A, and $\mathcal{C}_m^i$ denotes the number of combinations of m-node subset from a i-node set. The probability of Type I sensor nodes facing towards the center of their detectability area, A_1 is $p_1=\theta_1/(2\pi)$ where $A_1=\pi\rho_1^2$ and probability of Type II sensor nodes facing towards the center of their detectability area, A_2 is $p_2=\theta_2/(2\pi)$ where $A_2=\pi\rho_2^2$.

Case 1.1: $\theta_1\neq\theta_2$ where $\rho_1=\rho_2=\rho$ and $\lambda_{s_1}\neq\lambda_{s_2}$

In Case 1.1, it is assumed that Type I and Type II sensor nodes have different angle of view (i.e., $\theta_1\neq\theta_2$), different sensor density (i.e., $\lambda_{s_1}\neq\lambda_{s_2}$), and same sensing range (i.e., $\rho_1=\rho_2=\rho$) where their detectability area, A, is the same (i.e., $A_1=A_2=A=\pi\rho^2$), as shown in Fig. 4.16a. Therefore, Eq. 4.19 can be further derived as,

$$\begin{aligned}P(k) =&\sum_{j=k}^{\infty}\sum_{i=0}^{j}\sum_{m=0}^{k}\left(\frac{e^{-\lambda_{s_1}A}(\lambda_{s_1}A)^i}{i!}\mathcal{C}_m^i(p_1)^m(1-p_1)^{i-m}\times\right.\\ &\left.\frac{e^{-\lambda_{s_2}A}(\lambda_{s_2}A)^{j-i}}{(j-i)!}\mathcal{C}_{k-m}^{j-i}(p_2)^{k-m}(1-p_2)^{j-i-(k-m)}\right)\\ =&\sum_{j=k}^{\infty}e^{-A(\lambda_{s_1}+\lambda_{s_2})}A^j\lambda_{s_2}^j p_2^k(1-p_2)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}_m^i\mathcal{C}_{k-m}^{j-i}}{i!(j-i)!}\left(\frac{p_1}{p_2}\right)^m\left(\frac{\lambda_{s_1}}{\lambda_{s_2}}\right)^i\left(\frac{1-p_1}{1-p_2}\right)^{i-m}\\ \overset{(a)}{=}&\sum_{j=k}^{\infty}e^{-A(\lambda_{s_1}+\lambda_{s_2})}A^j\lambda_{s_2}^j p_2^k(1-p_2)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}\left(\frac{p_1}{p_2}\right)^m\left(\frac{\lambda_{s_1}}{\lambda_{s_2}}\right)^i\left(\frac{1-p_1}{1-p_2}\right)^{i-m}\end{aligned}$$

$$
\begin{aligned}
&=\sum_{j=k}^{\infty}\frac{e^{-A(\lambda_{s_1}+\lambda_{s_2})}}{k!(j-k)!}A^j\lambda_{s_2}^j p_2^k(1-p_2)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}\Big(\frac{\lambda_{s_1}p_1}{\lambda_{s_2}p_2}\Big)^m\Big(\frac{\lambda_{s_1}(1-p_1)}{\lambda_{s_2}(1-p_2)}\Big)^{i-m}\\
&\overset{(b)}{=}\sum_{j=k}^{\infty}\frac{e^{-A(\lambda_{s_1}+\lambda_{s_2})}}{k!(j-k)!}A^j\lambda_{s_2}^j p_2^k(1-p_2)^{j-k}\Big(1+\frac{\lambda_{s_1}p_1}{\lambda_{s_2}p_2}\Big)^k\Big(1+\frac{\lambda_{s_1}(1-p_1)}{\lambda_{s_2}(1-p_2)}\Big)^{j-k}\\
&=\frac{e^{-A(\lambda_{s_1}+\lambda_{s_2})}}{k!}\Big(A(\lambda_{s_1}p_1+\lambda_{s_2}p_2)\Big)^k\sum_{j=k}^{\infty}\frac{\Big(A\Big(\lambda_{s_1}(1-p_1)+\lambda_{s_2}(1-p_2)\Big)\Big)^{j-k}}{(j-k)!}\\
&\overset{(c)}{=}\frac{1}{k!}e^{-A(\lambda_{s_1}+\lambda_{s_2})}\Big(A(\lambda_{s_1}p_1+\lambda_{s_2}p_2)\Big)^k e^{A\lambda_{s_1}(1-p_1)+A\lambda_{s_2}(1-p_2)}\\
&=\frac{1}{k!}e^{-A(\lambda_{s_1}p_1+\lambda_{s_2}p_2)}\Big(A(\lambda_{s_1}p_1+\lambda_{s_2}p_2)\Big)^k\\
&=\mathcal{P}\Big(k;(\lambda_{s_1}p_1+\lambda_{s_2}p_2)\times A\Big)
\end{aligned}
\tag{4.20}
$$

where (a) follows the combination properties, $\frac{\mathcal{C}_m^i\mathcal{C}_{k-m}^{j-i}}{i!(j-i)!}=\frac{\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}$ (see the proof in Appendix B.5), and (b) follows the binomial coefficient property of statistical population where $(1+x)^k(1+y)^{j-k}=\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}(x)^m(y)^{i-m}$.

From the derivation result in Eq. 4.20, we observe that the visual coverage probability with visual occlusions also follows the Poisson point process with the sensor density $(\lambda_{s_1}p_1+\lambda_{s_2}p_2)$ in area A. If $p_1=p_2=p$ and $\lambda_{s_1}+\lambda_{s_1}=\lambda_s$, then Eq. 4.20 converges to Eq. 4.2 which means heterogeneous visual sensor deployment becomes homogeneous.

Case 1.2: $\rho_1\neq\rho_2$ where $\theta_1=\theta_2=\theta$ and $\lambda_{s_1}\neq\lambda_{s_2}$

In Case 1.2, it is assumed that Type I and Type II sensor nodes have different sensing ranges (i.e., $\rho_1\neq\rho_2$), different sensor densities (i.e., $\lambda_{s_1}\neq\lambda_{s_2}$), but same angle of view (i.e., $\theta_1=\theta_2=\theta$) where their detectability area, A_1 and A_2 equal to $A_1=\pi\rho_1^2$ and $A_2=\pi\rho_2^2$, respectively as shown in Fig. 4.16b. Therefore, Eq. 4.19 can be further

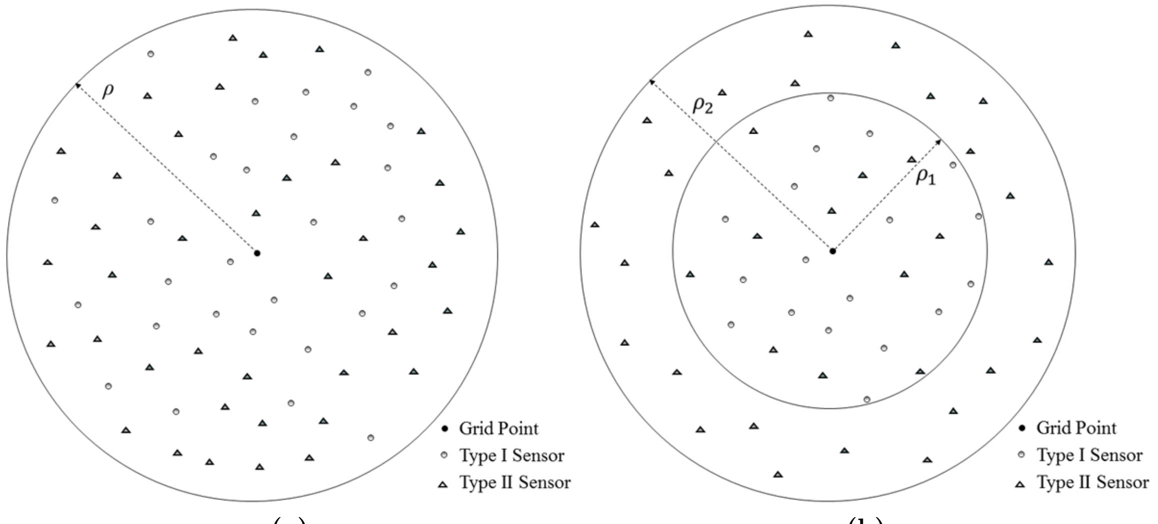


Figure 4.16: Heterogenous visual sensor deployment with different angle of view ($\theta_1 \neq \theta_2$) and (a) same sensing range ($\rho_1 = \rho_2 = \rho$) or (b) different sensing range ($\rho_1 \neq \rho_2$)

derived as,

$$
\begin{aligned}
P(k) =& \sum_{j=k}^{\infty}\sum_{i=0}^{j}\sum_{m=0}^{k}\Bigg(\frac{e^{-\lambda_{s_1}A_1}(\lambda_{s_1}A_1)^i}{i!}\mathcal{C}^i_m(p)^m(1-p)^{i-m}\times \\
&\frac{e^{-\lambda_{s_2}A_2}(\lambda_{s_2}A_2)^{j-i}}{(j-i)!}\mathcal{C}^{j-i}_{k-m}(p)^{k-m}(1-p)^{j-i-(k-m)}\Bigg) \\
=& \sum_{j=k}^{\infty} e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}(A_2\lambda_{s_2})^j p^k(1-p)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}^i_m\mathcal{C}^{j-i}_{k-m}}{i!(j-i)!}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^i \\
\overset{(a)}{=}& \sum_{j=k}^{\infty} e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}(A_2\lambda_{s_2})^j p^k(1-p)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}^k_m\mathcal{C}^{j-k}_{i-m}}{k!(j-k)!}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^i \\
=& \sum_{j=k}^{\infty}\frac{e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}}{k!(j-k)!}(A_2\lambda_{s_2})^j p^k(1-p)^{j-k}\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}^k_m\mathcal{C}^{j-k}_{i-m}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^{i-m}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^m \\
\overset{(b)}{=}& \sum_{j=k}^{\infty}\frac{e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}}{k!(j-k)!}(A_2\lambda_{s_2})^j p^k(1-p)^{j-k}\Big(1+\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^k\Big(1+\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^{j-k} \\
=& \frac{e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}}{k!}\Big(p(A_1\lambda_{s_1}+A_2\lambda_{s_2})\Big)^k\sum_{j=k}^{\infty}\frac{\Big((1-p)(A_1\lambda_{s_1}+A_2\lambda_{s_2})\Big)^{j-k}}{(j-k)!}
\end{aligned}
$$

$$
\begin{aligned}
&\overset{(c)}{=}\frac{1}{k!}e^{-(\lambda_{s_1}A_1+\lambda_{s_2}A_2)}\Big(p(A_1\lambda_{s_1}+A_2\lambda_{s_2})\Big)^k e^{(1-p)(A_1\lambda_{s_1}+A_2\lambda_{s_2})}\\
&=\frac{1}{k!}e^{-p(A_1\lambda_{s_1}+A_2\lambda_{s_2})}\Big(p(A_1\lambda_{s_1}+A_2\lambda_{s_2})\Big)^k\\
&=\mathcal{P}\Big(k;p\times(A_1\lambda_{s_1}+A_2\lambda_{s_2})\Big) \qquad (4.21)
\end{aligned}
$$

where (a) follows the combination properties, $\frac{\mathcal{C}_m^i\mathcal{C}_{k-m}^{j-i}}{i!(j-i)!}=\frac{\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}$ (see the proof in Appendix B.5), and (b) follows the binomial coefficient property of statistical population where $(1+x)^k(1+y)^{j-k}=\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}(x)^m(y)^{i-m}$.

From the derivation result in Eq. 4.21, we observe that the visual coverage probability with visual occlusions also follows the Poisson point process with the parameter $p\times(A_1\lambda_{s_1}+A_2\lambda_{s_2})$ where $A_1=\pi\rho_1^2$ and $A_2=\pi\rho_2^2$. If $\rho_1=\rho_2=\rho$ and $\lambda_{s_1}+\lambda_{s_1}=\lambda_s$, then Eq. 4.21 converges to Eq. 4.2 which means heterogeneous visual sensor deployment become homogeneous.

Case 1.3: $\theta_1\neq\theta_2$ and $\rho_1\neq\rho_2$ where $\lambda_{s_1}\neq\lambda_{s_2}$

In Case 1.3, it is assumed that Type I and Type II sensor nodes have different angle of view (i.e., $\theta_1\neq\theta_2$), different sensing range (i.e., $\rho_1\neq\rho_2$), and different sensor density (i.e., $\lambda_{s_1}\neq\lambda_{s_2}$) where their detectability area, A_1 and A_2 equal to $A_1=\pi\rho_1^2$ and $A_2=\pi\rho_2^2$, respectively as shown in Fig. 4.16b. Therefore, Eq. 4.19 can be further derived as,

$$
\begin{aligned}
P(k)=&\sum_{j=k}^{\infty}\sum_{i=0}^{j}\sum_{m=0}^{k}\Bigg(\frac{e^{-\lambda_{s_1}A_1}(\lambda_{s_1}A_1)^i}{i!}\mathcal{C}_m^i(p_1)^m(1-p_1)^{i-m}\times\\
&\frac{e^{-\lambda_{s_2}A_2}(\lambda_{s_2}A_2)^{j-i}}{(j-i)!}\mathcal{C}_{k-m}^{j-i}(p_2)^{k-m}(1-p_2)^{j-i-(k-m)}\Bigg)\\
=&\sum_{j=k}^{\infty}e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}(\lambda_{s_2}A_2)^j p_2^k(1-p_2)^{j-k}\times\\
&\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}_m^i\mathcal{C}_{k-m}^{j-i}}{i!(j-i)!}\Big(\frac{p_1}{p_2}\Big)^m\Big(\frac{1-p_1}{1-p_2}\Big)^{i-m}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^i
\end{aligned}
$$

$$
\begin{aligned}
&\overset{(a)}{=}\sum_{j=k}^{\infty} e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}(\lambda_{s_2}A_2)^j p_2^k(1-p_2)^{j-k}\times\\
&\quad\sum_{i=0}^{j}\sum_{m=0}^{k}\frac{\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}\Big(\frac{p_1}{p_2}\Big)^m\Big(\frac{1-p_1}{1-p_2}\Big)^{i-m}\Big(\frac{\lambda_{s_1}A_1}{\lambda_{s_2}A_2}\Big)^i\\
&=\sum_{j=k}^{\infty}\frac{e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}}{k!(j-k)!}(\lambda_{s_2}A_2)^j p_2^k(1-p_2)^{j-k}\times\\
&\quad\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}\Big(\frac{\lambda_{s_1}A_1p_1}{\lambda_{s_2}A_2p_2}\Big)^m\Big(\frac{\lambda_{s_1}A_1(1-p_1)}{\lambda_{s_2}A_2(1-p_2)}\Big)^{i-m}\\
&\overset{(b)}{=}\sum_{j=k}^{\infty}\frac{e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}}{k!(j-k)!}(\lambda_{s_2}A_2)^j p_2^k(1-p_2)^{j-k}\Big(1+\frac{\lambda_{s_1}A_1p_1}{\lambda_{s_2}A_2p_2}\Big)^k\Big(1+\frac{\lambda_{s_1}A_1(1-p_1)}{\lambda_{s_2}A_2(1-p_2)}\Big)^{j-k}\\
&=\frac{e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}}{k!}\Big(\lambda_{s_1}A_1p_1+\lambda_{s_2}A_2p_2\Big)^k\sum_{j=k}^{\infty}\frac{\Big(\lambda_{s_1}A_1(1-p_1)+\lambda_{s_2}A_2(1-p_2)\Big)^{j-k}}{(j-k)!}\\
&=\frac{1}{k!}e^{-\lambda_{s_1}A_1+\lambda_{s_2}A_2}\Big(\lambda_{s_1}A_1p_1+\lambda_{s_2}A_2p_2\Big)^k e^{\lambda_{s_1}A_1(1-p_1)+\lambda_{s_2}A_2(1-p_2)}\\
&=\frac{1}{k!}e^{-\lambda_{s_1}A_1p_1+\lambda_{s_2}A_2p_2}\Big(\lambda_{s_1}A_1p_1+\lambda_{s_2}A_1p_2\Big)^k\\
&=\mathcal{P}\Big(k;\lambda_{s_1}A_1p_1+\lambda_{s_2}A_2p_2\Big) \qquad (4.22)
\end{aligned}
$$

where (a) follows the combination properties, $\frac{\mathcal{C}_m^i\mathcal{C}_{k-m}^{j-i}}{i!(j-i)!}=\frac{\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}$ (see the proof in Appendix B.5), and (b) follows the binomial coefficient property of statistical population where $(1+x)^k(1+y)^{j-k}=\sum_{i=0}^{j}\sum_{m=0}^{k}\mathcal{C}_m^k\mathcal{C}_{i-m}^{j-k}(x)^m(y)^{i-m}$.

From the derivation result in Eq. 4.22, we observe that the visual coverage probability with visual occlusions also follows the Poisson point process with the parameter $\lambda_{s_1}A_1p_1+\lambda_{s_2}A_2p_2$ where $A_1=\pi\rho_1^2$ and $A_2=\pi\rho_2^2$. If $p_1=p_2=p$, $\rho_1=\rho_2=\rho$, and $\lambda_{s_1}+\lambda_{s_1}=\lambda_s$, then Eq. 4.22 converges to Eq. 4.2 which means heterogeneous visual sensor deployment become homogeneous.

Heterogeneous Sensor Deployment with Visual Occlusions

In previous three cases for heterogeneous sensor deployment with different combinations of the sensor related parameters (i.e., λ_s, ρ, and θ), we ignored the visual

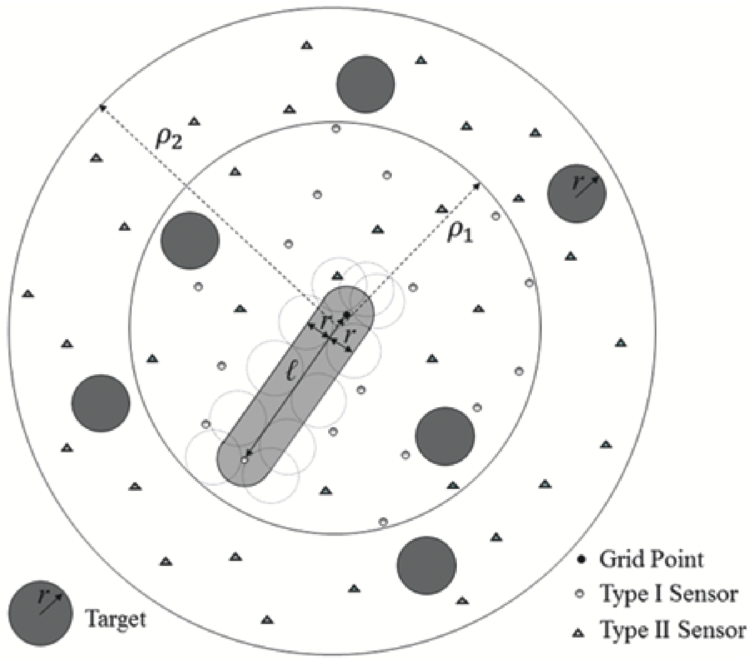


Figure 4.17: Occlusion zone in heterogenous visual sensor deployment

occlusion for simplicity in derivation of closed-form solution for visual coverage estimation. To make the scenario more realistic, we relax our assumption on target radius, r that becomes a finite value, i.e., $r > 0$. Therefore, it is not appropriate to ignore the visual occlusions. To cover a specific grid point of the sensing field and determine the target non-existence at that point, not only the corresponding grid point must be inside the FOV of the sensor node, the centers of all targets should also be outside of the occlusion zone between the corresponding grid point and the node which is illustrated as the bold-boundary region in Fig. 4.17.

As describe in Section 4.5, Q denote the probability of covering a specific grid point of the sensing field by heterogeneous visual sensor nodes and determining the target non-existence that depends on two independent factors, i.e., the grid point is within the FOV of the sensor, p, and that there is no occlusion between the sensor and the grid point, q. Q can be expressed as

$$Q(l) = p \times q = \frac{\theta}{2\pi} \times e^{-\lambda_t(\pi r^2 + 2rl)} \tag{4.23}$$

Thus, the visual coverage probability that exactly k nodes cover a specific grid point of the sensing field by heterogeneous visual sensor nodes and determine the target non-existence, $P(k)$, is

$$P(k) = \int P(k, Q_1(l), Q_2(l)) f(l) dl \tag{4.24}$$

where $P(k, Q_1(l), Q_2(l))$ is the probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence at that point with respect to $Q_1(l)$ and $Q_2(l)$ values of Type I and Type II sensor nodes. And $f(l)$ is the probability density function (pdf) of distance l between the corresponding grid point and each sensor node in the circular detectability areas A_1 and A_2 with sensing radius ρ_1 and ρ_2 centered at the grid point.

In order to derive the $P(k, Q_1, Q_2)$, Eq. 4.18 can be further derived as,

$$\begin{aligned} P(k, Q_1(l), Q_2(l)) = \sum_{j=k}^{\infty} \sum_{i=0}^{j} \sum_{m=0}^{k} \Big(& \mathcal{P}(i; \lambda_{s_1} \times A_1) \mathcal{C}_m^i (Q_1)^m (1 - Q_1)^{i-m} \times \\ & \mathcal{P}(j - i; \lambda_{s_2} \times A_2) \mathcal{C}_{k-m}^{j-i} (Q_2)^{k-m} (1 - Q_2)^{j-i-(k-m)} \Big) \end{aligned} \tag{4.25}$$

where $Q_1(l) = p_1 \times q = \frac{\theta_1}{2\pi} \times e^{-\lambda_t(\pi r^2 + 2rl)}$ and $Q_2(l) = p_2 \times q = \frac{\theta_1}{2\pi} \times e^{-\lambda_t(\pi r^2 + 2rl)}$.

The derivation of $P(k, Q_1, Q_2)$ in Eq. 4.25 is similar with Eq. 4.19. Only difference is replacement of p_1 and p_2 with Q_1 and Q_2, respectively. Therefore, derivation results for three cases will be similar with parameter replacement.

Case 2.1: $\theta_1 \neq \theta_2$ where $\rho_1 = \rho_2 = \rho$ and $\lambda_{s_1} \neq \lambda_{s_2}$

In Case 2.1, it is assumed that Type I and Type II sensor nodes have different angle of view (i.e., $\theta_1 \neq \theta_2$), different sensor density (i.e., $\lambda_{s_1} \neq \lambda_{s_2}$), and same sensing range (i.e., $\rho_1 = \rho_2 = \rho$) where their detectability areas, A, is the same (i.e., $A_1 =$

$A_2 = A = \pi\rho^2$) and $Q_1 \neq Q_2$. Therefore, derivation result of Eq. 4.25 is

$$P(k, Q_1, Q_2) = \mathcal{P}\Big(k; (\lambda_{s_1} Q_1 + \lambda_{s_2} Q_2) \times A\Big) \tag{4.26}$$

Case 2.2: $\rho_1 \neq \rho_2$ where $\theta_1 = \theta_2 = \theta$ and $\lambda_{s_1} \neq \lambda_{s_2}$

In Case 2.2, it is assumed that Type I and Type II sensor nodes have different sensing range (i.e.,$\rho_1 \neq \rho_2$), different sensor density (i.e., $\lambda_{s_1} \neq \lambda_{s_2}$), and same angle of view (i.e., $\theta_1 = \theta_2 = \theta$) where their detectability areas, A_1 and A_2 equal to $A_1 = \pi\rho_1^2$ and $A_2 = \pi\rho_2^2$ and $Q_1 = Q_2 = Q = \frac{\theta}{2\pi} \times e^{-\lambda_t(\pi r^2 + 2rl)}$. Therefore, derivation result of Eq. 4.25 is

$$P(k, Q_1, Q_2) = \mathcal{P}\Big(k; Q \times (A_1\lambda_{s_1} + A_2\lambda_{s_2})\Big) \tag{4.27}$$

Case 2.3: $\theta_1 \neq \theta_2$ and $\rho_1 \neq \rho_2$ where $\lambda_{s_1} \neq \lambda_{s_2}$

In Case 2.3, it is assumed that Type I and Type II sensor nodes have different angle of view (i.e., $\theta_1 \neq \theta_2$), different sensor density (i.e., $\lambda_{s_1} \neq \lambda_{s_2}$), and different sensing range (i.e., $\rho_1 \neq \rho_2$) where their detectability areas, A_1 and A_2 equal to $A_1 = \pi\rho_1^2$ and $A_2 = \pi\rho_2^2$ and $Q_1 \neq Q_2$. Therefore, derivation result of Eq. 4.25 is

$$P(k, Q_1, Q_2) = \mathcal{P}\Big(k; \lambda_{s_1} A_1 Q_1 + \lambda_{s_2} A_2 Q_2\Big) \tag{4.28}$$

4.10.2 Effect of Heterogeneous Target Existence on the Visual Coverage Estimation

In addition, when heterogeneous targets exist in the sensing field with different target radius, r, we have to consider all types of targets in order to compute the probability that there is no occluding target between a grid point and a sensor node, q. For example, if two types of targets exist in the sensing field with target density, λ_{t_1} and λ_{t_2}, and target radius, r_1 and r_2, to cover a grid point in the sensing field, and determine the target non-existence at that point, not only the corresponding grid

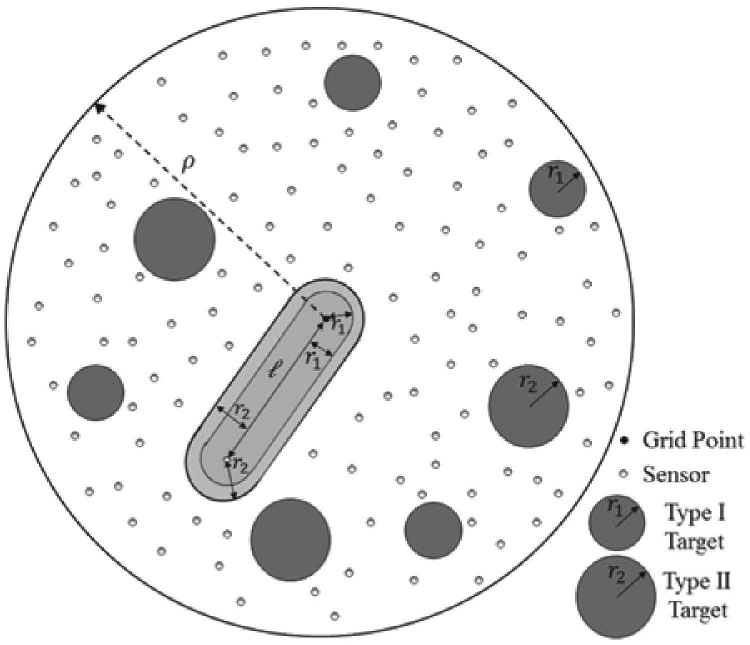


Figure 4.18: Occlusion zones, A_{o_1} and A_{o_2}, of Type I and Type II targets in case of heterogenous target existence.

point must be inside the FOV of the sensor node, the occlusion zones between the grid point and the sensor node, A_{o_1} and A_{o_2} should also be free of Type I and Type II targets, respectively as shown in Fig. 4.18. Therefore, the probability of having no Type I target in the occlusion zone, A_{o_1} equals to $q_1 = e^{-\lambda_{t_1} A_{o_1}}$ where $A_{o_1} = \pi r_1^2 + 2r_1 l$ and the probability of having no Type II target in the occlusion zone, A_{o_2} equals to $q_2 = e^{-\lambda_{t_2} A_{o_2}}$ where $A_{o_2} = \pi r_2^2 + 2r_2 l$.

The visual coverage probability that exactly k nodes cover a specific grid point of the sensing field and determine the target non-existence when heterogeneous targets exist in the sensing field, $P(k)$, is

$$P(k) = \int P(k, q_1(l), q_2(l)) f(l) dl \tag{4.29}$$

where $P(k, q_1(l), q_2(l))$ is the probability that exactly k sensor nodes cover a specific grid point of the sensing field and determine the target non-existence at that point with respect to $q_1(l)$ and $q_2(l)$ values of Type I and Type II targets, respectively. The derivation of $P(k, q_1(l), q_2(l))$ is

$$
\begin{aligned}
P(k,q_1,q_2) =& \sum_{j=k}^{\infty}\sum_{i=k}^{j}\sum_{h=k}^{i}\mathcal{P}(j;\lambda_s A)\mathcal{C}_i^j p^i(1-p)^{j-i}\mathcal{C}_h^i q_1^h(1-q_1)^{i-h}\mathcal{C}_k^h q_2^k(1-q_2)^{h-k} \\
=& \sum_{j=k}^{\infty}\sum_{i=k}^{j}\mathcal{P}(j;\lambda_s A)\mathcal{C}_i^j p^i(1-p)^{j-i}q_1^k q_2^k(1-q_1)^{i-k}\sum_{h=k}^{i}\mathcal{C}_h^i\mathcal{C}_k^h\Big(\frac{q_1(1-q_2)}{1-q_1}\Big)^{h-k} \\
=& \sum_{j=k}^{\infty}\sum_{i=k}^{j}\mathcal{P}(j;\lambda_s A)\mathcal{C}_i^j p^i(1-p)^{j-i}q_1^k q_2^k(1-q_1)^{i-k}\sum_{s=0}^{i-k}\mathcal{C}_{k+s}^i\mathcal{C}_k^{k+s}\Big(\frac{q_1(1-q2)}{1-q_1}\Big)^{s} \\
\overset{(a)}{=}& \sum_{j=k}^{\infty}\sum_{i=k}^{j}\mathcal{P}(j;\lambda_s A)\mathcal{C}_i^j p^i(1-p)^{j-i}q_1^k q_2^k(1-q_1)^{i-k}\sum_{s=0}^{i-k}\mathcal{C}_k^i\mathcal{C}_s^{i-k}\Big(\frac{q_1(1-q_2)}{1-q_1}\Big)^{s} \\
\overset{(b)}{=}& \sum_{j=k}^{\infty}\sum_{i=k}^{j}\mathcal{P}(j;\lambda_s A)\mathcal{C}_i^j p^i(1-p)^{j-i}q_1^k q_2^k(1-q_1)^{i-k}\mathcal{C}_k^i\Big(\frac{q_1(1-q_2)}{1-q_1}+1\Big)^{i-k} \\
=& \sum_{j=k}^{\infty}\mathcal{P}(j;\lambda_s A)p^k q_1^k q_2^k(1-p)^{j-k}\sum_{i=k}^{j}\mathcal{C}_i^j\mathcal{C}_k^i\Big(\frac{p(1-q_1q_2)}{1-p}\Big)^{i-k} \\
=& \sum_{j=k}^{\infty}\mathcal{P}(j;\lambda_s A)p^k q_1^k q_2^k(1-p)^{j-k}\sum_{s=0}^{j-k}\mathcal{C}_{k+s}^j\mathcal{C}_k^{k+s}\Big(\frac{p(1-q_1q_2)}{1-p}\Big)^{s} \\
\overset{(c)}{=}& \sum_{j=k}^{\infty}\mathcal{P}(j;\lambda_s A)p^k q_1^k q_2^k(1-p)^{j-k}\sum_{s=0}^{j-k}\mathcal{C}_k^j\mathcal{C}_s^{j-k}\Big(\frac{p(1-q_1q_2)}{1-p}\Big)^{s} \\
\overset{(d)}{=}& \sum_{j=k}^{\infty}\frac{e^{-\lambda_s A}(\lambda_s A)^j}{j!}p^k q_1^k q_2^k(1-p)^{j-k}\mathcal{C}_k^j\Big(\frac{p(1-q_1q_2)}{1-p}+1\Big)^{j-k} \\
=& \sum_{j=k}^{\infty}\frac{e^{-\lambda_s A}(\lambda_s A)^j}{j!}p^k q_1^k q_2^k\frac{j!}{k!(j-k)!}(1-pq_1q_2)^{j-k} \\
=& \frac{1}{k!}e^{-\lambda_s A}(\lambda_s Apq_1q_2)^k\sum_{j=k}^{\infty}\frac{\big((\lambda_s A)(1-pq_1q_2)\big)^{j-k}}{(j-k)!} \\
=& \frac{1}{k!}e^{-\lambda_s A}(\lambda_s Apq_1q_2)^k e^{\lambda_s A(1-pq_1q_2)} \\
=& \frac{1}{k!}e^{-\lambda_s Apq_1q_2}(\lambda_s Apq_1q_2)^k \\
=& \mathcal{P}(k;\lambda_s pq_1(l)q_2(l)A)
\end{aligned}
\tag{4.30}
$$

where (a) and (c) follows the combination properties, $\mathcal{C}_{k+s}^{i}.\mathcal{C}_{k}^{k+s} = \mathcal{C}_{k}^{i}.\mathcal{C}_{s}^{i-k}$, and (b) and (d) follows the binomial coefficient property, $(x+y)^n = \sum_{s=0}^{n} \mathcal{C}_{s}^{n} x^{n-s} y^s$ where $s = i - k$. Also, $A = \pi\rho^2$, $p = \theta/(2\pi)$, $q_1 = e^{-\lambda_{t_1}(\pi r_1^2 + 2r_1 l)}$, and $q_2 = e^{-\lambda_{t_2}(\pi r_2^2 + 2r_2 l)}$.

From the derivation result in Eq. 4.30, we observe that the visual coverage probability with visual occlusions also follows the Poisson point process with the parameter $\lambda_s p q_1 q_2$ in area $A = \pi\rho^2$. If $r_1 = r_2 = r$, and $\lambda_{t_1} + \lambda_{t_2} = \lambda_t$, then Eq. 4.30 converges to Eq. 4.2 which means heterogeneous visual sensor deployment become homogeneous.

4.11 Summary

In this chapter, we presented a closed-form solution for the visual coverage estimation problem in the presence of visual occlusions among crowded targets in a VSN. By assuming the uniform random deployment of sensor nodes into a large-scale sensing field and taking the visual occlusions and boundary effects into account, we derived the visual coverage estimation from a different point of view by modeling the target detection algorithm based on the certainty map approach. Then, we further estimated the minimum sensor density that suffices to ensure a visual K-coverage in a crowded sensing field by using the visual coverage estimation model.

Our major contributions in this chapter were two-fold. First, we adopted the certainty-based target detection model in coverage estimation in a randomly deployed VSN and derived a closed-form solution for visual coverage estimation. Therefore, the sensor related parameters (e.g., sensor density, sensing range, etc.) can be decided before deployment in order to have proper visual coverage in the sensing field. Second, since the visual coverage probability in a crowded environment depends not only on the sensor density and deployment but also on the target density and distribution, our proposed closed-form solution considers both the directional sensing nature of cameras and the visual occlusions among targets and provides more accurate and more realistic coverage estimation in a crowded VSN.

By comparing the simulation results and the theoretical values, we validated the proposed closed-form solution of visual coverage estimation and showed the effectiveness of our model to be deployed in practical scenarios. In order to make the scenario more realistic, we relaxed the assumptions on homogeneous sensor deployment and homogeneous target existence and extended the proposed closed-form solution for more general scenarios where heterogeneous visual sensors and targets are likely to be deployed into the sensing field.

Chapter 5

Conclusions and Future Work

5.1 Summary

In this dissertation, we presented collaborative solutions to visual sensor networks (VSNs) that are formed by significantly small size and low cost visual sensor platforms with imaging, on-board processing and communication capabilities. A visual sensor network covers a large surveillance area and is capable of solving computer vision problems through distributed sensing and collaborative in-network processing. Although many potential applications have been possible using these powerful visual sensor platforms, VSNs also present unique challenges that could hinder their practical deployment compared to conventional 1-D scalar sensor networks because of unique features of cameras, including the extremely higher data rate and the directional sensing characteristics with limited field of view and visual occlusions.

In order to address these challenges in visual sensor networks, we first presented an *energy-efficient* and *light-weight* approach to localize targets in a crowded environment using a visual sensor network through distributed sensing and collaborative in-network processing by taking the directional sensing and visual occlusion issues in visual sensors into account. Traditionally, the problem is solved by localizing the targets at the intersections of the back-projected 2D cones of each target. However, the existence of visual occlusion among targets would generate many false alarms. Instead of resolving the uncertainty about target existence at the intersections, we identify and study the non-occupied areas in the cone and generate the so-called *certainty map* of non-existence of targets.

Secondly, we proposed a data fusion algorithm to integrate certainty maps where not only each camera node transmits a very limited amount of data but that a limited number of camera nodes is involved. We introduced a dynamic itinerary for certainty map integration where the entire map is progressively clarified from sensor to sensor starting the integration with the sensor that has the greatest contribution information to the current certainty map. When the confidence of the certainty map is satisfied, targets are localized at the remaining unresolved regions in the certainty map.

In addition to target localization, we also focused on the design of a fault-tolerant target localization algorithm in VSNs that would not only accurately localize targets but also detect the faults in camera orientation, tolerate these errors and further correct them before they cascade. Based on the locations of detected targets in the final certainty map, we then constructed a generative image model in each camera that estimates the camera orientation, detect inaccuracies in camera orientations and correct them before the fault in the system cascades and reaches a point where the performance of the algorithm dramatically drops or sometimes the algorithm fails. We also presented the distributed implementation of the fault-tolerant collaborative target localization by selecting the voting threshold automatically.

Finally, we derived a closed-form solution for the visual coverage estimation problem in the presence of occlusions to guarantee the required coverage in a sensing field. In order to have proper sensor coverage in the sensing field, some sensor related parameters, such as sensor density, sensing range, etc., should be decided based on the estimated sensor coverage probability *before* deployment. According to the coverage estimation model, we further proposed an estimate of the minimum sensor density that suffices to ensure a visual K-coverage in a crowded sensing field. Simulation was conducted which shows extreme consistency with results from theoretical formulation, especially when the boundary effect is considered. In order to make the scenario more realistic, we extended the proposed closed-form solution for more general scenarios where heterogeneous visual sensors and targets are likely to be deployed into the sensing field.

5.2 Directions for Future Research

In a visual sensor network, many sensor nodes are randomly deployed to cover a large sensing field as long as possible and collect data from its surroundings as much as possible. However, these two concept are inversely proportional because collecting data consumes energy and in many scenarios, sensor nodes are powered by limited power supplies (i.e., batteries) which is not possible to exchange or recharge the batteries because of the harsh or inaccessible environments. When the battery is run out, the visual sensor node dies. Therefore, energy-efficiency and power-conservation are still important issues in a visual sensor network to maximize the network lifetime where it is defined as time duration when every point in the sensing region is covered by at least one camera.

In order to conserve energy in a visual sensor network and prolong its network lifetime, we can periodically wake up some sensor nodes from sleep mode to collect the information while others are in sleep, and sleep them back and wake up others. In Chapter 4, we derive a closed-form solution for the visual coverage estimation in visual sensor networks to guarantee the required coverage in a sensing field. As a further extension of this work for a future research, we can study an optimal wake-up/sleep scheduling algorithm for energy saving purpose. Our approach is based on that we can estimate the number of visual sensor node by using the closed-form solution for the visual coverage estimation that ensures the coverage requirement and organize the disjoint subsets of sensor nodes and put the redundant sensors into different subsets. Then, we leave the sensor nodes in one of the subsets in the active mode to cover the sensing field and put the rest of the redundant sensor subsets into the sleep mode.

Publications

Publications

1. **M. Karakaya** and H. Qi, "Distributed Target Localization using a Progressive Certainty Map in Visual Sensor Networks", Journal of Ad Hoc Networks, vol. 9, pp. 576-590, 2011.

2. **M. Karakaya** and H. Qi, "Coverage Estimation for Crowded Targets in Visual Sensor Networks", accepted by ACM Transactions on Sensor Networks, 2011.

3. R. A. Kerekes, R. A. Martins, D. Davis, **M. Karakaya**, S. S. Gleason and M. Dyer, "Automated Tracing of Horizontal Neuron Processes During Retinal Development", Neurochemical Research, vol. 36, no. 4, pp. 583-593, 2011.

4. **M. Karakaya**, R. A. Kerekes, S. S. Gleason, R. A. Martins, and M. Dyer, "Automatic Detection, Segmentation and Classification of Retinal Horizontal Neurons in Large-scale 3D Confocal Imagery", SPIE Medical Imaging Conference, Vol. 7962, 2011.

5. **M. Karakaya** and H. Qi, "Fault Detection, Correction, and Tolerance for Collaborative Target Localization in Visual Sensor Networks", 4th ACM/IEEE International Conference on Distributed Smart Cameras, 2010.

6. **M. Karakaya** and H. Qi, "Target Detection and Counting in Crowds with a Progressive Certainty Map in Distributed Visual Sensor Networks", 3rd ACM/IEEE International Conference on Distributed Smart Cameras, 2009. **(Best Paper Award)**

Bibliography

Bibliography

Abbo, A. and Kleihorst, R. (2002). A programmable smart camera architecture. In *Advanced Concepts for Intelligent Vision Systems*, Belgium. xi, 3, 4

Abramowitz, M., S. I. A. (1970). *Handbook of Mathematical Functions*. Dover, New York, 9 edition. 110

Aghajan, H. and Cavallaro, A. (2009). *Multi-Camera Networks*. Academic Press, Oxford. 2

Ahmed, N., Kanhere, S. S., and Jha, S. (2005). Probabilistic coverage in wireless sensor networks. In *LCN '05: Proceedings of the The IEEE Conference on Local Computer Networks 30th Anniversary*, pages 672–681, Washington, DC, USA. IEEE Computer Society. 100

Ai, J. and Abouzeid, A. A. (2006). Coverage by directional sensors in randomly deployed wireless sensor networks. *Journal of Combinatorial Optimization*, 11:21–41. 101

Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114. 97

Brass, P. (2007). Bounds on coverage and target detection capabilities for models of networks of mobile sensors. *ACM Trans. Sen. Netw.*, 3(2):9. 100

Cai, Y., Lou, W., Li, M., and Li, X.-Y. (2009). Energy efficient target-oriented scheduling in directional sensor networks. *IEEE Transactions on Computers*, 58:1259–1274. 98, 101

Chen, P., Ahammad, P., Boyer, C., Huang, S.-I., Lin, L., Lobaton, E., Meingast, M., Oh, S., Wang, S., Yan, P., Yang, A., Yeo, C., Chang, L.-C., Tygar, J., and Sastry, S. (2008). Citric: A low-bandwidth wireless camera network platform. In *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pages 1–10. xi, xii, 3, 4

Clouqueur, T., Saluja, K. K., and Ramanathan, P. (2004). Fault tolerance in collaborative sensor networks for target detection. *IEEE Trans. Comput.*, 53(3):320–333. 54, 55, 57, 60

Collins, R. (1996). A space-sweep approach to true multi-image matching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Proceedings CVPR '96.*, pages 358–363. 12

Devarajan, D. and Radke, R. J. (2007). Calibrating distributed camera networks using belief propagation. *EURASIP Journal on Advances in Signal Processing*, 2007(60696). 55, 56

Ding, M., Liu, F., Thaeler, A., Chen, D., and Cheng, X. (2007). Fault-tolerant target localization in sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 2007(1):19–19. 55

Feeney, L. and Nilsson, M. (2001). Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. *INFOCOM 2001: Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, 3:1548–1557. xi, 29, 30

Fleck, S. and Strasser, W. (2008). Smart camera based monitoring system and its application to assisted living. *Proceedings of the IEEE*, 96(10):1698 –1714. 3

Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P. (2008). Multicamera people tracking with a probabilistic occupancy map. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):267–282. 13, 61, 62, 71

Fusco, G. and Gupta, H. (2009). Selection and orientation of directional sensors for coverage maximization. In *6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 1 –9. 101

Geddes, K. O., Glasser, M. L., Moore, R. A., and Scott, T. C. (1990). Evaluation of classes of definite integrals involving elementary functions via differentiation of special functions. *Applicable Algebra in Engineering, Communication and Computing*, 1:149–165. 10.1007/BF01810298. 110, 167

Gui, C. and Mohapatra, P. (2004). Power conservation and quality of surveillance in target tracking sensor networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 129–143, New York, NY, USA. ACM. 100

Guvensan, M. A. and Yavuz, A. G. (2011). On coverage issues in directional sensor networks: A survey. *Ad Hoc Networks*, In Press, Corrected Proof:–. 101

Halvorson, E. and Parr, R. (2007). Counting objects with a combination of horizontal overhead sensors. *Technical Report, Department of Computer Science Duke University*. 13

Han, M., Xu, W., Tao, H., and Gong, Y. (2004). An algorithm for multiple object trajectory tracking. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages 864–871. 12

Haritaoglu, I., Harwood, D., and Davis, L. (1998). W4: Who? when? where? what? a real time system for detecting and tracking people. In *Third IEEE International*

Conference on Automatic Face and Gesture Recognition. Proceedings., pages 222–227. 12

Haritaoglu, I., Harwood, D., and Davis, L. (1999). Hydra: multiple people detection and tracking using silhouettes. In *International Conference on Image Analysis and Processing. Proceedings.*, pages 280–285. 12

Hengstler, S., Prashanth, D., Fong, S., and Aghajan, H. (2007). Mesheye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*, pages 360–369, New York, NY, USA. ACM. xi, xii, 3, 4

Huang, C.-F. and Tseng, Y.-C. (2003). The coverage problem in a wireless sensor network. In *ACM International Workshop on Wireless Sensor Networks and Applications*, pages 115–121. ACM Press. 100

Hynes, C. S., Corps, U. S. M., and Rowe, N. C. (2004). A multi-agent simulation for assessing massive sensor deployment. *Journal of Battlefield Technology*, 7:23–36. 33, 97

Isard, M. and MacCormick, J. (2001). Bramble: a bayesian multiple-blob tracker. In *Eighth IEEE International Conference on Computer Vision, ICCV 2001. Proceedings.*, volume 2, pages 34–41. 12

Isler, V. and Bajcsy, R. (2006). The sensor selection problem for bounded uncertainty sensing models. *Automation Science and Engineering, IEEE Transactions on*, 3(4):372 –381. 101

Kahn, J. M., Katz, R. H., and Pister, K. S. J. (1999). Next century challenges: Mobile networking for smart dust. *Human Motion, 2000. Proceedings. Workshop on*, pages 271–278. 5

Kandhalu, A., Rowe, A., and Rajkumar, R. (2009). Dspcam: A camera sensor system for surveillance networks. In *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pages 1 –7. xi, 4

Karakaya, M. and Qi, H. (2009). Target detection and counting using a progressive certainty map in distributed visual sensor networks. In *Third ACM/IEEE International Conference on Distributed Smart Cameras, ICDSC 2009*, pages 1–8. 9, 10, 55, 57

Karakaya, M. and Qi, H. (2010). Fault detection, correction, and tolerance for collaborative target localization in visual sensor networks. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC '10, pages 111–118, New York, NY, USA. ACM. 54

Karakaya, M. and Qi, H. (2011a). Coverage estimation for crowded targets in visual sensor networks. *Accepted by ACM Trans. Sen. Netw.* 97

Karakaya, M. and Qi, H. (2011b). Distributed target localization using a progressive certainty map in visual sensor networks. *Ad Hoc Networks*, 9(4):576 – 590. Multimedia Ad Hoc and Sensor Networks. 9, 98

Kleeman, L. and Kuc, R. (1995). Mobile Robot Sonar for Target Localization and Classification. *The International Journal of Robotics Research*, 14(4):295–318. 12

Klein, L. (1993). A boolean algebra approach to multiple sensor voting fusion. *Aerospace and Electronic Systems, IEEE Transactions on*, 29(2):317–327. 57

Krumm, J., Harris, S., Meyers, B., Brumitt, B., Hale, M., and Shafer, S. (2000). Multi-camera multi-person tracking for easyliving. In *Third IEEE International Workshop on Visual Surveillance.*, pages 3–10. 12, 55

Kumar, S., Lai, T. H., and Arora, A. (2005). Barrier coverage with wireless sensors. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 284–298, New York, NY, USA. ACM. 100

Lamport, L., Shostak, R., and Pease, M. (1982). The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401. 57

Laurentini, A. (1994). The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162. 13

Li, J.-S. and Kao, H.-C. (2010). Distributed k-coverage self-location estimation scheme based on voronoi diagram. *Communications, IET*, 4(2):167 –177. 98

Lin, Y.-T., Saluja, K. K., and Megerian, S. (2011). Adaptive cost efficient deployment strategy for homogeneous wireless camera sensors. *Ad Hoc Networks*, 9(5):713 – 726. 102

Liu, L., Ma, H., and Zhang, X. (2008). On directional k-coverage analysis of randomly deployed camera sensor networks. In *Communications, 2008. ICC '08. IEEE International Conference on*, pages 2707 –2711. 101

Meguerdichian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M. (2001). Coverage problems in wireless ad-hoc sensor networks. In *INFOCOM . Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1380 –1387. 100

Mittal, A. and Davis, L. (2003). M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. In *International Journal of Computer Vision*, volume 51, pages 189–203. 12

Munishwar, V. P. and Abu-Ghazaleh, N. B. (2010). Scalable target coverage in smart camera networks. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC '10, pages 206–213, New York, NY, USA. ACM. 101

Poelman, C. and Kanade, T. (1997). A paraperspective factorization method for shape and motion recovery. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(3):206–218. 56

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes The Art of Scientific Computing*. Cambridge University Press, 3 edition. 111

Qian, C. and Qi, H. (2008). A distributed solution to detect targets in crowds using visual sensor networks. In *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pages 1 –10. 9, 12, 102

Rahimi, M., Estrin, D., Baer, R., Uyeno, H., and Warrior, J. (2004). Cyclops, image sensing and interpretation in wireless networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 311–311, New York, NY, USA. ACM. xi, 3, 4

Rinner, B. and Wolf, W. (2008). A bright future for distributed smart cameras. *Proceedings of the IEEE*, 96(10):1562–1564. 1

Rowe, A., Goode, A., Goel, D., and Nourbakhsh, I. (2007). Cmucam3: An open programmable embedded vision sensor. In *tech. report CMU-RI-TR-07-13*, Robotics Institute, Carnegie Mellon University. xi, xii, 3, 4

Sankaranarayanan, A., Veeraraghavan, A., and Chellappa, R. (2008). Object detection, tracking and recognition for multiple smart cameras. *Proceedings of the IEEE*, 96(10):1606 –1624. 2

Sogo, T., Ishiguro, H., and Trivedi, M. (2000). Real-time target localization and tracking by n-ocular stereo. In *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*, pages 153–160. 12

Soro, S. and Heinzelman, W. (2009). A survey of visual sensor networks. *Advances in Multimedia*, (640386):1 –21. 2

Tards, J. D., Neira, J., Newman, P. M., and Leonard, J. J. (2002). Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 21:311–330. 12

Tsalatsanis, A., Valavanis, K., and Tsourveloudis, N. (2006). Mobile robot navigation using sonar and range measurements from uncalibrated cameras. In *Control and Automation, 2006. MED '06. 14th Mediterranean Conference on*, pages 1 –7. 12

Veltri, G., Huang, Q., Qu, G., and Potkonjak, M. (2003). Minimal and maximal exposure path algorithms for wireless embedded sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 40–50, New York, NY, USA. ACM. 100

Wan, P.-J. and Yi, C.-W. (2006). Coverage by randomly deployed wireless sensor networks. *IEEE/ACM Trans. Netw.*, 14(SI):2658–2669. 100

Wang, L., Wu, F., and Hu, Z. (2007). Multi-camera calibration with one-dimensional object under general motions. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–7. 101

Wang, Y., Li, F., and Fang, F. (2010). Poisson versus gaussian distribution for object tracking in wireless sensor networks. In *2nd International Workshop on Intelligent Systems and Applications*, pages 1–4. 66, 104

Xing, G., Wang, X., Zhang, Y., Lu, C., Pless, R., and Gill, C. (2005). Integrated coverage and connectivity configuration for energy conservation in sensor networks. *ACM Transactions on Sensor Networks*, 1(1):36–72. 100

Yang, D., Gonzalez-Banos, H., and Guibas, L. (2003). Counting people in crowds with a real-time network of simple image sensors. In *Ninth IEEE International Conference on Computer Vision, Proceedings.*, volume 1, pages 122–129. 12, 13, 17, 55, 62, 104

Yang, D., Shin, J., Ercan, A., and Guibas, L. (2004). Sensor tasking for occupancy reasoning in a network of cameras. In *In Proceedings of Workshop Broadband Advance Sensor Network*. 19, 101

Yen, L.-H., Yu, C. W., and Cheng, Y.-M. (2006). Expected k-coverage in wireless sensor networks. *Ad Hoc Networks*, 4(5):636 – 650. 100, 115

Zou, Y. and Chakrabarty, K. (2004). Sensor deployment and target localization in distributed sensor networks. *ACM Trans. Embed. Comput. Syst.*, 3(1):61–91. 55

Appendix

Appendix A

List of Selected Symbols

Symbol	Description	Section
s_i	Index of a visual sensor node	2.3
(x_{s_i}, y_{s_i})	Coordinates of the sensor node s_i in the sensing field	2.3
$\mathbf{v}_{s_i}$	Vector that describes the non-occupied areas within the FOV of the sensor node, s_i	2.3
$\varphi_{i,j}$	Starting angle of the j^{th} non-occupied area	2.3
$\psi_{i,j}$	Ending angle of the j^{th} non-occupied area	2.3
j^{th}	Index of non-occupied area	2.3
B_i	Total number of non-occupied areas of the image taken by node s_i	2.3
S	Set of the visual sensor nodes	2.3
$U(S)$	Union formed by all the local certainty maps in S	2.3
$f(\mathbf{v}_{s_i})$	Mapping function to convert $\mathbf{v}_{s_i}$ to the certainty map	2.3
$\|f(\mathbf{v}_{s_i})\|$	Total area that can be cleared from the current certainty map by sensor node s_i	2.5.1
δ	Threshold to determine if the sensor node holds adequate additional clarification information	2.5.1

Symbol	Description	Section
N	Number of visual sensor nodes	2.7
M	Number of targets	2.7
N_d	Number of involved sensor nodes that in dynamic itinerary	2.7
N_o	Number of involved sensor nodes that in fixed or random itineraries	2.7
D	Size of the data vector	2.7
$\mathbb{E}$	Total energy consumption during the transmission of certainty maps	2.7
c	Energy cost per byte	2.7
ξ	Fixed cost for different communication modes	2.7
$\mathbb{E}^{send}$	Energy consumption for sending data	2.7
$\mathbb{E}^{recv}$	Energy consumption for receiving data	2.7
$\mathbb{E}^{disc}$	Energy consumption for discarding data	2.7
θ_{s_i}	Actual (or inaccurate) orientation of the i^{th} sensor node, s_i	3.3
$\theta^*_{s_i}$	Ground truth (or calibrated) orientations of the i^{th} sensor node, s_i	3.3
$N_{s_i}(0,\sigma)$	Gaussian noise with zero-mean and standard deviation σ in the orientation	3.3
δ_{s_i}	Byzantine fault in the orientation	3.3
$C_{x,y}$	Number of sensor nodes that covers a grid pixel at coordinate (x, y)	3.3
$V_{x,y}(S)$	Normalized voting algorithm value a grid pixel at coordinate (x, y)	3.3
Ψ	Normalized pseudo-distance between two 1D scanline images	3.3
$\theta^e_{s_i}$	Expected camera orientation	3.3

Symbol	Description	Section
K	K-coverage requirement, i.e., at least K nodes covers a specific grid point	4.1
k	Number of sensors that covers a point	4.4
r	Target radius	4.3
ρ	Sensing range of a camera	4.3
θ	Camera angle of view in degrees	4.3
R	2D sensing field	4.3
λ_t	Target density	4.3
λ_s	Sensor density	4.3
A	Circular detectability area	4.4
(x, y)	Coordinates of a sensor node in the sensing field	4.4
$P(k)$	Probability that exactly k sensor nodes cover a specific grid point of the sensing field	4.4
p	Probability of facing the sensor node towards the center of detectability area	4.4
$\mathcal{C}_k^j$	Number of combinations of k-node subset from a j-node set	4.4
$\mathcal{P}(j; \lambda_s A)$	Probability that a detectability area A contains exactly j sensor nodes from a Poisson point process	4.4
A_o	Area of the occlusion zone	4.5
q	Probability that there is no visual occlusion between the grid point and the sensor node	4.5
l	Distance between the grid point and the node	4.5
$Q = p \times q$	Probability of covering a specific grid point of the sensing field R	4.5
s_i	Index of a visual sensor node	4.5
N_s	Number of visual sensor nodes	4.5
N_t	Number of targets	4.5

Symbol	Description	Section
$f(l)$	probability density function (pdf) of distance l_i between the corresponding grid point and each sensor node s_i	4.5
$f(Q)$	probability density function (pdf) of function Q	4.5
$\Gamma(k, \lambda)$	Upper incomplete gamma function	4.5
$\mathrm{T}(3, k, \lambda)$	A special case of Meijer G-function	4.5
$\mathrm{E}_1(\lambda)$	Exponential integral	4.5
$F_p(k; \lambda)$	Cumulative probability distribution (cdf) of Poisson distribution with parameter k and λ	4.5
A^C	Detectability area of the corner sub-regions	4.7
A^S	Detectability area of the side sub-regions	4.7
A^M	Detectability area of the middle sub-regions	4.7
$M \times N$	Size of the rectangle sensing field	4.7.1
(u, v)	Minimum distances from a grid point to two borders of the $M \times N$ rectangle sensing field	4.7.1
ε	Tolerance value	4.8
$\hat{\lambda}_s$	Minimum sensor density	4.8

Appendix B

Derivations and Proofs

B.1 Probability of Radial distance, l_i

Suppose sensor nodes appears at a point uniformly distributed at random on a circular plane, A, of radius ρ in the sensing field, R. Let L be the distance of the point from the center of the circular detectability area A.

The basic assumption is that the probability of sensor nodes appears in a particular region of the circular detectability area A is proportional to the area of the region. From the Fig. B.1, $0 \leq l \leq \rho$,

$$\begin{aligned} P(R \in dl) &= \frac{\text{Area of annulus from l to l+dl}}{\text{Total area}} \\ &= \frac{\pi(l+dl)^2 - \pi l^2}{\pi \rho^2} \\ &= \frac{2l}{\rho^2} dl \end{aligned} \tag{B.1}$$

by ignoring the term involving $(dl)^2$. Therefore, L has probability of density function as

$$f(l) = \begin{cases} \dfrac{2}{\rho^2} l & \text{for } 0 \leq l \leq \rho \\ 0 & \text{for otherwise} \end{cases} \tag{B.2}$$

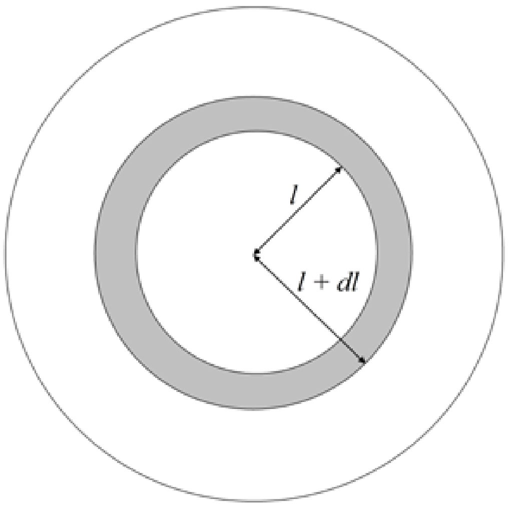


Figure B.1: Area at particular radial distance.

where $f(l)$ is the pdf of distance l_i between the corresponding grid point and each sensor node s_i. It follows linear distribution from 0 to ρ.

B.2 Probability of function $Q(l)$

To compute the pdf of the function Q, $f(Q)$ from the pdf of distance l, Eq. B.2, we utilized the change of variable property. Since Q is a monotonically decreasing function, $f(Q)$ is

$$f(Q) = f_L(g^{-1}(Q)) \times \left| \frac{dg^{-1}(Q)}{dQ} \right| \tag{B.3}$$

Let $Q = g(l) = p \times q = \dfrac{\theta}{2\pi} \times e^{-\lambda_t(\pi r^2 + 2rl)}$. Then,

$$g^{-1}(Q) = \frac{\ln\left(\frac{\frac{\theta}{2\pi} e^{-\lambda_t \pi r^2}}{Q}\right)}{2\lambda_t r}$$

and,

$$\frac{dg^{-1}(Q)}{dQ} = -\frac{1}{2\lambda_t r Q}.$$

Hence, $f(Q)$ is

$$f(Q) = \begin{cases} \dfrac{2}{\rho^2} \times \dfrac{\ln\left(\frac{\frac{\theta}{2\pi}e^{-\lambda_t \pi r^2}}{Q}\right)}{2\lambda_t r} \times \dfrac{1}{2\lambda_t r Q} & \text{for } Q(l=\rho) \leq Q \leq Q(l=0) \\ 0 & \text{otherwise} \end{cases} \tag{B.4}$$

B.3 Derivative of Incomplete Gamma Function

Let $\Gamma(k, \lambda)$, the upper incomplete gamma function, is

$$\Gamma(k, \lambda) = \int_{\lambda}^{\infty} u^{k-1} e^{-u} du. \tag{B.5}$$

The derivative of the upper incomplete gamma function is

$$\begin{aligned} \frac{\partial}{\partial k}\Gamma(k, \lambda) &= \frac{\partial}{\partial k}\left(\int_{\lambda_b}^{\infty} u^{k-1} e^{-u} du\right) \\ &= \int_{\lambda_b}^{\infty} \frac{\partial}{\partial k}\left(u^{k-1} e^{-u} du\right) \\ &= \int_{\lambda_b}^{\infty} u^{k-1} e^{-u} \ln u du \end{aligned} \tag{B.6}$$

To further derivation of derivative of the upper incomplete gamma function, we can use Meijer G-function as

$$\frac{\partial}{\partial k}\Gamma(k, \lambda) = \ln \lambda \Gamma(k, \lambda) + \lambda \mathrm{T}(3, k, \lambda) \tag{B.7}$$

where $\mathrm{T}(3, k, \lambda)$ is a special case of Meijer G-function, Geddes et al. [1990].

B.4 Recurrence Formula of the function $\mathbf{T}(m,k,\lambda)$

By using integration by parts in Eq.B.5, we find the simple recurrence formula of the upper incomplete gamma function as

$$\Gamma(k+1,\lambda) = k\Gamma(k,\lambda) + e^{-\lambda}\lambda^k \tag{B.8}$$

To find the simple recurrence formula of the function $\mathrm{T}(m,k,\lambda)$, we substitute Eq.B.8 into Eq.B.7 as

$$\begin{aligned}\frac{\partial}{\partial k}\Gamma(k+1,\lambda) &= \frac{\partial}{\partial k}\Bigg(k\Gamma(k,\lambda)+e^{-\lambda}\lambda^k\Bigg)\\ \ln\lambda \times \Gamma(k+1,\lambda) + \lambda\mathrm{T}(3,k+1,\lambda) &= \Gamma(k,\lambda) + k\Bigg(\ln\lambda\Gamma(k,\lambda)+\lambda\mathrm{T}(3,k,\lambda)\Bigg) + e^{-\lambda}\lambda^k\ln\lambda\\ \lambda\mathrm{T}(3,k+1,\lambda) &= k\lambda\mathrm{T}(3,k,\lambda)+\Gamma(k,\lambda) \qquad \text{(B.9)}\end{aligned}$$

By repeating the recursive formula of function $\mathrm{T}(3,k,\lambda)$, we find its generalized version as

$$\begin{aligned}\lambda\mathrm{T}(3,k+1,\lambda) =& k\lambda\mathrm{T}(3,k,\lambda)+\Gamma(k,\lambda)\\ =& k\Big((k-1)\lambda\mathrm{T}(3,k-1,\lambda)+\Gamma(k-1,\lambda)\Big)+\Gamma(k,\lambda)\\ =& k(k-1)\Big((k-2)\lambda\mathrm{T}(3,k-2,\lambda)+\\ &+\Gamma(k-2,\lambda)\Big)+(k-1)\Gamma(k-1,\lambda)+\Gamma(k,\lambda)\\ &\vdots\\ =& \frac{k!}{(k-1-j)!}\lambda\mathrm{T}(3,k-j,\lambda)+\sum_{i=0}^{j}\frac{k!}{(k-i)!}\Gamma(k-i,\lambda)\end{aligned} \tag{B.10}$$

For $k-1 \to k$ and $j \to k-2$,

$$\begin{aligned}
\lambda \mathrm{T}(3,k,\lambda) =& \frac{(k-1)!}{(k-1-1-k+2)!}\lambda \mathrm{T}(3,k-1-k+2,\lambda)+ \\
&+\sum_{i=0}^{k-2}\frac{(k-1)!}{(k-1-i)!}\Gamma(k-1-i,\lambda) \\
=&(k-1)!\lambda \mathrm{T}(3,1,\lambda)+\sum_{i=1}^{k-1}\frac{(k-1)!}{(i)!}\Gamma(i,\lambda) \\
=&(k-1)!\mathrm{E}_1(\lambda)+(k-1)!\sum_{i=1}^{k-1}\frac{\Gamma(i,\lambda)}{(i)!}
\end{aligned} \tag{B.11}$$

where $\lambda \mathrm{T}(3,1,\lambda)$ equals to $\mathrm{E}_1(\lambda)$ which is the exponential integral.

B.5 Derivation of Heterogeneous Visual Sensor Network

$\mathcal{C}_k^n$ denotes the number of combinations of k-element subset from a n-element set. The number of k-combinations is equal to the binomial coefficient which can be written using factorials as $\mathcal{C}_k^n = \frac{n!}{k!(n-k)!}$. By using this factorial enpension,

$$\begin{aligned}
\frac{\mathcal{C}_m^i \mathcal{C}_{k-m}^{j-i}}{i!(j-i)!} &= \frac{\frac{i!}{m!(i-m)!} \times \frac{(j-i)!}{(k-m)!(j-i-(k-m))!}}{i!(j-i)!} \\
&= \frac{1}{m!(i-m)!} \times \frac{1}{(k-m)!(j-i-(k-m))!} \\
&= \frac{\mathcal{C}_m^k}{k!} \times \frac{\mathcal{C}_{i-m}^{j-k}}{(j-k)!} \\
&= \frac{\mathcal{C}_m^k \mathcal{C}_{i-m}^{j-k}}{k!(j-k)!}
\end{aligned} \tag{B.12}$$

Vita

Mahmut Karakaya received the B.S. degree from Fatih University, Istanbul, Turkey in 2005 and the M.S. degree from University of South Alabama in 2007. To pursuit Ph.D., he enrolled the graduate program in the department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville in 2007. He is a graduate research assistant with the Advanced Imaging & Collaborative Information Processing (AICIP) Lab. During his Ph.D. study, he focused on computer vision, digital image processing, and distributed visual sensor networks. In Summer 2010, he worked as an intern research engineer at Imaging, Signals, and Machine Learning Group at Oak Ridge National Laboratory. He will complete the Doctor of Philosophy degree in Computer Engineering in Summer 2011. Mr. Karakaya received the Best Paper award from ICDSC 2009.