

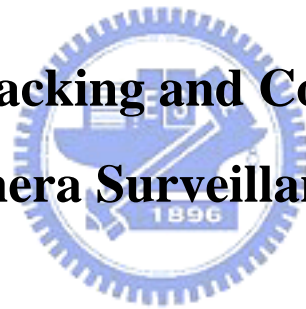
國立交通大學

電子工程學系 電子研究所碩士班

碩 士 論 文

多台攝影機監視系統下的物體追蹤與對應

**Moving Objects Tracking and Correspondence over
Multi-Camera Surveillance System**



研 究 生：王郁晴

指導教授：王聖智 博士

中 華 民 國 九 十 六 年 六 月

多台攝影機監視系統下的物體追蹤與對應

Moving Objects Tracking and Correspondence over Multi-Camera Surveillance System

研究生：王郁晴

Student：Yu-Ching Wang

指導教授：王聖智博士

Advisor：Dr. Sheng-Jyh Wang



A Thesis

Submitted to Department of Electronics Engineering & Institute of Electronics

College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of Master

in

Electronics Engineering

June 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

多台攝影機監視系統下的物體追蹤與對應

研究生：王郁晴

指導教授：王聖智 博士

國立交通大學

電子工程學系 電子研究所碩士班

摘要

在本文中，我們提出了一個基於三維粒子濾波器的多物體追蹤及對應技術，利用多個二維物體追蹤的結果，建立起物體在三維空間中的機率分佈，並透過二維三維資訊的交換，更新及預測這個分佈的變化。同時，這個三維的機率分佈，也可以幫助我們修正二維追蹤的結果，以及建立起各視野間物體的對應關係，並且能夠自動的修正錯誤的對應關係。有了物體的對應關係，就可以解決物體交錯所造成的追蹤錯誤，並且能夠更加正確的估測三維物體的機率分佈，使多物體的追蹤和對應更加準確而可靠。

Moving Objects Tracking and Correspondence over Multi-Camera Surveillance System

Student : Yu-Ching Wang Advisor : Dr. Sheng-Jyh Wang

Department of Electronics Engineering, Institute of Electronics

National Chiao Tung University

Abstract

In this thesis, we propose a 3-D particle filter based objects tracking and correspondence system. Through the 2-D tracking results, we can predict and update the probability distribution of moving objects in 3-D domain. With such probability, we can not only refine the 2-D tracking results but also construct the correspondence of objects in different camera views. In addition, our system is able to correct the correspondence automatically based on some 2-D and 3-D clues. With the object correspondence, the occlusion problem can be solved easily and the 3-D probability distribution of moving objects can also be estimated more precisely. These advantages make the results of objects tracking and correspondence more robust and reliable.

誌謝

我很慶幸當初決定來交大念研究所，能和指導教授 王聖智老師以及實驗室的成員們一起做研究。實驗室的氣氛融洽，每星期大家都會一起打球，有問題也會互相幫助，就像是一個大家庭，使我能很快融入新的學習環境。我很喜歡每星期的 group meeting，除了可以瞭解其他人的研究，一起討論，更可以磨練自己的表達能力。在此特別感謝王聖智老師，他總是站在學生的立場為我們著想，且對學生完全的信任。每每在研究遇到瓶頸的時候，老師都能適時的給予很多新的想法以及方向，透過不斷的討論和驗證，解決原本始終想不透的問題。讓我瞭解到要如何從無到有，解決一個困難的問題。這兩年來的日子，除了培養了研究的能力，更使我變得有自信且自動自發。



Contents

Chapter 1 Introduction	1
Chapter 2 Backgrounds	3
2.1 Motion Detection.....	3
2.1.1 Background Subtraction.....	3
2.1.2 Temporal Differencing	5
2.1.3 Optical Flow	6
2.1.4 Learned Classifier	6
2.2 Motion Tracking.....	8
2.2.1 Region-Based Tracking.....	8
2.2.2 Active-Contour-Based Tracking.....	9
2.2.3 Model-Based Tracking	10
2.2.4 Feature-Based Tracking.....	11
2.2.4.1 Mean-Shift.....	12
2.2.4.2 Particle Filter	13
2.3 Multi-Camera Correspondence	14
2.3.1 Region-Based Methods	14
2.3.2 Point-Based Methods	14
Chapter 3 Proposed Method.....	20
3.1 2-D Objects Detection and Tracking.....	21
3.1.1 Background Subtraction Detection	21
3.1.2 Mapping Objects	23
3.2 Transformation Between 2-D and 3-D	25
3.3 3-D Particle Filter.....	27
3.3.1 Particle Filter Algorithm.....	27
3.3.2 Proposed 3-D Particle Filter.....	31
3.3.2.1 First Generation of Particles.....	31
3.3.2.2 Correction of Particles.....	33
3.3.2.3 Shrink of Particles	35
3.3.2.4 Prediction of Particles	37
3.4 Information Exchange Between 2-D and 3-D.....	40
3.4.1 Construction of Correspondence.....	40

3.4.1.1 First Construction of Correspondence.....	40
3.4.1.2 Correspondence Update	43
3.4.2 Occlusion Handling.....	45
3.5 Overall System Structure	47
Chapter 4 Experimental Results.....	48
Chapter 5 Conclusion	56
Reference.....	57



List of Tables

Table 3-1 algorithm of particle filter[28]	29
Table 3-2 algorithm of resampling [28]	30

List of Figures

Figure 1-1 Multi-camera surveillance system	1
Figure 2-1 Horprasert T's method [2]	4
Figure 2-2 The illustration of the results before and after Gabor Filters [5].....	5
Figure 2-3 The example of detection result by applying Gabor Filters [5].....	5
Figure 2-4 Training data[6]	6
Figure 2-5 A example of training features[6]	7
Figure 2-6The original image[8]	8
Figure 2-7 The example of Wren's region-based tracking method	8
Figure 2-8 The example of McKenna's region-based tracking algorithm [9].....	9
Figure 2-9 The example of Paragios's active-contour-based algorithm[10].....	9
Figure 2-10 3-D generic model[11].....	10
Figure 2-11 Different models of human body [12]	10
Figure 2-12 An example of color histogram [20].....	11
Figure 2-13 The illustration of the convergence of mean-shift.....	12
Figure 2-14 the concept of particle filter.....	13
Figure 2-15 The comparison of mean-shift and particle filter[20].....	13
Figure 2-16 An example of indoor surveillance system[23]	15
Figure 2-17 An example of Utsumi's method [23]	15
Figure 2-18 The illustration of correspondence in 3-D domain by Utsumi's method[23].....	16
Figure 2-19 The illustration of epipole constraint [24]	17
Figure 2-20 Black, Jamesa's correspondence method based on the epipole constraint [24]	17
Figure 2-21 S. Khan's correspondence in 2-D method [25]	18
Figure 2-22 An example of J. Black's method [26]	19
Figure 3-1 a posterior process of background subtraction	21
Figure 3-2 The results before and after the morphological operations	22
Figure 3-3a illustration of objects detection applied in our system	22
Figure 3-4 the illustration of all kinds of tracking results	24
Figure 3-5 a illustration of camera setup[27]	25
Figure 3-6 a illustration of particle filter	27
Figure 3-7 a illustration of resampling	30
Figure 3-8 a illustration of tree.....	31
Figure 3-9an example of generating initial particles.....	32

Figure 3-10 3-D particles and the centers of gravity after shrink	36
Figure 3-11 2-D tracking results and the illustration of 3-D particles	38
Figure 3-12 Extra particles for new entering objects	39
Figure 3-13 extra particles for the objects still not establishing the correspondence relation.	39
Figure 3-14 a illustration of fuzzy adjustment	46
Figure 3-15 the flow chart of our system	47
Figure 4-1 the tracking and correspondence results in seq1 under occlusions	50
Figure 4-2 modification of size by a fuzzy adjustment (Seq1)	50
Figure 4-3 modification of size by a fuzzy adjustment (Seq2)	51
Figure 4-4 the demonstration of self-correction of our system.....	53
Figure 4-5 the establishment of the correspondence relation for detected new object	54
Figure 4-6 the reconstruction of correspondence relation when splitting	55



Chapter 1 Introduction

Surveillance systems have attracted more and more attention in recent years. With an intelligent surveillance system, we can automatically detect and track moving objects. Furthermore, we can recognize one person's identity and decide the occurrence of an abnormal event. Based on the number of cameras, surveillance systems can be classified into single-camera surveillance systems and multi-camera surveillance systems. A single-camera surveillance system cannot handle the occlusion problem whereas a multi-camera surveillance system can integrate the information from several cameras to solve this occlusion problem and make the tracking result more reliable. With calibrated cameras, we can use the internal and external parameters of cameras and 2-D tracking results to construct the correspondence relation and then to estimate the location of moving objects in the 3-D domain. Moreover, the heights [24] of the moving objects can also be estimated to give us extra information for objects tracking and correspondence. Even though we may also use 2-D tracking results to find the correspondence relation among uncalibrated cameras, our research focuses on a surveillance system equipped with multiple calibrated cameras. With calibrated cameras, the correspondence problem becomes much simpler. In *Figure 1-1(a)*, we show different views of multiple cameras; while in (b), we demonstrate the objects tracking and correspondence results of our system.



(a)The environment of our surveillance system

(b)Objects tracking and correspondence of our surveillance system

Figure 1-1 Multi-camera surveillance system

In this thesis, we propose a 3-D particle filter based objects tracking and correspondence system. The main idea is that the motion detection problem can be described as a probability distribution problem. Here, we assign higher probability values at those 3D positions where some moving objects are likely to be present. Since the probability distribution of moving objects in the 3-D domain may not be Gaussian distribution, we use a group of particles to approximate this distribution. Moreover, since this distribution combines all the information from different camera views, it can refine the 2-D tracking results and construct the correspondence relation by back-projecting all 3-D particles onto 2-D image planes. When the correspondence has been constructed, the occlusion problem in 2-D image planes can be solved easily by fusing other camera views' 2-D tracking results, which belong to the same 3-D moving object. In addition, we can classify all the 3-D particles into different moving objects in the 3-D domain. The major advantage is that we will neither just focus on the moving objects which have the most 3-D particles nor ignore the ones which have the fewest 3-D particles. By exchanging information between 2-D domain and 3-D domain, our system can automatically check if the correspondence relation is correct. The number of the 3-D particles varies dynamically according to the correspondence results. For example, if there is an object in a 2-D image plane which cannot build the correspondence with other camera' views, we will give more particles along the path where this object may appear in the 3-D domain. Also, if some 2-D image planes detect new objects, we will use the same method to generate 3-D particles. By adjusting the number of 3-D particles adaptively, we can dynamically update the 3-D probability distribution of moving objects and make our system more flexible and robust.

Chapter 2 Backgrounds

In this chapter, we will introduce some existing methods for moving objects detection, tracking and correspondence. In 2.1 and 2.2, we will integrate and discuss some moving objects detection and tracking methods, which are popular in recent years. In 2.3, we will discuss some methods which focus on how to fuse the information from different view of cameras and how to establish the correspondence of moving objects in each camera view.

2.1 Motion Detection

In this section, some popular methods about how to detect moving objects in a complex background are to be discussed. The detection process is very important since better detection algorithms support better tracking and correspondence results. Since all of these methods have advantages and disadvantages, we usually choose a suitable algorithm according to the environment of the surveillance system.

2.1.1 Background Subtraction

Background subtraction is a method to detect moving regions in an image by taking the difference between the current image and the reference background in a pixel-by-pixel fashion. This technique is developed especially for surveillance systems with static cameras since the reference background model is needed. After subtraction, each pixel of residuals is then classified as foreground if its intensity is larger than a given threshold; otherwise the pixel is classified as background. Although this method is quite simple and costs low computation, it is very sensitive to the image noise and the variation of illumination. Stauffer [1] uses a set of Gaussian distributions to represent a pixel's value. In other words, the intensity at each pixel in the reference background follows the distribution of a Gaussian mixture model (GMM). The difference between the current image and the reference background is obtained by measuring the distance of each point to its corresponded set of Gaussian distributions. Moreover, the parameters of Gaussian distributions will be adjusted as time goes on. As a result, the background model becomes more precisely and flexibly. However, the detection results are still degenerated by shadows and highlights. Horprasert T [2] utilizes the concept of color constancy of human eyes to separate the information of color and intensity. It can not only segment the moving regions in an image but also tell if some pixels of the moving regions belong to shadows or highlights. As *Figure 2-1* shows, the upper-left picture is the reference background and the upper-right picture is the current image. In the lower-left figure, the blue region represents the detected moving object, and the red region indicates the shadows and highlights. The lower-right picture shows the final result of detection.

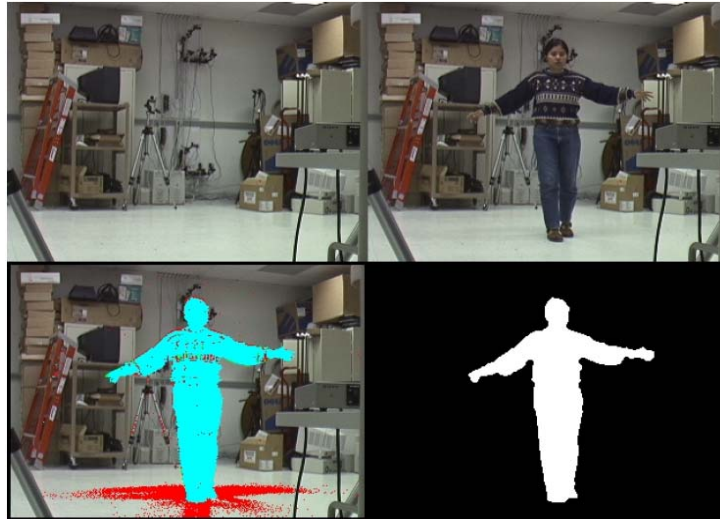


Figure 2-1 Horprasert's method [2]

Upper-left is the reference background, upper-right is the current image. In the lower-left picture, the blue region is the detected moving object and red region is shadows and highlights. The lower-right picture is the final detection result.

Duque [3] takes the temporal differencing results into consideration. The detected moving regions obtained by both temporal differencing results and background subtraction results can reduce the influence of noise.



2.1.2 Temporal Differencing

Temporal differencing is a method to detect moving regions by taking the difference between successive images. We can use a given threshold to distinguish moving parts from static parts. Although this approach needs low computation and is not sensitive to the environment, it can only detect moving objects. That is, if an object stops to move, then it will be misclassified as a part of background. In addition, sometimes the detection results are not reliable because the detected regions are usually not complete and have many holes. Fu-Yuan Hu [4] does some morphological operations and GMM detection on the results of temporal differencing to eliminate these holes. S Dubuisson [5] uses a particular set of Gabor filters to filter the residuals. The contribution of such filters is to emphasize the moving regions. As *Figure 2-2* shows, the left figure illustrates the original result, and the right figure shows the results after using Gabor filters. After the filtering, we put some particles randomly on the moving regions, with the number of particles being proportional to the magnitude of motion. As *Figure 2-3* shows, in the left figure the white particles represent the locations with larger motion. By applying some classification methods, we can cluster the small residuals to accomplish the detection task.

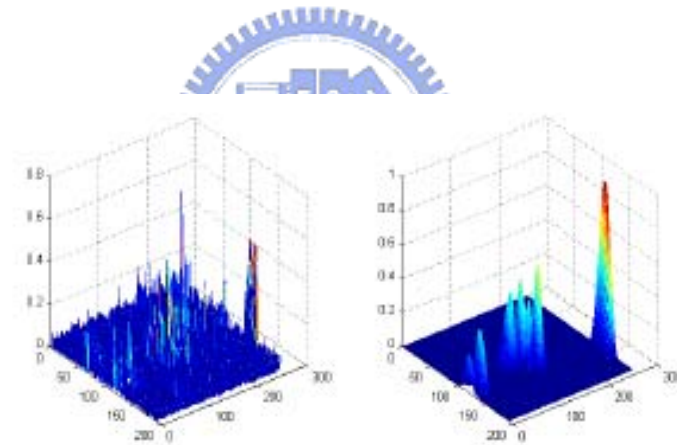


Figure 2-2 The illustration of the results before and after Gabor Filters [5]
Left one is the original results of time differencing, and right one is the results after applying Gabor filters.



Figure 2-3 The example of detection result by applying Gabor Filters [5]
Left figure is the detection results after applying Gabor filters, white particles represent the points used to clustering, and the right figure shows the final detection results.

2.1.3 Optical Flow

Optical flow is a method to represent each pixel by a function of time and location. If along the time a moving point only changes its location while keeps its intensity values unchanged, then we can estimate its motion vector. The pro of this method is that we can extract moving objects from the background even over a dynamic camera system. In addition, by classifying the motion vectors of each pixel we can further segment and distinguish different moving objects. The con of this method is that it costs a lot of computation and is not suitable for real-time surveillance systems.

2.1.4 Learned Classifier

Learned Classifier method needs to know the objects which are going to detect. That means we need to train a set of classifiers at first based on the features of objects. *Figure 2-4* illustrates some examples of training data for detection. However, there are lots of features that can be selected to train the classifiers and *Figure 2-5* illustrates one of the selections. The pro is that if the amount of the training data is large enough then the performance of this method can be very good. In addition, the application of this method is not limited to static camera systems. The con is that this method can only detect specific kinds of objects and it needs a large amount of training data to “learn” these objects. V Nair [6] proposes a method which doesn’t require much training data at first. Instead, his system can learn the features of objects online by combining the motion information of the moving objects. Hence, his system can learn the objects’ features adaptively.



Figure 2-4 Training data[6]

The left figure shows some training examples of “people”.

The right figure shows some training examples of “background”.

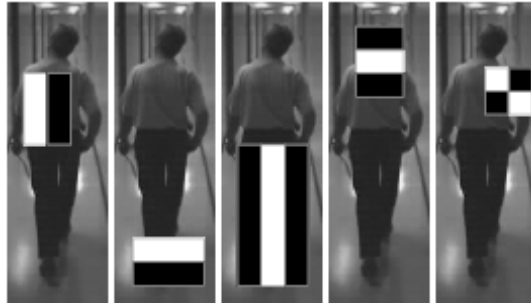


Figure 2-5 A example of training features[6]

Many detection methods contain more than one working principle. For example, Dongxiang Zhou [7] proposes a method which integrated all the aforementioned concepts. Since each method has its pros and cons, we may combine different methods to achieve better detection results.



2.2 Motion Tracking

In the previous section, we have introduced several detection methods. In this section, motion tracking methods are to be discussed. Motion tracking is to track moving objects from one frame to another in an image sequence and it is the key process of a surveillance system. Some major types of motion tracking techniques will be introduced in the following sections.

2.2.1 Region-Based Tracking

The basic component of a region-based tracking algorithm is a segmented region, called blob. The main idea is to track such a region in successive frames of an image sequence. Generally speaking, these kinds of tracking algorithms utilize the background subtraction to detect objects in the first place. Wren [8] describes the detected objects as small blobs. As *Figure 2-7* shows, by comparing the value with the reference GMM background model, each pixel of the foreground can be further classified into the corresponding blob. Then we can successfully track the object by tracking all of its small blobs.



Figure 2-6 The original image [8]

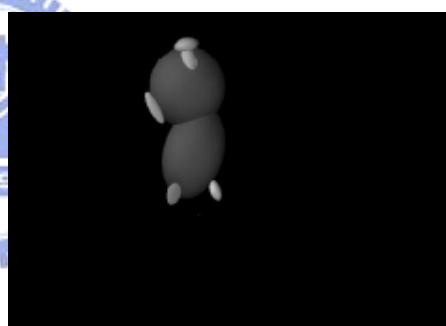
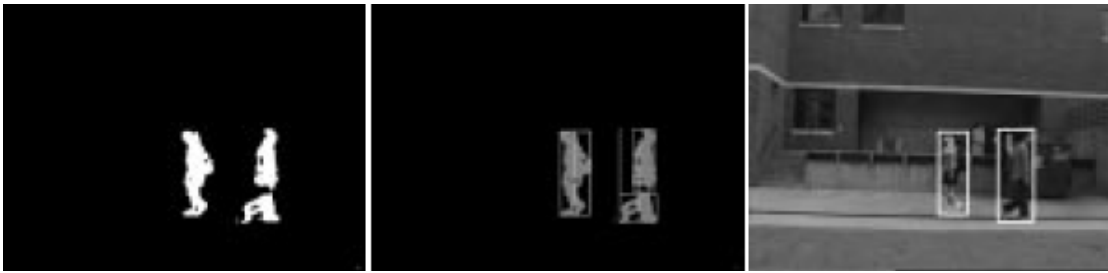


Figure 2-7 The example of Wren's region-based tracking method.

The 2-D illustration of small blobs belongs to the same object [8]

McKenna [9] also uses the background subtraction technique to get the regions which possibly belong to the parts of moving objects and uses bounding boxes to select these regions. As shown in *Figure 2-8*, the left picture shows the original results of background subtraction, and the middle picture shows the results after classification. Each bounding box contains a part of moving object. The right picture illustrates the final results of McKenna's algorithm. Based on the structures and color models of human bodies, we are able to classify the regions into different persons. Moreover, if the region involves only one person then it is denoted as "people"; otherwise, if it involves more than one human body, it will be denoted as "group". The task of objects tracking can be accomplished by analyzing and tracking these regions. By comparing the color information with their color models, moving objects can be tracked successfully without

losing their identities even if these objects are partially occluded.



*Figure 2-8 The example of McKenna's region-based tracking algorithm [9]
The left figure is the result of background subtraction, and the middle figure shows the segmentation of residuals which are marked by rectangles. There are three regions and two of them belong to the same person. The right one is the final tracking result.*

2.2.2 Active-Contour-Based Tracking

Active-contour tracking method is to track objects by representing their outlines as bounding contours and updating these contours over time. Paragios [10] tracks the moving objects in an image using a geodesic active contour objective function and a level-set formulation scheme. In *Figure 2-9*, there is a convergent process of active-contour-based algorithm. Active-contour-based algorithms can represent the moving objects more precisely than region-based algorithms. However, this method requires perfect detection results and needs manual selection at first. Consequently, it is difficult to start this kind of systems automatically.

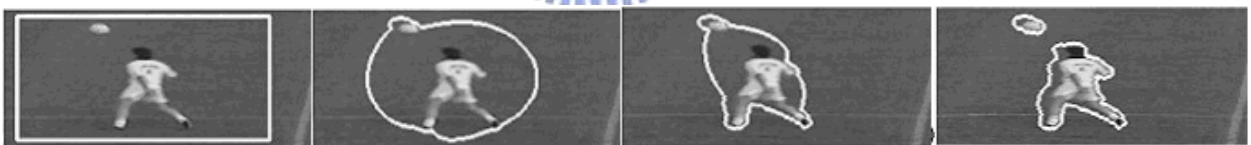


Figure 2-9 The example of Paragios's active-contour-based algorithm[10]

2.2.3 Model-Based Tracking

Model-based tracking algorithms need to build the prior structure models of the objects at first and then build the model for each candidate of target. Moving objects are tracked by matching the prior model with the model of the target candidates. The definition of rigid objects and non-rigid objects are quite different. Rigid objects such as cars can be tracked by comparing their 3D generic models with the reference models [11], as shown in *Figure 2-10*. On the other hand, the models of non-rigid objects, such as human bodies, are more difficult to build due to the possible deformations. Some models of human body [12] are shown in *Figure 2-11*. Model-based tracking methods usually need to maintain a lot of parameters to build the models. Hence, it is difficult to implement these model-based tracking methods due to their heavy computational loads. However, if we use fewer parameters to build the models, then these models may not be able to describe the objects precisely.

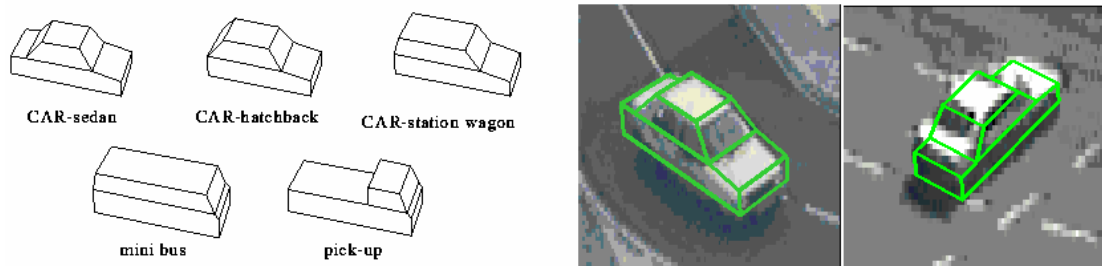


Figure 2-10 3-D generic model[11]

The left figure shows some 3-D generic models of cars. The middle and the right figures show the models of the detected cars.

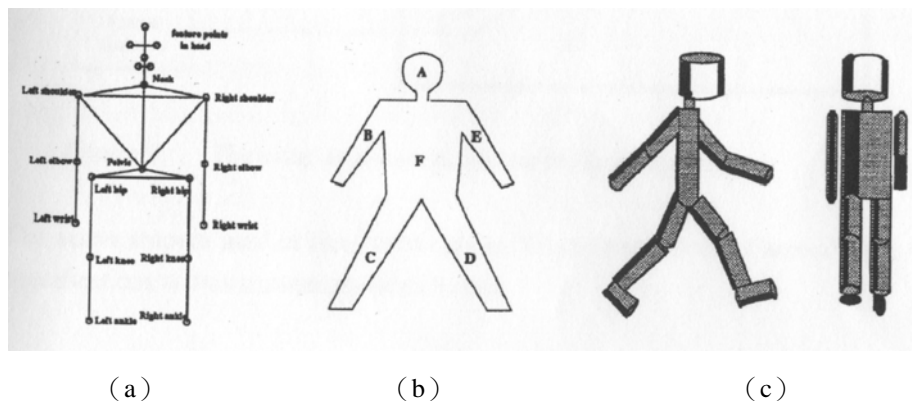


Figure 2-11 Different models of human body [12]

(a) stick-figure model of human body (Chen and Lee, 1995) (b) 2-D contour model of human body (Leung and Yang, 1994) (c) volumetric model of human body (Hogg, 1994)

2.2.4 Feature-Based Tracking

The main idea of feature-based tracking algorithm is to extract the features of moving objects for tracking. By comparing the features in an image sequence, the task of tracking is successfully done. Based on different features, the algorithms can be further classified as global feature-based methods, such as center of gravity[13], color[14], area[15]; local feature-based methods, such as line[16], vertex[17]; and dependence-graph-based methods, such as the change of structures among features.

Nowadays many real-time surveillance systems adopt feature-based algorithms for objects tracking. Mean-shift method [19] and particle filter [21] are two popular algorithms. Generally speaking, color is the common feature for these two algorithms. The tracked objects are modeled in terms of their color histograms. Color histogram is to calculate the color distribution of a selected area. As shown in *Figure 2-12*, (a) is the color histogram of the target calculated from the region inside the green rectangle, while (b) is the color histogram of the candidate calculated from the region inside the red rectangle. By comparing the color histogram with the reference color model, moving objects can be tracked since the same objects should have similar color distributions.

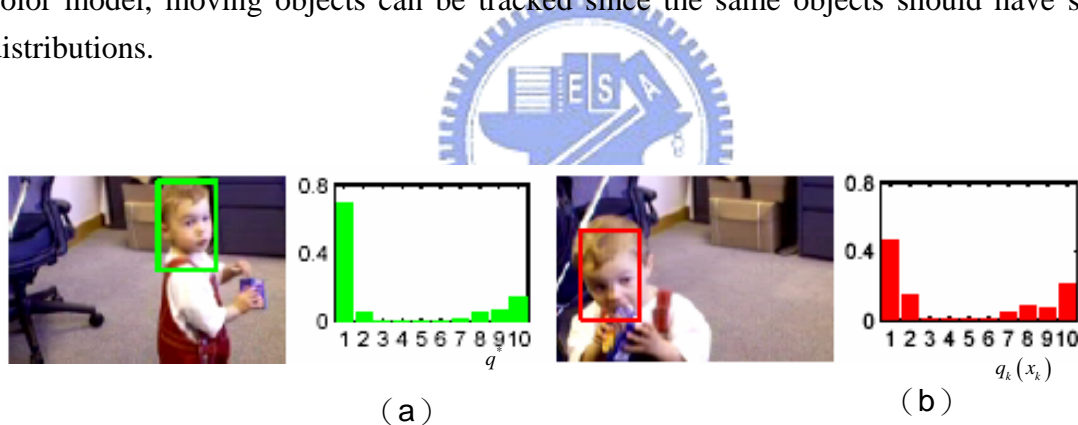


Figure 2-12 An example of color histogram [20]

(a) color histogram of the target (b) color histogram of the candidate

Bhattacharyya distance is a popular similarity measure of color histogram. The value of similarity measure is obtained by multiplying the color histograms of the reference model and the detected object in a point-by-point fashion and then summing up the products. The value of the similarity measure ranges from 0 to 1, with a larger value indicating a more similar matching. The advantage of color histogram is that it keeps the information of the target even if the target object is under expansion, shrink, rotation, or deformation. In addition, it requires low expense of computation. In the following section, we will introduce the mean-shift algorithm and particle filter, respectively.

2.2.4.1 Mean-Shift Algorithm

Mean-shift is a mathematic tool used to find the local maximum of any known or unknown distribution. As long as we have the well definition of object model such as color histogram and appropriate similarity measure such as Bhattacharyya coefficient, mean-shift can help us search for the top of the distribution to achieve the task of object tracking in an image sequence. In Figure 2-13, the surface is constructed by calculating the similarity between each point and its corresponding point in the reference model. The red point is the initial position and it will converge to the blue point which is the estimated position of moving object after applying mean-shift algorithm.

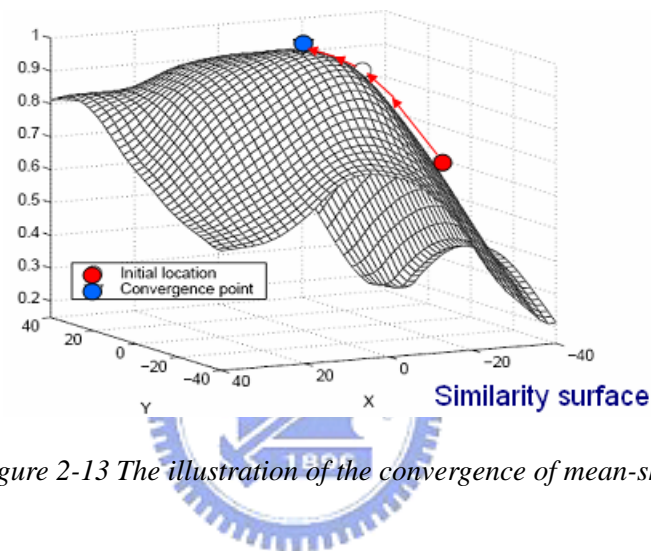


Figure 2-13 The illustration of the convergence of mean-shift

This method is appropriate to be applied to real-time surveillance system due to its low computation. However, since mean-shift algorithm can only find the local maximum, it must be careful to choose the initial position otherwise it will converge to the wrong position. So the system using mean-shift algorithm for objects tracking must have self-correction ability. That means it can detect and correct the wrong tracking result automatically to make the performance robust and reliable.

2.2.4.2 Particle Filter

If the moving objects in an image can be described as a probability distribution, then we can track the moving objects successfully by maintaining such a distribution precisely in an image sequence. Unfortunately, it is hard to estimate and update the distributions dynamically since these kinds of distributions usually do not belong to linear or Gaussian distributions. Particle filter algorithm provides a solution for such problem by using a group of particles to approximate the distribution. Through the prediction and update of each particle's weight, the distribution of moving objects will be described correctly over time. As shown in Figure 2-14, the position of each black point which is so-called particle sampled from the objects' distribution represents the possible position of the moving object. Each particle has different weight based on its similarity with the target model. The final position of moving object, the white point, is estimated by calculating the center of gravity of all these particles.

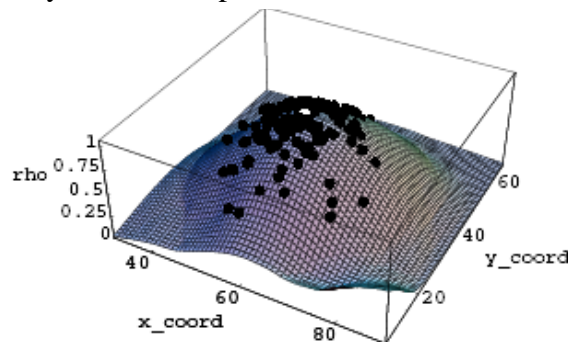


Figure 2-14 the concept of particle filter

The pro of particle filter algorithm is that the moving objects are described as a probability distribution approximated by lots of particles so the system has more chance to find the right positions of objects even under short-time occlusions. In Figure 2-15, top row shows the result of mean-shift algorithm while bottom row shows the result of particle filter and the red bounding box represents the final decision whereas yellow bounding box represents the decision of each particle with different weight. It is clear that the particle filter algorithm gives the better results than mean-shift algorithm.

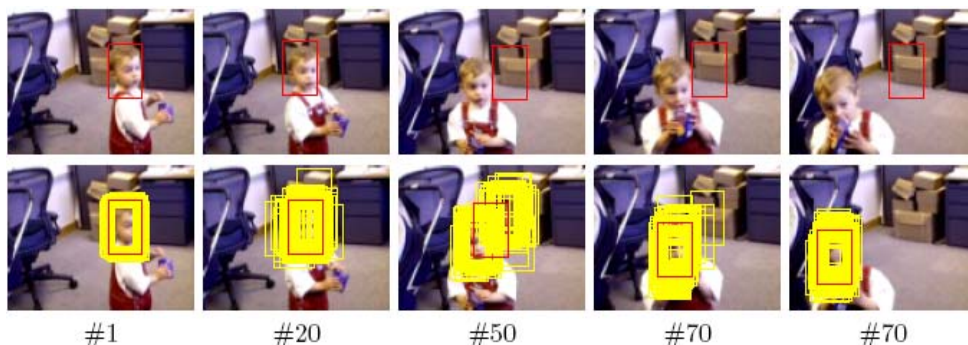


Figure 2-15 The comparison of mean-shift and particle filter[20]

Top row shows the result of mean-shift and bottom row shows the result of particle filter

2.3 Multi-Camera Correspondence

In previous sections, we have introduced how to detect and track moving objects over single-camera surveillance system. In the following section, we are going to introduce how to track moving objects and construct the correspondence between multiple cameras. The system will have a broader view for objects tracking through the cooperation of multiple cameras. In addition, when the correspondence relation has been established, the system will overcome some problems which are not able to be solved in single-camera surveillance system such as occlusions. Multi-camera correspondence can be roughly classified into region-based method and point-based method.

2.3.1 Region-Based Methods

Region-based methods usually represent the moving objects in an image as regions in one camera view and then compare the features of these regions with other regions in another camera view to construct the correspondence relations. Color information is the popular feature used to establish the correspondence such as color histogram [21] or GMM color model [22]. Although this method is really simple, it is not reliable due to the variation of illumination. Also, the color information of objects in different camera views has an important impact on the correspondence result. For example, if two persons in different camera views dressed in the clothes with same color or two sides of clothes are different colors, it is easy to make mistakes in objects correspondence.

2.3.2 Point-Based Methods

Point-based methods construct the correspondence by comparing the features between different camera views based on some constraints of cameras. According to different geometric constraints, we can classify the methods into correspondence in 2-D domain and correspondence in 3-D domain.

The concept of correspondence in 3-D domain is shown in Figure 2-16. The left figure is an example of indoor surveillance system and the right figure shows the top view of this indoor environment where the cameras mounted on the ceilings in a circle. Each camera monitors one view and then transmits the 2-D information of that view to 3-D domain. By fusing all 2-D information coming from cameras of different views, we can roughly estimate the positions of objects in 3-D domain. The correspondence relations are able to be constructed by back-projecting the 3-D estimations into each 2-D image plane. However, this method needs the camera calibration before the system starts to work.

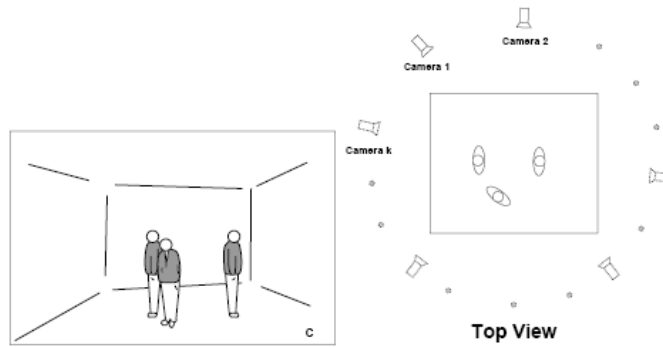


Figure 2-16 An example of indoor surveillance system[23]

Utsumi [23] finds the COG (center of gravity) of the detected objects in each 2-D image plane. The points inside a detected region are called COG if they have the longest distances to their closest boundaries. As shown in Figure 2-17, the left is the original residual after detection and the right figure illustrates the distance map of the residual and the black points are so-called COG. How to find the correspondence relations by COGs? First of all, we project COGs from all camera views to 3-D domain and use the Gaussian distributions to estimate the positions of objects in 3-D domain. Through the back-projection of 3-D Gaussian distributions to all 2-D image planes, we can compare the detected COGs with the projected COGs in each camera view to find the possible correspondence relations.

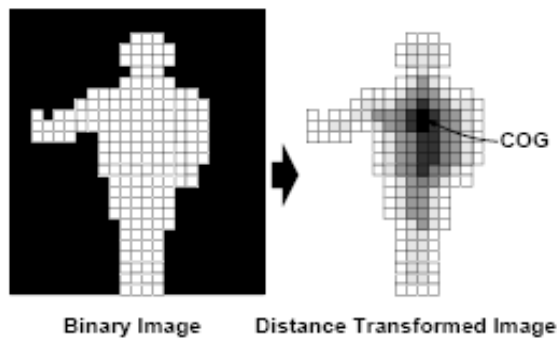


Figure 2-17 An example of Utsumi's method [23]

The left one is the original residual. And the right one is the distance map where the black points are so-called COG.

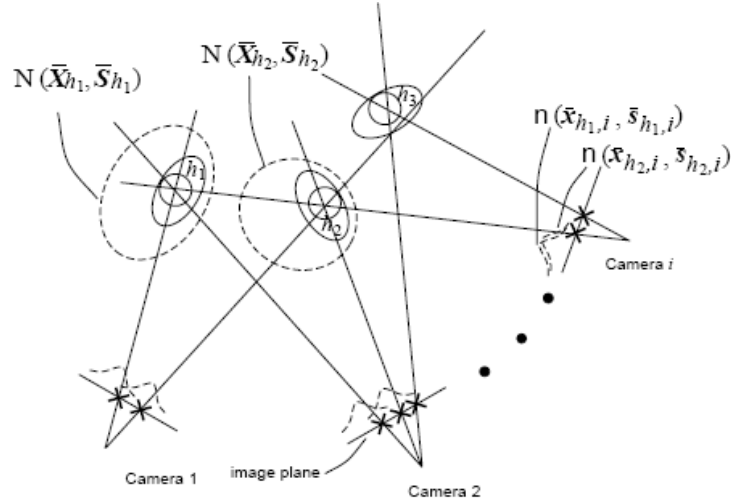


Figure 2-18 The illustration of correspondence in 3-D domain by Utsumi's method[23] The objects are modeled as Gaussian distributions with $N(\bar{X}_{h1}, \bar{S}_{h1})$ in 3-D domain where \bar{X}_{h1} and \bar{S}_{h1} are mean and covariance of Gaussian distribution respectively. It is still a Gaussian distribution after back-projection from 3-D to 2-D.

Black, Jamesa [24] uses the constraints of epipole plane to correspond objects in different camera views. As shown in Figure 2-19 it is an illustration of epipole plane where $m1$ and $m2$ are obtained by back-projecting the 3-D object located at M on image plane 1 and image plane 2 respectively. The back-projection lines, $Mm1$ and $Mm2$, will go through $Es1$ and $Es2$ which are camera focuses. The epipole plane constraint tells us that $Es1$, $Es2$ and M are in the same 2-D plane called epipole plane and $m2$ will lay on $ep11$ which is the intersection line of epipole plane and image plane 2.

Based on the epipole constraint, the center of each detected object in a 2-D image can project a line called epipole line on a 2-D image of another camera view and this epipole line will go through the center of the corresponding object belonging to the same 3-D object in that camera view. However, there may be some errors of 2-D tracking results. So the center of the corresponding object could not lay on the epipole line exactly. By combing this constraint and the 2-D tracking results, the task of multi-objects correspondence can be achieved.

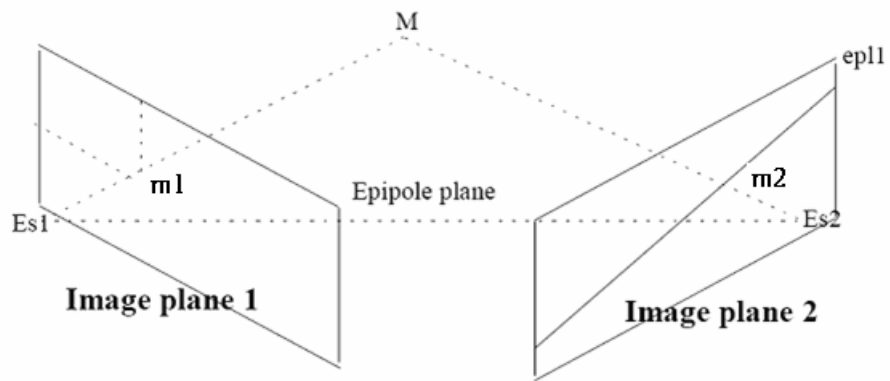


Figure 2-19 The illustration of epipole constraint [24]

Es_1, Es_2 are focuses of camera1 and camera2, M is the position of object in 3-D domain, m_1, m_2 are the back-projection of M to image plane 1 and image plane 2 respectively, and ep_{11} is the intersection line of epipole plane and image plane 2.

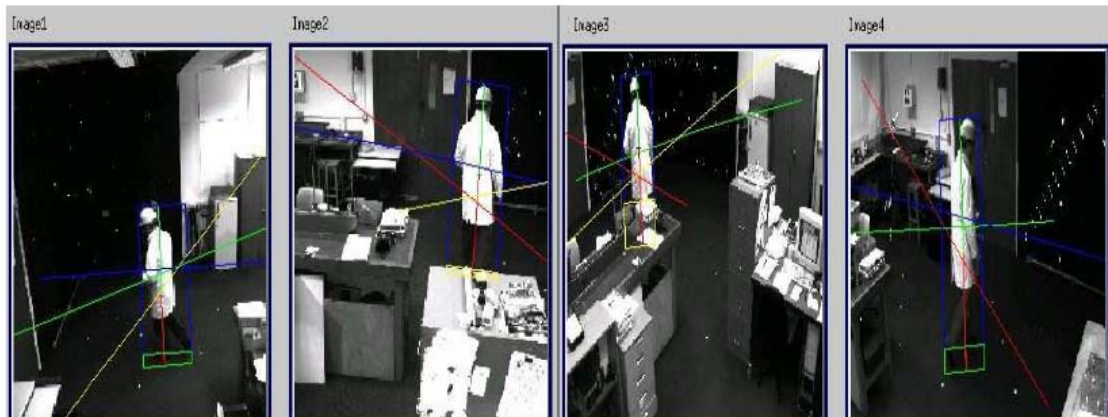


Figure 2-20 Black, Jamesa's correspondence method based on the epipole constraint [24]

There are epipole lines with different colors in four images above. The color of the epipole lines of detected object in image1 is red and green, blue, and yellow are the colors of the epipole lines of image2, image3, and image4 respectively. The green rectangles in image and image4 are ground plane regions, and the yellow rectangles in image2 and image3 are occlusion plane regions.

The correspondence method in 2-D domain doesn't need the calibrated cameras. It only needs some clues of 2-D images in different camera views to find the correspondence relations. S. Khan [25] finds the overlap regions of current image and other images of different camera views. Only when the moving object appears in such overlap regions, it has the correspondence relation to the objects detected in other camera views. As shown in *Figure 2-20*, the range of each camera view is shown in the left figure and the grey point represents the object. We can find that the object is visible in camera2's view but invisible in camera3's view. The right figure is camera1's view and the region of other camera's views projecting on it.

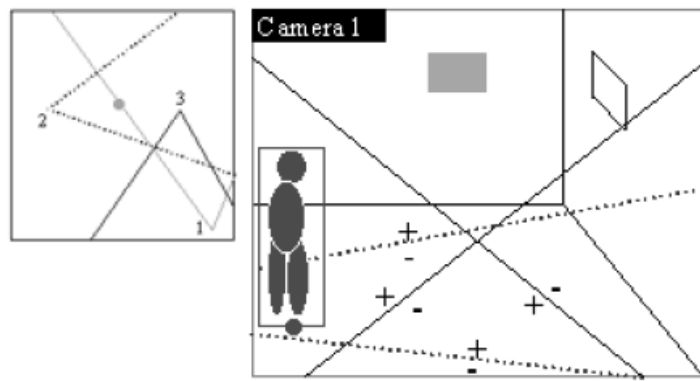


Figure 2-21 S. Khan's correspondence in 2-D method [25]

The left figure illustrates the ranges of three camera views. The right figure is the camera1's view. There are some lines indicated the regions of the overlap views from other camera's views. We can easily find that the black point in the bottom of object is visible in camera1's view and camera2's view but invisible in camera3's view.

J. Black [26] is to find the homography matrix between two 2-D images of different camera views. With this matrix, we can transform any point of current camera view to the point of another camera view. This method is much easier than the method using the epipole constraint. As shown in *Figure 2-22*, the blue lines are epipole lines and red points are the correspondence points using homography matrix transformation.



Figure 2-22 An example of J. Black's method [26]

Viewpoint correspondence using: epipole line analysis and homography alignment

The tasks of multi-objects correspondence in both 2-D domain and 3-D domain are hard due to that the results are easier degraded due to the influence of noise or the inaccuracy of 2-D tracking results.



Chapter 3 Proposed Method

In a multi-camera surveillance system, if we know the exact positions of moving objects in the 3-D domain, then the 2-D image planes are just the back-projection of the 3-D data onto different camera views. In other words, if we can track objects successfully in the 3-D domain, then the 2-D objects tracking problem is solved automatically. In practical situations, however, we only have 2-D images. Since the 2-D images have lost the depth information when projecting the 3-D objects onto 2-D images, it is difficult to deal with the occlusion problem when two or more objects are closer to one another. Hence, in our study, we try to find a method to combine the 2-D tracking results from different views of cameras and use the camera geometry constraints to estimate the 3-D positions of the moving objects.

In this thesis, we use a group of 3-D particles with different weights to describe the probability distribution of moving objects in the 3-D domain. This probability distribution can be constructed and updated by 2-D tracking results. In addition, this probability distribution can also help the refinement of the 2-D tracking results since it fuses the information coming from all camera views. The multi-objects correspondence can be established by some clues obtained by back-projecting all 3-D particles onto the 2-D image planes of different camera views. The correspondence relations will help us to solve the occlusion problem which is hard to be accomplished in single-camera surveillance systems. Besides, the establishment of correspondence is also an important clue for 3-D particles' classification and then the probability distribution will be more accurate to describe the moving objects in 3-D domain. The advantage of classification for 3-D particles is that the distribution will not only focus on some specific objects having better tracking results in 2-D image planes. When new objects enter, the probability distribution should be updated as soon as possible. So we can put extra 3-D particles along the paths where the objects may appear in 3-D domain based on the clues coming from the current 2-D tracking results. In addition, if the objects in some 2-D image planes are still not able to establish the correspondence relations, we can use the same method mentioned above to update the probability distribution. Consequently, the number of 3-D particles is variable which depends on the current situation. Moreover, our system has the ability to correct the correspondence relations automatically to make performance more robust and reliable.

3.1 2-D Objects Detection and Tracking

In this section, we will discuss the method of 2-D objects tracking applied in our system. Each image of the sequence is applied background subtraction for objects detection. Then by comparing and connecting the detection results between successive images, the task of objects tracking is accomplished successfully.

3.1.1 Background Subtraction Detection

Since our system belongs to the static-camera surveillance system, we can build the reference background model for each camera view before starting the system. Then we can extract the foreground by taking the difference between current image and the reference background in a point-by-point fashion. However, the residuals are neither complete nor correct and need to do some posterior processing due to the influence of noise, variation of illuminations, and shadows. First of all, we use a given threshold of intensity to remove some noise and apply the method proposed in [3] to reduce the influence of illuminations and shadows. As shown in *Figure 3-1*, (a) is the residuals after a given threshold, (b) indicates the illuminations and shadows, and (c) is the final result obtained by eliminating noise, illuminations and shadows. It is clear that the influence of the shadows is reduced in this example.



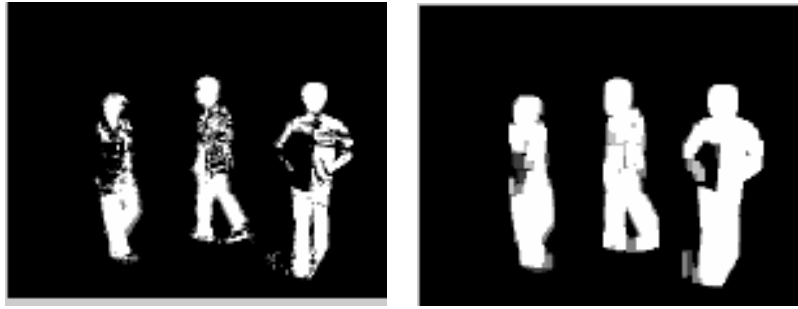
(a) the residuals after a given threshold

(b) illuminations and shadows

(c) final result

Figure 3-1 a posterior process of background subtraction

Then we apply the morphological operations to fill the holes to make the residuals more complete. As shown in *Figure 3-2*, figure (a) and (b) are the results before and after the morphological operations.

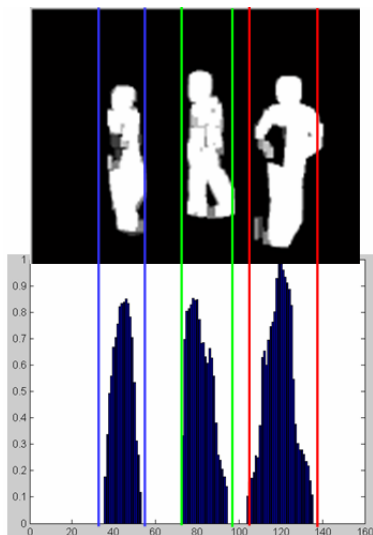


(a) the results before the morphological operations

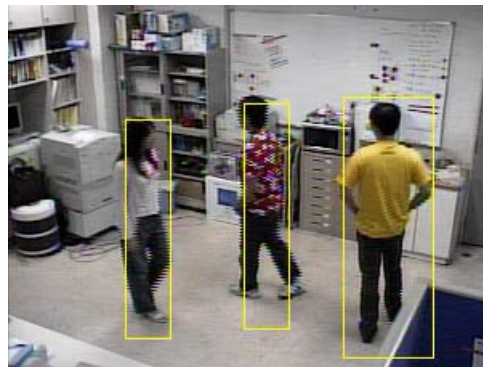
(b) the results after the morphological operations

Figure 3-2 The results before and after the morphological operations

Sometimes there is more than one object in an image. So we need to segment the objects from the residuals. First, we sum the residuals vertically to get the profile of moving objects in x-direction. The height of profile is proportion to the probability of moving objects appearance. So we can use a given threshold to eliminate some noise of the profile to segment the moving objects in x-direction, get the width of each moving object, and calculate the number of moving objects in that image. This process is shown in *Figure 3-3 (a)*. Then we can sum the residuals in each segmented region horizontally to get the length of each moving object. Finally, each moving object in the image is marked as a bounding box as shown in *Figure 3-3 (b)*. However, when two or more objects in an image become too closer, they will be detected as one object since there is no extra information to separate them.



(a)an example of vertical summation of the residual.

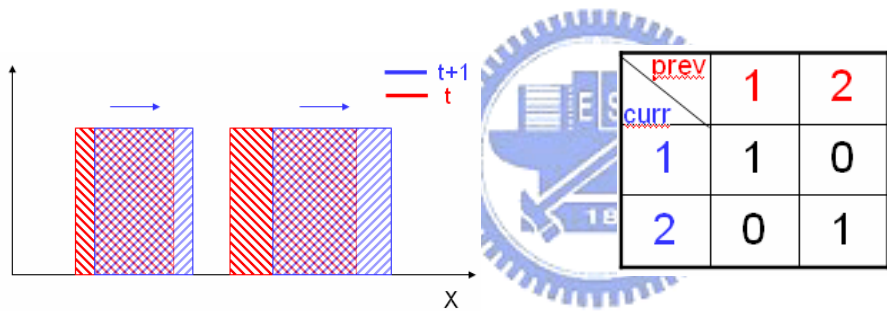


(b)final result

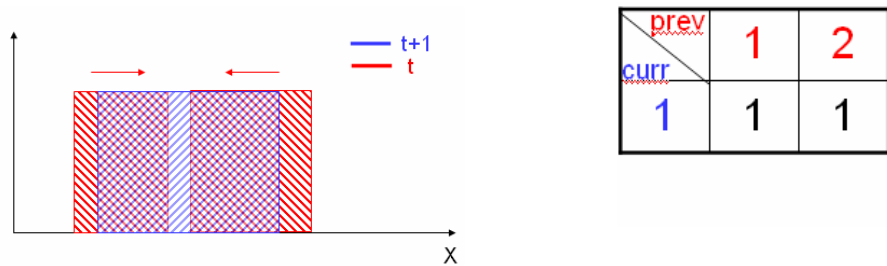
Figure 3-3a illustration of objects detection applied in our system

3.1.2 Mapping Objects

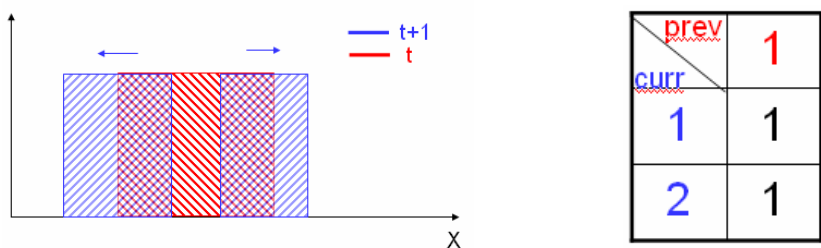
Through the method mentioned above, we can do objects detection for each image. In order to do objects tracking, the detection result of each image should be connected in time domain. Our objects tracking method is to compare the detection results between successive images to correspond the moving objects in time domain. The concept can be explained by *Figure 3-4*. Left column is the illustration of all possible tracking results between successive images and the red rectangle represents the detected object at time $t-1$ while blue rectangle represents the detected object at time t . We can establish the correspondence relations of moving objects in time domain by analyzing their overlap regions. Right column represents the correspondence tables of moving objects at time $t-1$ and time t . Figure (a) is the one to one case where the moving objects at time $t-1$ correspond to different objects at time t respectively, figure (b) is the mergence case where the moving objects become closer at time $t-1$ and then merge to the same object at time t , figure (c) is the split case where the moving object at time $t-1$ splits into two objects at time t , figure (d) is the new objects case where there are more moving objects at time t than time $t-1$, and figure (e) is the leave case where some of moving objects at time $t-1$ disappear at time t .



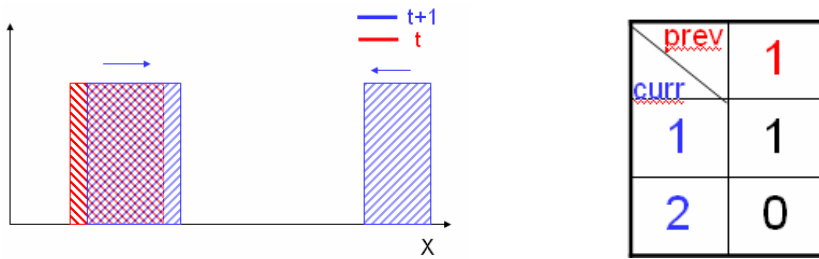
(a) one to one



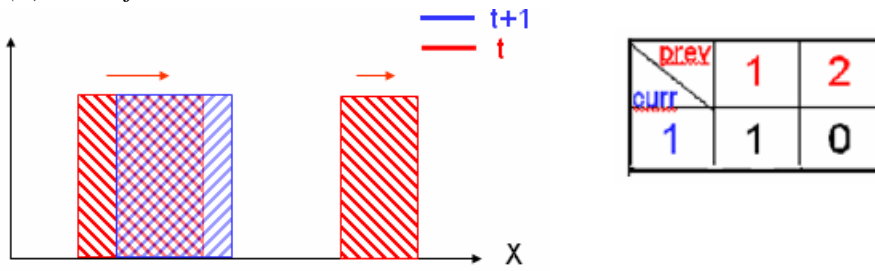
(b) mergence



(c) split



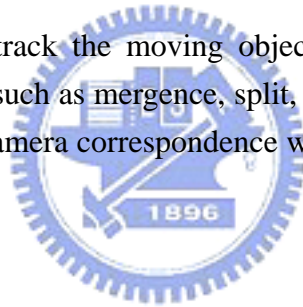
(d) new objects



(e) leave

Figure 3-4 the illustration of all kinds of tracking results

Through this way, we can track the moving objects in an image sequence and get the relations of these moving objects such as mergence, split, or new objects entering etc. It provides the useful information for multi-camera correspondence which will be discussed in the following sections. .



3.2 Transformation Between 2-D and 3-D

Before using multiple cameras for objects tracking and correspondence, the geometric relations between cameras should be established at first. With such relations we can project the 2-D tracking results into 3-D domain and then fuse the information to achieve the cooperation of multiple cameras for objects tracking and correspondence. After camera calibration, we can get the intrinsic and extrinsic parameters of cameras which can be used for transforming the information between 2-D domain and 3-D domain. *Figure 3-5 (a)* illustrates the world coordinate of PTZ camera. O_R and O_C are the rotation center and projection center of camera respectively, O_C' is the new projection center after the camera tilts ϕ degree and r is rotation radius of camera.

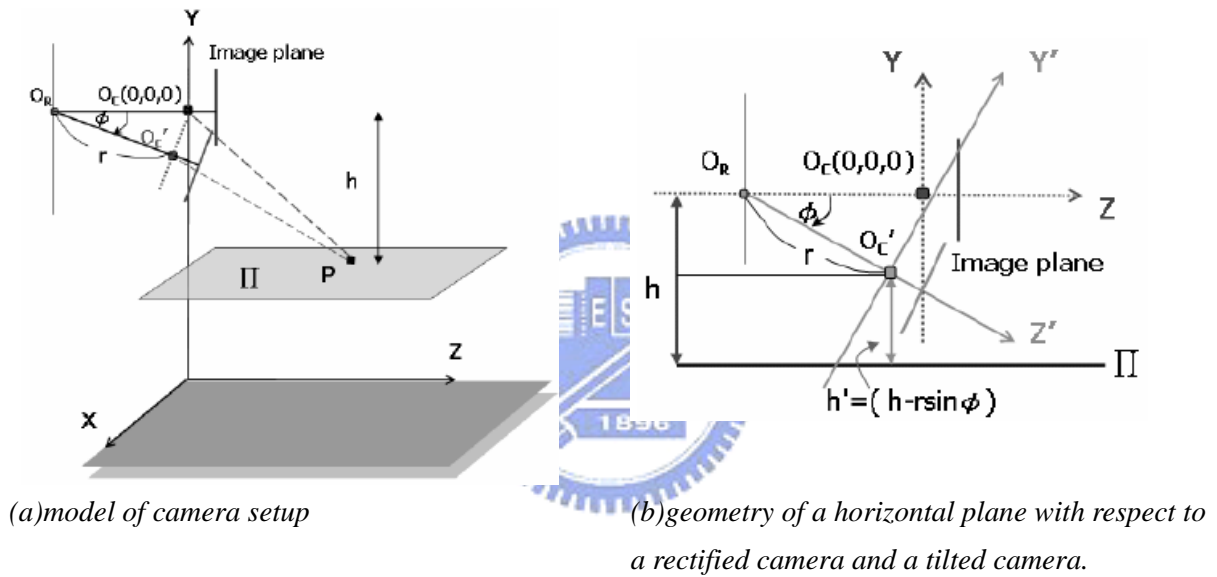


Figure 3-5 a illustration of camera setup[27]

Each camera has its own world coordinate and some intrinsic and extrinsic parameters obtained by camera calibration. With these parameters we can easily transform information between 2-D and 3-D domain. Also we can get the rotation matrix and translation vectors for transformation between different world coordinates. The formula of transformation from 3-D to 2-D is as follows,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \frac{X}{Z'} \\ \beta \frac{Y}{Z'} \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \frac{X}{-Y \sin \phi + Z \cos \phi + r(\cos \phi - 1)} \\ \beta \frac{Y \cos \phi + Z \sin \phi + r \sin \phi}{-Y \sin \phi + Z \cos \phi + r(\cos \phi - 1)} \\ 1 \end{bmatrix}, \quad \text{Eq 3-1}$$

where X, Y, Z represent the world coordinate, X', Y', Z' represent the world coordinate with camera tilted ϕ degree, x', y' are image coordinate and the formula of transformation from 2-D domain to 3-D domain is as follows,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{(x' - u_0)\beta(r \sin \phi - h)}{\alpha[(v_0 - y') \cos \phi - \beta \sin \phi]} \\ -h \\ \frac{[(v_0 - y') \sin \phi + \beta \cos \phi](r \sin \phi - h)}{(v_0 - y') \cos \phi - \beta \sin \phi} - r + r \cos \phi \end{bmatrix} \quad \text{Eq 3-2}$$

. Each pixel of an image can only emit a line instead of a corresponding point in the 3-D world coordinate since it loses the depth information. Giving different height h produces a corresponding point X, Y, Z on that line based on Eq 3-3.

We can get a set of 3-D lines emitted from the centers of bounding box in different camera views. If these detected objects correspond to the same 3-D object then those 3-D lines will intersect at the same point. But that intersection point may not exist due to the tracking errors. However, we can use the method proposed in [24]. Assume there are N objects in different camera views and their corresponding 3-D lines are $r_i = a_i + \lambda_i b_i$ and $1 \leq i \leq N$. Then the estimated point can be approximated as

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N 1 - b_{ix}^2 & \sum_{i=1}^N -b_{ix}b_{iy} & \sum_{i=1}^N -b_{ix}b_{iz} \\ \sum_{i=1}^N -b_{ix}b_{iy} & \sum_{i=1}^N 1 - b_{iy}^2 & \sum_{i=1}^N -b_{iy}b_{iz} \\ \sum_{i=1}^N -b_{ix}b_{iz} & \sum_{i=1}^N -b_{iy}b_{iz} & \sum_{i=1}^N 1 - b_{iz}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^N a_{ix} - b_{ix}a_i \cdot b_i \\ \sum_{i=1}^N a_{iy} - b_{iy}a_i \cdot b_i \\ \sum_{i=1}^N a_{iz} - b_{iz}a_i \cdot b_i \end{bmatrix}. \quad \text{Eq 3-4}$$

With these transformation formulas, it is easier and more convenient to develop the multi-camera tracking and correspondence algorithms.

3.3 3-D Particle Filter

Before discussing multi-camera correspondence methods, we will introduce particle filter algorithm in more formal way. First, we will deduce the basic formula of particle filter and then discuss how to apply both 2-D and 3-D information in our surveillance system based on this mathematic tool.

3.3.1 Particle Filter Algorithm

Particle filter algorithm has been applied in objects tracking extensively. The concept of particle filter is to use a group of particles with different weights to describe the probability distribution of moving objects. So we can use the probability instead of exact value to describe the location of the object. As shown in *Figure 3-6* the red points are the samples from the distribution and the more number of samples, the more accuracy of probability distribution. The blue point on the top of the distribution is the estimated location obtained by averaging all red points with different weights. Trough the prediction and update of the distribution, we can estimate the location of moving objects to achieve the tracking task.

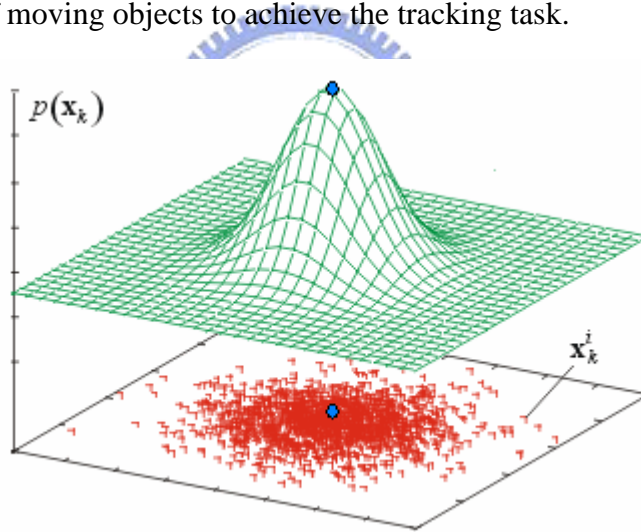


Figure 3-6 a illustration of particle filter

Then we will use the mathematic formula to express the algorithm of particle filter. We should define some variables at first. k is the time index, $\{x_k, k \in N\}$ is the set of state sequence, and $\{z_k, k \in N\}$ is the set of measurement sequence. The estimation of object state at time k can be represented as $x_k = f_k(x_{k-1}, v_{k-1})$ where $f_k(\bullet)$ is an estimation function which is related to the object state at time $k-1$ and v_{k-1} , i.i.d noise at time $k-1$. The measurement of object at time k can be expressed as $z_k = h_k(x_k, n_k)$ where $h_k(\bullet)$ is a measurement function which is related to the current state and n_k , i.i.d noise at time k . $f_k(\bullet)$ and $h_k(\bullet)$ are not restricted to be linear functions. So the probability distribution of object can be written as

$$p(x_k | z_{1:k}) = \frac{p(z_k | x_k)p(x_k | z_{1:k-1})}{p(z_k | z_{1:k-1})} \quad \text{Eq 3-5}$$

$$\text{where } p(x_k | z_{1:k-1}) = \int p(x_k | x_{k-1})p(x_{k-1} | z_{1:k-1})dx_{k-1} \quad \text{Eq 3-6}$$

. Based on these two equations, we can deduce $p(x_k | z_{1:k})$ from $p(x_{k-1} | z_{1:k-1})$. Eq 3-5 and Eq 3-6 are the equations for updating and predicting of probability distribution respectively. By calculating these two equations iteratively, we can update the distribution of the moving object.

Generally speaking, the probability distribution of the moving object doesn't belong to known distribution. Consequently, we use a group of particles with different weights written as

$\{x_{0:k}^i, w_k^i\}_{i=1}^{N_s}$ to express $p(x_{0:k} | z_{1:k})$. It can be represented as

$$p(x_{0:k} | z_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \cdot \delta(x_{0:k} - x_{0:k}^i) \quad \text{Eq 3-7}$$

where $\sum_{i=1}^{N_s} w_k^i = 1$

So we can regard these particles as the samples obtained from the distribution. Then the problem of how to predict and update the distribution becomes the problem of how to generate the particles and how to update their weights in the next time. Assume we can sample the particles, $\{x_{0:k}^i\}_{i=1}^{N_s}$, from the importance density, $q(x_{0:k} | z_{1:k})$, then $w_k^i \propto \frac{p(x_{0:k}^i | z_{1:k})}{q(x_{0:k}^i | z_{1:k})}$ in Eq 3-7.

Also, assume $q(x_{0:k}^i | z_{1:k}) = q(x_k | x_{0:k-1}, z_{1:k}) \cdot q(x_{0:k-1} | z_{1:k-1})$, that means we can sample $\{x_k^i\}_{i=1}^{N_s}$ and $\{x_{0:k-1}^i\}_{i=1}^{N_s}$ from $q(x_k | x_{0:k-1}, z_{1:k})$ and $q(x_{0:k-1} | z_{1:k-1})$ respectively. With replacement, w_k^i can be inferred as

$$\begin{aligned} w_k^i &\propto \frac{p(x_{0:k}^i | z_{1:k})}{q(x_{0:k}^i | z_{1:k})} \propto \frac{p(z_k^i | x_k^i)p(x_k^i | x_{k-1}^i)p(x_{0:k-1}^i | z_{1:k-1})}{q(x_k^i | x_{0:k-1}^i, z_{1:k})q(x_{0:k-1}^i | z_{1:k-1})} \\ &\propto w_{k-1}^i \frac{p(z_k | x_k^i)p(x_k^i | x_{k-1}^i)}{q(x_k^i | x_{0:k-1}^i, z_{1:k})} \end{aligned} \quad \text{Eq 3-8}$$

.We can generate new particles through $q(x_{0:k} | z_{1:k})$ and each particle's weight can be calculated by Eq 3-8. Hence, we can get the probability distribution over time. Most of time, we will assume $q(x_{0:k} | z_{1:k}) = p(x_k | x_{k-1})$ then the Eq 3-8 can be rewritten as $w_k \propto w_{k-1}^i \cdot p(z_k | x_k^i)$. Table 3-1 shows the algorithm of particle filter.

Table 3-1 algorithm of particle filter [28]

<p>Algorithm 1: SIS Particle Filter $[\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}] = \text{SIS}[\{\mathbf{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, \mathbf{z}_k]$ <ul style="list-style-type: none"> • FOR $i = 1; N_s$ <ul style="list-style-type: none"> - Draw $\mathbf{x}_k^i \sim q(\mathbf{x}_k \mathbf{x}_{k-1}^i, \mathbf{z}_k)$ - Assign the particle a weight, w_k^i, • END FOR </p>

We can estimate the location of the moving object, X_{mean} , by averaging these particles with different weights and where

$$X_{mean} = \sum_{i=1}^{N_s} w_k^i \cdot x_k^i, \quad \sum_{i=1}^{N_s} w_k^i = 1 \quad \text{Eq 3-9}$$

. However, the potential problem of this algorithm is that the variation of the weights will become more sever along the time. That means we will pay a lot of efforts to compute the particles with very small weights then the final distribution will still be dominated by some particular particles with large weights. The problem will be solved by the method of resampling. The idea of resampling is to remove the particles with excessively small weights and split the particles with large weights at each time. This concept is shown in *Figure 3-7* and Table 3-2 shows the algorithm of resampling. At first we evaluate the cdf of these particles according their weights and generate $u_i, 1 \leq i \leq N_s$, from a uniform distribution. By searching the cdf from small to large value, we will find the value which is the smallest value bigger than u_i then assign its corresponding particle to the new particle. So it is obvious that the particles with large weights will be sampled many times. On the contrary, the particles with small weights may not be sampled after all.

Table 3-2 algorithm of resampling[28]

Algorithm 2: Resampling Algorithm
 $[\{\mathbf{x}_k^{j*}, w_k^j, i^j\}_{j=1}^{N_s}] = \text{RESAMPLE } [\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}]$

- Initialize the CDF: $c_1 = 0$
- FOR $i = 2: N_s$
 - Construct CDF: $c_i = c_{i-1} + w_k^i$
- END FOR
- Start at the bottom of the CDF: $i = 1$
- Draw a starting point: $u_1 \sim \mathcal{U}[0, N_s^{-1}]$
- FOR $j = 1: N_s$
 - Move along the CDF: $u_j = u_1 + N_s^{-1}(j - 1)$
 - WHILE $u_j > c_i$
 - * $i = i + 1$
 - END WHILE
 - Assign sample: $\mathbf{x}_k^{j*} = \mathbf{x}_k^i$
 - Assign weight: $w_k^j = N_s^{-1}$
 - Assign parent: $i^j = i$
- END FOR

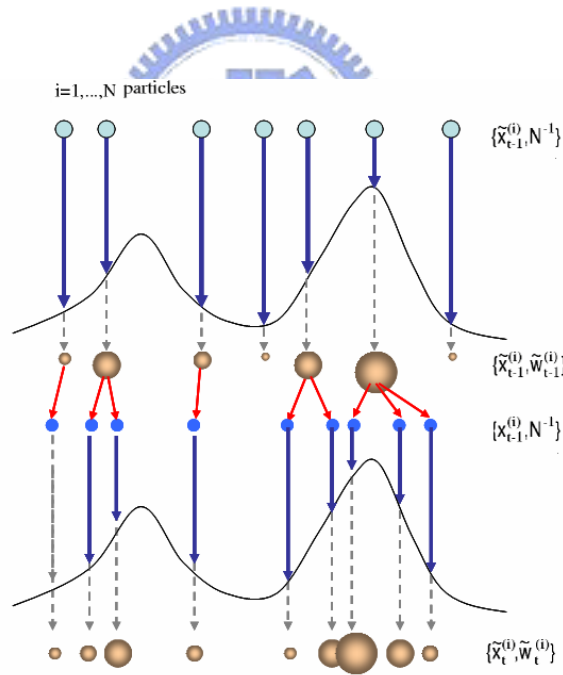


Figure 3-7 a illustration of resampling

3.3.2 Proposed 3-D Particle Filter

Nowadays, particle filter focuses on 2-D objects tracking. When having multiple cameras, we can extend the objects tracking from 2-D domain to 3-D domain and take the advantage of cooperation of multiple cameras by fusing their information together to achieve the mission of objects tracking. We propose a surveillance system based on 3-D particle filter to track moving objects in different camera views and construct their correspondence relations. Before starting to introduce our method, we define some variables and notations at first. $\{X_t^{(s)}\}_{s=1}^{N_{ts}}$ is the set of 3-D particles at time t where each $X_t^{(s)}$ represents a location in 3-D coordinate and $\{W_t^{(s)}\}_{s=1}^{N_{ts}}$ is the set of weights of $\{X_t^{(s)}\}_{s=1}^{N_{ts}}$. N_{ts} is the number of 3-D particles and it varies according to the current tracking and correspondence results. N_c is the number of cameras and $N_t(c)$ represents the number of detected objects in c th camera view at time t. M_t is number of detected objects in 3-D domain at time t. and $M_t \geq N_t(c), 1 \leq c \leq N_c$ since the objects in 3-D domain may not be visible in all camera views all the time.

3.3.2.1 First Generation of Particles

First of all, we should generate some particles in 3-D domain as the initial state. We utilize the 2-D tracking results of different camera views to estimate the possible locations of moving objects in 3-D domain. The concept is shown in Figure 3-8. This tree illustrates all possible correspondence relations of objects in all camera views and each branch represents one correspondence relation where $N_0(1) \sim N_0(N_c)$ is the number of detected objects in each camera view at time 0. We use each estimated correspondence relation from each branch to emit some 3-D lines from those detected objects' centers in different camera views and then use the approximation method in [24] to find the estimated intersection point of these 3-D lines.

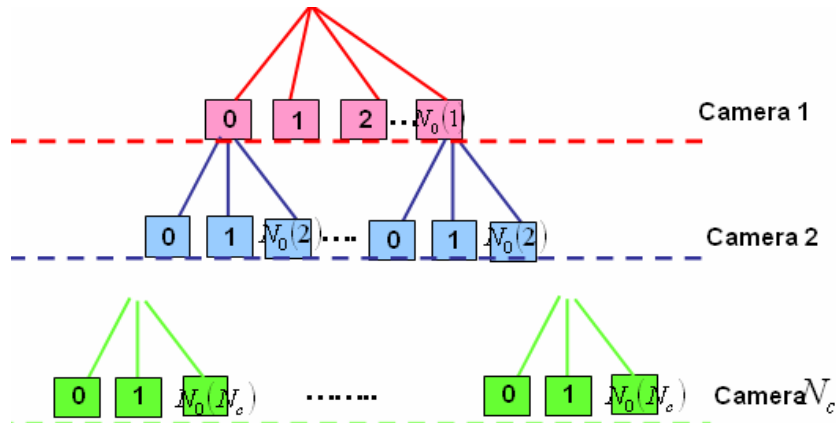
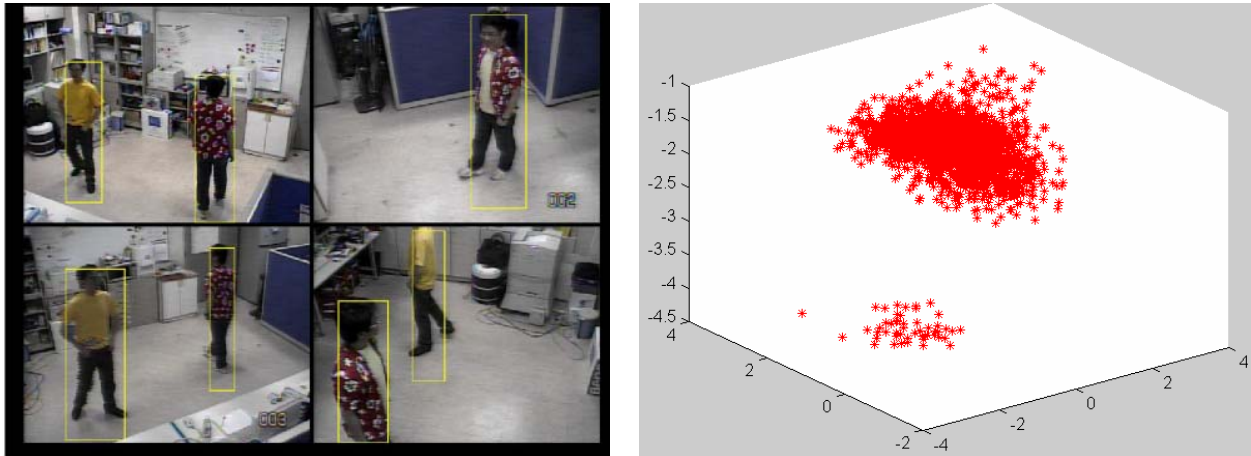


Figure 3-8 a illustration of tree
All possible correspondence relations

There will be a lot of particles obtained by above method and then we will put some new random particles around those particles to get a group of initial particles, $\{X_0^{(s)}\}_{s=1}^{N_{os}}$. In *Figure 3-9*, figure (a) demonstrates the 2-D tracking results in all camera views and figure (b) is the illustration of the initial particles in 3-D domain.



(a) 2-D tracking results in all camera views

(b) a illustration of initial particles in 3-D domain

Figure 3-9 an example of generating initial particles



3.3.2.2 Correction of Particles

We can get a group of 3-D particles, $\{X_t^{(s)}\}_{s=1}^{N_t}$, through first generation or prediction of particles and the objects probability distribution in 3-D domain can be established by updating those particles' weights. We update the weights of the particles by projecting those particles from 3-D domain to 2-D domain to compare with the 2-D tracking results. Before starting to deduce the formula of updating weights, we will define the mapping function $I_{32}(\bullet)$ more formally. It is a function mapping a 3-D particle into a 2-D image plane of c -th camera view and then find that particle's corresponding object and it can be formulated as follows:

$$I_{32}(X_t^{(s)}, c) = \begin{cases} i, & \Phi(X_t^{(s)}, c) \text{ falls in the bounding box of } i \text{ th object in } c \text{ th camera view} \\ 0, & \Phi(X_t^{(s)}, c) \text{ doesn't fall in any bounding box in } c \text{ th camera view} \end{cases} \quad \text{Eq 3-10}$$

, where $1 \leq c \leq N_c$ and $1 \leq i \leq N_t(c)$. $\Phi(X_t^{(s)}, c)$ which is a transformation function gives a 2-D coordinate value by back-projecting a 3-D particle, $X_t^{(s)}$, into the 2-D image plane in c th camera view. The idea of the definition of each 3-D particle's weight is that if the distance between $\Phi(X_t^{(s)}, c)$ and the center of its corresponding bounding box is small then the weight becomes large due to its accurate correspondence and vice versa. The mathematic formulation is as follows.

$$W_t^{(s)} = \frac{1}{\sqrt{2\pi}\sigma} e^{\left(-\frac{d_t^{(s)} - \bar{d}_t}{2\sigma^2}\right)} \quad \text{Eq 3-11}$$

$$\sum_{s=1}^{N_t} W_t^{(s)} = 1 \quad \text{Eq 3-12}$$

, where

$$d_t^{(s)} = \frac{\sum_{c=1}^{N_c} \sum_{i=1}^{N_t(c)} \left(\left| \frac{(\Phi(X_t^{(s)}, c))_x - (y_t^{c,i})_x}{wid^{c,i}} \right| + \left| \frac{(\Phi(X_t^{(s)}, c))_y - (y_t^{c,i})_y}{len^{c,i}} \right| \right) \cdot \delta(I_{32}(X_t^{(s)}, c) - i)}{\sum_{c=1}^{N_c} \sum_{i=1}^{N_t(c)} \delta(I_{32}(X_t^{(s)}, c) - i)}, \quad \text{Eq 3-13}$$

$$\bar{d}_t = \frac{\sum_{s=1}^{N_{ts}} \sum_{c=1}^{N_c} \sum_{i=1}^{N_t(c)} \left(\left| \frac{(\Phi(X_t^{(s)}, c))_x - (y_t^{c,i})_x}{wid^{c,i}} \right| + \left| \frac{(\Phi(X_t^{(s)}, c))_y - (y_t^{c,i})_y}{len^{c,i}} \right| \right) \cdot \delta(I_{32}(X_t^{(s)}, c) - i)}{\sum_{s=1}^{N_{ts}} \sum_{c=1}^{N_c} \sum_{i=1}^{N_t(c)} \delta(I_{32}(X_t^{(s)}, c) - i)}, \text{ Eq 3-14}$$

$(\bullet)_x$ and $(\bullet)_y$ represent the 2-D coordinate value in x-direction and y-direction respectively.

$len^{c,i}$ and $wid^{c,i}$ representing the length and width of i-th object's bounding box in c-th camera view are the normalization terms to make the definition of weight more reasonable. Without these two terms, the weight will be in proportion to the size of detected objects.

After these computation, we can finally get these 3-D particles with different weights, $\{X_t^{(s)}, W_t^{(s)}\}_{s=1}^{N_{ts}}$, at time t.



3.3.2.3 Shrink of Particles

In each camera view, every object has a subset of the 3-D particles. When projecting such subset of 3-D particles into the 2-D image plane, they will all fall in its corresponding 2-D object's bounding box. In other words, this subset of 3-D particles can be visualized as the distribution of that 2-D object in 3-D domain. Consequently, we can estimate the position of this 2-D object by finding the center of gravity of its corresponding subset of 3-D particles.

We condense entire 3-D particles to the centers of gravity of all subsets. The information of objects is still complete even if the number of particles reduces from N_{ts} to $\sum_{c=1}^{N_c} N_t(c)$. In more formal way, every 2-D object can get its center of gravity and weight in 3-D domain through the transformation function $X_M(X_t, c, i)$ and $W_M(X_t, c, i)$ as shown in Eq 3-15 and Eq 3-16 where $X_t = \{X_t^{(s)}\}_{s=1}^{N_{ts}}$ represent all 3-D particles, c and i represent i -th object in c -th camera view, $1 \leq c \leq N_c$ and $1 \leq i \leq N_t(c)$.

$$X_M(X_t, c, i) = \frac{\sum_{s=1}^{N_{ts}} X_t^{(s)} \cdot W_t^{(s)} \cdot \delta(I_{32}(X_t^{(s)}, c) - i)}{\sum_{s=1}^{N_{ts}} W_t^{(s)} \cdot \delta(I_{32}(X_t^{(s)}, c) - i)} \quad \text{Eq 3-15}$$

$$W_M(X_t, c, i) = \frac{\sum_{s=1}^{N_{ts}} W_t^{(s)} \cdot \delta(I_{32}(X_t^{(s)}, c) - i)}{\sum_{s=1}^{N_{ts}} \delta(I_{32}(X_t^{(s)}, c) - i)} \quad \text{Eq 3-16}$$

As shown in *Figure 3-10*, the red points are the 3-D particles after weight updating and the blue points are the centers of gravity after shrink. By back-projecting these centers of gravity to its corresponding 2-D image plane, we can get the refined location of 2-D tracking result, $\bar{y}_t^{c,i}$, which is written as

$$\bar{y}_t^{c,i} = \Phi(X_M(X_t, c, i), c) \quad \text{Eq 3-17}$$

. These centers of gravity also can be utilized to establish the correspondence relations between 2-D detected objects in all camera views since if some 2-D objects correspond to the same 3-D object then their centers of gravity in 3-D domain should be very close. The method of correspondence will be discussed in 3.4.

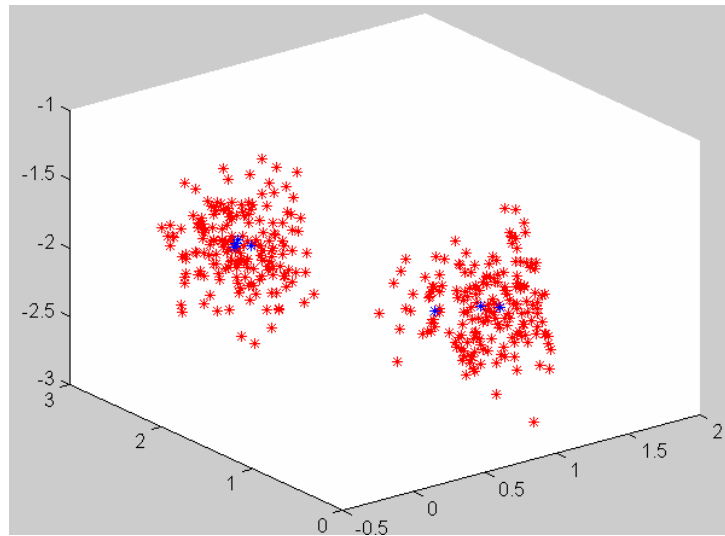


Figure 3-10 3-D particles and the centers of gravity after shrink. Red points are all 3-D particles and blue points are the center of gravity after shrink



3.3.2.4 Prediction of Particles

We can use the centers of gravity obtained by shrinking the 3-D particles to predict the objects probability distribution. Based on the correspondence relations, we can classify these centers of gravity to their corresponding 3-D objects and then use them to predict 3-D objects' positions in the next time.

We generate N_p new particles for each 3-D object by using the method of resampling mentioned in 3.1. Each center of gravity belonging to the same 3-D object will split as several new particles in proportion to its weight. We define an indicator, $I_{23}(\bullet)$, which gives the index of its corresponding 3-D object from a 2-D object. For example, $I_{23}(c,i) = k$ represents that the i -th object in c -th camera view corresponds to k -th object in 3-D domain. This indicator is obtained after establishing the correspondence relations. The algorithm of $Resample(\bullet)$ is referred in Table 3-2. The centers of gravity belonging to the same 3-D object with different weights will split into equal weight 3-D particles through this function. The estimated state of these particles can be written as

$$X_{t+1}^{(s)} = \hat{X}_t^{(s)} + n_{t+1} \quad \text{where } n_{t+1} \sim N(0, \Sigma), \quad 1 \leq s \leq N_{ts} \quad \text{Eq 3-18}$$

$$\left\{ \hat{X}_t^{(s)}, \frac{1}{N_p} \right\}_{s=(k-1)N_p+1}^{k \cdot N_p} = \text{Resampling} \left(\left\{ X_M(X_t, c, i), W_M(X_t, c, i) \mid I_{23}(c, i) = k \right\}_{c=1}^{N_c} \right)_{i=1}^{N_t(c)} \quad \text{Eq 3-19}$$

$$1 \leq k \leq M_t$$

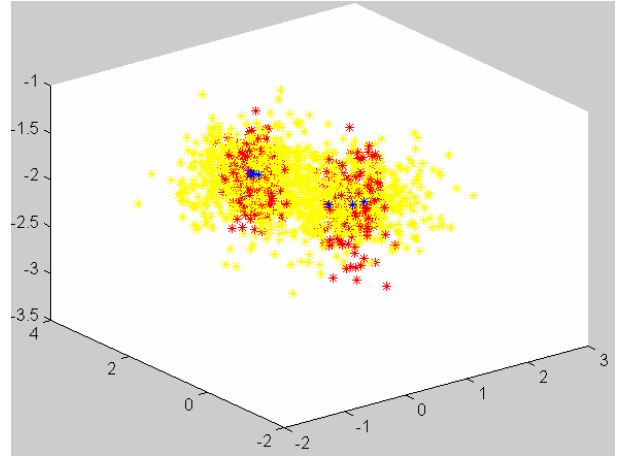
, where $W_M(X_t, c, i)$ is normalized as

$$W_M(X_t, c, i) = \frac{W_M(X_t, c, i)}{\sum_{c=1}^{N_c} \sum_{i=1}^{N_t(c)} W_M(X_t, c', i') \cdot \delta(I_{23}(c', i') - I_{23}(c, i))} \quad \text{Eq 3-20}$$

In *Figure 3-11*, figure (a) is the 2-D tracking results with objects correspondence. The 2-D objects belonging to the same 3-D object have their bounding box in the same color. Figure (b) demonstrates the 3-D particles under different situation. The red points describe the current probability distribution, the blue points represent the centers of gravity after shrink and the yellow points are the prediction particles describing the probability distribution in the next time.



(a) 2-D tracking results with objects correspondence



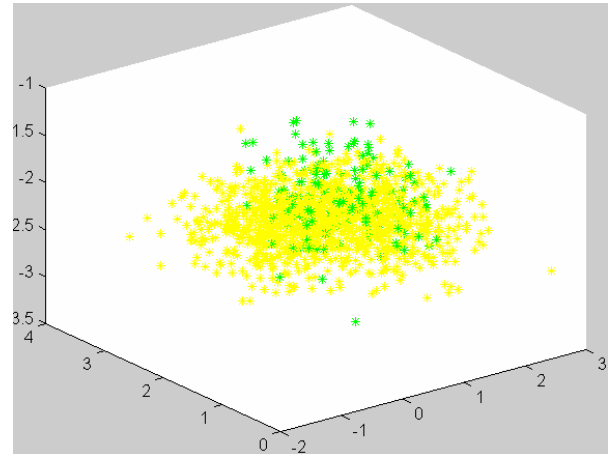
(b) 3-D particles under different situations

Figure 3-11 2-D tracking results and the illustration of 3-D particles

When detecting a new object entering a 2-D image, the probability distribution of 3-D objects should be updated immediately. As shown in *Figure 3-12(a)*, the 2-D image detects a new object in the right top sub-image with yellow bounding box. Then along the 3-D line emitting from the center of this bounding box, we will put N_p new particles on the particular locations where the object may appear based on some known information such as the normal height of a person. As shown in *Figure 3-12(b)*, the yellow points are the prediction particles and the green points represent the particles of a new object. By adjusting the weights of these particles through 2-D tracking results, the new probability distribution can be updated. As for the objects which can not establish the correspondence relation, we will use the same method to emphasize the positions where the 3-D objects of those corresponding 2-D objects may appear. As shown in *Figure 3-13(a)*, the object selected by the yellow bounding box in the right bottom sub-image still doesn't establish the correspondence relation yet. In *Figure 3-13(b)*, the yellow points are prediction particles and the green points represent the extra particles generated for that 2-D object. As a result, the total number of particles is $N_{t+1s} = (M_{t+1} + M_{new} + M_{uncrsp}) \cdot N_p$ in the next time where M_{new} and M_{uncrsp} are the summations of the detected new objects' number and the objects not establishing the correspondence relation in all camera views respectively.



(a) The right top sub-image detects a new object

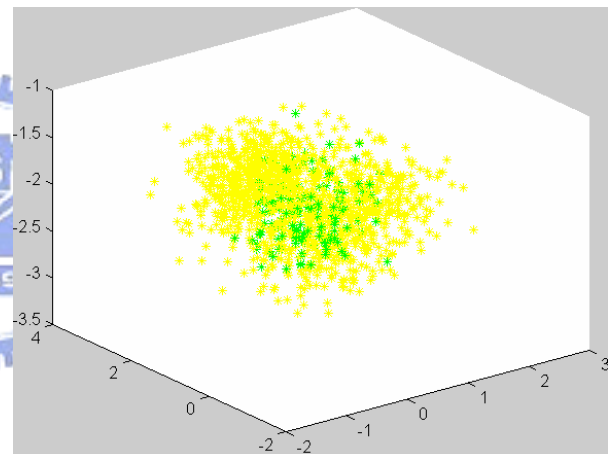


(b) the yellow points are the prediction particles and the green particles are the extra particles for the new entering objects

Figure 3-12 Extra particles for new entering objects



(a) The object in the right bottom sub-image still doesn't find the correspondence relation.



(b) The yellow points are prediction particles and the green points are the particles not establishing the correspondence relation yet.

Figure 3-13 extra particles for the objects still not establishing the correspondence relations.

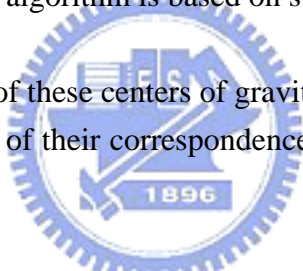
3.4 Information Exchange between 2-D and 3-D

The information of objects in 3-D domain is only obtained by the 2-D tracking results in different camera views. By using the clues of 2-D images, we can find the correspondence relations. The occlusion problem can be solved since we can project the 2-D information to 3-D domain, fuse them together and then back-project them to the 2-D image planes over the system with multiple calibrated cameras.

3.4.1 Construction of Correspondence

Based on the proposed 3-D particle filter, every object in each camera view has its corresponding center of gravity in 3-D domain. By back-projecting this center of gravity to each 2-D image plane, we can find which object it belongs to in all camera views to find the correspondence relation. If some 2-D objects belong to the same 3-D object then the distance between their centers of gravity in 3-D domain must be small. In theory, those centers of gravity will have similar correspondence relation in each camera view after back-projecting them to 2-D image planes. Our correspondence algorithm is based on such idea to develop.

How to decide the similarity of these centers of gravity? We use a score function to compare one center of gravity with another of their correspondence relation in each camera view and this function is written as


$$P(a,b) = \begin{cases} 1, & \text{if } a \neq 0, b \neq 0, \text{ and } a = b \\ -1, & \text{if } a \neq 0, b \neq 0, \text{ and } a \neq b \\ 0, & \text{otherwise} \end{cases} \quad \text{Eq 3-21}$$

, it means that if two centers of gravity map to the same 2-D object in one camera view, their similarity score adds one point and that if they map to different 2-D object, the similarity score minuses one point. But if any one of the centers of gravity doesn't map in that camera view, the similarity score will not change. In the following sections, our proposed method of correspondence will be discussed. We will introduce how to establish a correspondence table at initial state in 3.4.1.1 and how to update and correct the correspondence table after initial establishment in 3.4.1.2.

3.4.1.1 First Construction of Correspondence

At first, we must choose one camera, C_{start} , as the reference camera to decide the order of the 3-D objects based on the 2-D tracking result in that camera view. Then we can decide which index of the 3-D objects the 2-D object in other camera view belongs to. The formula of how to choose C_{start} is written as

$$C_{start} = \arg \max_c \frac{1}{N_0(c)} \cdot \sum_{i=1}^{N_0(c)} \sum_{c'=1}^{N_c} u(I_{32}(X_M(X_0, c, i), c')) \quad \text{Eq 3-22}$$

Since the number of 3-D objects can only be estimated by 2-D images in all camera views, we choose the camera with the most detected objects. If there is more than one camera satisfying this condition, we will calculate which camera's corresponding centers of gravity has the most corresponding objects in all 2-D images. Then we will decide that camera as the reference camera and assign the indicator $I_{23}(\bullet)$ as

$$I_{23}(C_{start}, k) = k \quad , \quad 1 \leq k \leq M_0 \quad \text{Eq 3-23}$$

The correspondence table is defined as $T_0^{(k)}(\bullet)$. For example, $T_0^{(k)}(c)$ gives the number of the ordered 2-D object in c-th camera view by back-projecting k-th 3-D object. So the correspondence table of C_{start} -th camera can be assigned as

$$T_0^{(k)}(C_{start}) = k \quad \text{Eq 3-24}$$

Further, we will define a buffer, $Buff^{(k)}(\bullet)$, saving for the all correspondence relations of the centers of gravity. For example, $Buff^{(k)}(j, 1 : N_c)$ represents a possible correspondence relation of k-th 3-D object in N_c camera views where $1 \leq j \leq Buff_num^{(k)}$. $Buff_num^{(k)}$ represents the number of possible correspondence relations saving in $Buff^{(k)}(\bullet)$ and it is increasing during the process. We put $T_0^{(k)}(C_{start})$ into $Buff^{(k)}(\bullet)$ at first. Then we will classify the 2-D objects in other camera views by the following equation,

$$I_{23}(c,i) = \arg \max_{\substack{k \\ 1 \leq k \leq M_0}} \frac{1}{\text{Buff_num}^{(k)}} \cdot \sum_{j=1}^{\text{Buff_num}^{(k)}} \sum_{c'=1}^{N_c} P(I_{32}(X_M(X_0,c,i),c'), \text{Buff}^{(k)}(j,c')) \quad \text{Eq 3-25}$$

, where $1 \leq c \leq N_c$, $c \neq C_{start}$ and $1 \leq i \leq N_0(c)$.

The idea of this equation is that we can project the center of gravity of i -th object in c -th camera view to each camera view c' then we can get the correspondence relation formulated as $I_{32}(X_M(X_0,c,i),c')$. By comparing $I_{32}(X_M(X_0,c,i),c')$ with $\text{Buff}^{(k)}(\bullet, c')$ one by one, we can find the most possible corresponding 3-D object of i -th object in c -th camera view. If it correspond to k -th 3-D object, then we put this correspondence relation, $I_{32}(X_M(X_0,c,i),c')$, in to $\text{Buff}^{(k)}(\bullet)$ and the size of $\text{Buff}^{(k)}(\bullet)$, $\text{Buff_num}^{(k)}$, also increases. The correspondence table is updated as

$$T_0^{(I_{23}(c,i))}(c) = i \quad \text{Eq 3-26}$$

. After establishing the correspondence relation for each 2-D object, we can finish the initial table and get the indicator, $I_{23}(\bullet)$, from 2-D domain to 3-D domain. However, not every 2-D object can find its correspondence relation in the initial state. For example, if there are more than one objects in a 2-D image corresponding to the same 3-D object or if one 2-D object correspond to more than one 3-D objects, these 2-D objects will try to find their correspondence relations in the next time.

3.4.1.2 Correspondence Update

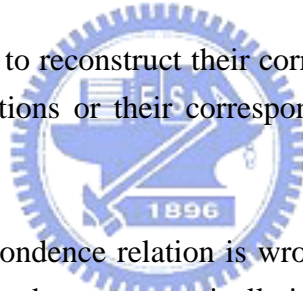
We need to check and update the correspondence table every now and then based on the current 2-D tracking results and the distribution of 3-D particles to keep the correctness of correspondence relations. The update method can be described as

$$I_{23}(c, i) = \arg \max_k \sum_{1 \leq k \leq M_t, c=1}^{N_c} P(I_{32}(X_M(X_t, c, i), c'), T_t^{(k)}(c')) \quad \text{Eq 3-27}$$

The concept of this equation is that by comparing the current center of gravity with each 3-D object's correspondence relation in the correspondence table, we can find the most possible corresponding 3-D object to update the indicator. The correspondence table can also be updated as

$$T_t^{(I_{23}(c, i))}(c) = i \quad \text{Eq 3-28}$$

. Sometimes the 2-D objects need to reconstruct their correspondence relations if they still don't establish the correspondence relations or their correspondence relations are considered as the failure cases.



How to decide if the correspondence relation is wrong is the main point of our algorithm. Our method to correct the correspondence automatically is classified into two parts. The first part is based on the 2-D objects tracking and correspondence and the second part is based on the distribution of 3-D particles. We will discuss as follows,

I. Based on 2-D objects tracking and correspondence results :

✚ One object splits into two objects :

If two objects are too closer at initial state, we will consider them as the same object. Along the time, they will split into two objects then our system will detect this situation and revoke the original correspondence relation. Besides, we will reconstruct these two objects' correspondence relations.

✚ More than one 2-D objects correspond to the same 3-D object :

When more than one 2-D objects in a 2-D image correspond to the same 3-D object, we will revoke their correspondence relations due to this contradiction and then reconstruct the correspondence relations in the next time.

II. Based on the distribution of 3-D particles :

When establishing the objects correspondence relations, we can use a classification function, $C(\bullet)$, to find the corresponding 3-D object for each 3-D particles, and it can be written as

$$C(X_t^{(s)}) = \arg \max_k \sum_{c=1}^{N_c} P(I_{32}(X_t^{(s)}, c), T_t^{(k)}(c)) \quad \text{Eq 3-29}$$

.Each 3-D object has a set of 3-D particles to describe its distribution. We can correct the correspondence relation by observing the projection results in all camera views of these 3-D particles. By calculating the distribution of the 3-D particles falling in a bounding box of 2-D object, we will revoke that 2-D object's correspondence relation if any case occurs as follows:

✚ There is no 3-D particles falling in the bounding box :

$$\text{If } \sum_{s=1}^{N_{ts}} \delta(I_{32}(X_t^{(s)}, c) - i) = 0 \quad \text{Eq 3-30}$$

, where $1 \leq c \leq N_c$, $1 \leq i \leq N_t(c)$, and N_{ts} represents the number of 3-D particles. That means the current 3-D probability distribution doesn't contain the information of this 2-D object which is i -th object in c -th camera view. So we will revoke its original correspondence result.

✚ The incoherent correspondence relations of the original one and the estimated result :

$$\text{If } I_{23}(c, i) \neq \arg \max_k \sum_{s=1}^{N_{ts}} \delta(I_{32}(X_t^{(s)}, c) - i) \cdot \delta(C(X_t^{(s)}) - k) \quad \text{Eq 3-31}$$

Assume the 2-D object correspond to k -th 3-D object, but the most number of 3-D particles inside its bounding box belongs to k' -th 3-D object. Then we will revoke its original correspondence relation due to the incoherence of correspondence results.

3.4.2 Occlusion Handling

When two or more objects in a 2-D image become closer, they will be covered by each other since they lose the information of depth when back-projecting from 3-D domain to 2-D domain. It is so-called occlusion problem which will degrade the 2-D tracking results. In order to solve this problem, we use the multiple cameras to estimate the positions of the moving objects in 3-D domain. Then the 2-D objects under occlusions can be orientated by the 3-D information and the performance of 2-D tracking results will be good and stable.

If the correspondence relations have been established before occlusions, then we can however use the 3-D particles belonging to their corresponding 3-D objects to locate their positions in a 2-D image. Moreover, the occluded objects in a 2-D image will be selected within the same bounding box. This bounding box provides the important information for positioning objects. The occlusion problem can be solved by two steps:

I. Orientation by 3-D particles :

Since the correspondence relations have been established, the location of the occluded object can be obtained by averaging the centers of gravity of other 2-D objects belonging to the same 3-D object. For example, assume the occluded object is i -th object in c -th camera view, then its estimated center of gravity is written as:

$$\mu_occlusion^{c,i} = \frac{\sum_{c'=1}^{N_c} \sum_{i'=1}^{N_i(c')} X_M(X_t, c', i') \cdot W_M(X_t, c, i) \cdot \delta(I_{23}(c', i') - I_{23}(c, i))}{\sum_{c'=1}^{N_c} \sum_{i'=1}^{N_i(c')} W_M(X_t, c, i) \cdot \delta(I_{23}(c', i') - I_{23}(c, i))} \quad \text{Eq 3-32}$$

II. Correction of location and modification of size by using the merging bounding box :

We use the boundary of the merging bounding box to correct the location of the occluded objects. If A and B are both inside the same bounding box and A is near the left side of the bounding box in respect to B, then we assign this left side of the merging bounding box as A's then move A's bounding box here and vice versa. After correcting the locations of the occluded objects, we will adjust the size of the bounding boxes. If the distance of A and B is small, it means that they encounter the sever occlusion so the merging bounding box can even approximate these two objects' bounding box. Then we expand their bounding boxes through a fuzzy adjustment function. The size adjustment of the bounding box is as follows,

$$wid_esti_t^{c,i} = \left(\frac{1}{e^{(a \cdot (d-T))} + 1} \right) \cdot wid^{c,i} + \left(1 - \frac{1}{e^{(a \cdot (d-T))} + 1} \right) \cdot wid_occlusion_t \quad \text{Eq 3-33}$$

,where $wid^{c,i}$ and $wid_esti^{c,i}$ are the widths before and after adjusting, $wid_occlusion_i$ is the width of merging bounding box, d is the distance between the centers of occluded objects and a, T are the parameter for fuzzy function.

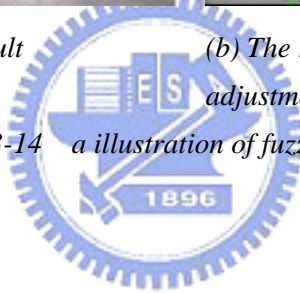
The original result is shown in *Figure 3-14 (a)*, we can find that the performance is not good due to the estimation error and the inappropriate size of the bounding box. The better result is shown in *Figure 3-14 (b)* after fuzzy adjustment. We can find that the locations and the sizes of bounding box are more appropriate than the original result.



(a) The original result

(b) The result after fuzzy adjustment

Figure 3-14 a illustration of fuzzy adjustment



3.5 Overall System Structure

The flow chart of our system is shown in *Figure 3-15*. When our system starts to work, first, we will spray lots of initial particles on the locations where objects may appear in 3-D domain based on the current 2-D tracking results in all camera views. Then we continue to track moving objects in each camera view. By back-projecting all particles from 3-D domain to 2-D domain, the weights of particles will be updated by comparing their projection results with the 2-D tracking results. Condensing all the 3-D particles into the centers of gravity which have the one to one relations to the 2-D objects, we can establish and update the correspondence table and refine the 2-D tracking results. Finally, we can use these centers of gravity to predict the next positions of the 3-D particles. The concept of our system is to exchange the information between 2-D domain and 3-D domain. The objects probability distribution in 3-D domain is updated by passing the 2-D tracking results to 3-D domain and the 2-D tracking results can also be refined by the back-projecting their centers of gravity in 3-D domain. Through this cooperation, our system can accomplish the multiple objects tracking and correspondence over the multi-camera surveillance system.

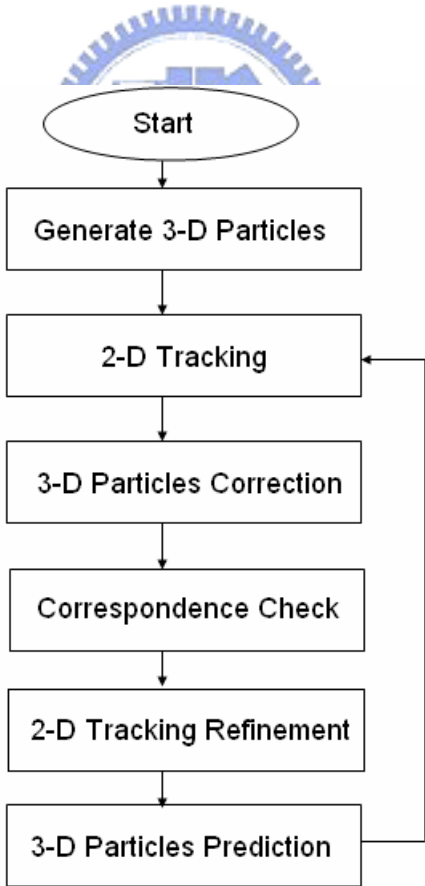


Figure 3-15 the flow chart of our system

Chapter 4 Experimental Results

Our system is developed in an indoor environment with four static and calibrated PTZ cameras mounted on the ceiling. In the following sections, we will demonstrate some experimental results of our algorithm. There are two sequences, seq1 and seq2 demonstrating a two-people case and a three-people case, respectively. The bounding boxes of the detected objects in different camera views t are in the same color for the same 3-D object. Moreover, a yellow bounding box represents an object, whose correspondence relation hasn't been found yet.

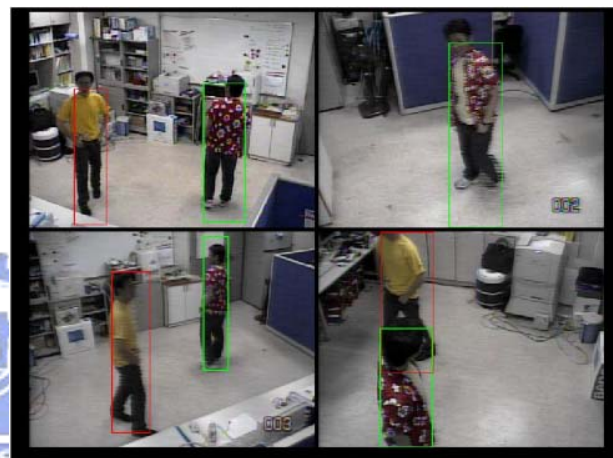
The first experimental result demonstrates the tracking results under occlusions. The location of an occluded people can be estimated by the cooperation of other 2-D objects belonging to the same 3-D people. In *Figure 4-1*, the left column shows the detection results obtained by background subtraction and there is no correspondence relation at all. The right column shows the results of our system. At $t = 603$ the correspondence relations have been constructed. At $t = 606$ two people in the lower-right image become closer and the occlusion problem occurs. At $t = 642$ these two people separate from each other but other two people in the lower-left image start to occlude each other. We find that our system can handle the occlusion problem in a reasonable and efficient way.



Seq1 , $t = 603$



Seq1 , t =606



Seq1 , t = 618



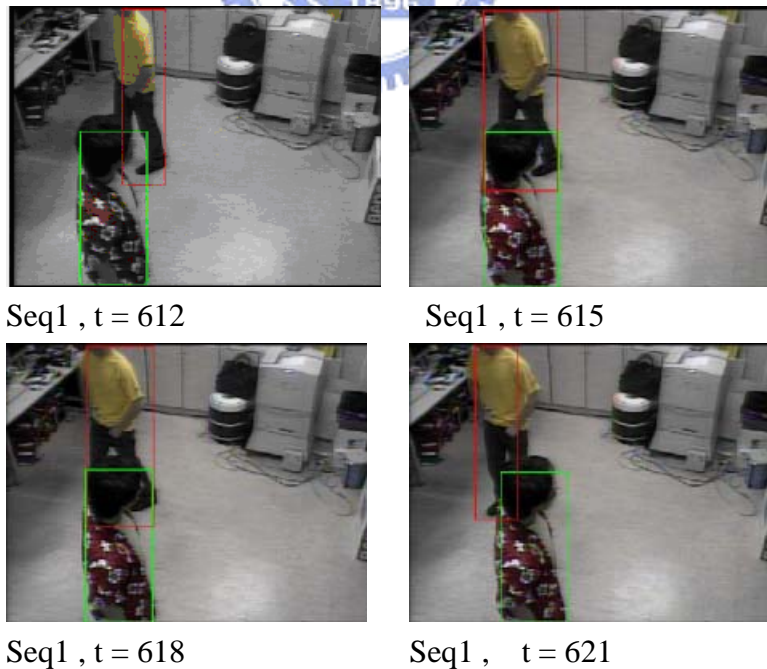
Seq1 , t =642



Seq1 , t = 654

Figure 4-1 the tracking and correspondence results in seq1 under occlusions

When occlusion occurs, the size of the bounding box only keeps the same size of the bounding box before occlusion. However, we can use the information of the merged bounding box to modify the size of the occluded objects' bounding boxes through a fuzzy adjustment. In *Figure 4-2*, two people encounter the occlusion situation from $t = 612 \sim 621$. At $t = 615$ and 618 the occlusion become so severe that the size of each occluded object approximates the size of the merged bounding box.



Seq1 , t = 612

Seq1 , t = 615

Seq1 , t = 618

Seq1 , t = 621

Figure 4-2 modification of size by a fuzzy adjustment (Seq1)

In *Figure 4-3*, it is another occlusion example for three people. We can find that the sizes of the bounding boxes of the occluded people become more similar to the merged bounding box and then recover to the original size when these two people separate from each other.

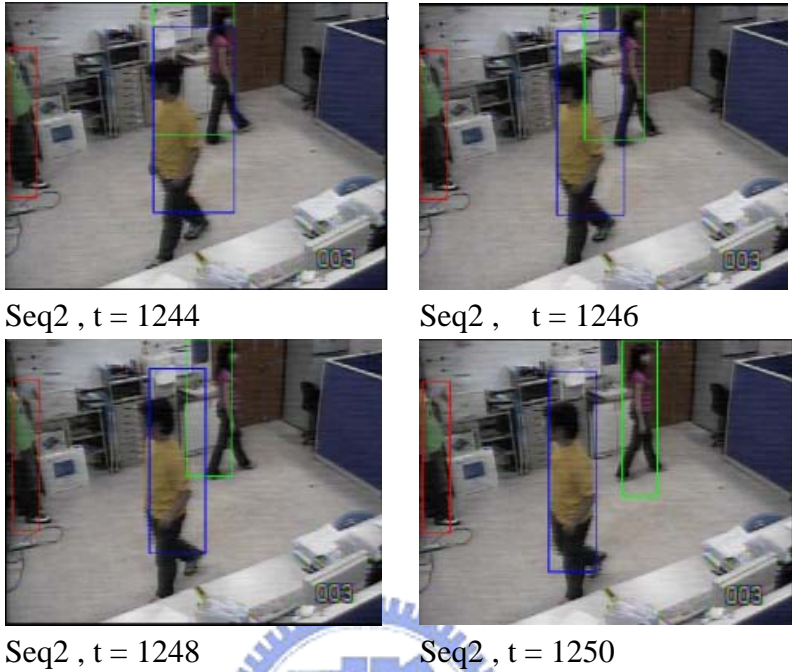


Figure 4-3 modification of size by a fuzzy adjustment (Seq2)



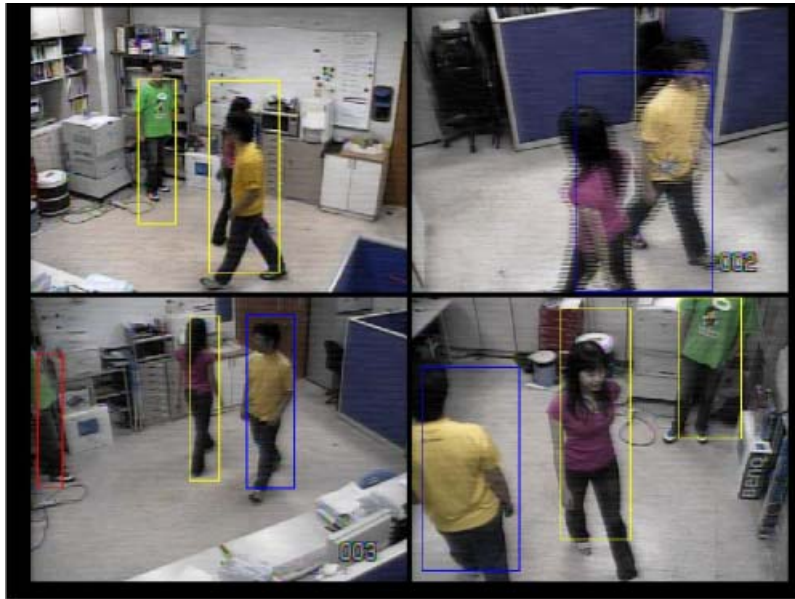
In the following demonstration, it shows that our system can check the correctness of the correspondence relations and revoke the unreasonable correspondence relations automatically. In *Figure 4-4*, at $t = 1196$ the object with the green bonding box in the upper-left image has a wrong correspondence relation. At $t = 1198$ the objects with the same green bounding box in the other camera views are revoked their original correspondence relations and at $t = 1120$ the object in the upper-left image is also revoked its correspondence. At $t = 1202$ most of the objects have established their correspondence relations.



Seq2 , t = 1196



Seq2 , t = 1198



Seq2 , t = 1200



Seq2 , t = 1202

Figure 4-4 the demonstration of self-correction of our system

When a 2-D image detects a new object, our system will find its correspondence relation as soon as possible. In *Figure 4-5*, at $t = 669$ a new object is detected in the lower-right corner. After that, our system will spay some new 3-D particles over the locations where this object may appear. At $t = 675$, we can find that object has established its correspondence relation.

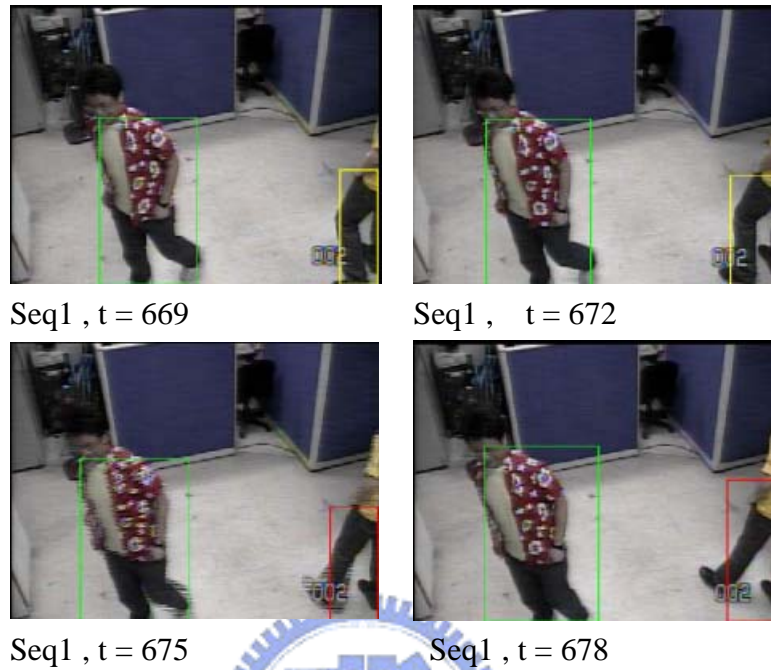


Figure 4-5 Establishment of the correspondence relation for the newly detected object

If two objects were initially thought to be a single object due to occlusion, as shown in the lower-left image in *Figure 4-6* at $t = 675$, and then separate from each other, our system will revoke the original correspondence of the merged object and reconstruct the correspondence relation for each of them, as shown in the lower-right image of *Figure 4-6* at $t = 678$.



Seq1 , $t = 675$



Seq1 , t = 678

Figure 4-6 the reconstruction of correspondence relation when splitting



Chapter 5 Conclusion

We construct a surveillance system with multiple cameras for moving objects tracking and correspondence. Our system uses the 2-D tracking results from different camera views and a lot of 3-D particles with different weights to estimate the probability distribution of moving objects in the 3-D domain. Although Utsumi[23] also uses a similar concept to describe the moving objects as some Gaussian distributions in the 3-D domain, our method is more flexible and reasonable since the distribution of moving objects usually doesn't follow a Gaussian distribution. Through the information exchange between the 2-D domain and 3-D domain, this probability distribution can be correctly updated. This distribution can also refine the 2-D tracking results and help each object in the 2-D domain to find its correspondence relation. Moreover, our system may check and update the correspondence relations automatically based on some proposed constraints. After establishing the correspondence relations, the occlusion problem can be solved easily and these correspondence relations can be used to maintain the correctness of the probability distribution of the moving objects in the 3-D domain.



Reference

- [1] Stauffer, C. and Grimson, W.E.L., “Adaptive Background Mixture Models for Real-time Tracking”, Conference on Computer Vision and Pattern Recognition, Volume 2, pp23-25, June 1999.
- [2] Horprasert T., Harwood D., and Davis L.S., “A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection”, IEEE International Conference on Computer Vision frame-rate workshop. 1999.
- [3] Duque, D., Santos, H., and Cortez, P., “Moving Object Detection Unaffected by Cast Shadows, Highlights and Ghosts”, International Conference on Image Processing, Volume 3, pp11-14, September 2005.
- [4] Fu-Yuan Hu, Yan-Ning Zhang, and Lan Yao, “An Effective Detection Algorithm for Moving Object with Complex Background”, Proceedings of 2005 International Conference on Machine Learning and Cybernetics, Volume 8, pp18-21, August 2005.
- [5] S Dubuisson, “An Adaptive Clustering for Multiple Object Tracking in Sequences in and Beyond the Visible Spectrum”, International Conference on Computer Vision and Pattern Recognition, pp 17-22, June 2006.
- [6] V Nair, and JJ Clark, “An Unsupervised, Online Learning Framework for Moving Object Detection”, International conference on Computer Vision and Pattern Recognition, Volume 2, pp317-324 , June 2004.
- [7] Dongxiang Zhou, and Hong Zhang, “Modified GMM Background Modeling and Optical Flow for Detection of Moving Objects”, IEEE International Conference on Systems, Man and Cybernetics, Volume 3, pp2224-2229, October 2005.
- [8] Wren, C.R., Azarbayejani, A., Darrell, T., and Pentland, A.P., “Pfinder: Real-Time Tracking of the Human Body”, IEEE Transactions on Volume 19, Issue 7, July 1997.
- [9] Stephen J .McKenna, Sumer Jabri, Zoran Duric, Azriel Rosenfeld, and Harry Wechsler , “Tracking Groups of People”, Conference on Computer Vision and Image Understanding, Volume 80, October 2000.
- [10] Paragios N. and Deriche R., “Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, Issue 3, pp. 266-280, March 2000.
- [11] D. Roller, K. Daniilidis and H. H. Nagel, “Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes”, International Journal of Computer Vision, pp257-281, October 1993.
- [12] C. J. Li, “A Study of Change Detection and Motion Tracking on an Active Surveillance Camera”, Master thesis, June, 2003.
- [13] R. Polana and R. Nelson, “Low Level Recognition of Human Motion,” Proceedings of IEEE Workshop on Motion of Non-Rigid and Articulated Objects, pp. 77–82, 1994.

- [14] C. A. Pau and A. Barber, "Traffic Sensor Using a Color Vision Method," Proceedings of SPIE—Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS, Vol. 2902, pp. 156–165, 1996.
- [15] B. Schiele, "Voxel-free Tracking of Cars and People Based on Color Regions," Proceedings IEEE International Workshop Performance Evaluation of Tracking and Surveillance, pp. 61–71, 2000.
- [16] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A Real-time Computer Vision System for Vehicle Tracking and Traffic Surveillance," *Transportation Res.: Part C*, vol. 6, no. 4, pp. 271–288, 1998.
- [17] J. Malik and S. Russell, "Traffic Surveillance And Detection Technology Development: New Traffic Sensor Technology Final Report", January 1, 1997. California Partners for Advanced Transit and Highways (PATH). Research Reports: Paper UCB-ITS-PRR-97-6.
- [18] T. J. Fan, G. Medioni, and G. Nevatia, "Recognizing 3-D objects using surface descriptions", *IEEE Transactions of Pattern Recognition Machine Intelligence*, Vol. 11, pp. 1140–1157, Nov. 1989.
- [19] Comaniciu D., Ramesh V. and Meer P., "Kernel-based Object Tracking", in *Proceedings on IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. 25, No. 5, pp. 564-575, May 2003.
- [20] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-Based Probabilistic Tracking," *Proceedings of European Conference Computer Vision*, Vol. 1, pp. 661-675, 2002.
- [21] Krumm J., Harris S., Meyers B., Brumitt B., Hale, M. and Shafer S., "Multi-Camera Multi-Person Tracking for EasyLiving", *Third IEEE International Workshop on Visual Surveillance*, pp3-10, July 2000.
- [22] Anurag Mittal and Larry S. Davis, "M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene Using Region-Based Stereo", *International Journal of Computer Vision* on Volume 51, Number 3, February, 2003.
- [23] Utsumi A., Mori H., Ohya J. and Yachida M., "Multiple-Human Tracking using Multiple Cameras", *Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp14-16, April 1998.
- [24] Black, Jamesa, Ellis and Tima, "Multi Camera Image Measurement and Correspondence", *Measurement* Vol. 32, Issue: 1, pp. 61-71, July, 2002.
- [25] S. Khan, O. Javed, and M. Shah, "Tracking in Uncalibrated Cameras with Overlapping Field of View," *Proceedings of Performance Evaluation of Tracking and Surveillance 2001*, (with CVPR 2001), Dec. 2001.
- [26] J. Black and T. J. Ellis, "Multi Camera Image Tracking", *Technical Report No. MVG99/1*, Dept. EEIE, City University, London, 1999.
- [27] I. H. Chen and S. J. Wang, "A Vision-based Approach To Extracting The Tilt Angle and Altitude of A PTZ Camera", *Proceedings of SPIE*, 2006.
- [28] Arulampalam, M. S., Maskell, S., Gordon, N. and Clapp, T., "A tutorial on particle filters for

online nonlinear/non-Gaussian Bayesian tracking”, IEEE Transactions on Signal Processing, Volume 50, pp 74 – 188, Issue 2, Feb. 2002.

