12-2006

A Distributed Solution for Visual Sensor Networks to Detect Targets in Crowds

Cheng Qian
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Cheng Qian entitled "A Distributed Solution for Visual Sensor Networks to Detect Targets in Crowds." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Hairong Qi, Major Professor

We have read this thesis and recommend its acceptance:

Lynne Parker, Itamar Elhanany

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

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

To the Graduate Council:

I am submitting herewith a thesis written by Cheng Qian entitled "A Distributed Solution for Visual Sensor Networks to Detect Targets in Crowds." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Hairong Qi

Hairong Qi, Major Professor

We have read this thesis
and recommend its acceptance:

Lynne Parker

Itamar Elhanany

Accepted for the Council:

Anne Mayhew

Vice Chancellor and Dean of
Graduate Studies

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

A Distributed Solution for Visual Sensor Networks to Detect Targets in Crowds

A Thesis
Presented For The
Master of Science Degree
The University Of Tennessee, Knoxville

Cheng Qian
December 2006

Acknowledgement

I would like to thank Dr Qi who gave me the chance to complete this thesis work and guided me through the amazing VSNs research. I would like to thank Dr. Parker and Dr. Elhanany for their services on my committee. I would like to thank my friends, Jingqiao Zhang, Li Yong, and JianMing Wu for their help with my research. I would like to thank all the AICIP members, especially Chris Beall for his free technical support. I would like to thank my teachers and friends in UT, Forrest Chen, Yuanyuan Li, Chunghao Chen, Mohammad S Huq, Joshua S Yuan, Yue Zheng, Yao Yi, Chang Cheng, Sijie Yu, Shaoyu Liu, Yunqiang Yang, and Lin Song who have been supporting me so much through this special year.

Lastly, I would like to thank my family, advisors, mentors, and friends back in China as well as my aunt in US whose understanding and encouragement make everything possible for me.

Abstract

Visual sensor networks (VSNs), a novel concept about fulfilling vision tasks by a network of collaborative visual sensors, has been attracting more and more attentions these days. This thesis introduces some pioneering research on developing a distributed algorithm for VSNs to detect targets in a cluttered scene. The algorithm is aimed to achieve excellent performances on both detection accuracy and energy efficiency.

Based on a statistical model of the cluttered scene, the development starts with a centralized version where all the nodes send visual data to a central node and the central node invokes an iterative prioritization strategy (IPS) to make globally optimal detecting decisions. Although resulting in excellent detection accuracy, the centralized fashion causes poor performance on energy utilization.

The algorithm is then transformed into a distributed version where the entire scene is partitioned into a Voronoi diagram and each node is only responsible for detecting targets inside its local polygon area. There are two challenges in realizing such a transformation. The first challenge is to design an energy-efficient method to exchange visual data among relevant nodes. A "back-projecting" strategy (BBR) is therefore created to tackle this challenge. Instead of sending request to nodes that have relevant data, the method initiates the data communication from source nodes. Each packet of visual data is then relayed towards the place where is located the target corresponding to the visual data. All the relevant data about the target will finally reach there and thereafter can be fused. This strategy enables the parallelism between transmitting visual data and integrating visual data for detection. With this parallelism, knowledge from partial detection results can be used to guide the transmission and therefore improve energy efficiency. The second challenge is to design a method to fuse decisions independently made by each node through small amount of mutual communication. A modified one-shot threshold strategy (MOTS) is proposed to tackle this challenge. By receiving small amount of data from related nodes, a local measure can be constructed to validate or invalidate local decisions.

Compared with the centralized algorithm, this distributed algorithm demands less energy cost for a large-scale VSN and at same time sustains satisfactory detection accuracy.

An experiment is presented in the end and the experimental results are analyzed.

Contents

List of Figures

1 Introduction

With nowadays advances in CMOS imaging devices, system-on-chip (SOC) technologies, and wireless communications, the visual sensor networks (VSNs) concept surfaced and has been attracting increasing attention. In a VSNs, each node carries sensing, computing, and wireless communication capacities. Once those nodes are deployed, they are able to connect to each other to form an ad-hoc network automatically, and then execute certain tasks with collaborations. Within reasonable budget, swarm of visual motes can be deployed in various environments to take on various tasks such as environmental surveillance, object detection/tracking, and remote videography [Obr02, Aky02].

This thesis is focused on designing a target detection algorithm for VSNs, especially for the cases where targets are crowded and occluding one another. The algorithm is expected to be generic and be able to tackle different type of objects or backgrounds, which could be vehicles in a parking lot, persons in an office, or wild horses resting in a valley. The algorithm should make full use of the visual information captured by each node so as to enable a VSN system to be aware of target's existence or emergence and track them down to their locations.

1.1 VSNs vs Multi-Perspective Systems

As a matter of fact, target detection has been heatedly discussed over decades but limited within the computer vision society [Yan03, Yan04, Mit02, Har98, Kru00]. There, cameras are placed at different positions and with different orientations. Images from different cameras are transmitted to a central computer where, by using multi-perspective geometry, targets in the scene can be detected, localized, and reconstructed. Seemingly similar to this multi-perspective system approach, a VSN system has some fundamentally different characteristics:

Firstly, in a multi-perspective system, those cameras are only responsible for capturing images and transmitting images to a highly powerful computer for processing. While in a VSN, each node is more than a camera. They also carry certain data processing capacities, though limited, and are capable of collaborating with one another to fulfill the computation of the task.

Secondly, even for large scale multi-perspective systems, the number of cameras is relatively limited, and cameras are never supposed to be densely deployed. Moreover, a strategic plan is usually made in advance. Most of time, cameras are placed at some advantageous positions in the scene, such as along a bounding circle or regular grid placement to ensure a full coverage. In contrast, nodes in a VSN can be assumed to be densely and arbitrarily deployed. Imagine an ad-hoc VSN for collecting urban information built among the sensors carried by thousands of pedestrians randomly walking in the Times Square, or a VSN for military spying formed by the sensors air-dropped from an airplane.

Thirdly, the main concern for multi-perspective systems is how to make full use of information collected by all the cameras so that interferences of occlusion can be overcome and locations of the targets can be accurately estimated. There are no assumed constraints for the amount of time when cameras are turned on or amount of data transmitted from the cameras to the central computer. While in VSNs, we have to achieve good information processing results with calculated energy utilization. As a matter of fact, the concern for energy efficiency has been haunting around the development of sensor networks since day one. In most cases, the node maintains itself with on-board battery. Once the node is deployed, it is very difficult or sometimes impossible to replace the battery. Therefore, how to improve energy efficiency and thereby prolong the lifetime of nodes is a crucial obstacle for the prospect of this new technology. With the well known fact that most energy is consumed by radio transmission [Zha04], this energy concern sometimes ends up with how to minimize the amount of data communications

between nodes. It is unacceptable that a node ignores its ability to process visual information and directly outputs raw images for communication.

These different characteristics highlight a set of principles for us to comply with when we are designing a detection algorithm for VSNs. In other words, to ensure that netting a population of so-called sensors with limited computing abilities over-performs a traditional vision system composed of several high quality cameras and powerful central computer, the algorithms designed for VSNs must be able to achieve following objectives:

(1) Fully exploit the resources of all the nodes in the network. Besides energy, "resources" also refers to other assets and capacities possessed by nodes that can contribute to the fulfillment of the task, such as capacities of CCD sensor, capacities of the processor, size of the memory, and bandwidth of the wireless channel. This thesis only focuses on energy utilization.
(2) Balance the exploitation over all the nodes.

1.2 Distributed vs Centralized

To achieve these objectives we are encouraged to design the target detection algorithm in a different fashion. Most of the algorithms for multi-perspective systems follow a centralized (client/server) fashion, by which nodes in the entire network all report their data to a central server and the server integrates all the data and executes the detection task on its own (figure 1.1.a) . The centralized fashion has some advantages such as in accuracy of detection. It may be able to achieve globally optimal results, because the server can make the detection upon the information from all the nodes. However, this fashion also suffers from the requirement for having a super powerful node play for the server that takes on most of the computing work, which does not quite match the team-working style propagandized by VSNs. Moreover, this fashion results in large amount of

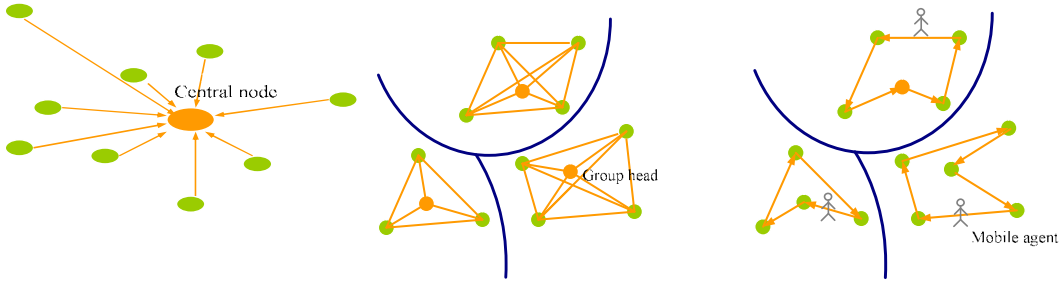


Figure 1.1 Algorithm fashions for VSNs. (a) Centralized fashion. (b) Group-based distributed fashion. (c) Mobile-agent distributed fashion.

data swarming into the server. As we know, wireless communication consumes most of the energy in a large-scale VSN. Therefore this fashion may easily wear out the batteries of those nodes near the server.

On the other side against this centralized fashion is a distributed fashion, where the task is decomposed into subtasks that are distributed to different nodes to finish. A typical tactic to realize this fashion is to group related nodes and thereby localize most of computation for the subtasks into the groups. For instance, in each group, a group head is elected that receives data from the members and executes the detection algorithm for all the members (figure 1.1.b) [Sor05, Ghi02], or a mobile agent that carries target detection software migrates among nodes progressively, collects information, and updates detection decision (figure 1.1.c) [Xuy04, Qih02, Rab04]. This fashion brings more nodes to work for the computation of the task. Moreover, since most of the visual data are processed within local groups, it also offers more space for improving the efficiency in energy exploitation. However, under this fashion it is a challenge to achieve as good detection results as those centralized algorithms, because each group executes the subtasks based on local data, and the decision made is only locally optimized.

Thus, the goal of this thesis comes down to whether we can decentralize a centralized algorithm serving for multi-perceptive systems to be a distributed algorithm serving for

VSNs, and the resulting distributed algorithm should not only have better performance regarding to utilizing energy resources, but also achieve excellent detection results comparable to the centralized version. In this thesis, we propose such a decentralization strategy. In this algorithm, the entire network is partitioned into a Voronoi diagram, and each node is responsible for detecting the targets inside local polygon. The algorithm consists of two main steps. The first step is to route visual information to related nodes for integration with well-controlled energy consumption, and the second step is to remove those false local decisions made by individual nodes.

1.3 Contributions and Document Organization

The ultimate goal of this thesis is to design a distributed solution for VSNs to detect targets in crowds. However, to approach this goal, we have to conquer a sequence of difficult problems that have never been addressed before. We list what this thesis contributes to the VSNs research as follows.

(1) Model VSNs problems in a statistical way. The statistical model we propose in this thesis can facilitate the design of VSNs algorithms and evaluation of their performances. The model is especially designed for the situation where targets are massively and crowdedly located. There are two major characteristics about the situation: One, occlusions between targets cannot be ignored. Two, features of targets cannot be considered as being selective. Both the characteristics are addressed in the model.

(2) Create a so-called Iterative Prioritization Strategy (IPS) to “reason out” targets based on ambiguous visual information. IPS is designed for detection in a centralized fashion. We can prove that IPS is equivalent to a sequence of binary Bayesian classifications, which is actually trading a small increase in missed error for a large decrease in false error and over-performs the straightforward “one-shot” threshold strategy with lower total detection errors

(3) Tackle a challenge in designing a distributed VSN algorithm, i.e. to design an energy-efficient method to exchange visual data among relevant nodes, by a so-called "Broadcasting Through Back-Projecting Relays" (BBR) strategy. In BBR, efforts to search for relevant nodes are discarded, but relevant visual data can still converge to the same place for integration within efficient energy consumption. This broadcasting strategy also enables a parallelism between transmitting visual data and integrating visual data for detection.

(4) Tackle another challenge in designing a distributed VSN algorithm, i.e. to fuse local decisions made by different nodes with small amount of wireless communication, by a so-called Modified One-Shot Threshold Strategy (MOTS). This strategy is about constructing a measure based on limited-scope of knowledge from related nodes in order to validate or invalidate local decisions. We can prove that, under a certain hypothesis, detecting decisions made by MOTS based on limited scope of knowledge are statistically related with detecting decisions made by IPS based on global knowledge.

The thesis is organized as follows. Chapter 2 presents a statistical model for a cluttered scene as a basis for developing the following algorithms. Chapter 3 describes a method to extract target features from images. Chapter 4 presents a centralized detection algorithm that adopts IPS to detect targets, and specially discusses IPS's advantageous performance on detection accuracy. Chapter 5 first proposes two challenges in decentralizing the centralized detection algorithm and then presents two strategies, i.e. BBR and MOTS, as the solutions. Chapter 6 evaluates the algorithms' performances on energy efficiency. Chapter 7 presents the results of an experiment. This thesis is concluded by chapter 8.

2 Statistical Model of a Cluttered Scene

Specifics about the environmental scene, such as the density of nodes, density of targets, and resolution of visual sensors, have apparent impact on algorithms' performances, and it does justice for algorithms if we evaluate them in an unbiased scene. Here we define such a radius *R* scene where nodes and targets are statistically uniformly distributed with node density as ρ_s and target density as ρ_t. The scene area is supposed to be large enough that statistical rules can hold and boundary situations can be ignored in discussion. We also assume that the ground of the scene is horizontal, and visual sensors on all the nodes are mounted at same height and pointed horizontally.

In the following, we describe some formulations which will be frequently related in later chapters when it comes to evaluate algorithms' performances.

2.1 Statistics about Nodes and Targets

Without occlusions how many nodes capture a same target? Let us first assume the scene area is limited, i.e. A. The total number of nodes in the scene is $N_s = \rho_s A$, and the number of targets is $N_t = \rho_t A$. Let d and θ be the radius and angle of the field of view (FOV) of visual sensors on the nodes, and therefore the area of the FOV is $A_{FOV} = d^2\theta/2$. See figure 2.1. For a specific target, without considering occlusions among targets, the probability of its being seen by a specific node is $q = A_{FOV}/A$, and the probability of its being seen by *n* nodes is

$$p_s(n, A) = C_{N_s}^n q^n (1-q)^{N_s - n} \tag{2-1}$$

As $A \to \infty$, $p_s(n, A)$ converges to

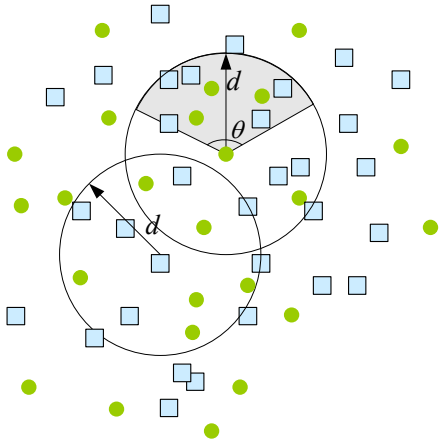


Figure 2.1 A scene where nodes (dot) and targets (cube) are uniformly deployed.

$$p_s(n) = \lim_{A\to\infty} p_s(n, A) = \lim_{A\to\infty} \frac{C_{N_s}^n}{\left(\frac{1-q}{q}\right)^n} (1-q)^{N_s} = \frac{\left(\rho_s A_{FOV}\right)^n}{n!} e^{-\rho_s A_{FOV}} \tag{2-2}$$

where $\rho_s A_{FOV}$ is equal to the average number of nodes that capture this target.

Without occlusions how many targets are captured by a single node? Without interference of occlusions, the event that a node captures a target is independent from events that the node captures other targets, and therefore we have the probability that a specific node capturing *n* targets as

$$p_t(n, A) = C_{N_t}^n q^n (1-q)^{N_t - n} \tag{2-3}$$

As $A \to \infty$, $p_t(n, A)$ converges to

$$p_t(n) = \lim_{A\to\infty} p_t(n, A) = \frac{\left(\rho_t A_{FOV}\right)^n}{n!} e^{-\rho_t A_{FOV}} \tag{2-4}$$

where $\rho_t A_{FOV}$ is equal to the average number of targets that is captured by a node.

With occlusions how many nodes capture a same target? Occlusions block a sensor to see some targets that, however, still stay inside its FOV. For a scene where targets are crowded, occlusions must be considered, and the resulting statistics is much more

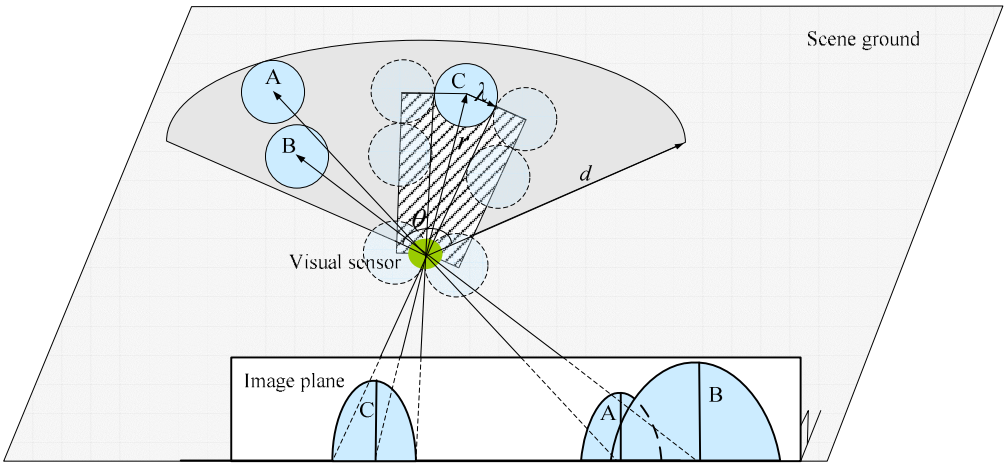


Figure 2.2 Occlusion model. The hatched region represents the occlusion zone for target C to be seen by the node. Target A is occluded by target B, because target B stays inside the occlusion zone of target A.

complicated. Here we consider that all the targets have similar height and isotopic horizontal scale of radius λ. To make sure a target can be fully captured by a node, the zone between the target and the node must be clear of other targets. The hatched region in figure 2.2 illustrates such an occlusion zone. The generalization of this occlusion model can be justified by those cylinder-shape objects, such human bodies, and those cone-shape objects that can be approximately considered as having a vertical axis and having most of "shape energy" contained in a cylinder space around the axis, such as automobiles.

Let us first consider the occurrence that a specific node captures a specific target located in the FOV at a distance *r*. The area of the occlusion zone is $\Delta A(r) = 3\lambda\sqrt{r^2 - \lambda^2}$ (figure 2.2). The probability of the occurrence is

$$\tilde{q}(r) = \frac{\theta r}{A}\left(\frac{A - 3\lambda\sqrt{r^2 - \lambda^2}}{A}\right)^{N_t - 1} \tag{2-5}$$

and as $A \to \infty$, $\tilde{q}(r)$ converges to

$$\lim_{A\to\infty}\tilde{q}(r)=\lim_{A\to\infty}\frac{\theta r}{A}\times\lim_{A\to\infty}\left(\frac{A-3\lambda\sqrt{r^2-\lambda^2}}{A}\right)^{N_t-1}=\lim_{A\to\infty}\frac{\theta r}{A}\times e^{-3\rho_t\lambda\sqrt{r^2-\lambda^2}} \tag{2-6}$$

The minimum *r* is reached when the target occludes the entire FOV, i.e. $r_1=\lambda/sin(\theta/2)$. The maximum *r* is reached when the contour of the target touches the distant edge of the FOV, i.e. $r_2=\sqrt{d^2+\lambda^2}$. The probability that the target is seen by the node is expressed as:

$$\tilde{q}=\int_{r_1}^{r_2}\tilde{q}(r)dr \tag{2-7}$$

Then we can express the probability that a specific target is seen by *n* nodes as

$$\tilde{p}_s(n)=\lim_{A\to\infty}C_{N_s}^{n}\tilde{q}^n(1-\tilde{q})^{N_s-n}=\frac{\left(\rho_s\tilde{A}_{FOV}\right)^n}{n!}e^{-\rho_s\tilde{A}_{FOV}} \tag{2-8}$$

where $\tilde{A}_{FOV}=\frac{\theta}{b^2}\left[e^{bd}(bd-1)-e^{b\lambda ctg(\theta/2)}(b\lambda ctg(\theta/2)-1)\right]$, and $b=-3\rho_t\lambda$. $\tilde{A}_{FOV}$ can be viewed as an effective FOV when occlusions are considered, which transforms the complicated occlusion interferences to a simple influence on the FOV of the node. When the occlusion interference vanishes, i.e. $\lambda\to 0$, $\tilde{A}_{FOV}\to\theta d^2/2$ and $\tilde{p}_s(n)\to p_s(n)$.

With occlusions how many targets are captured by a single node? This statistics is even more complicated to obtain, because, once occlusions are considered, the visibilities of different targets to a node are no longer independent from each other. In this thesis, we give up the efforts to calculate the specific probability $p_t(n)$ but give an average measure.

The average number of targets captured by a single node $\overline{n}_t$ is related to the average number of targets that capture a same target $\overline{n}_s$. The straightforward relationship between $\overline{n}_t$ and $\overline{n}_s$ is

$$\rho_s\overline{n}_t=\rho_t\overline{n}_s \tag{2-9}$$

where $\overline{n}_s=\sum_{n=0}^{\infty}np_s(n)=\rho_s\tilde{A}_{FOV}$. Therefore we have

$$\overline{n}_t = \rho_t \tilde{A}_{FOV} \tag{2-10}$$

Formula (2-10) again shows that $\tilde{A}_{FOV}$ acts as an effective FOV for a node when occlusions are considered.

2.2 Statistics about Target Features

Besides the distribution of targets and nodes, evaluating algorithms' performances, especially detection accuracy, also involves statistics about target features. As will be discussed later, each node extracts certain space-invariant target features such as shape, color, and texture, from the images it captures. These features are the basis of fusing visual information from different nodes. We always hope that each target can bear selective features, so that features corresponding to different targets can be distinguished, and features corresponding to same targets, though extracted by different nodes, can still coincide with each other. However, the truth is that, in a cluttered environment with massive targets, there is always possibility that some targets may take on close features, such as persons are dressed in similar color. More likely, features may be projected onto the image with errors, for instance, because the color performance of visual sensors is poorly calibrated. Therefore mistakes such as failing to attribute a feature to the right target, attributing a feature to a wrong target, and integrating features actually from different targets, might happen.

Considering only one kind of feature is to be used, we define the statistics of three mistakes as follows. Let f_A denote the real feature that target A bears, which is assumed to be a uniformly random value in the range $[F_1, F_2]$. Let $f_{i,A}$ denote the feature extracted from the visual information of target A that node i captures, which is assumed to be a random value between $[f_A - \Delta f/2, f_A + \Delta f/2]$, where Δf denotes measurement error of node s.

Mistake 1. A feature fails to be attributed to the right target. Suppose a feature $f_{i,A}$ is actually corresponding to target *A* but fails to be attributed to target *A*. This mistake happens when

$$\left|f_{i,A} - f_A\right| > \delta / 2 \tag{2-11}$$

where δ is a threshold we use to define similar features. The probability that this mistake happens on a specific feature value $f_{i,A}$ is

$$p_{f1} = \begin{cases} \left|\delta - \Delta f\right| / \Delta f & if\ \delta < \Delta f \\ 0 & otherwise \end{cases} \tag{2-12}$$

Mistake 2. A feature is misattributed to a wrong target. The mistake that feature $f_{i,A}$ is misattributed to target *B* occurs when

$$\left|f_{i,A} - f_B\right| < \delta / 2 \tag{2-13}$$

The probability that this mistake happens on a specific feature value $f_{i,A}$ is

$$p_{f2} = \frac{\delta}{(F_2 - F_1)} \tag{2-14}$$

Notice that δ regulates the trade between the two probabilities p_{f1} and p_{f2} above. A small δ leads to small risk of mistake 2 but big risk of mistake 1. There is no doubt that we should always choose a δ less than Δf, since larger δ will not help to reduce either mistake.

Mistake 3. Similar features in *n* random features. Features that should be attributed to different targets may be similar to each other by chance and considered as being attributed to a "false" target by fault. For arbitrary two features $f_{i,A}$, $f_{j,B}$ among *n* random features, they are similar to each other when

$$\left|f_{i,A} - f_{j,B}\right| \le \delta \tag{2-15}$$

$p_{f3}(n,m)$ represents the probability that, among *n* random features, there are at most *m* mutually similar features. Since $p_{f3}(n,m)$ is very difficult to calculate, we provide an

approximate version by assuming that the whole range $[F_1, F_2]$ is divided into intervals of width δ, and only features staying in a same interval are considered as mutually similar.

The mistake is therefore converted to a multi-nominal problem, and the probability can be expressed as

$$p_{f3}(n,m) = \sum_{(m_1,m_2,...m_L)\in\upsilon} \frac{n!}{m_1!m_2!...m_L!}(p_{f1})^n, \quad \upsilon: m_1+m_2...+m_L = n \text{ and } m_1, m_2...m_L \le m \tag{2-16}$$

where $L = \left|F_2 - F_1\right|/\delta$ and $ceil(n/L) \le m \le n$ [1]. However, this formula contains a sum over m dimensional space and is very difficult to implement on computer since m is a variable. Here we provide a recursion form for this probability, which can be easily implemented on computer. Let $C(n,L,m,l)$ denote number of ways to distribute n features to L intervals so that l intervals have m features, and we have

$$C(n,L,m,l) = C_L^l C_n^m C_{n-m}^m ... C_{n-(l-1)m}^m = C_L^l \frac{n!}{(m!)^s (n-lm)!} \tag{2-17}$$

Let $D(n,L,m)$ denote number of ways to distribute n features to L intervals so that all the intervals have no more than m features[2].

$$D(n,L,m) = \sum_{l=0}^{floor(n/m)} C(n,L,m,l) D(n-lm, L-l, m-1) \tag{2-18}$$

Then we have

$$p_{f3}(n,m) = \frac{D(n,L,m)}{L^n} \tag{2-19}$$

The three dimensional recursion in calculating $D(n,m,k)$ can be implemented by a dynamic programming routine.

Figure 2.3 illustrate the shape of this function $p_{f3}(n,m)$ when L= 20.

[1] *ceil*(*x*) function is to round *x* up to the next integer .
[2] *floor*(*x*) function is to round *x* down to the next integer

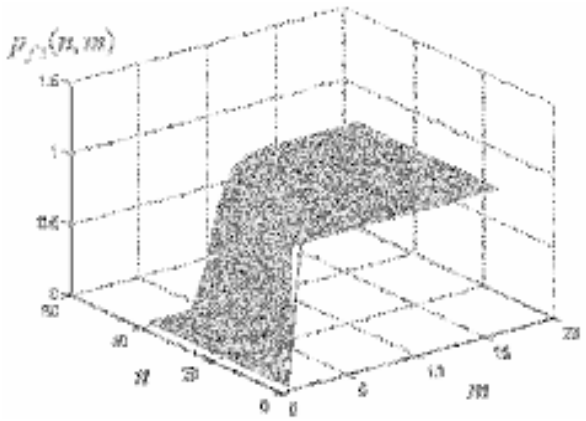


Figure 2.3 $p_{f3}(n,m)$ when L =20.

3 Feature Extraction

As the pre-stage of detecting targets, each node extracts target features from image and dispatches them into the network. Because of limited processing ability each node possesses, the extraction method should be lightweight. Some complicated methods may achieve better results but may not be suited in this case.

Step 1. This scenario of feature extraction starts with silhouette extraction. A silhouette image can be directly generated by a subtraction from a background model. The background model resides in the sensor buffer, which can be a background image captured when the background scene is static, or a statistical representation trained by sequences of background images [Dar98,Har98]. A silhouette image contains most of the shape and texture information about the targets captured by the node.

Step 2. The silhouette image can be roughly segmented into different regions associated with different targets. As we hypothesize in the previous chapter, nodes are placed pointed horizontally, and targets are modeled as cone-shape objects with uniform heights. Therefore, if essential portion of a target is unconcluded to a node as described in the occlusion model, the target will be projected to be a ridge-shape region in the image of the node, and in return a ridge region in the image can be associated with a target in the scene. A silhouette image may contain a sequence of such ridges. Ridges may overlap with each other, which corresponds to partial occlusion between targets. To elaborate more specifically, along the contour of the silhouette a sequence of local maximum pixels x_i and local minimum pixels z_i can be detected. The region between x_i and its neighboring minimum pixels or pixels touching the ground can be defined as a ridge that indicates existence of a target (figure 3.1).

However, we ignore those ridges with narrow width. These ridges are either projected from minor structures of the targets or generated due to inconsistence between real background scene and the background model. Another kind of source for these minor

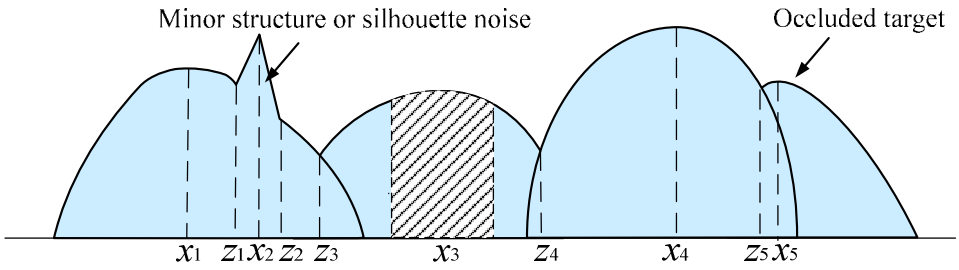


Figure 3.1 A silhouette image that contains a sequence of ridges at $x_1, x_2 ... x_5$. Ridges with enough width can be associated with targets, such as the ridge at x_3 that is bounded by z_3 and z_4. Features can be extracted form central areas (the hatched area) of ridges. Ridges with narrow width are excluded from consideration, such as the ridge at x_2 (background inconsistence) and the ridge at x_5 (occluded target).

ridges is those heavily occluded targets that only leak limited shape or texture information to the node. These ridges are not expected to contain accurate target information.

Step 3. Based on pixels in the ridges, some space-invariant features can be extracted. However if a ridge is overlapping with neighboring ridges, it may also contain pixels leaked from the targets corresponding to the neighboring ridges. Therefore features should be extracted based on the pixels in the central part of the ridge regions (figure. 3.1). The feature can be color. For instance, we find three major colors inside the ridge. The feature can also be texture. For instance, we can store several texture patterns in each node so that the node can label the ridge with the tag of the most similar texture pattern.

Some features, though vary with different perspectives, can also be used. Combined with positions or camera parameters of the nodes, they still indicate certain space-invariant characteristics of the targets. For instance we can use height of the ridge as feature *f*. For a specific spot in the scene, rf / κ indicates height of the target at that spot, where κ denotes focal length of the visual sensor, and r denotes the distance from the node to the spot.

We should point out that features discussed in this thesis are to be dispatched to other nodes for integration. Due to the concern for the energy consumption by wireless transmission, the representations of these features should be compact. Those features that need bulky data to describe, though more informative, are not appropriate to use in this case.

Step 4. In the end, feature of each ridge is packed into a structure, and the structure is to be sent to related nodes for integration. The structure is defined as $v\{i, f, (x_0, y_0), \vec{r}\}$, where i is the global identification of the feature structure, which is composed of the global identification of the node and the local identification of the feature structure in the node, f is the feature value, (x_0, y_0) is the position of the node, and $\vec{r}$ represents the orientation of the target supposed to be associated with this feature towards the node in global coordinate system. Since visual sensors of all the nodes are assumed to be on the same vertical level, we only need a 2D vector on the scene ground, and therefore $\vec{r}$ is defined as

$$\vec{r} = R^{-1}\begin{bmatrix} x \\ \kappa \end{bmatrix} \tag{3-1}$$

where R denotes the rotation matrix in the extrinsic camera model for visual sensors, and x is the central location of the ridge corresponding to this feature on horizontal dimension of the image. In this thesis, we also assume each visual sensor is calibrated beforehand, and therefore (x_0, y_0) and R are known to each node. Parameter $\vec{r}$ is also called as back-projecting direction and will be frequently cited in later chapters.

4 Centralized Detection

Within centralized fashion, all the nodes send their feature structures to a central node for the computation of detecting targets. As for the detection scheme embedded in the central node, the basic idea is related to the research in [Yan03] and can be explained as follows. Imagine each feature structure is projected from its source node back to the ground of the scene and covers a cone-shape area (figure 4.1). An intersection of these cone areas can be speculated as being occupied by a target. Although the speculation turns questionable if there are occluding targets existing between the intersection and nodes, it gets more confirmed with number of intersecting cone areas increased.

Let us explain the idea more specifically. We use a grid structure to mimic the ground of the scene. Spots in the grid are small enough so that we can assume each spot can only be occupied by at most one target. For each feature structure *v*, we define a 2D cone area on the ground that extends from the source node along the back-projecting direction $\vec{r}$ in *v* (figure 4.1). The cone area contains all possible locations of the right target associated with the feature structure. The cone has radius d and angular width $\Delta\theta$. d should be equal to the radius of the visual sensor's FOV, and $\Delta\theta$ should be equal to the angular error in calculating $\vec{r}$, which combines errors in estimating T and determining x. $\Delta\theta$ is assumed to be small enough so that the widest part of the cone area, i.e. $d\Delta\theta$, is much smaller than the horizontal scale of the target. We drop the structure into spots inside the cone. After dropping all feature structures, we look into each spot and search for structures with similar features. Let $\Psi(x, y)$ denote the largest subset of structures in spot (x, y) in which any two structures have mutually similar features, where (x, y) is the index of the spot in the grid. We use $|\Psi(x, y)|$, number of structures in $\Psi(x, y)$, to measure the indication of spot (x, y) being occupied by a target. A spot that has large $|\Psi|$ can be speculated as being occupied by a target, and this speculation also means that all the feature structures in Ψ should be associated with the target at the spot.

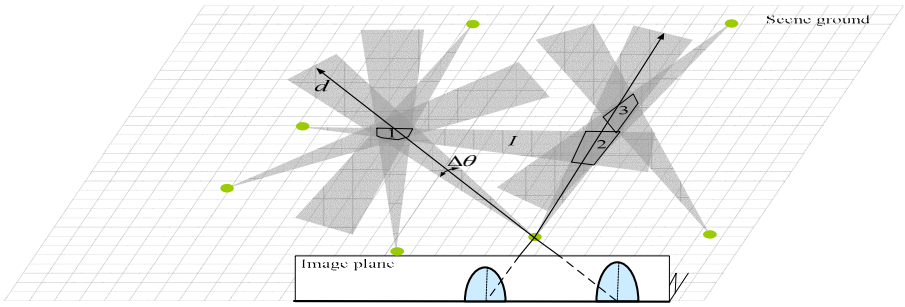


Figure 4.1 Illustration of detecting targets in the centralized algorithm. The ground of the scene is implemented by a grid structure. For each feature structure, a cone area extends from the source node along the back-projecting direction. The cones form intersections 1,2, and 3, which are speculated to be occupied by a target.

4.1 Iterative Prioritization Strategy (IPS)

There is some degree of uncertainty in detecting targets based on $|\Psi|$ values, which is related with feature ambiguities discussed in chapter 2. Due to feature ambiguities, a feature structure may appear in Ψ of multiple spots inside its corresponding cone area. Some of them may be occupied by the real target that is indeed associated with this structure, some may be occupied by targets that just bear similar features, and others, which may turn to be false targets, are intersected by a bunch of structures that accidentally contain similar features. A feature structure is only supposed to be associated with one target, and has to be used for the spots occupied by the right target. Failure to do so may not only suppress real targets from being detected, but lead to generate false targets.

In this centralized algorithm, we create a so-called Iterative Prioritization Strategy (IPS) to "reason out" targets based on distribution of feature structures in the grid. This strategy

gives priority to detect those spots showing strong indications of being occupied, and at same time prefers to use the structures for those detections. More specifically, IPS iteratively searches for spots having maximal $|\Psi|$ values. Once those spots are decided as being occupied, all the structures in their Ψ are associated with the targets on those spots, and then removed from the grid so that they will not be used for other spots. The strategy is written as follows

---

Iterative Prioritization Strategy (IPS)

Repeat:

Search for maximal $|\Psi|$ across the grid, i.e. $\hat{n}$.

IF $\hat{n} < H_{IPS}$, return.

ELSE, decide that spots with $|\Psi| = \hat{n}$ are occupied by a target. Among these occupied spots, declare that an individual target is detected on the spots that share the same Ψ, and all the feature structures in the Ψ are associated with the target. Moreover, remove each associated feature structure v from spots (x', y') inside the cone area of v, that is, if $v \in \Psi(x', y')$, $\Psi(x', y') = \Psi(x', y') - \{v\}$.

Notice that if $v \in \Psi(x', y')$, removing v from spot (x', y') may disqualify the rest of Ψ from being the largest subset. Strictly speaking, the spot has to reshuffle the structures to update Ψ, which however requires large amount of computation. Therefore, in IPS, we simply update Ψ by excluding v from Ψ. Also notice that more than one targets detected in the same iteration might happen to be associated with the same structure. When it happens, we accept all the associations. We consider this situation a rare event and ignore it in the following theoretic analysis.

Let us look at IPS's performance on detection accuracy. In IPS, once a target is detected on some spots, each associated feature structure will be withdrawn from the grid, which may change Ψ of nearby spots that also have the structure in their Ψ and therefore

influence detections afterwards. The influence can be negative, if the withdrawn structure is not indeed associated with the detected target, or the detected target is false. As a result, the target indeed associated with this structure loses the support of this structure and may be suppressed from being detected in the end. The influence can also be positive, if the withdrawn structure is indeed associated with the detected target. Removing the structure from other spots abates the possibility that the structure is used for qualifying some false targets. In the instance shown in figure 4.1, we suppose feature values in all structures are similar to each other. In the beginning, spots of intersection A collect four structures in Ψ, and spots in intersection B and C both collect three structures. The first target is detected on intersection A, which leads to structure I withdrawn from intersection B. Therefore, if H_{IPS} is set as 3, the last target will be detected on intersection C, instead of intersection B. The key element in this process is the structure I. If structure I is indeed associated with the target on intersection A, the decisions about intersection B and C would be reasonable. Otherwise, if there is a target on intersection B which is also what structure I should be associated with, IPS reaches a wrong decision. As a matter of fact, IPS can tip the influence to the positive side in a statistical sense. Next section gives detailed analysis.

4.2 In-Depth Analysis of IPS

It is easy to relate the iterative prioritization strategy (IPS) to a straightforward "one-shot" threshold strategy (OTS). OTS only focuses on those regions on the grid in which spots have local maximal $|\Psi|$ value, such as intersections 1, 2, and 3 in figure 4.1. We call them Local Maximal regions (LMR). If $|\Psi|$ of a LMR is larger than a predefined threshold H_{OTS}, OTS declares that an individual target is detected on the LMR. Therefore the question is: Compared with OTS, will the complication of executing IPS be paid off by better detection results? In this section, we will reveal the IPS's advantageous performance on detection accuracy. We first discuss the statistics about feature structures in each spot and the influence detecting each target exerts on these statistics, which paves

the way for equalizing IPS to a sequence of binary Bayesian classifications. Then we prove that IPS is actually trading a small increase in missed error for a large decrease in false error, and over-performs OTS with lower total detection errors.

To simplify the analysis of IPS, we make three assumptions about the distribution of feature structures on a grid.

(1) All the spots in a LMR have the same Ψ

(2) Ψ of an arbitrary spot outside all LMRs is a subset of Ψ of a nearby LMR.

(3) Any two LMRs are distant enough from each other on the grid, so that their Ψ share no more than one same feature structure.

Based on the first assumption, if a spot in a LMR is decided as being occupied, other spots in the same LMR must be decided as being occupied by the same target in the same iteration. Based on the second assumption, if a spot outside LMRs is decided as being occupied, spots in a nearby LMR must be decided as being occupied by the same target in the same iteration. Based on the third assumption, if spots in two different LMRs are decided as being occupied and $H_{IPS} > 1$, they must be decided as being occupied by two different targets. This means that, if we pick up an arbitrary spot from each LMR, the picked spots are one-to-one corresponding to all the targets that IPS is able to dig out. Thus, we can consider that IPS only works on these picked spots, and each spot, if decided as being occupied, must be occupied by an individual detected target. Trivially, this consideration can be also applied to OTS.

We define two categories C_1 and C_2 for these picked spots. We assume that, for each real target existing in the scene, there is an intersection on the grid formed by cone areas of all the feature structures that are indeed associated with the target, and the intersection contains one LMR. This means that, a spot picked from the LMR would contain all the feature structures associated with the target. We define that such a picked spot belongs to C_1, and the target detected on a spot in C_1 is a real target. Therefore real targets in the scene are one-to-one corresponding spots in C_1. For other spots, we consider that all the

feature structures they collect are random, and based on the third assumption above, these random structures must be indeed associated with different targets. We define that these spots belong to C_2, and the target detected on a spot in C_2 is a false target. Let $p(n \mid C_1)$ denote the likelihood that $|\Psi| = n$, given that the spot belongs to C_1, and $p(n \mid C_2)$ denote the likelihood that $|\Psi| = n$, given that the spot belongs to C_2. Let the miss error *ME* denote the probability that spots belonging to C_1 fail to be decided as being occupied, and the false error *FE* denote the probability that spots belonging to C_2 are decided as being occupied.

4.2.1 Initial Statistics

We start the discussion with the statistics of feature structures in spots of two different categories at the beginning of IPS, when all feature structures have been dropped and no targets has been detected. To ease following discussion, let Z denote the set Ψ at the beginning of IPS.

Compared with the occurrence that a feature is correctly attributed to the right target, occurrences of mistake 1 and mistake 2 can be assumed to be rare, that is, $1 - p_{f1} >> p_{f1}$, and $1 - p_{f1} >> p_{f2}$. Therefore, as for a spot of C_1, we ignore those structures brought into the spot by mistake 2. At the beginning, the likelihood $p(n \mid C_1)$ is equal to

$$p_{[0]}(n \mid C_1) = \sum_{m=0}^{\infty} p_{real}(m,n), \quad p_{real}(m,n) = C_m^n (1 - p_{f1})^n {p_{f1}}^{m-n} p_s(m) \tag{4-1}$$

where $p_{real}(m,n)$ represents the probability that, at the beginning of IPS, a spot of C_1 has collected m feature structures and its Z contains n of them. As for spots of C_2, we assume the statistics about the structures they have collected at the beginning of IPS are not only mutually independent, but also independent from the statistics of the structures spots of C_1 have collected. The likelihood $p(n \mid C_2)$ at the beginning is

$$p_{[0]}(n \mid C_2) = \sum_{m=0}^{\infty} p_{f_3}(m,n) p_{s'}(m) \tag{4-2}$$

where $p_{s'}(m)$ represents the probability that a spot of C_2 has collected *m* structures at the beginning of IPS.

4.2.2 Change in Statistics after Detecting a Target

Once a target is detected, each associated structure will be withdrawn from the grid, which may influence Ψ of other related spots. As mentioned above, the influence can be negative if the feature structure is indeed associated with another target, because the spot occupied by this "another target" may thereby lose the support of this feature structure. Consider a spot of C_1 and a feature structure v in its Z. Any detected target in the cone area of v (more precisely, the spot that the target is detected on is in the cone area of v) , no matter whether real or false, may result in withdrawing v from the spot due to mistake 2, and the probability for such occurrence is ,

$$u_1 = p_{f2} \tag{4-3}$$

On the other side, the influence can be positive if the feature structure is indeed associated with the detected target, because spots of C_2 nearby may thereby lose the support of this feature structure. Consider a spot of C_2 and a feature structure v in its Z. As we know, v is indeed associated with a real target located nearby, which means if the real target is detected, the probability of removing v from this spot will be $1-p_{f1}$, instead of p_{f2}. As for an arbitrary target detected in the cone area of v, the probability of removing v from the spot is actually related with the number of previously detected targets in the cone area. Therefore for the *l*th target detected in the cone area of v, we express the probability as $u_2(l)$. To calculate $u_2(l)$, let us first consider the probability that this spot still keeps v in Ψ after α real targets, β false targets detected in the cone area of v, i.e.

$$\begin{aligned} U_2(\alpha,\beta) &= \left[\frac{\alpha}{\Delta N_t} p_{f1}\left(1-p_{f2}\right)^{\alpha-1} + \frac{\Delta N_t-\alpha}{\Delta N_t}\left(1-p_{f2}\right)^{\alpha}\right]\left(1-p_{f2}\right)^{\beta} \\ &= \left[1-\frac{\alpha}{\Delta N_t}\left(1-\frac{p_{f1}}{1-p_{f2}}\right)\right]\left(1-p_{f2}\right)^{\alpha+\beta} \end{aligned} \tag{4-4}$$

where ΔN_t represents the number of real targets located inside the cone area, and $\frac{\alpha}{\Delta N_t}$ represents the probability that the real target indeed associated with v is included in the previous detections. Since

$$U_2(\alpha,\beta)=\prod_{l=1}^{\alpha+\beta}\left[1-u_2(l)\right] \tag{4-5}$$

we have

$$u_2(l)=\begin{cases}1-\dfrac{U(\alpha,\beta)}{U(\alpha-1,\beta)}=\chi(\alpha)p_{f2} & \text{if the } l\text{th target is real}\\ 1-\dfrac{U(\alpha,\beta)}{U(\alpha,\beta-1)}=p_{f2} & \text{if the } l\text{th target is false}\end{cases} \tag{4-6}$$

where $\chi(\alpha)=1+\dfrac{1-p_{f2}}{p_{f2}\left(1+\dfrac{\Delta N_t(1-p_{f2})}{1-p_{f1}-p_{f2}}-\alpha\right)}>1$. Eq (4.6) shows that compared with a false detection, a real detection in the cone area has higher probability of withdrawing v, and the probability increases with the increase of α, that is, the number of real targets have been dig out of the cone area.

By comparing Eqs (4.3) and (4.5), we can conclude that, confronted with a false detection in the cone area of a structure in *Z*, spots of both categories have equal probability of losing the structure. However, confronted with a real detection, spots of C_2 are more likely to lose the structure, and the likelihood increases with more real targets to be detected in the cone area. This difference can be clearly shown from the change of the two likelihood functions by detecting a target. Assuming that detected targets are randomly located across the scene, statistically speaking, detecting a target influences λ_1 percent of spots of C_1 and λ_2 percent of spots of C_2. These influenced spots are referred to as spots with one structure in *Z* whose corresponding cone area covers the detected target, and λ_1 should be equal to λ_2. Notice that, based on the third assumption made

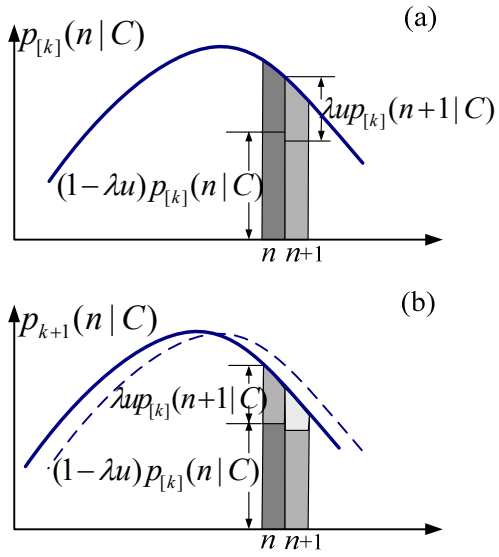


Figure 4.2 The influence that detecting a target puts on likelihood function $p(n\,|\,C)$. The likelihood curve is moved a step leftward after a target is detected, and the moving rate is proportional to λ and u. (a) The likelihood before the *k*th target is detected. (b) The likelihood after the *k*th target is detected. The dashed curve represents the likelihood before the *k*th target is detected.

above, each influenced spot has only one structure whose corresponding cone area covers the detected target.

Let $p_{[k+1]}(n\,|\,C_1)$ and $p_{[k]}(n\,|\,C_1)$ denote the likelihood function $p(n\,|\,C_1)$ before and after the *k*th target is detected, and $\hat{n}_{[k]}$ denote the maximal $|\Psi|$ value across the grid before the *k*th target is detected. We have

$$\begin{aligned} p_{[k+1]}(n\,|\,C_1) &= \lambda_1\left[(1-u_1)p_{[k]}(n\,|\,C_1)+u_1 p_{[k]}(n+1\,|\,C_1)\right]+(1-\lambda_1)p_{[k]}(n\,|\,C_1) \\ &= (1-\lambda_1 u_1)p_{[k]}(n\,|\,C_1)+\lambda_1 u_1 p_{[k]}(n+1\,|\,C_1) \qquad n<\hat{n}_{[k]}-1 \end{aligned} \tag{4-7}$$

Similarly for the spots of C_2,

$$p_{[k+1]}(n\,|\,C_2) = (1-\lambda_2 u_2)p_{[k]}(n\,|\,C_2)+\lambda_2 u_2 p_{[k]}(n+1\,|\,C_2) \qquad n<\hat{n}_{[k]}-1 \tag{4-8}$$

where u_2 represents the mean of $u_2(l)$ over all possible combinations of previously detected targets in a single cone area before the *k*th target is detected. Figure 4.2 depicts

the change of the likelihood functions after detecting a target. The detection transports $\lambda_1 u_1$ ($\lambda_2 u_2$) portion of the likelihood at n+1 a step leftward to n. Therefore if a real target is detected, $u_2 > u_1$ and $p(n \mid C_2)$ is transported leftward much faster than $p(n \mid C_1)$. Otherwise, $u_2 = u_1$ and the two likelihoods are transported leftward with same rate.

4.2.3 Statistics of the Entire Detection Process

Based on the analysis of the initial state and the change caused by detecting each target, we can track $p(n \mid C_1)$ and $p(n \mid C_2)$ through the whole process of executing IPS. With conditional probability $p(C \mid n) = p(n \mid C)P(C)$, the whole process of executing IPS can be explained as a sequence of binary Bayesian classification. In the first iteration, the basic shape of $p_{[0]}(C_1 \mid n)$ and $p_{[0]}(C_2 \mid n)$ is shown in figure 4.3.a. Although $P_{[0]}(C_2) > P_{[0]}(C_1)$ even for a cluttered environment, $p_{[0]}(C_1 \mid \hat{n}_{[0]}) > p_{[0]}(C_2 \mid \hat{n}_{[0]})$ is still expected to hold when $\hat{n}_{[0]}$ is large enough. Therefore the resulting decision that spots with $\mid \Psi \mid = \hat{n}_{[0]}$ are decided as being occupied is equal to the decision from a Bayesian classification that spots with $\mid \Psi \mid = \hat{n}_{[0]}$ are classified into C_1. Similarly, in an arbitrary iteration afterwards, suppose k targets have been detected before that iteration. If $\hat{n}_{[k]}$ satisfies $p_{[k]}(C_1 \mid \hat{n}_{[k]}) > p_{[k]}(C_2 \mid \hat{n}_{[k]})$, the resulting decision that spots with $\mid \Psi \mid = \hat{n}_{[k]}$ are decided as being occupied is equal to the Bayesian decision that spots with $\mid \Psi \mid = \hat{n}_{[k]}$ are classified into C_1.

Detections made in each iteration not only change the prior probabilities, but also change the likelihood functions as described above. Assuming most of the detected targets are real, the general changing trend is that $P(C_1)$ decreases and $P(C_2)$ increases, both likelihood functions $p(n \mid C_1)$ and $p(n \mid C_2)$ move leftward, and $p(C_1 \mid n)$ moves faster than $p(C_2 \mid n)$ (figure 4.3.b).

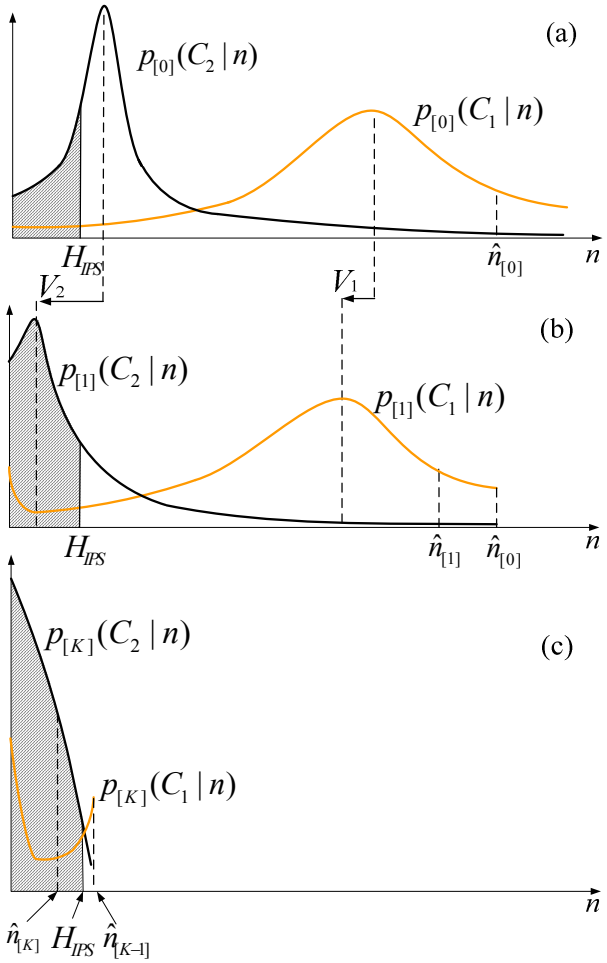


Figure 4.3 Change of conditional probability $p(C_1 \mid n)$ (grey) and $p(C_2 \mid n)$ (black) over iterative detections. (a) The conditional probability functions at the beginning of step 3. (b) The conditional probability after detecting a real target. Notice that $p(C_2 \mid n)$ is moved leftwards faster than $p(C_1 \mid n)$, which causes more likelihood enter the left side of H. (c) The termination point where $p(C_1 \mid \hat{n}) = p(C_1 \mid \hat{n})$.

In the final iteration where the process is terminated, suppose K targets have been detected, and $p_{[K]}(C_1 \mid \hat{n}_{[K]}) < p_{[K]}(C_2 \mid \hat{n}_{[K]})$. Decision that spots with $|\Psi| = \hat{n}_{[K]}$ cannot be decided as being occupied is equal to the Bayesian decision that these spots are classified into C_2. Therefore the threshold H_{IPS} can be explained as the n that satisfies $p_{[K]}(C_1 \mid n) = p_{[K]}(C_2 \mid n)$ (It is trivial to prove that this n must be smaller than $\hat{n}_{[K-1]}$). By assuming that $p_{[K]}(C_1 \mid n) < p_{[K]}(C_2 \mid n)$, for $n \le \hat{n}_{[K]} - 1$, excluding remaining spots from further consideration for detecting targets is also equal to the Bayesian decision that all the remaining spots are classified into C_2 (figure 7.3.c).

Therefore the miss error can be interpreted as the probability that spots belonging to C_1 are classified into C_2, and the false error FE can be interpreted as the probability that spots belonging to C_2 are classified into C_1.

4.2.4 Advantageous Performance of IPS on Detection Accuracy

The trick of IPS lies in the different change in $p(C_1 \mid n)$ and $p(C_2 \mid n)$ caused by detecting each target. After a target detected, there are some spots of both C_1 and C_2 categories that have their $|\Psi|$ values dropped below H_{IPS} and will be classified into C_2. This means detecting a target causes both miss error and false error to increase. The increase in miss error by detecting a target is

$$\Delta E_{miss} = \lambda_1 u_1 p_{[k]}(C_1 \mid H_{IPS}) \tag{4-9}$$

The decrease in false error by detecting a target is

$$\Delta E_{false} = \lambda_2 u_2 p_{[k]}(C_2 \mid H_{IPS}) \tag{4-10}$$

Suppose H_{IPS} is such a small value that $p(C_1 \mid H_{IPS}) < p(C_2 \mid H_{IPS})$ holds through all the detections until the terminating iteration. Since $u_2 \ge u_1$, $\Delta E_{false} > \Delta E_{miss}$ holds when each target is detected. This betrays that IPS is actually using a small increase in miss error to trade for a large decrease in false error.

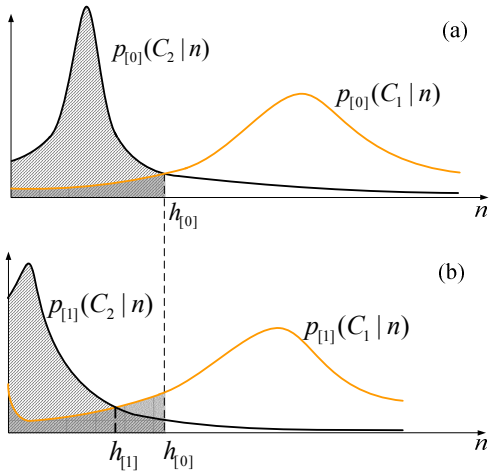


Figure 4.4 Proof for the superiority of IPS in terms of detection accuracy. The shaded region represents $B_{[k]}(h \mid C_1)$, and the hatched region represents $B_{[k]}(h \mid C_2)$. (a) Before a detection. (b) After the detection.

Now we use sum of miss error and false error as a metric E to evaluate the detection accuracy. We can reach the conclusion that IPS is superior to OTS in terms of E, i.e. $E_{IPS} < E_{OTS}$, if H_{IPS} is set at the n that satisfies $p_{[K]}(C_1 \mid n) = p_{[K]}(C_2 \mid n)$. The proof is written as follows.

***Proof*:** After k targets have been detected, let $h_{[k]}$ denote the critical value that satisfies $p_{[k]}(C_1 \mid h_{[k]}) = p_{[k]}(C_2 \mid h_{[k]})$, $B_{[k]}(h \mid C_1)$ denote the area under curve $p_{[k]}(C_1 \mid n)$, $n < h$, i.e. $\sum_{n=0}^{h-1} p_{[k]}(C_1 \mid n)$, and $B_{[k]}(h \mid C_2)$ denote the area under curve $p_{[k]}(C_2 \mid n)$, $n < h$, i.e. $\sum_{n=0}^{h-1} p_{[k]}(C_2 \mid n)$. Let $D_{[k]}(h)$ denote the difference between these two areas, i.e. $B_{[k]}(h \mid C_1) - B_{[k]}(h \mid C_2)$ (Figure 4.4).

To achieve minimal E by OTS, the "one-shot" threshold H_{OTS} should be set at $h_{[0]}$, and the resulting E is equal to

$$E_{OTS} = P(C_2) + \left[B_{[0]}(h_{[0]} \mid C_1) - B_{[0]}(h_{[0]} \mid C_2) \right] = P(C_2) + D_{[0]}(h_{[0]}) \tag{4-11}$$

In IPS, all the remaining spots in the last iteration are classified into C_2. Among them, spots indeed belonging to C_1 are classified into C_2, and the miss error is equal to $B_{[K]}(H_{IPS} \mid C_1)$. As for spots indeed belonging to C_2, although they are correctly classified, their complement within C_2 have been classified into C_1 falsely, so the false error is equal to $P(C_2) - B_{[K]}(H_{IPS} \mid C_2)$. Therefore, if $H_{IPS} = h_{[K]}$, the value of E can be expressed as

$$E_{IPS} = P(C_2) + \left[B_{[K]}(h_{[K]} \mid C_1) - B_{[K]}(h_{[K]} \mid C_2) \right] = P(C_2) + D_{[K]}(h_{[K]}) \tag{4-12}$$

In IPS, once the kth target is detected,

$$B_{[k+1]}(h_{[k]} \mid C_1) = B_{[k]}(h_{[k]} \mid C_1) + \lambda_1 u_1 p_{[k]}(h_{[k]} \mid C_1) \tag{4-13}$$

$$B_{[k+1]}(h_{[k]} \mid C_2) = B_{[k]}(h_{[k]} \mid C_2) + \lambda_2 u_2 p_{[k]}(h_{[k]} \mid C_2) \tag{4-14}$$

$$D_{[k+1]}(h_{[k]}) = D_{[k]}(h_{[k]}) + \left[\lambda_1 u_1 p_{[k]}(h_{[k]} \mid C_1) - \lambda_2 u_2 p_{[k]}(h_{[k]} \mid C_2) \right] \tag{4-15}$$

If the kth target is real, then $u_1 < u_2$, and since $p_{[k]}(C_1 \mid h_{[k]}) = p_{[k]}(C_2 \mid h_{[k]})$, we have $D_{[k+1]}(h_{[k]}) < D_{[k]}(h_{[k]})$. If the target is false, then $u_1 = u_2$ and $D_{[k+1]}(h_{[k]}) = D_{[k]}(h_{[k]})$. In addition, since $h_{[k+1]} < h_{[k]}$, $D_{[k+1]}(h_{[k+1]}) < D_{[k+1]}(h_{[k]})$. Therefore, no matter whether the kth detected target is real or false,

$$D_{[k+1]}(h_{[k+1]}) < D_{[k]}(h_{[k]}). \tag{4-16}$$

Based on Eq (4.11) (4.12) and (4.16), we can reach the conclusion that

$$E_{IPS} = P(C_2) + D_{[K]}(h_{[K]}) < P(C_2) + D_{[0]}(h_{[0]}) = E_{OTS} \qquad \square \tag{4-17}$$

The proof above also reveals the miss error and false error of IPS. The miss error is equal to

$$ME = B_{[K]}(H_{IPS} \mid C_1) \tag{4-18}$$

The false error is equal to

$$FE = P(C_2) - B_{[K]}(H_{IPS} \mid C_2) \tag{4-19}$$

4.3 Centralized Algorithm

In the centralized algorithm, all the nodes extract target features, pack features into feature structures, and send structures to the central node. When the central receives structures from all the nodes, it begins to execute the following procedure to detect the targets.

Centralized Target Detection Procedure (embedded in the central node)

Step 1. Receive and access all the feature structures arriving at the receiving port.

Step 2. Back-project each feature structure on the grid, and drop the structure into the spots inside of the back-projecting cone.

Step 3. Search for Ψ in each spot.

Step 4. Execute IPS.

---

5 Distributed Detection

The centralized detection requires the central node to take over the visual data of all the nodes and carry on the computation of target detection alone. This fashion not only leads to inefficiency in overall energy usage, but causes data to flood towards the central node, which may easily deplete the batteries of the central node as well as nodes nearby. Especially for a homogeneous VSN where each node possesses equal energy resource and processing capacity, it is wise to design a distributed solution so that all the nodes, instead of a single central node, can contribute to the fulfillment of the task, and therefore the energy resource of the network can be utilized fully and evenly.

In the distrusted fashion, each node is assigned with a subtask. Each node works on limited amount of local and temporal information to make decisions. Before designing a distributed solution for integrating visual information, two questions arising from the energy concern must be answered. Since for a large-scale VSN, wireless communication consumes most of the energy, this energy concern in this thesis also comes down to an expectation to minimize the amount of wireless communication between nodes. The first question is how to exchange visual data between nodes. Usually a subtask requires correlating data from multiple nodes, so a single node cannot finish the subtask alone and must request relevant data from others. Therefore an energy-efficient method to exchange visual information between nodes has to be figured out. The second question is how to fuse decisions made by different nodes independently. Because subtasks assigned to different nodes may be mutually interrelated, nodes should negotiate with one another to align their decisions. Can we implement such a negotiation procedure by paying a reasonable cost of wireless communication?

The following two sections introduce two strategies as the answers for our specific target detection case. The first strategy is an energy-efficient method of broadcasting visual data. Instead of sending request to nodes that have relevant data, the method initiates the data communication from source nodes where the visual information is generated. Each

packet of visual data is then relayed towards the place where is located the target corresponding to the visual data. All the relevant data about the target will finally reach there and thereafter are fused. The second strategy is about reinterpreting the IPS principles to be a measure for judging local decisions, and constructing this measure needs limited amount of data communications with neighboring nodes.

5.1 Broadcasting Visual Information through Back-Projecting Relays (BBR)

[Zha00] defines the problem of energy-efficient information exchange between nodes as "sensor selection" that a node should only request data from those nodes that have relevant data, and request in an order according to their relevance. The nodes having relevant data can be those within short geographical or routing distance, or overlapping FOV. However, in some cases such as the one we are facing, the relevance is more about the content of the data inside the nodes rather than those known characteristics. This fact prevents a node from having an explicit relevance estimate of other nodes before accessing them, and therefore makes the node unable to decide whether it is a good deal or not to access them by paying the prescribed energy cost. More specifically, in this detection case, nodes are considered as having mutual relevant data if they capture same targets. In a homogenous VSN where each node has equal sensing and computing abilities, the only factors left to define the relevance are distance and FOV. However, when targets are crowded and nodes are densely deployed, the bar of having short distance and overlapping FOV is so low that would qualify too many nodes as relevant. As we know, in a cluttered environment, occlusions widely exist, and therefore neighboring nodes may capture different targets even in their overlapping view. Then is it possible to refine the definition of relevance by incorporating occlusion information? The answer will not be encouraging because the occlusion information will only be revealed after triangulating the relevant data from different nodes.

We tackle this problem by choosing to broadcast the visual data but in a novel and energy-efficient way. We call this strategy as "Broadcasting Through Back-Projecting

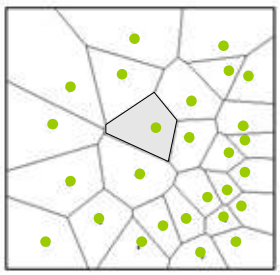


Figure 5.1 Voronoi partition

Relays" (BBR). In BBR, efforts to search for relevant nodes are discarded. Instead, each feature structure is relayed along the back-projecting direction towards the place where the target that the structure is indeed associated with is located. If each structure can be relayed along the back-projecting path without getting lost, we can expect that all the relevant structures will finally meet at the place of the target and form a high $|\Psi|$ value. This broadcasting strategy also enables parallelism between transmitting visual data and integrating visual data for detection. With this parallelism, knowledge from partial detection results can be used to guide the rest of data transmission, which will improve energy efficiency further.

5.1.1 BBR Algorithm

In BBR, the entire ground is partitioned into a Voronoi diagram, and each node is only responsible for detecting targets inside its local polygon (figure 5.1). Each node receives relevant feature structures from other nodes, and makes decisions about local targets on its own. More specifically, the scenario starts with all the nodes dispatching their feature structures into the network. Each structure is then relayed along the back-projecting direction, and joins the detection procedure running in the nodes on the way. A feature structure will be stopped by a node from being relayed further, if the node finds out that one of the following three conditions is satisfied.

(1) The structure triggers a local target to be detected.

(2) The structure is compatible with a previous detected target.

(3) The structure has been relayed beyond the FOV of the source node of the structure. The scenario ends when all the structures have been stopped.

To implement such a scenario, in each node, a patch of memory is allocated to manage the spots in the local polygon, and a procedure of relaying feature structures is initiated and kept running until the node is sure that the network has finished relaying all the feature structures. The procedure is presented as below.

BBR Procedure (embedded in each node)

A process is created to listen to the arrival of feature structures. Once a new structure v arrives, the node processes it through the following five steps.

Step 1. Receive v, and read v.

Step 2. If spots in local polygon are all beyond the cone area of v, return.

Step 3. If the feature value in v is similar to the feature value of a previous detected target right inside the cone area of v, return.

Step 4. Sweep through the area intersected by the cone area of v and local polygon, drop v into spots inside the area, and update Ψ of those spots.

Step 5. Search for spots with $|\Psi| = H_{BBR}$ in the intersecting area, and if they exist, decide that they are occupied by a target. Among those occupied spots, declare that an individual target is detected on spots that share the same Ψ, and all the structure features in the Ψ are associated with the target. Moreover, remove each associated feature structure from spots inside the intersecting area. Return

Step 6. Pass v to neighboring nodes along the back-projecting direction.

The port-listening process is terminated, if it has been so long since the arrival of the last structure that the node considers that the network must have finished relaying feature structures and there will be no any structures to receive. At same time, this procedure ends.

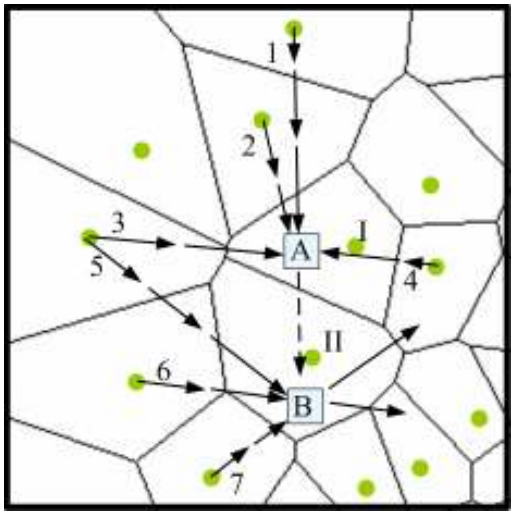


Figure 5.2 BBR in the first three rounds. All the nodes dispatch their feature structures at same time and are synchronized to relay them. Feature values in structures 1-7 are similar to each other. $H_{BBR} = 3$.

A simple case is illustrated in figure 5.2. Suppose all the nodes dispatch their feature structures at same time and are synchronized to relay them. We further assume that feature values in structures 1-7 are similar to each other, and $H_{BBR} = 3$. In the second round, node I receives structures 2, 3, and 4 and detects target A. The journeys of these structures are thereafter ended. In the third round, node II receives structure 5, adding to the structures received from 6 and 7 in the second round, triggers target B to be detected. At that same time structure 1 reaches node I and is found to be compatible with target A. Structure 1 is therefore stopped from being relayed to node II.

As a matter of fact, executing the procedure above does not require the nodes to synchronize their relaying the feature structures. Each node can receive, process, and pass the feature structures according to its own clock. However, timing of structures' arrival at the node has a profound impact on the detection accuracy of this procedure. In this thesis, we simply assume the network is working in a synchronized mode and expect more comprehensive discussions in the future.

The major characteristic of BBR is to broadcast the visual information through back-projecting relays, which achieves energy efficiency from three aspects. Firstly, a feature structure is only relayed by those nodes along the back-projecting direction, which, and only which, are possible to supervise the target the structure is indeed associated with. Secondly, the structure is relayed through the nodes by an increasing order according to their distances to the source node, which conforms to the fact that the ground truth target is more likely to be located nearby if occlusions between targets widely exist. Thirdly, the structure will be stopped from being relayed, if it is found to be compatible with certain local detection or has been relayed too far to be informative. BBR's performance on energy efficiency will be evaluated by specific metrics presented in chapter 6.

The third aspect mentioned above actually shows a parallelism between transmitting visual data and integrating visual data for detection. Because of this parallelism, knowledge from partial detection results can be immediately applied to regulate the transmission afterwards, which saves energy from those meaningless data transmission and computation that may occur otherwise. However this parallelism also causes more problems when mistakes 2 and 3 occur. As designed in BBR, if a passing-by structure triggers a false detection or takes a feature accidentally similar to a local detected target, the structure will be stopped from being sent downstream and therefore lose the chance to contribute to detecting the real target that the structure is indeed associated with. In figure 5.2, suppose the ground truth is that structure 1 is associated with target B. Structure 1 is stopped at node I, because it is found to be compatible with target A due to mistake 2. Thus, structure 1 is unable to reach node II to contribute to detecting target B. The detection accuracy of BBR may deteriorate if these two mistakes can not be well controlled. Therefore when we invoke BBR to detect targets, we should play conservative, that is, use large H_{BBR} and small threshold δ_{BBR} to define similar features. BBR should only be used to discover those targets that either have selective features or are slightly occluded and therefore captured by a number of nodes. As for those targets missed by BBR, we should rely on other methods to find them out.

5.2 Modified One-Shot Threshold Strategy (MOTS)

Let us consider another problem. Suppose, in BBR, each node turns off the computation of detecting targets and only relays passing-by feature structures and drops those structures into local spots. All the feature structures are relayed until they are stopped because of being beyond the FOV of their source nodes. With each local spot filled with all relevant feature structures, we encourage each node to make detecting decisions on its own. "One-shot" threshold strategy (OTS) discussed in section 4.2 is a way to make independent decisions. There each node searches for spots with local maximal $|\Psi|$ values, and if the local maximal $|\Psi|$ values are larger than a predefined threshold, these spots are decided as indicating existences of targets. OTS may be free from any extra expenses on communications, but creates conflicts between the decisions separately made by different nodes. For instance, a feature structure may be associated with multiple targets detected in different nodes, and moreover, false targets may be generated even based on feature structures associated with those targets that have been detected in other nodes.

To remove those conflicts, nodes are allowed to spend a small amount of communications on negotiating their independent decisions. Prioritizing those "local-maximal-$|\Psi|$" spots according to their $|\Psi|$ values, as used in IPS, can do this job. However adapting IPS into a distributed version faces several difficulties. Firstly, IPS requires each target to be detected on spots with globally maximal $|\Psi|$, which means, in the distributed version, in order to validate a target on local spots, each node must possess the Ψ data of all the "local- maximal-$|\Psi|$" spots across the grid to be sure that $|\Psi|$ of those local spots is globally maximal. Secondly, IPS requires to update Ψ of those influenced spots after each detecting decision is made, which means, in the distributed version, each detecting decision made in the network has to be reported to those nodes supervising the corresponding influenced spots. To tackle these difficulties, we need to pay extra communication costs, the amount of which will be by no means decent.

In this section, we present a so-called Modified One-Shot Threshold Strategy (MOTS) to achieve satisfactory independent decisions, which combines OTS to create "suspicious spots" that might indicate existence of a target, and a following "negotiation" procedure between related nodes to qualify or disqualify these local "suspicious spots".

5.2.1 MOTS Algorithm

At first, based on structures dropped in local spots, each node searches for the intersections in which spots have local maximal $|\Psi|$, and picks an arbitrary spot from each such intersection if the corresponding local maximal $|\Psi|$ is larger than a predefined threshold H_{MOTS}. By using a large threshold δ_{MOTS} to define similar features and a small H_{MOTS}, the node considers conservatively that each picked spot might indicate an individual target. These spots are called "suspicious spot", and their qualification in indicating a real target should be further investigated.

We still wish to check the qualifications of those suspicious spots in indicating a real target based on the feature structures in their Ψ. For a suspicious spot o_i, v_k denotes a feature structure in Ψ_i, and ST_k denotes the set of suspicious spots that all have v_k in their Ψ, though may be located in different polygons (figure 5.3). The real target indeed associated with v_k may - and only may - be indicated by one of spots in ST_k. Suppose $|\Psi|$ values of all the spots in ST_k are known to the node that supervises o_i. Similarly to IPS, the node is inclined to believe that those spots in ST_k with large $|\Psi|$ values are more likely to indicate a real target and v_k should be used for them, but it does not want to rule out the chance of o_i if $|\Psi_i|$ is small. The node defines a factor w_{ik} to weight v_k's contribution to qualifying o_i, that is,

$$w_{ik} = \frac{|\Psi_i|}{\sum_{o_j \in ST_k} |\Psi_j|} \tag{5-1}$$

w_{ik} is equal to 1 when o_i is the only suspicious spot in ST_k, that is, the only option that

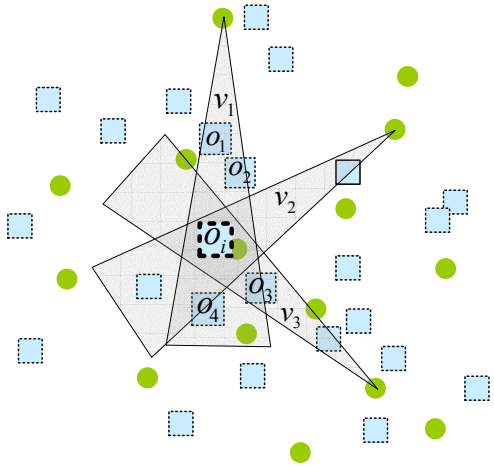


Figure 5.3 Definitions in MOTS. As for a suspicious spot o_i, $\Psi_i = \{v_1, v_2, v_3\}$. As for a feature structure v_1, $ST_i = \{o_1, o_2, o_3, o_i\}$.

v_k can be used for. w_{ik} decreases with the increase in the number of suspicious spots in ST_k that are competing for v_k, i.e. $|ST_k|$, as well as the increase in other spots' competences, i.e. $|\Psi_j|$, $j \neq i$. After obtaining the weight factors of all the feature structures in Ψ_i, the node can assess the qualification of o_i in indicating existence of a real target as

$$g_i = \sum_{v_k \in \Psi_i} w_{ik} \tag{5-2}$$

If g_i is larger than a threshold H_g, o_i is qualified in indicating a real target. Otherwise, o_i is considered as indicating a false target

Implementation of MOTS only requires limited amount of data communication. To assess the qualification of a local suspicious spot o_i, a node only needs to obtain the sum numbers, i.e. $W_k = \sum_{o_j \in ST_k} |\Psi_j|$, of all the feature structures v_k in Ψ_i by requesting information from related nodes. Therefore, we design a message for each feature structure v_k in the network and send the message through a round trip to bring back W_k. The message contains the identification of v_k and a one-byte long data field for carrying

the sum number W_k or intermediate sum number. The message starts from the source node of v_k, and follows the back-projecting route of v_k. At each relay node, it asks for local suspicious spots with v_k in their Ψ, and if those spots are provided, the data field in the message is updated by adding the $|\Psi|$ values of those spots. When the message reaches the node where v_k is stopped, the message realizes that it has finished integrating relevant $|\Psi|$ values and W_k is right in the data field. The message then turns around and brings W_k to those related nodes on the way back that need W_k to assess the qualifications of their local suspicious spots. See figure 5.4. The MOTS procedure is written as follows.

MOTS Procedure (embedded in each node)

Step1. Calculate Ψ of local spots with relaxed δ_{MOTS} and search for suspicious spots with relaxed H_{MOTS}.

Step 2. For each structure created by this node, dispatch a message into the network for a round trip along the back-projecting route. Create a process to listen to arrivals of messages from other nodes. Once a message is received, execute following instructions:

Look for local suspicious spots whose Ψ contain the corresponding feature structure *v* of the message.

IF there are those suspicious spots

IF the message is on the forward trip, update *W* value of the message by adding $|\Psi|$ values of those spots.

ELSE, save *W* value of the message for calculating *g* values of those spots.

IF this node is where *v* was stopped from being relayed, pass the message to the neighboring node on the reverse back-projecting route.

ELSE, pass the message to the neighboring node on the back-projecting route.

Step 3. Calculate *g* values for local suspicious spots and decide their qualifications in indicating a real target.

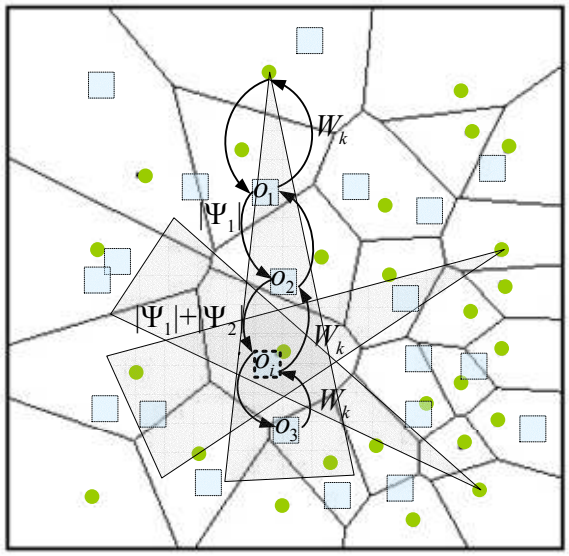


Figure 5.4 Implementation of MOTS. For each feature structure, a message is dispatched for a round-trip to act as the medium of negotiating conflicting decisions about the feature structure. In the forward trip, the message sums relevant $|\Psi|$ values, and in the backward trip, it brings the sum number W to related nodes that need this W to calculate g values of local suspicious spots.

Note that, in step 2, to prevent nodes from failing to receive the messages, each node should open the listening process at first, and then wait several milliseconds before dispatching messages, to make sure that other nodes have also finished step 1 and opened their listening processes.

5.2.2 MOTS and IPS

Let us relate this modified one-shot threshold strategy (MOTS) to the iterative prioritization strategy (IPS) used in the centralized algorithm. We can prove that, under a certain hypothesis, the g value of a suspicious spot in MOTS is related with the $|\Psi|$ value of the spot in the iteration where that $|\Psi|$ value becomes global maximum over all the suspicious spots in IPS.

Figure 5.5 Hypothesis about the meaning of w.

IPS gives the priority to qualify those suspicious spots with larger $|\Psi|$ values, and at the same time prefers to use structures for qualifying those spots. However, since each feature structure may be included in Ψ of multiple suspicious spots, decisions about different spots are correlated with one another. Therefore, as for a specific feature structure v, if we want to find out the exact suspicious spot that v is used for in IPS, we need to have global knowledge about Ψ of all the suspicious spots in the grid. Let us move back to the context of a distributed fashion. Is it possible to "reason out" the suspicious spot that v is used for in IPS, just based on limited scope of knowledge contained inside the cone area of v? There should be no definite answers but may be a probable one if we consider knowledge outside the cone area is randomized. With the understanding that, in IPS, v is more likely to be used for qualifying those suspicious spots with larger $|\Psi|$, we make the following hypothesis about the representation of the weight factor w (figure 5.5).

Hypothesis: Consider a suspicious spot o_i and a feature structure $v_k \in \Psi_i$. If the knowledge outside the cone area of v_k is assumed to be random, we hypothesize that w_{ik} represents the probability that v_k is used for qualifying o_i in IPS. In other words, in IPS, when it comes to the iteration where $|\Psi_i|$ becomes global maximum over all the suspicious spots, the probability that o_i still keeps v_k in Ψ_i is equal to w_{ik}.

Therefore we can express the probability that $|\Psi_i| = n$, when it comes to the iteration where $|\Psi_i|$ becomes global maximum over all the suspicious spots, as

$$\mu_i(n) = \sum_{\Psi_i(n)} \prod_{v_k \in \Psi_i(n)} w_{ik} \prod_{v_l \in \Psi_i - \Psi_i(n)} (1 - w_{il}) \tag{5-3}$$

where $\Psi_i(n)$ is an arbitrary combination of *n* structures in Ψ_i. It is easy to yield that

$$\sum_{n=0}^{|\Psi_i|} n\mu_i(n) = \sum_{k=1}^{|\Psi_i|} w_{ik} = g_i \tag{5-4}$$

Eq (5-4) shows that g_i in MOTS can be interpreted as the mean of $|\Psi_i|$ in the iteration where $|\Psi_i|$ becomes global maximum over all the suspicious spots.

Let us set H_g equivalent to H_{IPS}, and look into the situation when MOTS makes different decisions from IPS. Consider a suspicious spot o_i with $g_i \geq H_g$. Therefore o_i is considered by MOTS as indicating a real target. While in IPS, if $|\Psi_i| \geq H_{IPS}$ in the iteration where $|\Psi_i|$ becomes globally maximal, o_i is also considered by IPS as indicating a real target. Otherwise, IPS reaches a different decision that o_i indicates a false target (figure 5.6.a), and the probability of such an occurrence is

$$P_{1i} = \sum_{n=0}^{H_g} \mu_i(n) \tag{5-5}$$

Consider a suspicious spot o_i with $g_i < H_g$. Therefore, o_i is considered by MOTS as indicating a false target. While in IPS, if $|\Psi_i| < H_{IPS}$ in the iteration where $|\Psi_i|$ becomes globally maximal, o_i will also be decided by IPS as indicating a false target. Otherwise, IPS reaches a different decision that o_i indicates a real target (figure 5.6.b), and the probability of such an occurrence is

$$P_{2i} = \sum_{n=0}^{H_g} \mu_i(n) \tag{5-6}$$

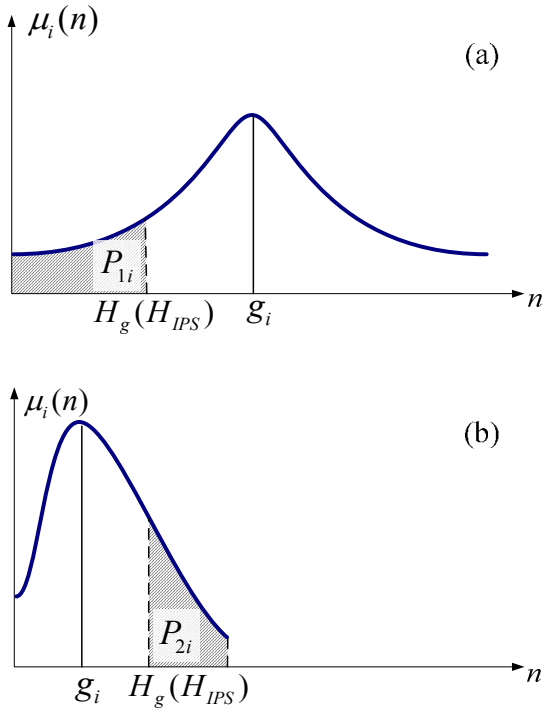


Figure 5.6 When IPS and MOTS make different decisions. (a) $\mu_i(n)$ function of a suspicious spot o_i when $g_i \geq H_g$, and MOTS decides o_i as indicating a real target. The hatch region represents the probability that IPS decides o_i as indicating a false target. (b) $\mu_i(n)$ function when $g_i < H_g$, and MOTS decides o_i as indicating a false target. The hatch region represents the probability that IPS decides o_i as indicating a real target.

By incorporating statistics of w, we can generalize the two probabilities above for a specific suspicious spot to the probabilities for a arbitrary suspicious spot. We denote the two generalized probability as ξ_1, ξ_2. Thus, the miss error and false error of MOTS can be expressed as

$$ME_{MOTS} = ME_{IPS}(1-\xi_1) + (1-ME_{IPS})\xi_1 \tag{5-7}$$

$$FE_{MOTS} = FE_{IPS}(1-\xi_2) + (1-FE_{IPS})\xi_2 \tag{5-8}$$

and the difference between MOTS and IPS in detection accuracy is

$$ME_{MOTS} - ME_{IPS} = \xi_1(1-2ME_{IPS}) \tag{5-9}$$

$$FE_{MOTS} - FE_{IPS} = \xi_2(1-2FE_{IPS}) \tag{5-10}$$

5.3 Distributed Algorithm

We design a distributed algorithm based on the two strategies presented above. The algorithm is divided into two stages. In the first stage, BBR is used to direct feature structures to related nodes and at same time dig out those targets that either have selective features or are slightly occluded. In the second stage, MOTS is used to find out the rest of targets.

As a matter of fact, we did wish that those structures that have been associated with targets detected in the first stage could be excluded from being considered in the second stage. However, in BBR, detecting decisions are made by different nodes independently, and some local spots in a node may still keep a feature structure that actually has been associated with a target detected in another node. Therefore, it is very hard to discern those "associated" structures without paying extra communication costs. In this distributed algorithm, we only expect that MOTS can withstand the interferences of these "associated" structures by itself, which turns out to be the truth.

Suppose in the second stage, we also pick up a suspicious spot from each region that is decided as indicating a target in the first stage. (Obviously it is not necessary to qualify those suspicious spots once again in the second stage). Like other suspicious spots, those spots also have their Ψ reshuffled by the relaxed threshold δ_{MOTS}. Since most of the targets detected in the first stage are real, $|\Psi|$ of those spots are normally very large. This means that each structure in the Ψ of those spots will bring very small weight factors w to other suspicious spots that also compete for this structure, and therefore provide very limited contribution to qualifying other competing spots. Thus, MOTS not only successfully suppresses the interferences of those structures that have been associated with targets under strict criterion in the first stage, but also suppresses the interferences of those structures that should be associated with targets detected in the first stage from the perspective of MOTS itself.

To detect targets in a distributed fashion, each node should execute the following procedure in a synchronized mode.

Distributed Target Detection Procedure (embedded in each node)

Step 1. Execute the BBR Procedure**.**

Step 2. Execute the MOTS Procedure.

.

6 Performance on Energy Efficiency

In this chapter, we evaluate the algorithms' performances on utilizing energy resource from two aspects: overall energy cost and degree of decentralizing the overall cost. Compared with its centralized version, an effective distributed algorithm should achieve better performances on both aspects.

Overall energy cost (*OEC*) is defined as the sum of energy cost spent by nodes over the entire network.

$$OEC = \sum_{i=0}^{N-1} e_i \tag{6-1}$$

where e_i is the energy cost that node i spends on executing the algorithm, and N is the number of nodes in the network.. For a large-scale VSN, most energy is consumed by wireless communication, and we can simplify *OEC* as

$$OEC = \sum_{i=0}^{N-1} c^s b_i^s + c^r b_i^r \tag{6-2}$$

where b_i^s (b_i^r) is the number of bytes that node i sends (receives) when executing the algorithm, and c^s (c^r) is the coefficient indicating the amount of energy consumed by sending (receiving) 1 byte of data. c^r can be considered as identical among all the nodes, and, for a VSN where each node has same transmission range, c^s can be also considered as identical. *OEC* measures the overall energy cost demanded by the algorithm.

In the centralized algorithm, each node sends feature structures to a central node. Suppose nodes have uniform transmission range that covers all one-hop neighboring nodes and send data to the central node via a sequence of relay nodes. Therefore each feature structure is received and then sent once by per relay node, and the number of relay nodes is proportional to the distance r from the source node of the structure to the central node, which we approximately estimate to be $\sqrt{\rho_s} r$. Let b_f denote the number of

bytes of a feature structure, and the average number of structures a node may generate is $\overline{n}_t$. Then we have an estimate for *OEC* of the centralized algorithm as

$$OEC_{cen} = (c_s + c_r)\int_0^{2\pi}\int_0^R b_f \overline{n}_t \sqrt{\rho_s} r \rho_s r dr d\phi = \frac{2}{3}(c_s + c_r) b_f \overline{n}_t N_s \sqrt{\rho_s} R \qquad (6\text{-}3)$$

As to the distributed algorithm, in the first stage of executing BBR, with strict criterion of checking feature similarities and qualifying a target, most targets are real and most associations between structures and targets are correct. Therefore, we assume that those associated structures are relayed until reaching the nodes that supervise their corresponding targets. However, there are still some structures that either miss their targets and travel too far or get stopped before they can reach their targets. We consider these structures constitute *w* percent of the total, and further assume the worst situation in terms of energy cost for these structures that they all finally reach the end of their source nodes' FOV. Therefore we have an estimate for the overall energy cost in this stage as,

$$OEC_{dis\_I} = (c_s + c_r)\overline{n}_t b_f N_s \sqrt{\rho_s}\left[(1-w)\overline{r}_t + wd\right] \qquad (6\text{-}4)$$

where $\overline{r}_t$ represents the average distance from nodes to the targets they have captured. In the second stage, each feature has a message that experiences a round trip from the source node to the node located at the end of the source node's FOV. The energy cost in the second stage is

$$OEC_{dis\_II} = (c_s + c_r)\overline{n}_t b_w N_s \sqrt{\rho_s}\,(2d) \qquad (6\text{-}5)$$

where b_w denotes the number of bytes of the message used for calculating W. Therefore, *OEC* of the distributed algorithm can be estimated as

$$OEC_{dis} = (c_s + c_r)\overline{n}_t N_s \sqrt{\rho_s}\left[b_f (1-w)\overline{r}_t + b_f wd + 2b_m d\right] \qquad (6\text{-}6)$$

By comparing Eq (6-3) and (6-6), we can conclude that, in a wide scene with area scale *R* much larger than the visible range *d* of each node, the distributed algorithm demands less energy consumption. Moreover, since $\overline{r}_t < d$, OEC_{dis} decreases with more targets

detected in the first stage. Therefore, if there are less ambiguities in visual information between different targets, such as in the scene where targets all bear selective features or slightly occlude each other, this distributed algorithm will save more energy consumption.

Maximum energy cost (*MEC*) is the maximum energy cost spent by a node in the entire network, that is,

$$MEC = \underset{i}{Max}(e_i) \tag{6-7}$$

To finish a task, some nodes may take more responsibilities and thereby consume more energy than others, and this metric indicates the energy requirement for those most strained ones. However, for a homogeneous VSN where each node possesses equivalent resources and the role of each node is randomly decided, this metric unfortunately becomes the energy requirement for all the nodes. The ratio between *MEC* and *OEC* indicates the degree of how evenly the algorithm distributes the energy cost, which comes down to $1/N$ when each node takes on same energy cost. An effective distributed algorithm is expected to have a small ratio so that no nodes will be overburdened and wear out their batteries quite ahead of the others.

In the centralized algorithm, nodes close to the central node have to play the relay node for nodes far away from the central node, and therefore have more data to receive and send. Let us consider those nodes located in a most inner circle around the central node with radius τ. Each of those nodes is supposed to consume much more energy on data transmission because of relaying average $b_f \rho\pi R^2 / \rho\pi\tau^2 = b_f R^2 / \tau^2$ bytes to the central node (figure 6.1). Thus we have

$$MEC_{cen} = \frac{1}{\rho\pi\tau^2}(c_s + c_r)\int_0^{2\pi}\int_0^{\tau}\left(\frac{b_f \overline{n}_t R^2}{\tau^2}\right)\sqrt{\rho_s}\, r\rho dr d\phi = \frac{2(c_s + c_r) b_f \overline{n}_t \sqrt{\rho_s} R^2}{3\tau} \tag{6-8}$$

and $MEC_{cen} / OEC_{cen} = R / N_s\tau$. In a wide scene, nodes close to the central node are very likely to use up their energy shortly.

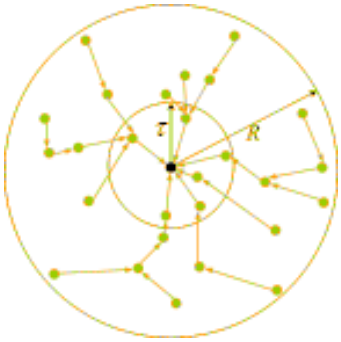


Figure 6.1 For the centralized algorithm, those most energy-strained nodes are located inside a most inner circle around the central node.

In the distributed algorithm, we assume nodes and targets are randomly located in the scene. Statistically speaking, each node is supposed to take on uniform energy cost. Therefore

$$MEC_{dis} = (c_s + c_r)\bar{n}_t\sqrt{\rho_s}\left[b_f(1-w)\bar{r}_t + b_f wd + 2b_m d\right] \tag{6-9}$$

and $MEC_{dis} / OEC_{dis} = 1/N_s$.

It is also interesting to look at the memory cost each node should spend on fulfilling the detection algorithm. The maximum size of memory allocated among all the nodes over the entire network (*MMC*) is what we should pay attention to, because, in a homogeneous VSN, *MMC* becomes the size requirement for the memory each node should be equipped with. The feature extraction method presented in chapter 3 only involves pixel-based computations, which can directly work on the image residing in the sensor buffer and does not need to allocate any extra memory. In fact, most of the memory cost of the two detection algorithms comes from "those spots", for each spot should have a pool to store feature structures. In the centralized algorithm, the central node should allocate all the spots in the grid mimicking the entire scene ground, and therefore $MMC_{cen} = \Theta(R^2)$. Meanwhile in the distributed version, the entire ground is partitioned into a Voronoi diagram, and each node is only responsible for detecting local targets, i.e. inside the local polygon area. Therefore, the memory cost is $MMC_{dis} = \mathrm{O}(R^2 / N_s)$ in a statistical sense.

7 Experimental Results

In this preliminary experiment, we place ten toy cars on the floor and use a camera to capture the scene from 21 different perspectives to mimic 21 scattered visual nodes. The camera is always pointed horizontally. The cars are crowded together, which creates lots of occlusions in the scene. Figure 7.1.a shows a top-view of the scene, and figure 7.1.b shows the locations of targets and nodes.

The rotation matrices R and the positions (x_0, y_0) of the "nodes" are precisely estimated by a calibration method proposed by [Bou05]. In the method, planar patterns are placed in the scene, which are projected to be different sizes and orientations in images taken by different nodes. These differences are taken advantage of to estimate the relative positions and orientations between nodes. The radial lens distortion is compensated in estimation (figure 7.2).

Each node captures one image (640 ×480) of the scene, and by removing the upper part (calibration pattern) and the bottom part (the ground), the image is processed through four steps. Figure 7.3 shows the result after each step.

(1) Subtract the background to obtain a silhouette image, which is easy in this experiment since we use pure white boards as the ground.

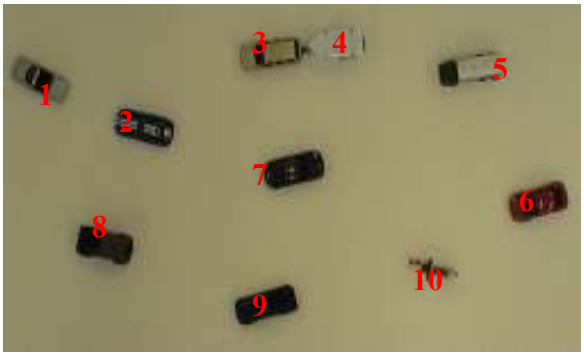
(2) Extract a silhouette contour from the silhouette image.

(3) Remove noises from the contour by median filtering with a 5-pixel wide window.

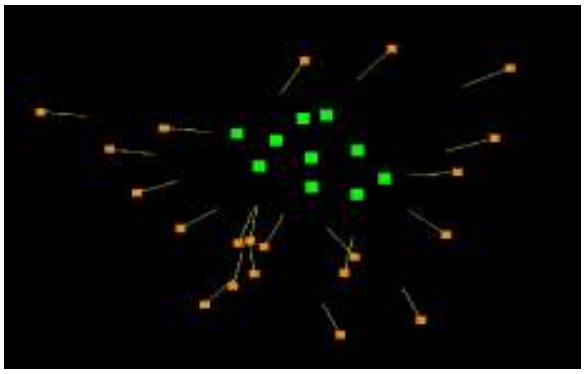
(4) Level off spikes of the contour whose width is less than 7 pixels.

(5) Define ridge regions based on the contour.

(6) Extract the features from ridge regions.



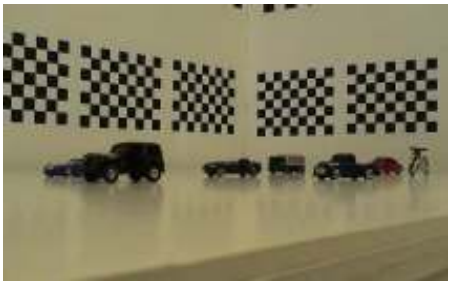
(a)



(b)

Figure 7.1 The experimental setting. (a) A top-view of the cluttered scene. (b) Visualization of the scene. Points in the center represent the targets (cars). Other points around represent the nodes, and the directions the lines pointing to represent orientations of the visual sensors on the nodes.

(a)



(b)

Figure 7.2 Node calibration [Bou05]. (a) and (b) are images captured from two different perspectives. Sizes and orientations of the grids vary with images captured from different perspectives, which provides information about relative positions and orientations between nodes.

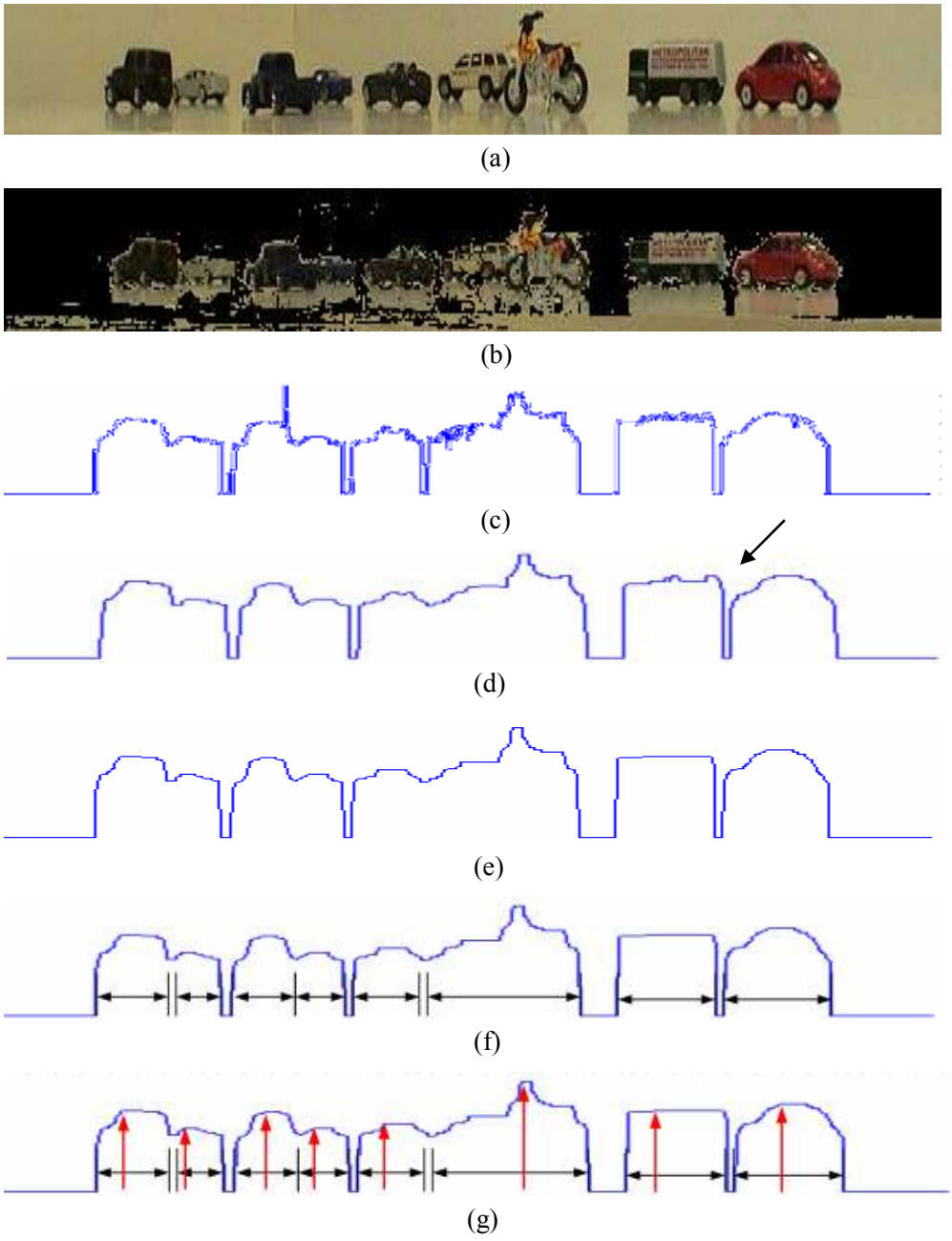


Figure 7.3 Feature extraction (a) Raw image (b) Silhouette image (c) Silhouette contour (d) Noises are removed. (e) Spikes are removed. (f) Ridge regions (g) Features from ridge regions.

Instead of using color information which is too selective in this experiment, we use the height of the ridge as the feature. As mentioned in chapter 3, this is not a space-invariant feature, and to use this feature, we need to make a little tweak to the algorithms. When a feature structure is dropped into a spot, the feature value f should be replaced by rf/κ, where r denotes the distance from the source node of the feature to the spot, and κ denotes the focal length of the visual sensor. Moreover, we should point out that all the discussion related with features in this thesis still holds, if some definitions are revised as follows:

Let f_A denote the height of target A, which is assumed to be an uniform random value in range $[F_1, F_2]$. Let $f_{i,A}$ denote the height of the ridge indeed associated with target A in the image of node i, and at spot (x, y) where stands the axis of target A, $r_{i,(x,y)} f_{i,A}/\kappa_i$ is assumed to be a random value between $[f_A - \Delta f/2, f_A + \Delta f/2]$, where $r_{i,(x,y)}$ denotes the distance from node i to spot (x, y), and κ_i denotes the focal length of the visual sensor mounted on node i. Consider a spot (x, y) where stands the axis of target A. At that spot, mistake 1 happens when

$$\left| r_{i,(x,y)} f_{i,A}/\kappa_i - f_A \right| > \delta/2 \tag{7-1}$$

and mistake 2 happens when

$$\left| r_{i,(x,y)} f_{i,B}/\kappa_i - f_A \right| < \delta/2 \tag{7-2}$$

Consider an arbitrary spot (x, y). At that spot, mistake 3 happens when

$$\left| r_{i,(x,y)} f_{i,A}/\kappa_i - r_{j,(x,y)} f_{j,B}/\kappa_j \right| \le \delta \tag{7-3}$$

The three corresponding probabilities p_{f1}, p_{f2} and p_{f3} are still defined in the same way as Eq (2-12) and (2-14), and (2-16). Regarding to the analysis of IPS, we assume that each spot in C_1 is the place where stands the axis of a real targets, which has collected all the feature structures associated with the target. Each spot in C_2 is the place where stands the axis of a false target, which has collected random feature structures

whose corresponding rf/κ values are randomly distributed in the range $[F_1, F_2]$ (we can exclude those rf/κ values outside that range from being considered for target detection) . Here comes the implementation of detection algorithms. The area where cars are crowded is $1.5m \times 1.5m$ square, and we use a 200×200 grid to mimic it. The height of the cars ranges from $F_1 = 0.014m$ to $F_2 = 0.038m$. The total 21 nodes generate 122 feature structures totally, and $\overline{n}_t = 5.8$. In the centralized algorithm, we set $\Delta\theta = 4^o$, $d = 1m$, $\delta = 0.005m$, and $H_{IPS} = 3$. Totally 11 "targets" are detected. The 12 iterations in IPS are shown in figure 7.4.

By comparing figure 7.1.b and 7.4.l, we can see that all the cars are detected, and the only confusion is that car 7 gets two detections. The reason is that car 7 has two humps as shown in figure 7.5. Although the two humps are close to each other, they are still separated by the nodes. We also notice that there are some spurious areas with high $|\Psi|$ values at the beginning of IPS, as shown in 7.4.a. However, we cannot find a detected target in those areas in figure 7.4.l, which means that the algorithm succeeds in preventing them from being detected as false targets. Therefore, there are no miss targets or false targets in the centralized detection results.

As for the distributed algorithm, in the first stage of executing the BBR strategy, we set $\Delta\theta = 4^o$, $d = 1m$, $\delta = 0.003m$, and $H_{BBR} = 5$. The BBR takes four rounds of relaying feature structures. In the second stage of executing MOTS, we set $\delta = 0.007m$ and $H_g = 3$. The results are shown in figure 7.6.

However, the performance of BBR does not live up to our expectations. In figure 7.6.d when relaying feature structures enters the third round, many problems occur. Firstly, one false target is dug out, which is pointed to by arrow 1. Secondly, car 8 gets double detections, which are pointed to by arrow 2. We looked into our program and found out that one of the detections is associated with 6 feature structures that actually should be

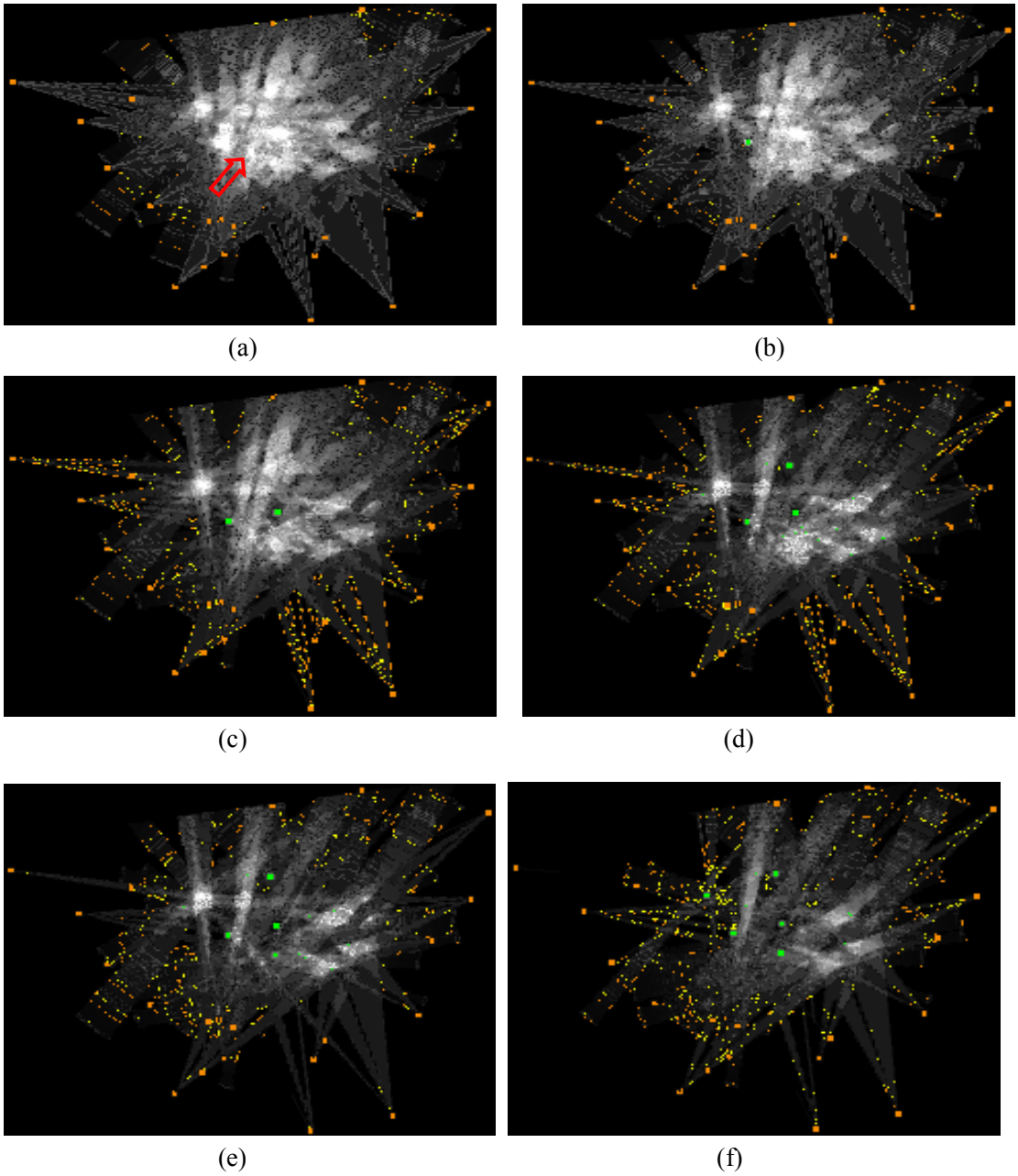
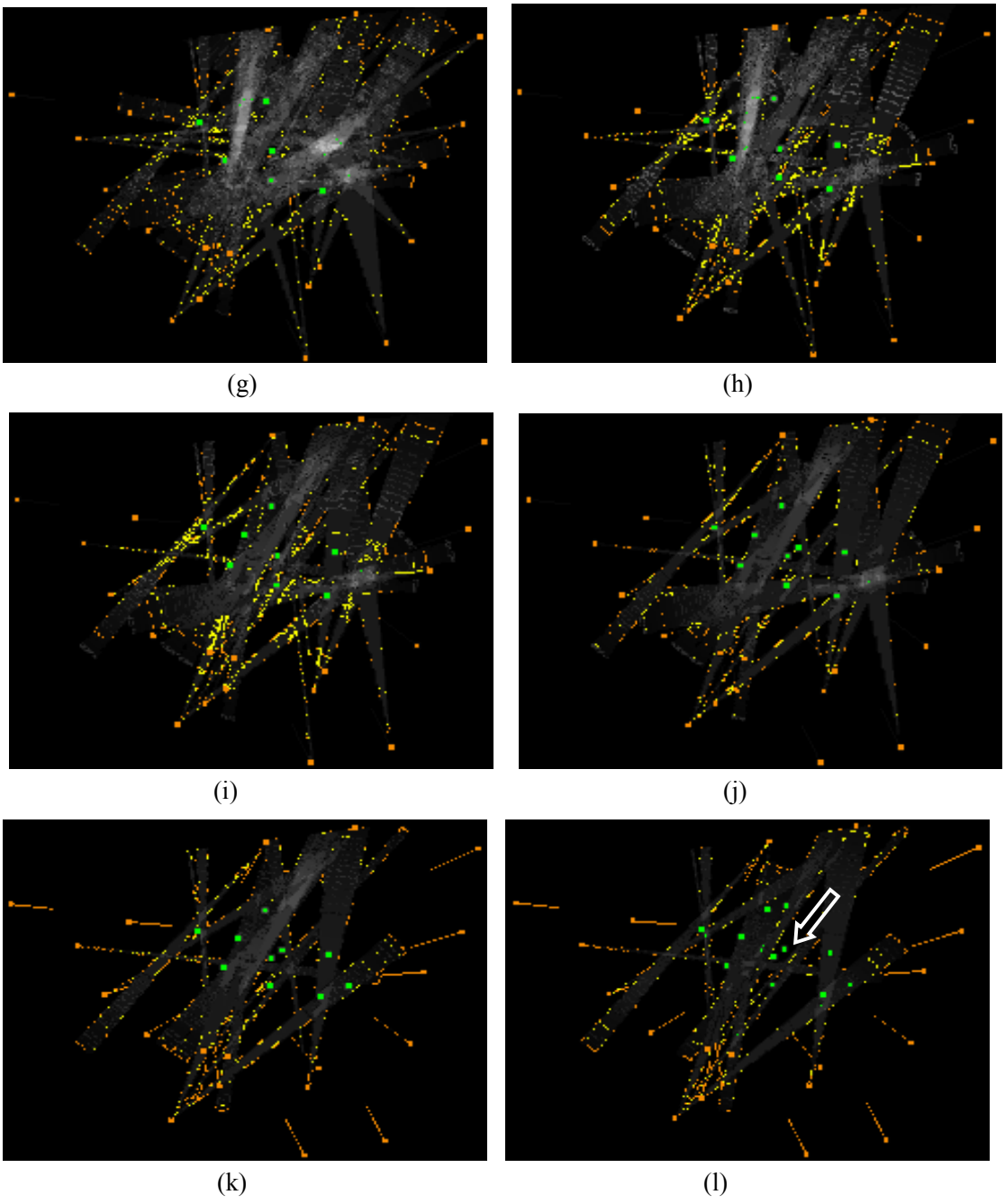


Figure 7.4 Centralized detection. Grey intensity in the image represents the $|\Psi|$ value of the spot. At the central part of the image, each point represents the center of the spots decided as indicating a car. (a) At the beginning of IPS. The arrow points to a spurious area. (b) After the first iteration in IPS.… (l) After the 11^{th} iteration in IPS. The arrow points to two crowded targets, which happens because the car at that place has two humps.



(g) (h)

(i) (j)

(k) (l)

Figure 7.4 Continued.

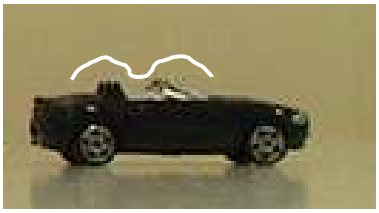


Figure 7.5 Car 7 has two humps

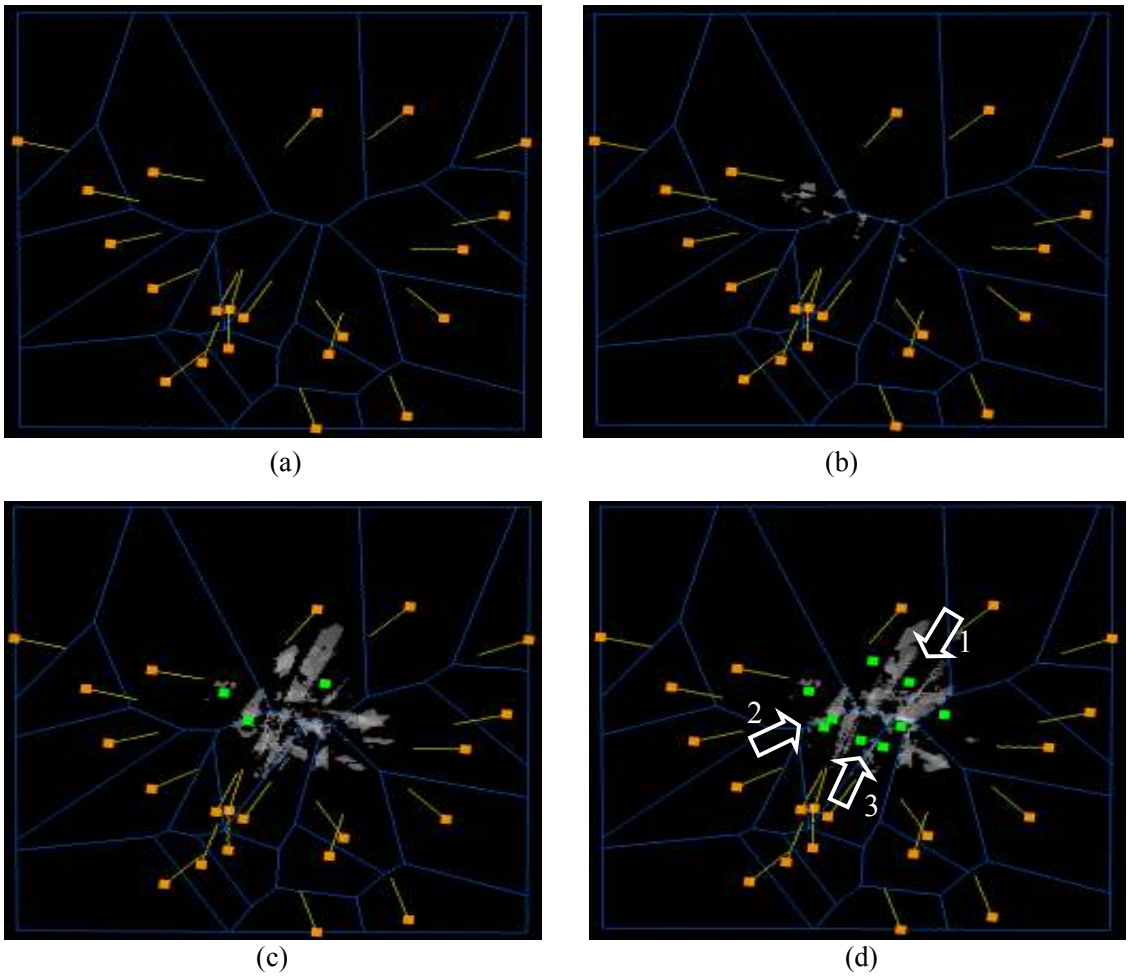
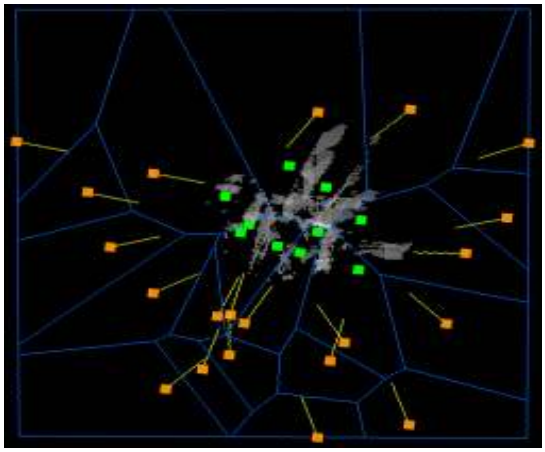
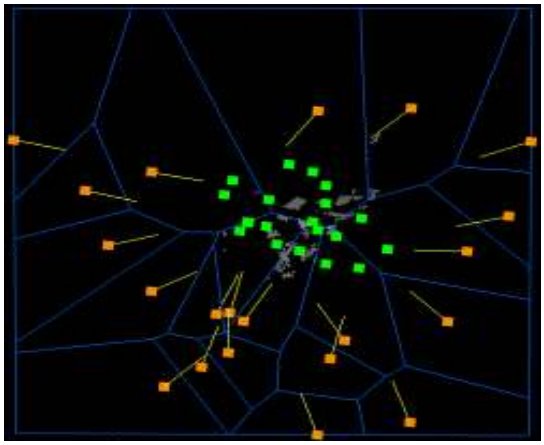


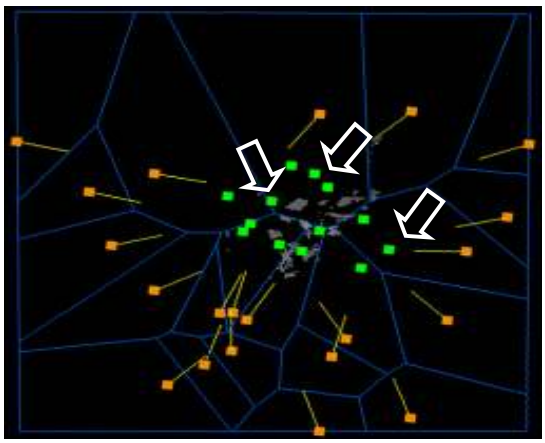
Figure 7.6 Distributed detection. Grey intensity in the image represents the $|\Psi|$ value of the spot. At the central part of the image, each point represents the center of the spots decided as indicating a target. (a) At the beginning of BBR. (b) After the first round of relaying feature structures in BBR. (c) After the second round in BBR. (d) After the third round in BBR. (e) After the fourth round in BBR, and all the feature structures have been stopped. (f) "Suspicious" spots are created in MOTS. (g) Three "suspicious" spots are qualified in indicating a real target.



(e)



(f)



(g)

Figure 7.6 Continued.

associated with other cars. Thirdly, car 9 gets double detections, which are pointed to by arrow 3. We looked into our program and found out that both detections are mainly based on feature structures that should be associated with car 9, but the area occupied by car 9 overlaps the supervising zones of two neighboring nodes and therefore the two nodes declare the same detections. The ultimate reason for these mistaken occurrences is that each node makes local decision on its own without considering if the decision is compatible or identical with decisions in other nodes. BBR strategy needs to be refined in future research. MOTS digs out car 2, 4, and 6, which are pointed to by arrows in figure 7.6.g. Fortunately, MOTS has not generated any false targets. Therefore the detection results of the distributed algorithms include two false targets and one duplicate target, which are all generated in the first stage by BBR. The detection results include no missed targets

Let us check the amount of consumed energy after executing the two algorithms. In the centralized algorithm, the central node is the one pointed to by arrow 1 in figure 7.7, and each feature structure experiences average 2.5 times of being received and sent before being accessed by the central node. A feature structure contains 11 bytes: one for the identification of the feature structure (a integer number), four for position of the source node (two float numbers), four for vector $\vec{r}$ (two float numbers), and two for feature value (one float number). Therefore $b_f = 11$, and we have

$$OEC_{cen} = (c_s + c_r) \times 11 \times 122 \times 2.5 = 3355(c_s + c_r) \quad (7\text{-}4)$$

The most energy-strained node during executing the centralized algorithm is the node pointed to by arrow 2 in figure 7.7, which has to relay totally 38 feature structures. The node also generates 6 structures by itself, so

$$MEC_{cen} = c_r \times 11 \times 38 + c_s \times 11 \times 44 = 418c_r + 484c_s . \quad (7\text{-}5)$$

Notice that the energy cost of the central node is also large, i.e. $c_r \times 11 \times 122 = 1342c_r$, which, however, is not considered as the maximum cost because normally $c_r << c_s$. As

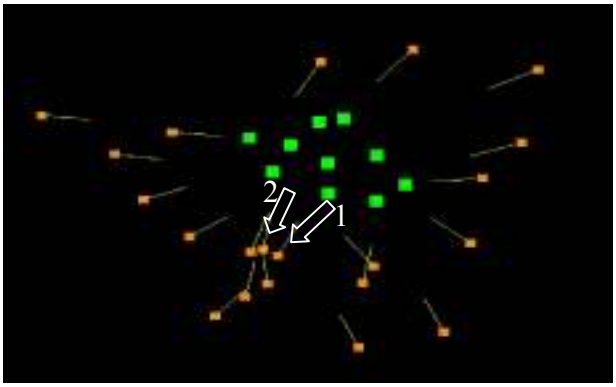


Figure 7.7 The central node (pointed to by arrow 1) and the most energy-strained node (pointed to by arrow 2) in the centralized algorithm.

for the memory cost, the central node should allocate a memory for 200×200 spots, and $MMC_{cen} = \Theta(4\times10^4)$

As for the distributed algorithm, each feature structure experiences average 2.7 times of being received and sent during the first stage of executing BBR.

$$OEC_{dis\_I} = (c_s + c_r)\times11\times2.7\times122 = 3623(c_s + c_r) \tag{7-6}$$

In the second stage of executing MOTS, the message for calculating W value contains two bytes, one for the W value and the intermediate sum value, and the other for the global identification of the corresponding feature structure. Notice that we do not need to insert the parameters about the back-projecting route into the message, because each node the message reaches can find the parameters in the corresponding feature structure that has ever passed by during the first stage. Therefore $b_f = 2$. The average number of relay nodes along the round-trip for a message is 3.8, and we have

$$OEC_{dis\_II} = (c_s + c_r)\times2\times2\times3.8\times122 = 1854(c_s + c_r) \tag{7-7}$$

Thus the overall energy consumed by the distributed algorithm is

$$OEC_{dis} = 5477(c_s + c_r) \tag{7-8}$$

The most energy-strained node is the one pointed to by arrow 1 in figure 7.8. The node has to relay totally 17 feature structures for once and 23 messages for twice, and also generates 6 structures by itself, so

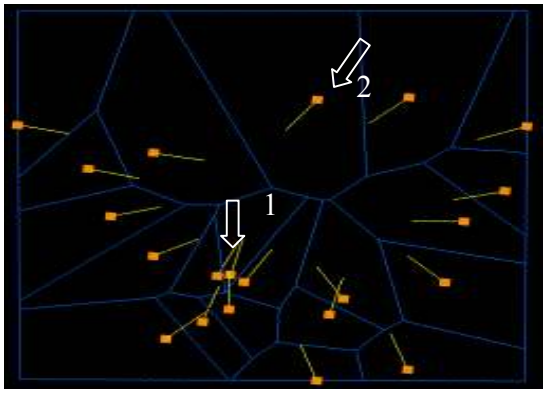


Figure 7.8 The most energy-strained node (pointed to by arrow 1) and the most memory-strained node (pointed to by arrow 2) in the distributed algorithm.

$$MEC_{dis} = c_r \times (11 \times 17 + 2 \times 2 \times 29) + c_s \times (11 \times 23 + 2 \times 2 \times 29) = 303c_r + 369c_s \qquad (7\text{-}9)$$

The most memory-strained node is the one pointed to by arrow 2 in figure 7.8. The node has to allocate a memory for 4867spots, and therefore $MMC_{dis} = \Theta(4.8 \times 10^3)$

The reason why OEC_{dis} is even larger than OEC_{cen} is that this experimental VSN is not a large scale VSN, and R=1.5 is comparable with d=1. However the distributed algorithm still over-performances the centralized algorithm on *MEC* and *MMC*, which proves that this distributed algorithm distributes the usage of the resources in the network to some extent.

8 Conclusions

This thesis initiates the research on designing algorithms for VSNs to fulfill vision tasks. The specific task in this thesis is to detect targets in a cluttered scene. The algorithm should achieve low miss rate and low false rate, and at same time be carried on in a distributed fashion where nodes collaborate with each other to fulfill the computation task and pay for resource consumption. The collaboration among nodes should be balanced between the benefits to detection accuracy brought by related information exchange and the cost in energy consumption caused by wireless data communication.

At first, the detection task is set in a scene where,
(1) Targets are massively and randomly located.
(2) Different targets may bear similar features.
(3) Targets are crowded and occlude one another.
(4) Nodes are massively and randomly deployed
Statistics of such a scene are analyzed with occlusions between targets especially considered.

A centralized detection algorithm is then proposed. The detection is based on triangulating features supposed to be associated with the same target but captured by different nodes. However, similarities among different targets as well as spurious triangulations caused by occlusions create ambiguities in features' "identities". IPS is such a strategy to remove these ambiguities by choosing to associate the feature with the target that shows the globally strongest indication of existence. Theoretic analysis proves that IPS actually uses a small increase in miss detection error to trade for a large decrease in false detection error.

The thesis is then working on a distributed detection version, where the entire scene is partitioned into a Voronoi diagram and each node is only responsible for detecting targets inside local polygon. The distributed algorithm consists of two stages. The first stage is

aimed to detect those targets that either have selective features or are slightly occluded. In that stage, each node runs two parallel processes: One is to relay feature information from source nodes to other related nodes, and two is to detect local targets based on received feature information. The second stage is to detect the remaining targets missed by the first stage. A so-called MOTS strategy is proposed, which consists of a straightforward procedure to select "suspicious spots" that might indicate existence of a target, and a following "negotiation" procedure between related nodes to qualify or disqualify those local "suspicious spots". It is proved that, under a certain hypothesis, detecting decisions made by MOTS are statistically related with detecting decisions made by IPS.

This thesis also opens many interesting topics for future research.
(1) Based on the statistics about a VSNs scene presented in this thesis, we can research on optimal clustering strategies, such as to what is the cluster size to achieve optimal detection accuracy under certain energy constraints, and what is the cluster size to achieve minimum energy cost on the condition of satisfying certain detection accuracy?

(2) Besides IPS where each feature is associated with the target that shows the strongest indication of existence, is there any other criterion to remove the ambiguities in visual information? How about associating the feature with the target bearing the most similar feature? Can this criterion be decentralized?

(3) As shown in this thesis, local decisions can be refined by incorporating limited scope of knowledge from other nodes. We can further define the relationship among the knowledge scope, the accuracy of local decisions, and the energy costs.

References

[Aky02] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," Computer Networks, 38, pages 393-422, 2002.

[Bou05] J.Y. Bouguet, "Camera Calibration Toolbox for Matlab," http://www.vision.caltech.edu/bouguetj/calib_doc/, 2005

[Dar98] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated Person Tracking Using Stereo, Color, and Pattern Detection," Proceedings IEEE Conf. on Computer Vision and Pattern Recognition, pages 601-608, 1998.

[Ghi02] S.Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, "Optimal Energy Aware Clustering in Sensor Networks," Sensors Magazine, MDPI, Issue'1, pages 258-269, January, 2002.

[Gup00] P. Gupta and P.R Kumar, "The Capacity of Wireless Networks," IEEE. Transactions on Information Theory, vol. 46, no. 2, pages 388- 404, Mar, 2000.

[Har98] I. Haritaoglu, D. Harwood, and LS Davis, "W4S: A Real-Time System for Detecting and Tracking People in 2-1/2 D," Proc. Europeen Conf. Computer Vision, June 1998.

[Kru00] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. "Multi-Camera Multi-Person Tracking for EasyLiving," 3rd IEEE International Workshop on Visual Surveillance, 2000.

[Mit02] A. Mittal and LS Davis, "M2Tracker: A Multi-View. Approach to Segmenting and Tracking People in a Cluttered Scene Using Region-Based Stereo," Proc. of the. 7th European Conf. on Computer Vision (ECCV), 2002

[Obr02] K. Obraczka, R. Manduchi, and J. Garcia-Luna-Aveces. "Managing the Information Flow in Visual Sensor Networks," Proceedings of the Fifth International Symposium on Wireless Personal Multimedia Communications, pages 27-30, 2002.

[Qih02] H. Qi, X. Wang, S. S. Iyengar, and K. Chakrabarty, "High Performance Sensor Integration in Distributed Sensor Networks Using Mobile Agents," International Journal of High Performance Computing Applications, 16(3), pages325-335, Fall, 2002.

[Rab04] M. Rabbat and R. Nowak, "Distributed Optimization in Sensor Networks," Proceedings of the Third International Symposium on Information Processing in Sensor Networks, pages 20-27, ACM Press, (New York), 2004.

[Sor05] S. Soro, W. R. Heinzelman, "Prolonging the Lifetime of Wireless Sensor Networks via Unequal Clustering," IEEE International Parallel and Distributed Processing Symposium, pages 236b, 2005.

[Xuy04] Y.Y Xu, H. Qi, "Distributed Computing Paradigms for Collaborative Signal and Information Processing in Sensor Networks," International Journal of Parallel and Distributed Computing, 64 (8) pages 945-959, August 2004.

[Yan03] D.B Yang, H.G-Banos, and L.J Guibas, "Counting People in Crowds With a Real-Time Network of Simple Image Sensors," Ninth IEEE International Conference on Computer Vision (ICCV '03), vol. 1, 2003.

[Yan04] D.B Yang, J. Shin, A.O Ercan and L.J Guibas, "Sensor Tasking for Occupancy Reasoning in a Network of Cameras," Proceedings of BASENETS'04, San Jose, CA, October 2004.

[Zha04] F. Zhao and L. Guibas. "Wireless Sensor Networks: An Information Processing Approach," Elsevier/Morgan-Kaufmann, 2004

Vita

Cheng Qian was born on Mar 14, 1978 in Suzhou, Jiangsu, P.R.China. He received the Bachelor of Science in Electrical Engineering in 2000 from Nanjing University of Aeronautics and Astronautics, P.R.China, and the Master of Science in Electrical Engineering in 2003 from Tsinghua University, P.R.China. He is currently a graduate student in the Advanced Imaging & Collaborative Information Processing Lab at the University of Tennessee, TN.