2006

Feature-based tracking of multiple people for intelligent video surveillance.

Mohammad Ahsan Ali
University of Windsor

FEATURE-BASED TRACKING OF MULTIPLE PEOPLE FOR INTELLIGENT VIDEO SURVEILLANCE

by

Mohammad Ahsan Ali

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada

2006

Your file Votre référence
ISBN: 978-0-494-17022-9
Our file Notre référence
ISBN: 978-0-494-17022-9

ABSTRACT

Intelligent video surveillance is the process of performing surveillance task automatically by a computer vision system. It involves detecting and tracking people in the video sequence and understanding their behavior. This thesis addresses the problem of detecting and tracking multiple moving people with unknown background. We have proposed a feature-based framework for tracking, which requires feature extraction and feature matching. We have considered color, size, blob bounding box and motion information as features of people. In our feature-based tracking system, we have proposed to use Pearson correlation coefficient for matching feature-vector with temporal templates. The occlusion problem has been solved by histogram backprojection. Our tracking system is fast and free from assumptions about human structure. We have implemented our tracking system using Visual C++ and OpenCV and tested on real-world images and videos. Experimental results suggest that our tracking system achieved good accuracy and can process videos in 10-15 fps.

DEDICATION

To my parents,

who enlighten my heart with knowledge

and my wife,

who always stays beside me in need

ACKNOWLEDGEMENTS

I like to take this opportunity to thank my thesis supervisor, Dr. Bubaker Boufama, for his caring, guidance and cooperation throughout my graduate studies. This work could not have been achieved success without his help and support. My heartiest gratitude goes to Dr. Nader Zamani, Department of Mechanical Automotive and Material Engineering and Dr. Imran Ahmad, School of Computer Science for being in the committee and spending their valuable time and to Dr. Ahmed Tawfik, School of Computer Science for serving as the chair of the defense.

I also would like to express my sincere gratitude to my learning peers who played an important role along the journey, as we engaged in fruitful discussion in solving various challenges we faced.

I extend my deepest appreciation to Dr. Dmitry Gorodnichy from National Research Council of Canada (NRC-IIT) for his valuable suggestions and advice regarding OpenCV.

Finally, I wish to express my sincerest gratitude to my wife for her understanding and cooperation all the way.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This chapter gives an overview of this thesis. First, the origin of computer vision and its relation to other fields is discussed with several examples. Then, a brief overview of computer vision and its high and low-level activities are illustrated. Following this, the motivation of this thesis and problem statement is introduced. Finally, the chapter ends outlining the layout of the rest of this thesis.

1.1 The Origin of Computer Vision

Computer Vision is a combination of multi-disciplinary fields. Even though some earlier works exists, it was only late 1970's that this area was explored when computers could manage to process a large amount of data sets such as images [wiki3]. Figure 1.1 shows a relationship between Computer Vision and various other fields. Consequently, solution of a computer vision problem is very task-specific and requires applying methods from different fields. For example, some of the learning techniques used in Computer Vision are based on techniques developed in Artificial Intelligence (AI). Some of the work in Computer Vision deals with light sources and image formation techniques by camera, which requires concept of physics. Sometimes Computer Vision tries to imitate biological vision system and requires study of neurobiology. Finally, many methods of Computer Vision are based on statistics, optimization and geometry. In fact, Computer vision, Image processing, Image analysis, Robot vision and Machine vision are closely related fields and has significant overlap in terms of techniques they apply and applciations they cover. Computer Vision is considered as a subfield of AI, but it is widely accepted that Computer Vision originated from AI and Robotics [O'Connell02]. In robotics, robots gather information about the scene by senses and perceive the information to perform a specific task. One such sense people rely on heavily is vision. Consequently, the need for processing visual information has led to evolve the field of Computer Vision.

People sometime confuse computer vision with computer graphics. Computer graphics deals with modeling a scene from its geometry, so information about the objects is known over there. On the other hand, computer vision deals with images and tries to extract information about geometry of objects from the image.

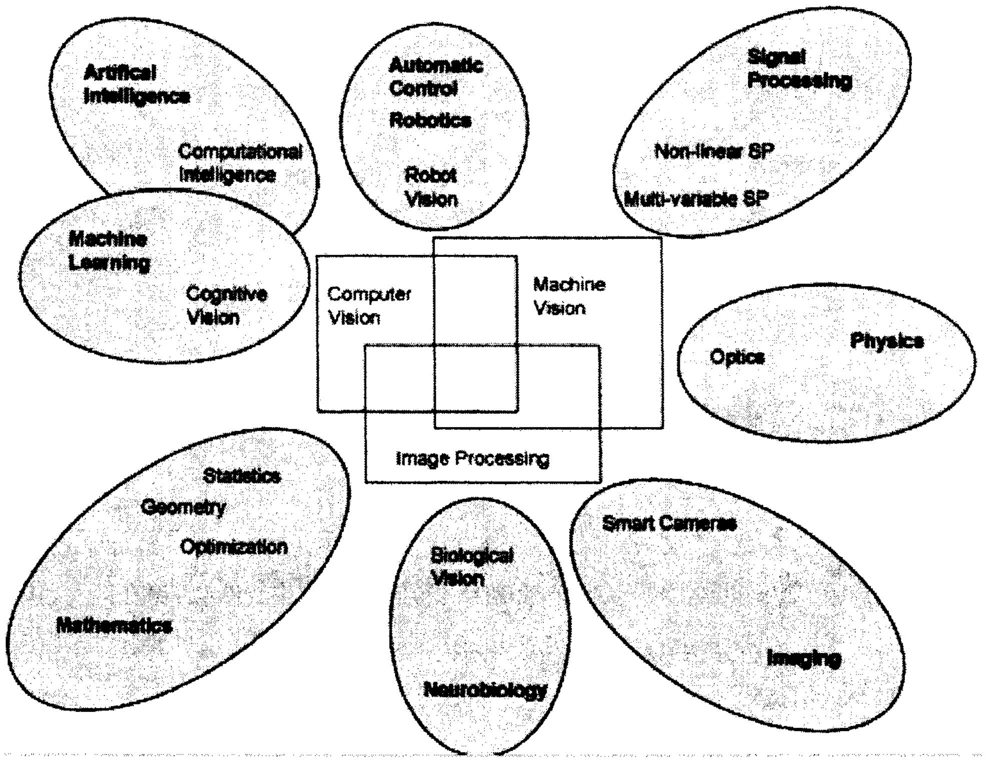


Figure 1.1: Relation between computer vision and various other fields [Wiki3]

1.2 Overview of Computer Vision

Computer Vision is the study of understanding content of image by computer or extracting information from visual stimuli. The extracted information can be used by human operator or to control processes. The input to a computer vision system is a digital grey scale or color image. Sometime video can be input to the system, but images are extracted from video by hardware (i.e. frame grabber) or software (i.e. API) before any processing.

According to Faugeras, Computer Vision tries to answer the following questions [Faugeras93]:

1. What information should be extracted from the outputs of the visual sensors?
2. How is this information to be extracted?
3. How should this information be represented?

4. How must the information be used to perform specific task?

The first three questions deal with low-level aspects of computer vision. Commonly used low level information includes edges, regions, texture, and depth. The final question deals with high-level vision and utilizes low-level information in applications like robotics, motion tracking, surveillance etc.

1.3 Motivation

In the last decade, a new branch of Computer Vision has emerged, called intelligent video surveillance. This domain of computer vision that analyzes images involving human is sometimes called "Looking at People" [Gavrila99]. Research in this area got considerable interest to researchers due to its numerous applications. Nowadays, video camera is cheap and widely used in public and private places for security surveillance. Fortunately, the increased processing capability of computer now allows processing a large amount of image within a short time. As the infrastructure is there, an efficient and accurate intelligent video surveillance will create enormous business opportunities. Specially, human motion analysis is receiving increasing attention from computer vision researchers [Aggarwal99]. Such a system is not developed yet in the industry, as the challenges associated with different levels are not solved yet properly. They are still an open problem and currently an active research area. This motivated me to conduct a research on a state of the art technology.

1.4 Overview

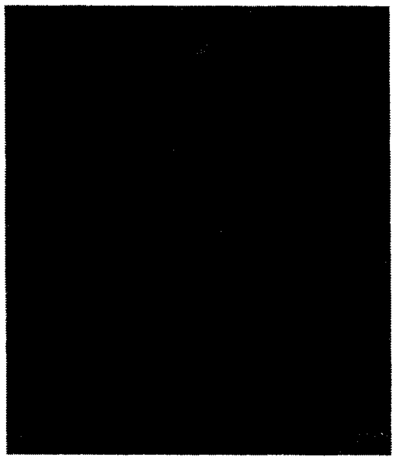
This thesis addresses the problem of detecting and tracking multiple people in complex environment with unknown background. The thesis is organized in six chapters. Chapter 2 gives a background about intelligent video surveillance and its components. In particular, the challenges of tracking are discussed in this chapter with examples. Chapter 3 presents an overview of previous work on detecting and tracking people. Chapter 4 describes a feature-based framework for tracking multiple people and solving occlusion problem. Some experimental results and its analysis are presented in chapter 5. Finally, the thesis concludes in chapter 6 with some future direction of current work.

CHAPTER 2
BACKGROUND OF VIDEO SURVEILLANCE

This chapter gives a background of intelligent video surveillance. First, the term video surveillance and intelligent video surveillance is explained with some potential applications. Then, the components of intelligent video surveillance are elaborated with examples. Finally, the major challenges in developing an intelligent video surveillance are discussed.

2.1 Video Surveillance

Video surveillance, also known as visual surveillance, is the process of monitoring people or object's (i.e. car) behaviour in a scene for conformity to expected norm [Wiki4]. The conventional approach of video surveillance requires capturing indoor or outdoor scene information by video capturing devices (i.e. closed-circuit television (CCTV) camera), transmitting video data and displaying in a terminal or a television, monitoring by a human observer and optionally recording video information in a storage medium. Nowadays, there is a widespread deployment of video capturing devices in both public and private areas, such as banks, airports, casinos, shopping areas etc [Xu04].



(a) Surveillance cameras

(b) The men believed to have been responsible for the 7 July attacks on London

Figure 2.1 Video surveillance [Wiki4]

Recently security has become a big issue for organizations. Specially after several terrorist bomb attacks in different places of the world, the organizations have increased their security by strict authentication process at the entrance and increasing number of surveillance cameras in all over the places. Deploying security cameras is not a big problem for them, as they are cheap. But the main problem is the huge cost associated with employing human operators to monitor activities in the scene. Moreover, they make mistakes due to fatigue or negligence. Another limitation is that, it is difficult for one operator to monitor several places at the same time. That is why it has become urgent to develop an intelligent system that can perform the surveillance task automatically. This kind of system will create enormous business opportunities for organizations.

The purpose of automated video surveillance system is not to replace human eyes with cameras, but to reduce human involvement significantly and assist human operators for better monitoring (to accomplish the entire surveillance task as automatic as possible [Hu04]). Human eye has a superior surveillance capability, which is unparallel to any other artificial monitoring system. With the help of computer-based video surveillance system, one particular operator will be able to monitor several places simultaneously.

2.2 Intelligent Video Surveillance

An intelligent video surveillance system should be capable to detect movement of objects in the scene, identify type of moving objects, track the moving objects in the scene, detect unusual events and warrant the security personnel to take preventive measures. The term "Automatic Video Surveillance" and "Smart surveillance" is used interchangeably with intelligent video surveillance. Intelligent video surveillance requires computing facility to capture frames from video, process video images and take intelligent decision based on the logic provided in the software. Intelligent video surveillance in dynamic scenes (both in indoor and outdoor environment) has become an active research area due to its applicability and the challenges associated with it [Hu04].

2.3 Applications of Intelligent Video Surveillance

Intelligent video surveillance has a wide range of potential applications, especially involving people and vehicles in the scene. Video surveillance can be considered as a subclass of video processing. There are other subclasses of video processing; they are control and analysis [Moeslund01]. Some of the applications of video processing are described here.

1. Measuring traffic flow to crowd flux statistics

Video surveillance can be used to categorize objects in the scene into human and vehicles and then monitor expressways and junction of roads to analyze traffic flow and congestion [Beymer97]. It can also be used to compute flux of people in important public areas such as shopping mall and subways. This information is helpful for management of people [Hu04].

2. Tracking people and interpreting their behavior

Video surveillance can be used to track a group of people in the scene and analyze their movements such as walking, running, dancing etc [Fujiyoshi98]. It can also be used to recognize patterns of human activity [Stauffer00] and behavior such as stalking [Niu04]. There is a popular and widely used system for tracking people and interpreting their behavior, called 'Pfinder' (Person finder) [Wren97]. W^4 is another popular surveillance system for tracking and understanding human activity [Haritaoglu98]. Recently, much research has been focused on *activity analysis* in videos [Gao04].

3. Understanding movement of human body

In kinesiology, the goal is to develop model of human body that explains how it functions mechanically and how one might increase its movement efficiency. A typical procedure involves obtaining 3D joint data, performing kinematic analysis and computing the corresponding forces and torques for a movement of interest [Gavrila99]. Video surveillance can be used for developing personalized training system and helping athletes to understand and improve their performance [Gavrilla99]. In choreography, there has been long-term interest in devising high-level descriptions of human movement for the notion of dance, ballet and theatre. Computer vision can be used to analyze movement of human body parts.

4. Understanding sign language

During mid 90s, some useful work has been done on gesture driven control and interpreting sign-language [Starner95]. A system was developed by Cui et al. called SHOSLIF-M to learn and recognize hand gestures from intensity image sequences using most discriminating features (MDF) [Cui95]. Other gesture-based controlled systems include ALIVE [Wren97] and SURVIVE [Wren97]. Sign language can be used by people who can not speak to communicate with computer.

5. Detecting robbery at bank or secured places

Applications of intelligent video surveillance in security range from simply detecting presence of a human in secured places (i.e. Museum) to people carrying guns or other kinds of weapons in the scene [Collins00]. It can be used to recognize a classic holdup position of armed robbery by analyzing silhouettes [Dever02].

6. Identity checking (biometrics)

Video surveillance can be used for access control in secured places such as military bases, government and corporate offices etc. Identity of a person can be verified by different biometric features such as height, facial appearance and walking gait [Chellappa04]. Another possible application of person identification is in the playground for identifying a player and type of his actions at a distance [Efros03].

7. Suspect identification

Using intelligent video surveillance, criminals or suspects can be identified in public or suspected places by analyzing biometric features of that suspect. This kind of system will be of great interest to police department.

8. Miscellaneous applications

Other applications of intelligent video surveillance includes preventing theft at parking lot and shopping areas [Xu04], patrolling national borders or airports [Collins00a], detecting suspicious appearance or camouflage [Tankus01], virtual reality, content-based indexing of videos [Gavrilla99], predict unusual event etc.

2.4 Components of Intelligent Video Surveillance System

For intelligent video surveillance, every frame of the video has to go through several levels of processing. Usually, following four levels of processing is applied on every frame:

1. Foreground object segmentation/Change detection
2. Classification of objects (i.e. car, human)
3. Tracking multiple objects (human or its body parts, vehicles etc.)
4. High-level processing to understand unusual events or human activities

In addition to these, some intermediate stages are required, such as noise removal, blob construction, shadow removal etc. Each of these stages is considered as an independent research area due to their difficulty of problems.

For real-time processing, all of the above mentioned stages should be processed several times within a second. If we consider processing the video in 10fps, then a very small amount of time is allocated for tracking. The accuracy of tracking is very much dependent on accuracy of foreground object segmentation, which largely affects the accuracy of overall surveillance task.

2.4.1 Foreground Objects Segmentation

Foreground objects in a scene are those objects that appear and/or move in front of a static background. All the pixels in a video scene can be classified into two groups; they are pixels belong to foreground objects and pixels belong to background objects. Foreground object segmentation requires low-level (pixel-wise) processing of video frames to distinguish foreground pixels from background pixels. Figure 2.4 shows two foreground objects (the person and the moving car) moving in front of a static background.

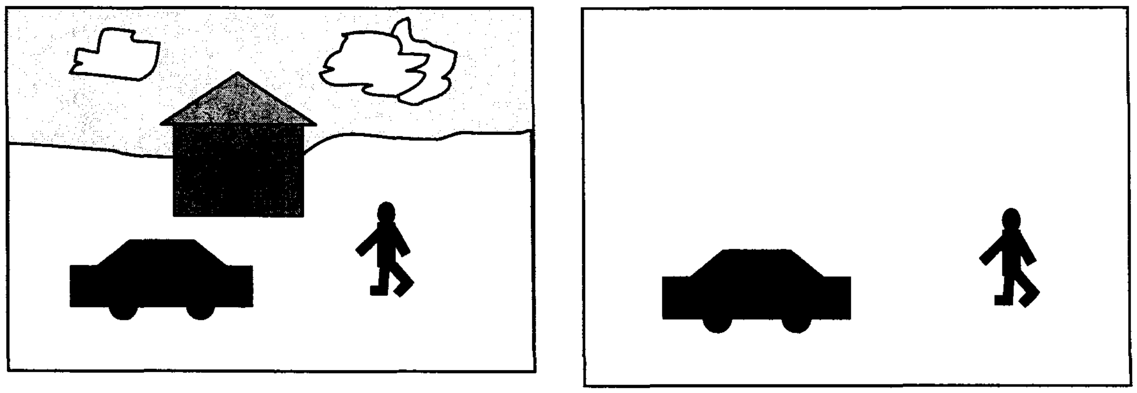
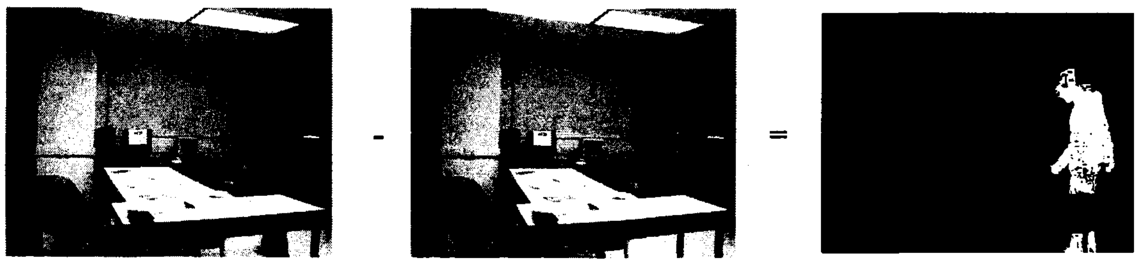

(a) Video frame or Image (b) Foreground pixels

Figure 2.2: Foreground pixels separated from background pixels [vsam]

There are three conventional approaches to identify foreground pixels; they are background subtraction, temporal differencing and optical flow.

Background Subtraction

Background subtraction requires previously stored background image. Motion in front of a background can be detected by subtracting background image from the current image. If the color of a pixel in the current image is different by certain amount than the color of that pixel in the background, then the pixel is considered as foreground pixel. Otherwise, the pixel is considered as background pixel.



(a) Frame extracted from video (b) Background model (c) Result of background subtraction

Figure 2.3: Background subtraction

The value of certain amount is called Threshold (T). Mathematically, the background subtraction can be written as:

$$FG_{x,y} = \begin{cases} 1, & if \mid I_{x,y} - BG_{x,y} \mid > T \\ 0, & otherwise \end{cases}$$

Here, $I_{x,y}$ is the color of a pixel at 2D (x, y) location in the current image and $BG_{x,y}$ is the color of the same pixel in the background.

Temporal Differencing

Temporal differencing is sometimes called frame differencing. In temporal differencing, the color intensity of a pixel in two or more consecutive frames (separated by constant time) is compared for change detection. If the change is more than a threshold, then that pixel is considered as foreground; otherwise it is considered as background. Mathematically, it can be written as:

$$FG_{x,y} = \begin{cases} 1, & if \mid I^{t}{}_{x,y} - I^{t-1}{}_{x,y} \mid > T \\ 0, & otherwise \end{cases}$$

Here, $I^{t}{}_{x,y}$ is the intensity of a pixel at location (x, y) in the current frame captured at time t . Sometimes more than two consecutive frames are considered to increase accuracy of detection.

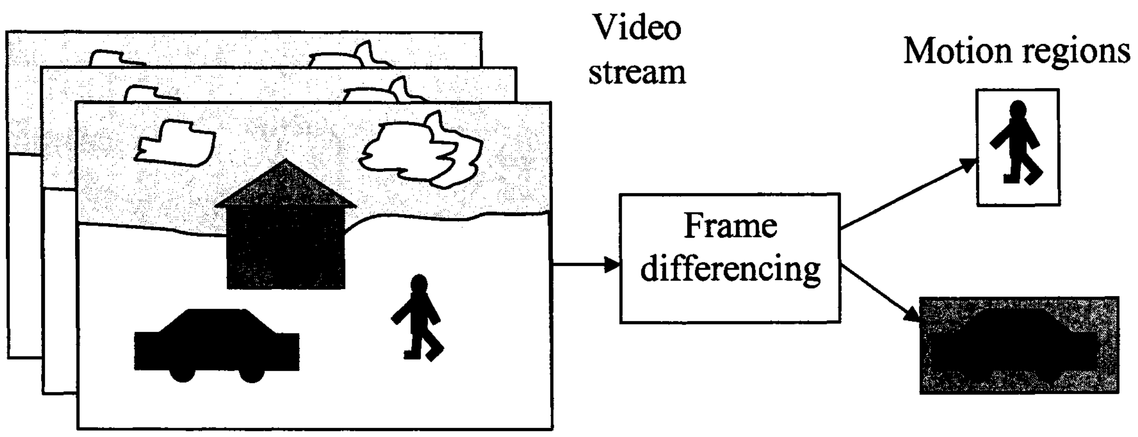


Figure 2.4: Frame differencing

Optical Flow

Optical flow represents the motion of objects in the scene as vectors originating or terminating at pixels in a digital image sequence. It is the apparent motion of the brightness pattern in an image and tells us how the position of the image point changes over time.

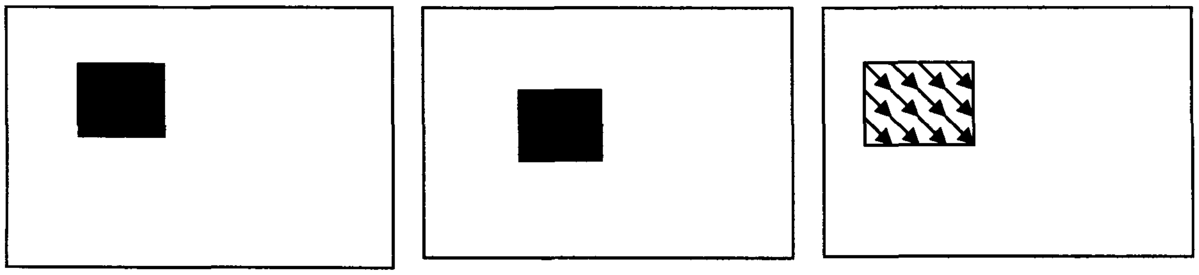
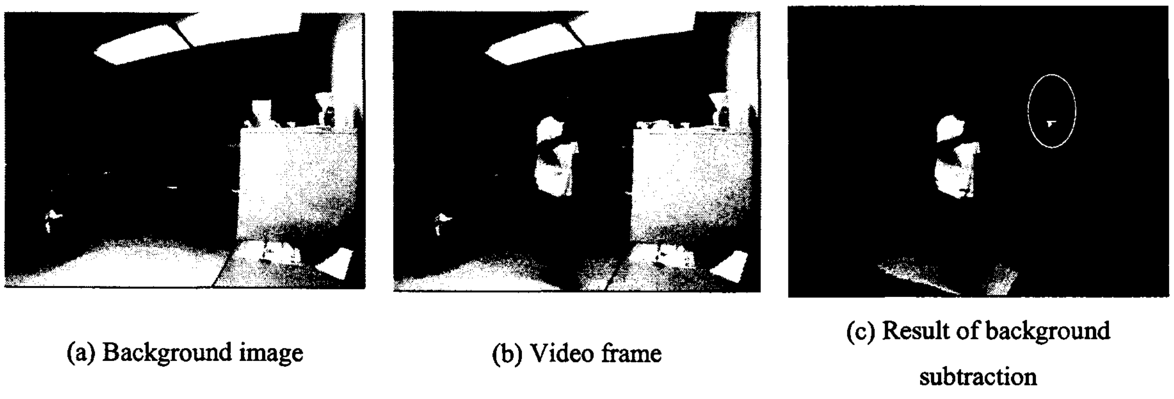


Figure 2.5: Optical flow

2.4.2 Challenges in Foreground Object Segmentation

Detecting foreground regions in the scene and separating them from background image is a challenging problem. In the real world, some of the challenges associated with foreground object segmentation are illumination changes, noise, shadow, camouflage in color, background and foreground apertures etc [Toyama99].

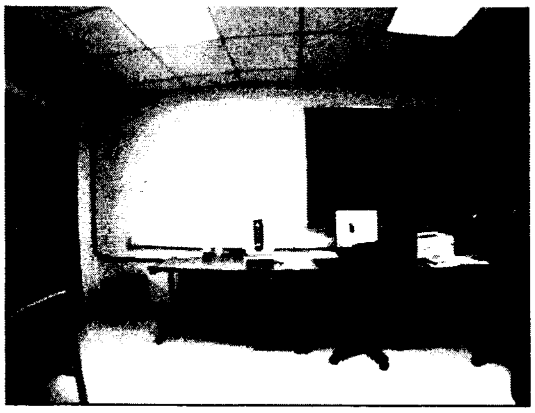


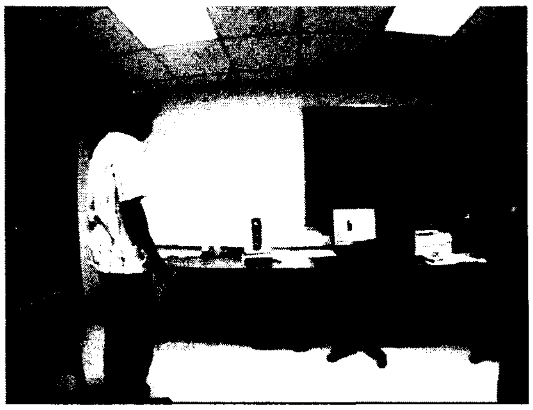
(a) Background image (b) Video frame (c) Result of background subtraction

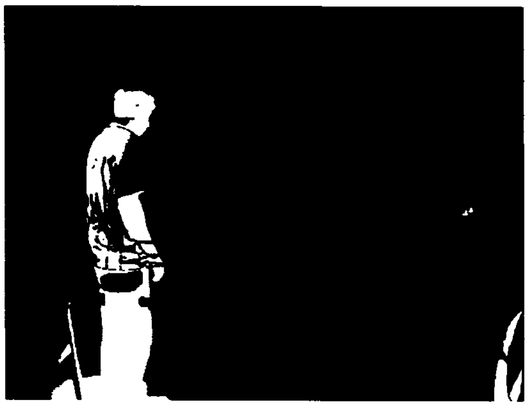
Figure 2.6: Example of cast-shadow and background aperture

1. If a background object starts moving (i.e. a parked car starts moving), then color of pixels in that region does not match with the color of background and as a result considered as foreground pixel. In other words, moving background objects creates an aperture in the scene. Also, when a foreground object stops moving (i.e. a car is

parked) for a long time, it is considered as foreground objects, although it should be considered as part of background after certain time. Figure 2.6 and 2.8 shows some examples of background aperture.


(a) Background


(b) A video frame


(c) Background subtraction with high value for T (50 in this case) creates foreground aperture

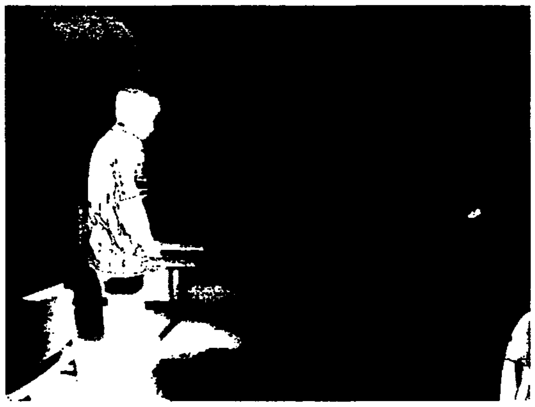

(d) Background subtraction with small value for T (10) includes self shadow and noises

Figure 2.7: Result of background subtraction with different value for T

2. Finding the value of T is a difficult problem. If we consider a smaller value for T, then some background pixels will be considered as foreground pixels, which are basically noise. If we consider a higher value for T, then some foreground pixels will be considered as background pixels, which results inaccurate foreground object segmentation. In this case, the foreground region might consist of apertures inside and/or get segmented into several parts. If foreground region is not segmented properly, then it will create problem during tracking and will degrade the performance of overall surveillance task. For a detail description problems in

foreground region segmentation, see [Toyama99]. Figure 2.7 shows an example of consequence for variation of T.

The main reason behind this problem is that, the color of all the pixels of a foreground object does not change equally due to different lighting conditions. Also, the same value of T will be inappropriate for other scenes and/or for the same scene but at different time of the day.

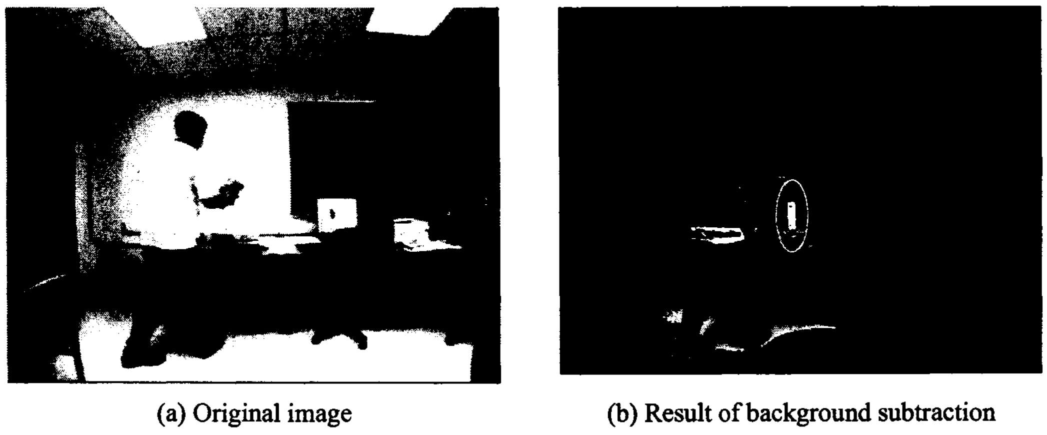

(a) Original image (b) Result of background subtraction

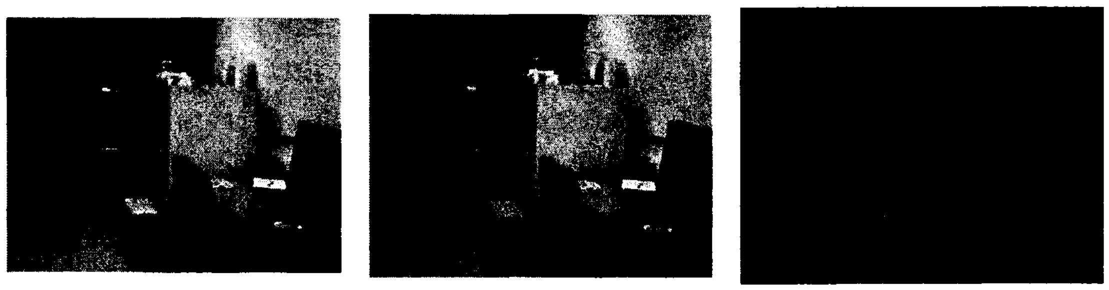
Figure 2.8: Example of cast shadow, saturation of light and background aperture

3. Shadow is another big problem in foreground object segmentation. Shadow region gets included in the foreground region, because intensity of pixels in that region differs with background. Shadows are two types: cast shadow or highlights and self shadow. The cast shadow is the shadow that occurs on a surface as a result of something between the light source and the surface. The self shadow is the shadow that occurs in foreground region due to variation of lighting condition. During the process of cast shadow removal, some of the self shadow regions might get deleted, which is not desirable [Xu04]. Figure 2.8 shows an example of cast shadow. It also shows that saturation of light deletes some part of foreground object.
4. Another major problem in foreground segmentation is camouflage. When a foreground region has similar color and intensity as background region, then it is really difficult to identify foreground pixels and foreground apertures are created inside the foreground region. Figure 2.9 shows that the leg is split into parts due to similarity with background color.

(a) Background image (b) Video frame (c) Result of background subtraction

Figure 2.9: Example of noise and split of foreground region due to camouflage

5. There are some other challenges associated with foreground object segmentation, such as waving trees in the background, light switching between on and off, bootstrapping, haziness of images, saturation of color, frequent variation of color of a pixel due to poor quantization by video camera etc. Bootstrapping is the availability of an ideal background without any moving object, which is not possible sometimes in some busy areas.



(a) Background image (b) Video frame (c) Result of background subtraction

Figure 2.10: Example of self-shadow, cast shadow, noise and camouflage

2.4.3 Noise Removal

Noise should be removed from foreground region to avoid unpredictable result. Noise can be removed in two levels: pixel level and region level. In the previous discussion, we have seen that some isolated noisy pixels are observed all over the places in the resultant foreground region. These pixels are generated due to small variation in lighting condition or variation of quantization of color by camera. In pixel level, noise can be removed by morphological operations such as opening (erosion followed by dilation) and closing (dilation followed by erosion). In region level, noise can be pruned after blob construction. Due to large variation of lighting condition, a group of noisy pixels might be observed in some places, which constitute a blob. These blobs can be removed by size thresholding, consistency constraints and shape constraints. Size thresholding ensures that blobs having size smaller than a threshold removed from the final list of blobs. Some spurious blobs appear in the scene temporarily and go away in the next frame. The consistency constraint can identify these blobs by checking their consistency for more that one frames. The shape constraints check the shape of a blob with some shape model of human. Most cases, the noisy blobs have random shapes and can be identified easily. Sometime noise is observed inside the foreground region, which create foreground aperture. It can be filled out by color of neighboring pixels, but this approach has the danger of filling a foreground region which really has an aperture (i.e. a human standing with hand holding the heap).

2.4.4 Blob Construction

Blob is a spatially coherent group of pixels clustered together that represent a moving object or person in the scene. Blobs can be constructed by checking neighbors of a foreground pixel. Usually 8-connectivity or 4-connectivity analysis is performed to identify blobs. Figure 2.11 shows that 3 blobs are identified in the scene.

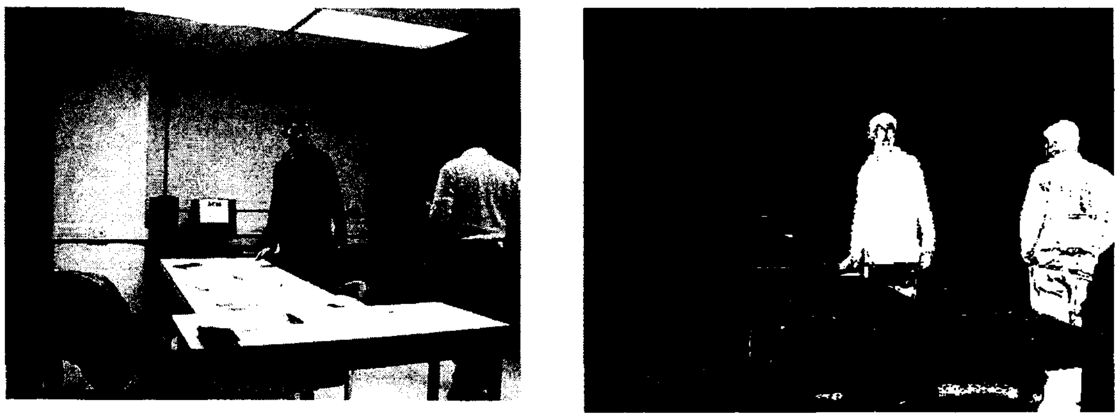


Figure 2.11: Group of foreground pixels form a blob

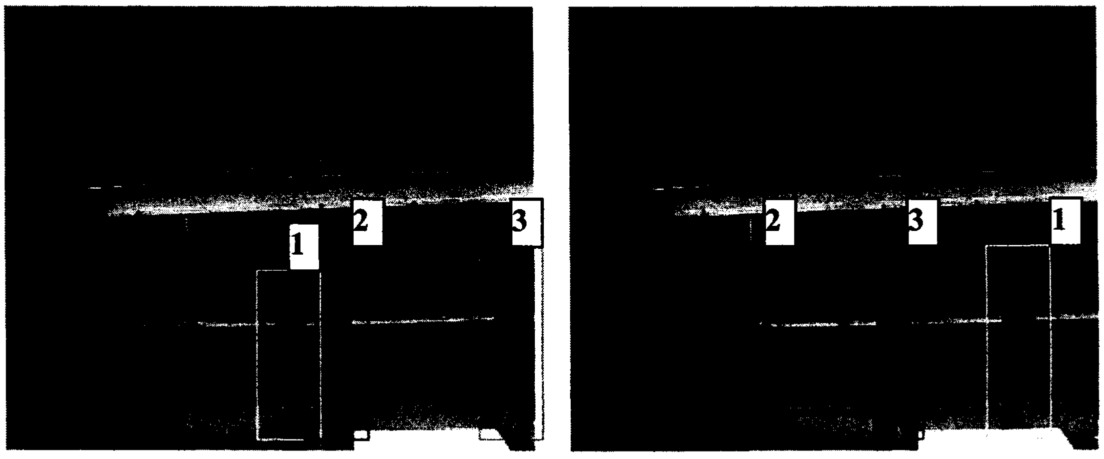
2.4.5 Classification of Objects

In this level, the type of object (i.e. human and car) is determined and focus of attention is given based on the decision. Classification of foreground objects is an independent research area and is not discussed in the thesis. The challenges associated with this stage are:

- Heuristics about size, width and height of objects (i.e. human) does not work all the time. The size of a child and the size of an adult do not match.
- Usually human body is considered as a no-rigid object and vehicles are considered as rigid objects. The classification based on rigidity does not work all the time (i.e. a handicapped person moving on his moving cart).
- The size and shape of objects change with rotation and zooming.

2.4.6 Tracking Multiple Objects

Tracking multiple moving objects in cluttered video sequences is considered as a difficult problem to solve for automated video surveillance. Tracking multiple human is the building block of understanding high-level events and complex actions such as detection of walking, running, dancing, stalking etc. The problem of tracking can be stated as determining the appearance and location of a particular object in the sequence of frames. In other words, tracking people in the current frame is answering the question "Who is who?" [Haritaoglu99].



(a) Frame 1 (b) Frame 5

Figure 2.12: Tracking 3 people

Figure 2.12 shows a result of tracking three people in a classroom. Each person is shown within a bounding rectangular box and given a label during tracking. The trajectory of movement of a person is of importance to the higher level analysis of human activity.

Challenges in Tracking

The challenges associated with tracking are:

1. Similarity of people in shape, color and size
2. Proximity of other people
3. Occlusion
    a. Occlusion by other moving people
    b. Occlusion by background component
4. Maintenance of appearance or disappearance of objects in the scene (which changes total number of objects being tracked)
5. Split of foreground region due to segmentation error and partial occlusion

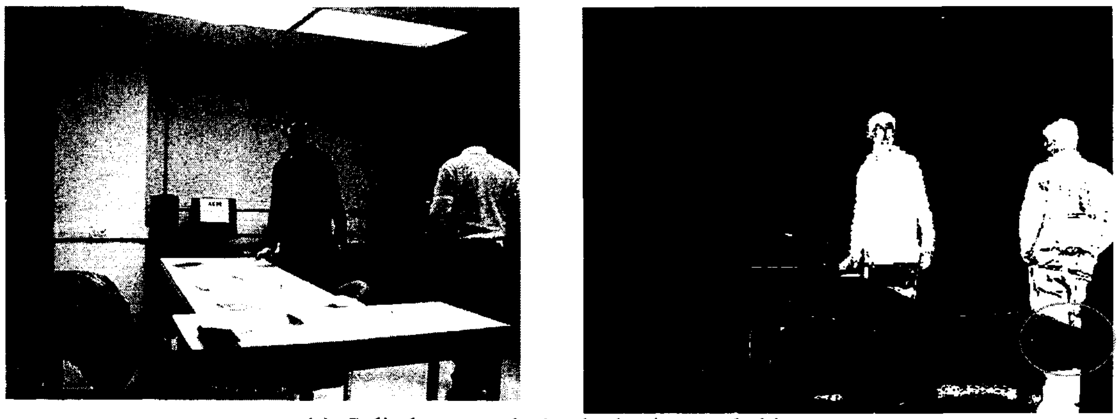
Following figures explain some challenges of tracking. Figure 2.13 shows different types of occlusion. Figure 2.14 shows different reasons of split in the foreground region.

(a) One moving object occludes the other

(b) Occlusion merge two blobs into a single blob

(c) Occlusion by background object

Figure 2.13: Different types of occlusion


(a) Split due to occlusion by background object

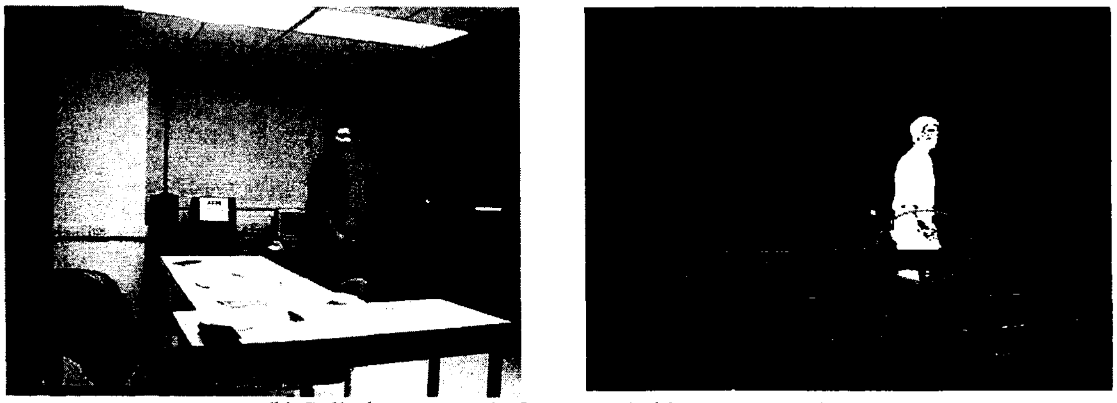

(b) Split due to error in foreground object segmentation

Figure 2.14: Different reasons of split in foreground region

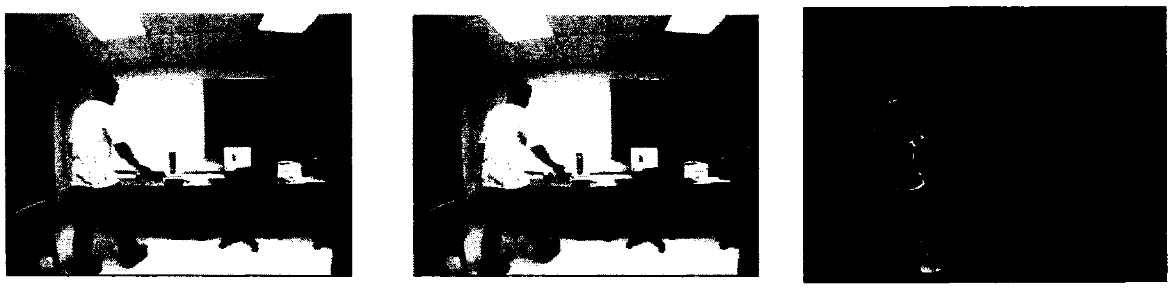
2.5 Conclusion

Automatic video surveillance requires developing a complete system with several components. Each of these components is associated with different levels of difficulty. Specially, foreground region segmentation and tracking are two most difficult components of intelligent video surveillance. This chapter has given an overview of all the components of intelligent video surveillance and addressed some of the problems of foreground region segmentation and multi-object tracking.

CHAPTER 3
PREVIOUS WORK

Video surveillance in dynamic scenes (both for indoor and outdoor environments) is an active research area in computer vision and recently got considerable interest to researchers [Hu04]. It attempts to detect, recognize and track certain number of objects from image sequences and understand their behaviour. This thesis addresses the problem of detecting and tracking multiple moving people in a complex environment with unknown background, assuming that the objects are mostly human. The problem of classifying and understanding object behaviour is not addressed in this thesis.

Nearly every video surveillance system starts with detecting people in the scene. Accuracy of human detection largely affects the accuracy of tracking result and behaviour recognition. As mentioned in chapter 2, foreground region segmentation can be done by three basic approaches; frame differencing [Anderson85], optical flow [Barron94] and background subtraction [Haritaoglu98]. Frame differencing is very fast and adaptive to dynamic scene changes. It does not require any background modeling, but suffers from the problem of foreground aperture. More generally, it cannot extract all foreground pixels. When a foreground object contains homogeneous color, then it mostly extracts pixels in the boundary region. Figure 3.1 shows that only boundary pixels are detected due to homogeneity of color of moving object. It also shows that it finds overlapping regions in consecutive frames.



(a) Previous frame (b) Current frame (c) Result of temporal differencing

Figure 3.1: Result of temporal differencing

Another disadvantage of frame differencing is the inability to detect stationary foreground objects, which might happen when a person is standing still in the scene for sometime. Optical flow is the most robust technique for detecting all moving pixels, even in the presence of camera motion. But, it is computationally expensive and not feasible for real-time system. Background subtraction is very fast and simple. It can find all foreground pixels, even though the object is stationary for sometime. The main problem of background subtraction is the requirement of an ideal background and a good threshold value to eliminate false positive. Another problem of background subtraction is the sensitivity to dynamic scene changes due to lighting conditions and extraneous events. Movement of background object also creates background aperture. Both frame differencing and background subtraction assumes that the camera is stationary. This assumption is not unrealistic for surveillance in real-world.

Tracking methods in the literature can broadly be classified into 5 categories [Hu04]; they are region-based tracking [Collins00, Huang05, Xu02], active-contour-based tracking [Israd99, Paragios99], model-based tracking [Zhao04], stereo-based tracking [Beymer99] and feature-based tracking [Xu05]. Region-based tracking is performed based on the variation of the image regions in motion. It does not require computation of image blobs and feature extraction, but it suffers from computational complexity, as it matches a window region with all candidate windows in the next frame. Moreover, it cannot reliably handle occlusion between objects [Hu04]. In addition, it fails to a match an object when it moves beyond a predictive region.

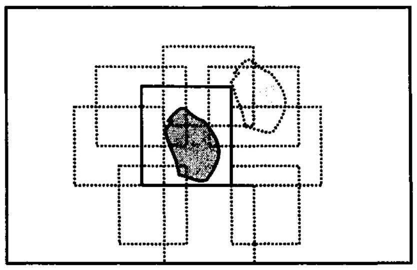


Figure 3.2: Region-based tracking

In active contour-based tracking, objects are more simply described. Here, bounding contours are used to represent object's outline, which are updated dynamically in successive frames [Hu04]. It is sensitive to initialization and limited to tracking precision. Model-based tracking requires developing a 2D or 3D model of human and tracking components of model. This is a robust approach for tracking and performs well under occlusion, but requires high computational cost [Hu04].

In stereo-based tracking, the stereo disparity is used for tracking movement of objects. A disparity background model is maintained to segment foreground disparity image. It is less sensitive to shadow, lighting changes and camera dynamics [Beymer99]. It also returns 3D shape of objects, which helps to find depth ordering of occluding surfaces. Stereo helps to minimize problem of scale in detection and provides useful state information (i.e. velocity). A camera model can be used to estimate 3D quantities [Zhao04]. This approach is computationally expensive and requires two cameras to calculate disparity.

In feature-based tracking, features of image blobs are extracted for matching in sequence of frames. In this method, several features of blobs are used in feature-vector for matching, such as size, position, velocity, ratio of major axis of best-fit ellipse [Xu04], orientation, blob bounding box etc. The feature-vectors can be compared by several techniques such as Euclidean distance [Xu04] and correlation-based approach [Haritaoglu99]. The histogram of RGB color components of image blobs can also be used as feature and those histograms can be compared for matching [Comaniciu00]. The motivation behind using feature-based approach in our tracking system is that, it is faster compared to other methods and a very simple and straightforward approach for tracking. Another advantage of feature-based approach is the remembrance of the feature-vectors in memory for a long period of time to identify a previously tracked person. The main disadvantage of feature-based tracking is the requirement of blob construction in every frame and extracting features from there, but it takes reasonable amount of time to develop a real-time tracking system.

In the literature, some of the tools used to track objects are Kalman filter [Xu04], geodesic method [Paragios99], CONDENSATION method [Israd98] and dynamic Bayesian network [Hu04]. Kalman filter is a method used to track a time-varying signal in the presence of noise. For a detail description of Kalman filter, see Appendix I and [Welch95]. Kalman filter has been used by many researchers in the literature for tracking objects, especially for tracking active contours [Peterfreund99]. The pure Kalman filter has a disadvantage. It is limited to use, because they are based on unimodal Gaussian densities that cannot support simultaneous alternative motion hypothesis [Collins00]. Israd & Blake proposed a method called CONDENSATION and showed that this method can successfully track active contours in cluttered scene and can handle multi-modal and non-Gaussian distributions [Israd98]. The main disadvantage of CONDENSATION is computational cost.

In this thesis, we have addressed the problem of foreground object segmentation by background subtraction, background modeling, removing noise and shadow, blob construction and feature-based multi-object tracking. Feature-based object tracking requires feature selection, feature extraction, feature matching, maintenance of total objects and occlusion handling. This chapter gives a literature review of different approaches to solve these problems.

3.1 Foreground Object Segmentation

Detecting regions of change in images is of widespread interest due to its large number of applications in diverse fields, starting from video surveillance to remote sensing, medical diagnosis and treatment, civil infrastructure etc [Radke05]. Detection of moving object in video stream is a difficult research problem [Toyama99]. Research on moving object detection in the scene first started in early 80's, when the technology was developed to capture digital images from video. In the literature, most of the works on foreground object segmentation is based on background subtraction technique [Intille97, Haritaoglu98, and Zhou01]. Background subtraction requires a threshold and such a threshold should be updated dynamically [Xu04]. One hard threshold for the entire image can be used or different threshold value can be used for each pixel [Collins00]. Maintaining a threshold for every pixel is computationally expensive. Anderson *et al.* used frame differencing for change detection and tracking objects [Anderson85]. Very few works are found in the literature on foreground object segmentation for tracking based on optical flow technique. For a detail description of optical flow, see [Barron94].

Some authors used a variation and/or combination of techniques used in foreground object segmentation. Collins *et al.* used a hybrid of frame differencing and background subtraction for effective foreground object segmentation [Collins00]. This approach face the initial problem of frame differencing and cannot segment foreground region reliably. Zhou and Aggarwal used a combination of color and texture in background subtraction [Zhou01]. The texture is represented by horizontal and vertical gradient and they are calculated by Sobel Gradient Operator. Their approach is limited to speed, becasue use of texture in segmentation requires much computational time. Ivanovic and Huang used a combination of Euclidean distance and Mahalanobis distance between color components in Lu*v* color space for background subtraction, instead of simple subtraction [Ivanovic04]. Euclidean distance has the problem of scaling and Mahalanobis distance requires calculation of covariance matrix, which is computationally expensive.

McKenna *et al.* proposed to use 3 times of standard deviation of a pixel as a threshold in background subtraction [McKenna00]. The significance of 3σ is that, if the data is normally distributed, then 99.7% of the data are within 3σ. Stauffer and Grimson used 2.5σ as the threshold [Stauffer99]. Chebyshev's theorem guarantees that 88.9% of the data are within 3σ of the mean regardless of distribution. This motivates the researchers to use 3σ as threshold. McKenna *et al.* [McKenna00] assumed that background changes slowly, which may not be true in some cases. A substantial amount of variation in lighting condition will invalidate the three-sigma rule.

Background subtraction technique requires modeling the scene background. To accommodate lighting changes in the scene, background should be updated continuously with scene changes to adapt [Xu04]. In the literature, a wealth of work can be found on modeling dynamic background. Researchers usually use statistical model [Zhou01, Haritaoglu01], Gaussian model [Xu05, McKenna00], a mixture of Gaussian [Stauffer99, Bunyak05, McKenna99, Xu02] and kernel density function [Elgamal02] for modeling background. Zhou *et al.* used a temporal median filtering technique for modeling background [Zhou01]. Haritaoglu *et al.* [Haritaoglu00] built a statistical model by representing each pixel with three values: its minimum and maximum intensity values and the maximum intensity difference between consecutive frames observed during the training period. These three values are updated periodically. McKenna *et al.* [McKenna00] used Gaussian model for each pixel of the background, assuming that the pixels in the background change independently following normal distribution (see equation 3.1).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2} \right) \qquad \ldots 3.1$$

In Gaussian model, the mean and variance of each Red-Green-Blue (RGB) color component of a pixel is maintained (i.e. μ_r, μ_g, μ_b, σ_r^2, σ_g^2, σ_b^2). The parameters of Gaussian model (mean and variance) can be updated dynamically using the equations 3.2 and 3.3 [McKenna00, Stauffer99]:

$$\mu_{t+1} = \alpha\mu_t + (1-\alpha)z_{t+1} \qquad \ldots. 3.2$$

$$\sigma_{t+1}^2 = \alpha(\sigma_t^2 + (\mu_{t+1} - \mu_t)^2) + (1-\alpha)(z_{t+1} - \mu_{t+1})^2 \qquad \ldots 3.3$$

Here α is a constant that determines how quickly the model is allowed to change and z_{t+1} is the value of latest measurement. Calculating and updating three means and standard deviation for each pixel is computationally expensive. To handle variations in lighting, moving scene clutter and arbitrary changes to the observed scene, one particular distribution of the background is not enough [Stauffer99]. This motivated the researchers to use the Gaussian mixture model (GMM), instead of one Gaussian model. GMM is calculated by pixel process (time series of pixel values, some recent values are used to determine Gaussian parameters). A pixel is matched against all background Gaussians to confirm foreground pixel. Stauffer *et al.* used the equation 3.4 and 3.5 of GMM to represent background [Stauffer99]:

$$P(X_t) = \sum_{i=1}^{K} w_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \qquad \ldots 3.4$$

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)} \qquad \ldots 3.5$$

Here K is the number of Gaussians, w is the weight of i^{th} Gaussian at time t and Σ is the covariance matrix. They assumed that RGB values are independent to each other and have the same variance, so that Σ can be estimated by $\sum_{k,t} = \sigma_k^2 \mathrm{I}$. In GMM, finding a correct number of Gaussians is difficult. Another problem of this approach is the computational cost and not feasible for real-time processing. Ridder *et al.* [81] modeled each pixel value with a Kalman Filter to compensate for illumination variance [Ridder95]. Maintaining a Kalman filter for every pixel is not feasible for real-time tracking. Toyama *et al.* proposed the 'Wallflower' algorithm in which background maintenance and background subtraction are carried out at three levels: the pixel level, the region level, and the frame level [Toyoma99]. Background can be modeled in different color spaces for handling some of the challenges associated with background issues.

Segmented foreground region may contain noise, so they should be removed before tracking. For pixel-level noise removal, Xu *et al.* used majority voting of neighbouring pixels to remove false detection [Xu04]. Morphological operations (i.e. erosion and dilation) are very popular method in the literature for noise removal and many researchers used these techniques [Intile97, Zhou01, Hunag05, Xu02, Zhao04, Haritaoglu98]. See Appendix I for a detail description of morphological operations. Huang *et al.* used median filter to smooth the foreground region [Huang05]. For blob-level noise removal, some higher level information is used to identify legitimate blobs. Any blob having size less than a threshold is considered as noise and is discarded consequently [McKenna00, Bunyak05, Huang05, Xu02]. McKenna *et al.* used 30 pixels as size threshold in their tracking system [McKenna00]. The threshold depends on the properties of the scene and placement and/or zooming of the camera. Sometime blobs are split into several parts due to segmentation error. To solve this problem, Intille *et al.* suggested to merge blobs having overlapping bounding box into one cluster to consider broken parts together [Intille97]. This approach has the danger of merging split part of other object. McKenna *et al.* used different levels of abstraction (object/region, people and group) to find relationship among blobs [McKenna00]. In this case, one or more blobs/regions constitute a person and one or more persons constitute a group.

Shadow is a big problem during foreground region segmentation. In the literature, different techniques are used to remove shadow from the foreground region. Among them, structure or shape analysis of foreground region, modeling the background with gradient information and use of different color spaces are most popular. Zhao *et al.* used shape analysis to distinguish human from its shadow [Zhao04].We know that the cast-shadow region retain similar texture (edge) properties. Any significant intensity change without significant chromaticity change is due to shadow. McKenna *et al.* calculated the chromaticity (of red and green color) and intensity or brightness of color using equations 3.6, 3.7 and 3.8 [McKenna00]:

$$r_c = \frac{r}{r+g+b} \qquad \ldots 3.6$$

$$g_c = \frac{g}{r+g+b} \qquad \ldots 3.7$$

$$I = \frac{r+g+b}{3} \qquad \ldots 3.8$$

They classified pixels into background, foreground and shadow regions by examining chromaticity distortion and brightness distortion. When background and foreground have same chromaticity, then first-order image gradient information (both for background and foreground) can be used to classify pixels [McKenna00, Xu04]. In this case, each background pixel's gradient is modeled using gradient means and variance. The distinction between foreground and shadow pixel is determined by calculating Euclidean distance of horizontal and vertical gradients. This distance is much less for a shadow pixel than a foreground pixel. This approach does not work for shadows with hard edges. The self-shadow region can get deleted while removing cast shadow region. In that case, Xu *et al.* proposed to use morphological operations to recover deleted regions [Xu04]. When a large portion of foreground region is deleted, then this approach will not work.

Instead of RGB color space, other color spaces are also used in the literature for detecting and tracking object, such as:

- HSV/HIS [McKenna99, Yazdi02]
- CMYK
- CIE LU*V* [Ivanovic04]
- CIE L*a*b* [Mckenna99]
- YIQ
- UCS
- YUV [Intille97, Wren97] etc.

In [Chen04], Chen *et al.* modeled the background in HSV color space to eliminate shadows. Horprasert *et al.* used a color constancy model for shadow detection, assuming that the chromaticity remains same while only intensity differs between shadow and background [Horprasert99]. Ivanovic *et al.* showed that LU*V* color space is perceptually more uniform than RGB color space [Ivanovic04]. Among these color spaces, HSV/HIS color space works well against shadow [Cucchiara05]. For a detailed description of HSV color space see Appendix I.

After finding the foreground pixels, blobs are identified by 8-connectivity [Xu04, Zhou01, Xu02] or 4-connectivity [Huang05] connected component analysis (CCA). The blobs (or pixels) which are in motion can be identified by multi-frame frame differencing [Haritaoglu01]. Some researchers classify objects at this stage for tracking specific objects (i.e. car or human). There are several techniques used for classifying objects, such as star skeletonization [Fujiyoshi98], neural network , Linear Discriminant Analysis (LDA), boosted classifiers, bayesian classifiers, MLE [Lipton98] etc. Zhao *et al.* constructed a human model from training data and matched against the blobs to identify human [Zhao04]. They used human head model to locate appearance of human in the scene. Often human shape is estimated by an ellipse while tracking.

3.2 Feature-based Object Tracking

A feature is a useful and persistent characteristic of an object or a blob. A feature-based object tracking algorithm requires feature selection, feature extraction, tracking by feature matching, proper handling of object's appearance and disappearance and tracking during occlusion. Usually object's features like color [Xu04], shape [Intille97], velocity, compactness/dispersedness [Ivanovic04], orientation [Zhou01] etc are used for tracking in video. Some features are more persistent, while others may be more susceptible to noise. According to Zhou *et al.*, motion and color are most reliable features [Zhou01]. Table 3.1 shows a summary of features used by different researchers.

Table 3.1: Summary of features used by researchers

	Dominant Color	Avg. Color	Shape /Size/ Area	Velocity of centroid	Orientation	Ratio of major axis (ellipse)	Position/ Centroid	Bounding box	Compactness	Color histogram
Xu04	√		√	√	√	√				
Intille 97		√	√	√			√			
Zhou01	√		√	√			√	√	√	
Bunyak05			√				√	√		
Ivanovic04		√	√				√	√	√	√
Xu02							√	√		

In table 3.1, it is observed that shape and centroid are mostly used features. Zhou *et al.* used PCA to extract the color feature of the object and compared blobs by their principle axis [Zhou01]. They also maintained a template of features for every blob. New template is created for new blobs and old template is deleted when blobs are lost for a while. The templates are updated by feature vector in the every frame using the equation 3.9.

$$T_{i,t} = \beta T_{i,t-1} + (1-\beta)R_{i,t} \qquad \ldots 3.9$$

Here, $T_{i,t-1}$ is the template of ith blob in previous frame and $R_{i,t}$ is the feature vector of ith blob in current frame. Feature vectors are calculated in every frame from the computed blobs. Some tracking system use prediction of features in the next frame and compare the predicted value with the estimated value to update the model. Many tracking systems in the literature use a model like Kalman filter for prediction. Kalman filter is used to update the tracking model such as feature template, location of an object in the next frame etc [Xu04]. Some contour-based tracking approach used Kalman filter to predict boundary and shape of an object in 3D [Zhao04]. In this case, a mean and variance of the feature vector is maintained based on last L frames [Xu04].

Templates are not updated during occlusion. Xu *et al.* suggested to update object states using partial observations whenever available [Xu02]. In feature-based object tracking, care should be taken in case of template drift due to change in lighting and orientation, background noise and occlusion [Beymer99].

After computing feature vectors successfully, the template of an object is matched against feature vectors of all candidate blobs. Accuracy and efficiency of tracking depends on accuracy and efficiency of matching of a template with feature vectors. Xu *et al.* used an exhaustive search to find best candidate blob of an object by matching a template with all feature vectors [Xu04]. Kalman filter can be used to identify possible location of an object in next frame and narrow down the search region [Xu04, Xu02]. In [Comaniciu00], the authors proposed a mean-shift technique to calculate most probable target position. Haritaoglu *et al.* employed a mean-shift tracker to recover trajectory of each individual in the scene [Haritaoglu01]. Velocity can be used to estimate position of an object in next frame. At high frame rate, objects usually move a little bit and remain close to their blob in the next frame. Size difference also varies slowly at high frame rate. Intille *et al.* used a maximum distance constraint to eliminate false match in their closed-world tracking system [Intille97]. McKenna *et al.* proposed that matching of an object with blobs having overlapping bounding boxes is effective and sufficient. We don't need to do any prediction, as visual motions of regions are always small relative to their spatial extents [McKenna00].

Regarding techniques used in matching feature vectors, different functions and hierarchy of feature matching is used. Xu *et al.* used a scaled Euclidean distance function for matching. The matching pair having lowest distance and less than a threshold indicates best match [Xu04]. Different features normally assume numerical values of different scales and variances. That is why features are usually normalized before matching. Xu *et al.* suggested using Mahalanobis distance in order to solve the problem of scale and variance of features [Xu04]. In [Collins00], Collins *et al.* used a correlation function for matching regions in motion. 'Hydra' used sum of absolute difference (SAD) for matching head regions in a 5x3 search window [Haritaoglu99]. Huang *et al* used a combination of region-level and object level tracking approach. The region level tracking is considered as global optimization problem and performed by Genetic Algorithm [Huang05]. Zhou *et al.* used a hierarchy of feature matching [Zhou01]. The order of matching they proposed is centroid, shape, color and so on. This approach is called multi-hypothesis matching [Bunyak05].

To store the matching result, a 2D matrix S_{ij} is usually maintained which store match score of object/template i with blob j [Bunyak05, Huang05]. Intille *et al.* maintained separate score matrices. A weighted sum of those matrices is considered as final match score matrix [Intille97]. Bunyak *et al.* maintained a confidence matrix Conf_{ij} that denotes confidence of matching [Bunyak05].

Some of the works in the literature used graph theory in their tracking system. Haritaoglu *et al.* detected a shopping group having similar motion cues [Haritaoglu01]. They employed graph partitioning approach to identify shopping groups from strongly connected component. Bunyak *et al.* used a graph structure (called ObjectGraph) which stores object features in successive frames [Bunyak05]. The graph is traced to find object trajectories. A combination of ObjectGraph and Dynamic programming can be used to represent all realizable tracking decision [Khalaf01].

The accuracy of tracking can be improved by delaying tracking decision. Khalaf *et al.* suggested that inaccurate matching result due to weak features can be improved by a temporal consistency throughout 1-5 sec window for reliable multiple-object tracking [Khalaf01].

To perform the object tracking successfully, we need to manage appearance and disappearance of objects in the scene, which changes total number of objects being tracked. Collins *et al.* proposed a basic object tracking algorithm and tracking hypothesis [Collins00]. According to them, the possible scenarios during tracking are:

- Continuation of an object (perfect match)
- An object can disappear
- New object can appear
- Two or more objects can merge
- A object can separate into several objects

In addition to these, another possible scenario that can happen is merging and splitting of blobs at the same time, which makes total number of blobs in the current frame unpredictable.

An effective management of object entry and exit was proposed by Stauffer [Stauffer03]. Intille *et al.* used context-sensitive information about the scene to make hypothesis about an object (i.e. closed world). An object can appear or disappear only through a particular region (i.e. door) which makes the maintenance of total objects easier [Intille97]. To remove spurious noisy blobs, a time stamp or a counter of current track length and lost track length can be maintained [Xu04, McKenna00]. If an object is successfully tracked for N frames, then that object is confirmed as a valid moving object and the template for that object is created. Typical values used for N are 3 [McKenna00], 5 [Haritaoglu01], 10 [Xu04], 50 [Zhou01] etc. If an object is lost for M frames, then the object is considered as occluded or went out of scene and the template is deleted. When a region splits, all the resulting regions inherit their parent's timestamp and status. When regions merge, the resultant regions template is updated from combination of merging regions. Khalaf *et al.* used some constraints to make the tracking task easier. They used spatial continuity, continuity in motion and continuity in appearance for reliable tracking [Khalaf01]. Spatial continuity ensures that people who enter a visual merge are the same people who leave the merge. Continuity in appearance ensures that people will tend to look similar.

Tracking during occlusion requires detection of occlusion and tracking under occlusion. When an object fails to find a matching candidate blob, a test on occlusion is carried out. Bunyak *et al.* used ObjectGraph to detect occlusion [Bunyak05]. Occlusion can be predicted in the next fame by several techniques. If the object's predicted bounding box overlaps with another object's predicted bounding box, then we can assume an occlusion in the next frame.
We can also predict occlusion in the next few frames by observing increase in correlation result.

In 'Hydra', Haritaoglu *et al.* detected occlusion by counting number of people (i.e. by detecting heads from local shape, global shape and texture matching) in a group (or in a blob) [Haritaoglu99]. A vertical projection histogram of each silhouette is constructed and matched against prior model to find single or multiple person. They used a recursive least square estimation method to predict location of head during total occlusion. 'Hydra' is limited to the visibility of head region for occlusion handling, but in real world the head region might not be visible for sometime. Xu *et al.* detected occlusion by counting total number of blobs [Xu05]. This information is only valid at the beginning of occlusion. After occlusion, total number of blobs remains same and it is not possible to detection occlusion in the current frame anymore. Zhou *et al.* detected occlusion by background object by checking reduction of size exceeding a certain threshold for few successive frames [Zhou01]. Finding an average size of a person is a difficult task, it varies drastically among people in different depth and different age group.

For tracking objects during occlusion, Haritaoglu *et al.* maintained a silhouette-based shape model and an appearance model [Haritaoglu99]. In their approach, they segmented the occluded blob (or assigned pixels) into individuals by analyzing path distance constraints among body parts (or by analyzing horizontal distance map from each pixel to each vertical torso axis, see figure 3.5). The local motion of a person in the occluded blob was estimated by edge information.

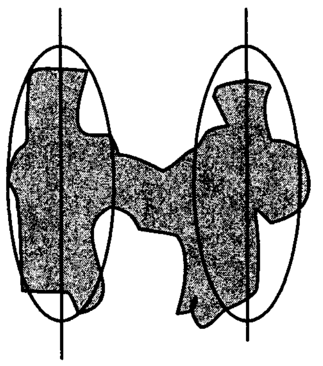


Figure 3.3: Separating pixels in occluded blob by horizontal distance map

Yazdi *et al.* proposed to use hue and intensity separately for object segmentation during occlusion [Yazdi02]. The depth information is helpful for tracking during occlusion. Xu *et al.* used an estimation of relative depth ordering using contact point of an object into the ground plane [Xu05]. Contact point might not be visible all the time due to partial appearance of a person in the scene or due to partial occlusion by background object.

Although color can be used as a feature in feature vector for feature-based tracking, some authors gave more emphasis on using color information extensively in their human tracking system. They argued that the color distribution of cloth is quite stable under rotation in depth, scaling and partial occlusion [McKenna00, Swain91]. Each persons appearance model can be build and updated based on color information and edge information. This appearance model can be used to track people successfully during occlusion [Haritaoglu01].

Color distributions can be effectively modeled by both color histogram and GMM [McKenna00]. When adaptation is not needed and data set is large, then histogram is preferable. When color samples are small and color range is large, then GMM is more appropriate. Histogram is slightly more effective for modeling skin color from large image database [McKenna00]. Huang *et al.* used only color as distinguishing feature [Huang05]. In their system, the appearance model is a color histogram built over H and S channels of HSV color space. H channel is quantized into 16 values and S into 8 values. A histogram can also be built based on color and edge information [Haritaoglu01]. Comaniciu *et al.* also calculated similarity of objects in their Mean-shift tracker by comparing histograms of target model and target candidates [Comaniciu00]. Similarity was expressed by a metric derived from the Bhattacharyya coefficient. Bunyak *el al.* used N RGB color vector, corresponding to N peaks in the histogram, as color descriptors of an object [Bunyak05].

The histogram of an object and a blob can be matched by histogram intersection and histogram back projection method. Histogram intersection tells “How many of the pixels in the model histogram are found in the image?”[Swain91, Khalaf01]. Histogram intersection is calculated by equation 3.10.

$$H(I,M)=\frac{\sum_{j=1}^{n}\min(I_j,M_j)}{\sum_{j=1}^{n}M_j} \qquad \ldots 3.10$$

Xu *et al.* used a normalized distance function, shown in equation 3.11, for matching histogram of a model I and a candidate I' [Xu05].

$$D(I,I')=\frac{\sum_{j=1}^{m}|h_j(c_i)-h'_j(c_i)|}{\sum_{j=1}^{m}|h_j(c_i)+h'_j(c_i)|} \qquad \ldots 3.11$$

Here m represents total number of bins and c_i is the i^{th} quantified color. Elgammal *et al.* used a different approach to represent color distribution of each region, which is called Non-parametric Kernal density estimation [Elgammal01]. Kernal density is a sample taken from a set of samples (of d-dimensional vector) that best describe the sample set.

McKenna *et al.* proposed using appearance models for tracking objects. The appearance model is a discrete probability distribution constructed from the color histogram of the individual objects [McKenna00]. They used equation 3.12 for constructing appearance model for person *i*.

$$P(x \mid i) = \frac{H_i(x)}{A_i} \qquad \ldots 3.12$$

Here A_i is the area of the person blob. In fact, appearance model is a color model of person I having pixels x. For histogram, each color channel is quantized into 16 values, so the histogram has a total of 16^3 or 4096 bins. The appearance model can be updated adaptively in each frame by equation 3.13.

$$P_{t+1}(x \mid i) = \beta P_t(x \mid i) + (1-\beta) P_{t+1}^{new}(x \mid i) \ldots 3.13$$

Here P_{t+1}^{new} is the probability computed using only new image and β is 0.8. Xu *et al.* used 0.9 for β [Xu05]. Elgammal *et al.* suggested to initialize the appearance model when objects are isolated [Elgammal01]. The adaptation is suspended when a person is in a group. They proposed using different appearance model for head, torso and leg. For tracking during occlusion, McKenna *et al.* first calculated a prior probability of a pixel corresponding to the i^{th} person in the group using equation 3.14 [McKenna00].

$$P(i) = \frac{A_i}{\sum_{j \in G} A_j} \qquad \ldots 3.14$$

Afterwards, they calculated posterior probabilities of every pixel within a group in respect to person *i* and separated all pixels into individual persons. Equation 3.15 was used for calculating posterior probabilities.

$$P(i \mid x_{x,y}) = \frac{P(x_{x,y} i) P(i)}{\sum_{j \in G} P(x_{x,y} \mid j) P(j)} \qquad \ldots 3.15$$

Posterior probability describes probability of belongingness of a pixel *x* into person *i*. A higher value for posterior probability indicates that a pixel in the group with those color values has a higher probability of belongingness to the unoccluded part of the target person. They computed a visibility index for each person in the group by equation 3.16 and used it to estimate a depth ordering of that person in the group.

$$v_t(i) = \frac{\sum_{x,y} P(i \mid x_{x,y})}{A_i} \qquad \text{...3.16}$$

When a group of people separate into smaller groups, then they used histogram intersection method proposed by Swain and Ballard [Swain91] to find which person belongs to which group.

Swain and Ballard also proposed another technique called 'histogram backprojection' for segmenting object during occlusion [Swain91]. Histogram backprojection answers the question "Where in the image are the colors that belong to the object being looked for (the target?)".

Color histogram lacks spatial layout information and may fail to characterize objects when they have similar colors. Huang *et al.* suggested incorporating spatial distribution of the object area [Huang05]. Elgammal *et al.* and Wren *et al.* used spatial distribution of pixels with respect to the whole body along with their appearance (color distribution) [Elgammal01, Wren97]. Xu *et al.* proposed a spatial-depth affinity matrix (SDAM) in histogram backprojection for improved segmentation [Xu05].

3.3 Conclusion

In this chapter, we have discussed different approaches proposed in the literature for foreground object detection and tracking. We have described the previous work in the context of different steps required for segmentation and tracking. From the discussion of this chapter, it is evident that the foreground region segmentation and object tracking is still an unsolved problem, because of their limitation in assumptions and application in specific context. Specially, there is no robust solution for shadow, camouflage and occlusion problem.

CHAPTER 4
FEATURE-BASED TRACKING OF MULTIPLE PEOPLE

This chapter describes in details all the processing steps of a feature-based tracking system. First, the background modeling approach and foreground region segmentation technique is discussed. Then, the noise and shadow removal technique followed by blob construction method is presented. Finally, the feature-based tracking of multiple people is explained with examples. The occlusion handling technique by histogram backprojection is discussed. The overall human detection and tracking system is illustrated by flow-charts.

4.1 Background Subtraction and Background Modeling

Feature-based tracking of multiple people starts with detection of people in the scene. In our tracking system, we have used a combination of background subtraction and clustering techniques for human detection. Background subtraction requires an ideal background and an appropriate threshold value to segment foreground region. In case of surveillance in a busy area (i.e. shopping mall or intersection of roads), the perfect background may not be available. To overcome this problem, we have used a statistical method proposed by Indupalli *et al.*[1]. The idea is that, an ideal background model can be build by observing the value of a pixel in previous frames and delaying the tracking task. Table 4.1 describes the algorithm for modeling background by statistical method.

The background was modeled in HSV color space. The reason of using HSV color space will be described later in this chapter. Figure 4.1 shows the processing steps for constructing histogram of a pixel from sequence of frames. In HSV color space, the value of H component varies from 0 to 360. We have used 18 bins for constructing histogram of H component, so that each bin can have 20 different values [Broek04]. We have used 16 bins in the histogram of S and V components.

[1]Indupalli, S.; Ali, M. A.; Boufama, B.; "A Novel Clustering-Based Method for Adaptive Background Segmentation", First International Workshop on Video Processing for Security (VP4S-06) in Proceedings of Third Canadian Conference on Computer and Robot Vision (CRV'06), June 7-9, Quebec City, Canada

Table 4.1: Background modeling by statistical method

Input: Sequence of frames (in Red-Green-Blue or RGB color space)

Output: An ideal background model (in Hue-Saturation-Value or HSV color space)

1. Convert each frame in HSV color space
2. Calculate the histogram of H, S and V component for a particular pixel in all the fames
3. Find the histogram bin with highest value and assign the median of this bin to the H, S and V component of the pixel in the background model
4. Perform step 1,2 and 3 for all the pixels in the background model until the pixel values are stabilized

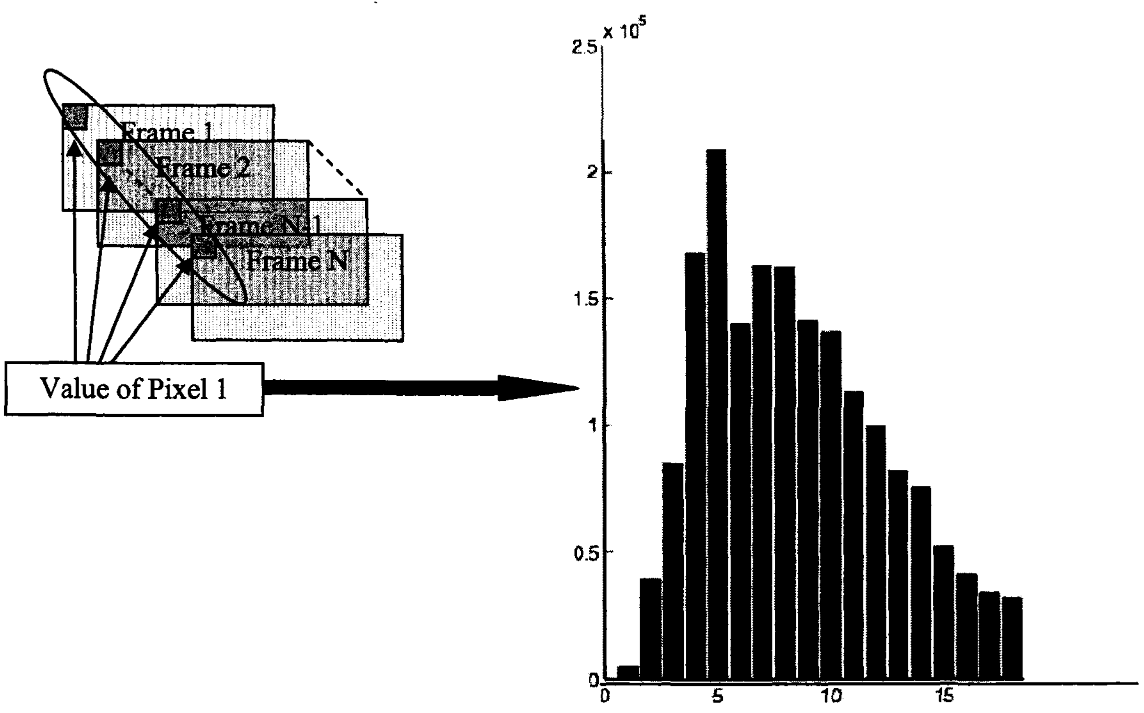
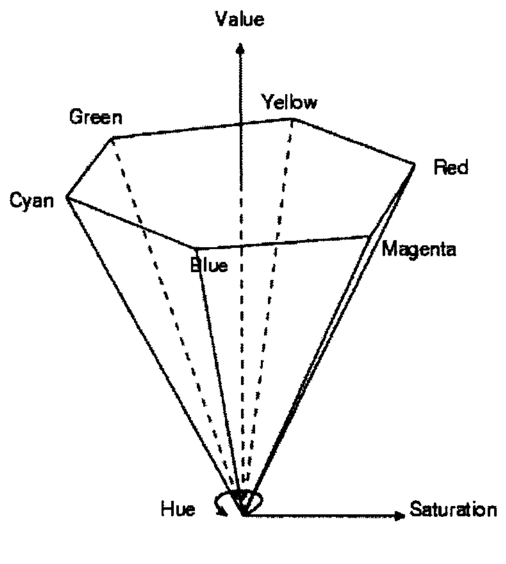
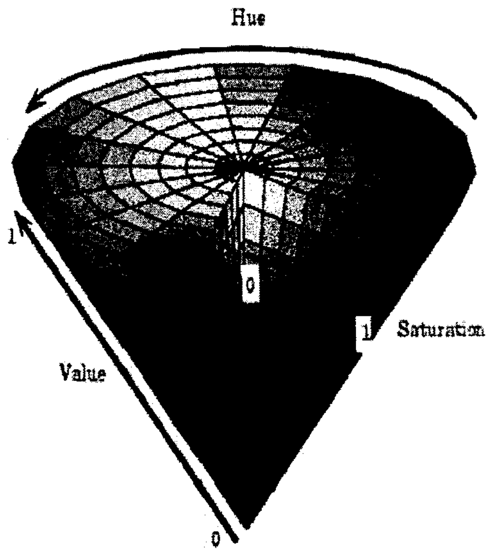


Figure 4.1: Histogram of a pixel constructed from sequence of frames

Figure 4.2 shows a visualization of HSV color space. From figure 4.2 we can see that the hue circle of HSV color space consists of the three primaries red, green and blue, separated by 120 degrees. In between red, green and blue, there are other three major colors; they are yellow, magenta and cyan. A circular quantization at 20 degree steps sufficiently separates the hues such that the three primaries and yellow, magenta and cyan are represented each with three sub-divisions. This is the main motivation behind using 18 bins for H in the histogram.



(a) Hexcone of HSV color space



(b) Hexagonal surface of hue contains 6 major colors

Figure 4.2: Quantization of H component [wiki1]

To adapt with dynamic scene changes, we have updated our background model continuously in every frame using equation 4.1 [Mckenna00].

$$BG_{i,t} = \alpha BG_{i,t-1} + (1-\alpha)x_{i,t} \qquad \ldots 4.1$$

Here $BG_{i,t}$ is the value of pixel *i* in background model and $x_{i,t}$ is the value of pixel *i* in the current frame. The value of α determines how quickly we want to change our background model. In our experiments, we have used $\alpha = 0.9$. For a very dynamic scene, the value of α should be less than 0.5.

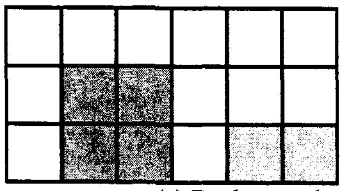
The next problem of background subtraction is finding a good threshold value to segment foreground region reliably. We have overcome this problem by using a clustering-based segmentation technique proposed by Indupalli *et al.* [Indupalli06]. The algorithm is shown in table 4.2. In this approach, k-means clustering is applied to the difference image to find foreground and background clusters.

Table 4.2: K-means clustering for segmenting foreground region

Input: background model and a video frame

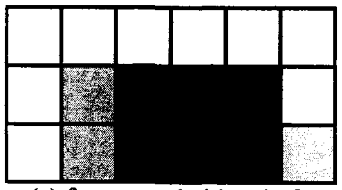
Output: Segmentation of all pixels into foreground and background clusters

1. Convert video frame into HSV color space
2. Subtract the H, S and V component of background model from the H, S and V component of the video frame and store the absolute value into a difference image
3. Find minimum and maxim values in the difference matrix. The minimum value corresponds to the seed of the background cluster, denoted by M_1 and maximum value corresponds to the seed of foreground cluster, denoted by M_2.
4. For every pixel *i* in video frame, calculate $D_{ij}=|x_i - M_j|$, j=1 & 2 and x_i is the value of pixel *i*
5. If $D_{i1}>D_{i2}$, then assign *i* to FG cluster, otherwise assign *i* to background cluster
6. Calculate mean of background cluster, M_1 and mean of foreground cluster, M_2.
7. Repeat step 4 and 5 until M_1 and M_2 does not change significantly.
8. Report pixels in foreground cluster as foreground region and vice versa


(a) Background model

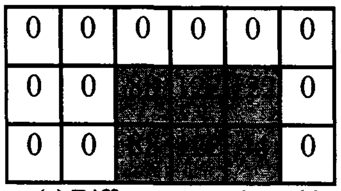
78	78	78	78	78	78
78	112	112	78	78	78
78	112	112	78	156	156

(b) Value of pixels in the background model


(c) foreground object in front of a background

78	78	78	78	78	78
78	112	200	200	200	78
78	112	200	200	200	156

(d) Values of pixels in the current frame



0	0	0	0	0	0
0	0	[illegible]	[illegible]	[illegible]	0
0	0	[illegible]	[illegible]	[illegible]	0

(e) Difference matrix and k-means clustering

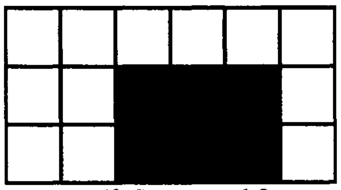

(f) Segmented foreground region

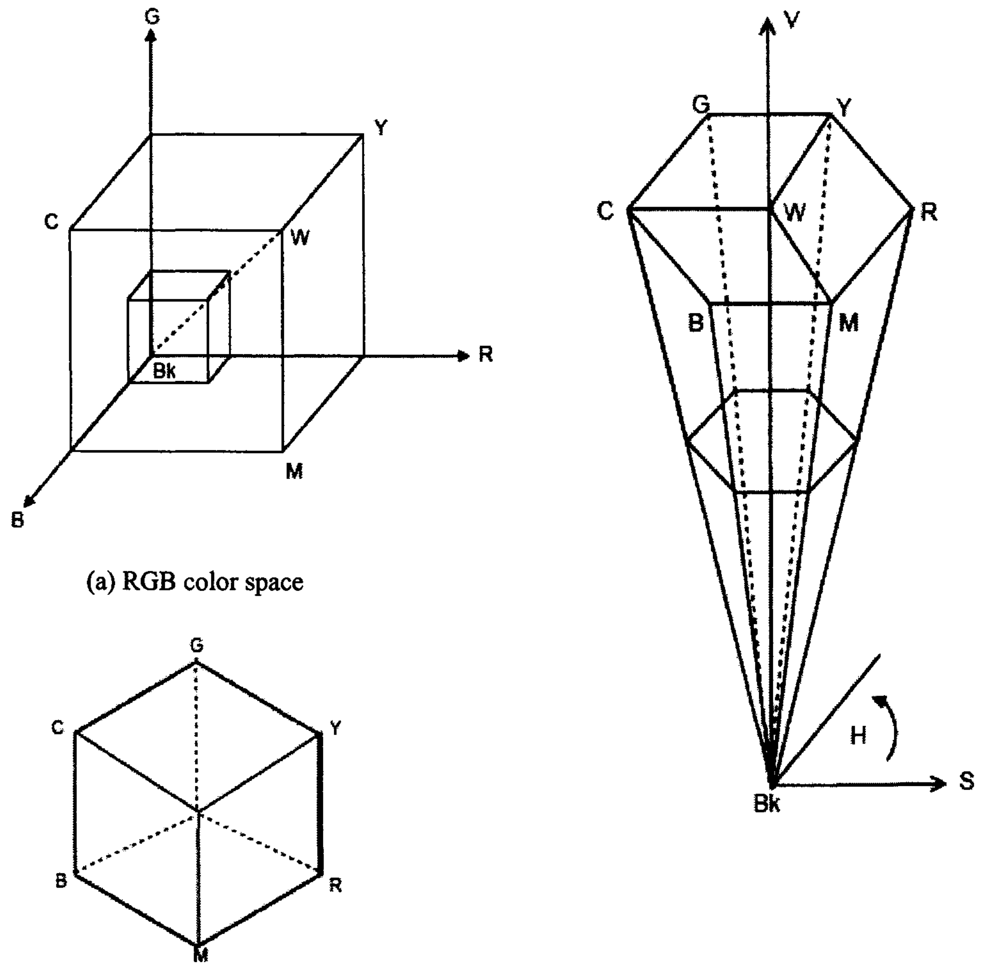
Figure 4.3: Example of segmentation by k-means clustering

Figure 4.3 shows an illustration of the segmentation process by k-means clustering using an example. K-means clustering has a drawback for segmenting foreground region. When there is no foreground object in the scene, then k-means clustering might produce unpredictable result. We can solve this problem by using a threshold value in the difference image or checking the size of each cluster. If one cluster is too big compared to other, then we assume that there is no foreground object in the scene and skip further processing of that frame.

A problem of background subtraction is background aperture. When a background object starts moving, then it creates a static aperture in the scene. To solve this problem, we have checked the motion vector of all foreground objects. If any object is stationary for several frames, then it is considered as part of background and background model is updated with the color of this foreground object.

In ideal situation, we are supposed to get perfect foreground regions, but in real-life this is not the case. Some noise appears in foreground regions and they should be removed before tracking. To remove noise, we have used morphological operations erosion and dilation [Xu04]. We have applied 3 erosions followed by 3 dilations using a 3x3 symmetric structuring element. The required number of morphological operation depends on the amount of noise in the scene. In our experiments, we have found that 3 erosion-dilation combination works well to remove discrete noise. For a detail description of morphological operations, see Appendix I.

Shadow creates problem during tracking; they should be removed in segmentation process. As suggested by many researchers [Cucchiara05], we have utilized the advantage of HSV color space for eliminating shadow. HSV color space can describe color more uniformly than RGB color space. For a comparison of RGB and HSV color space, refer to figure 4.4. The RGB color space represents the basic approach in representing color digitally. It uses a Cartesian coordinate system and forms a cube. The origin represents black where all color values are zero. The diagonal emanating from the origin to the top-right corner of the cube (representing white) is the locus of points with equal amounts of each color. This is also referred to as the gray diagonal. In HSV color space, the hexcone of hue with constant V can be viewed as the orthogonal projection of the RGB cube along the gray diagonal from black to white [Sattar05].



(a) RGB color space

(b) Orthographic projection of RGB color space

(c) HSV color space

Figure 4.4: Comparison of RGB and HSV color space (Bk=black, W=white, C=cyan, Y=yellow, M=magenta) [Sattar05]

To find an alternative of HSV color space, we have investigated the properties of other color spaces. The Cyan-Magenta-Yellow-Black (CMYK) color space is almost similar to RGB color space, except their difference in dimension. CMYK color space is mostly suited for printing [Ford98]. Hue-Saturation-Lightness or HSL color space is very similar to HSV color space, except for the range of values in L component. HSV color space is represented by a double cone. The two apexes of the HSL double hexcone correspond to black and white [Wiki1]. The Luminance-Bandwidth-Chrominance or YUV color space has one luminance and two chrominance component. This color space is mostly suited for television broadcasting. Although YUV color space can represent color more closely than RGB color space, not like HSV or HSL color space

[Wiki2]. YIQ (perceived luminance, color/luminance information) color space is similar to YUV color space, but described in more complex way. CIE L*U*V* and CIE L*a*b* are the variants of CIE 1931 XYZ color space, which is the most primitive color space for modeling human color perception. CIE L*a*b* color model was developed to represent an absolute color space with full gamut [Ford98] of colors, but they are inherently inaccurate.

When noise and shadow free foreground pixels are identified, we need to construct blob before tracking. We have applied the popular connectivity component analysis (CCA) for blob construction [Xu04]. CCA is similar to seeded region growing algorithm proposed by Adams *et al.* [Adams94]. We have used 8-conenctivity CCA in this case. The algorithm of CCA is described in table 4.3.

Table 4.3: Connected Component Analysis

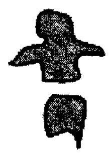
Input: foreground pixels

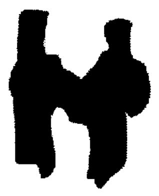
Output: list of blobs

1. Find a foreground pixel which is not considered in any blob. Consider this pixel as a seed for a new blob. Take a buffer and a queue for the new blob. Insert the seed into the queue.
2. Dequeue a pixel from the front of the queue and insert into the buffer. Check 8 neighbours of the current pixel and insert the foreground pixels into the queue.
3. Repeat step 2 until the queue becomes empty.
4. Report all the pixels in the memory as one blob
5. Repeat step 1 until all foreground pixels are assigned to any blob

CCA detects all the blobs in the scene that can be a noise, a human, a split part of a human due to segmentation error or a group of people. Figure 4.5 show different types of blobs produced by CCA.

(a) A human


(b) Part of a human


(c) A group of people

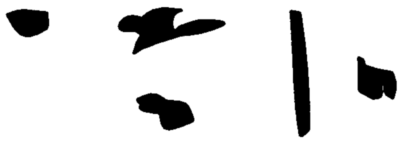
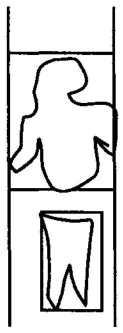
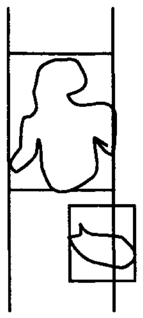

(d) Noise

Figure 4.5: Different representations of Blob

The morphological operations can remove discrete noise, but they cannot remove a group of noisy pixels that cluster together and form a blob. Before further processing, these noisy blobs should be removed by a size threshold to identify legitimate blobs. We have deleted those blobs having size (sum of height and width) less than a threshold. The size threshold can be determined dynamically by taking 10% of average size of blob. This approach has the problem of finding initial size of blob. The size threshold also depends on size of image and placement of camera or zooming of the camera. In our experiments, a threshold of 50 to 100 pixels worked well. Although the size threshold may delete a small split part of a person, it does not affect the accuracy of tracking significantly.


(a) 100% overlap


(b) 60% overlap

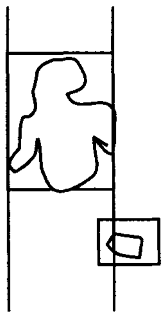

(c) 20% overlap; not merged

Figure 4.6: Different types of blob bounding box overlap

As we know that a blob can be a split part of a human body, we need to consider those parts together. We have merged some of the blobs in the scene using a heuristic. If two blob's bounding boxes overlap vertically for at least 25%, then they are merged together [Bunyak05]. Figure 4.6 shows different types of overlap between two blobs. In figure 4.7, a flow-chart summarizes all the major steps required for foreground object segmentation.

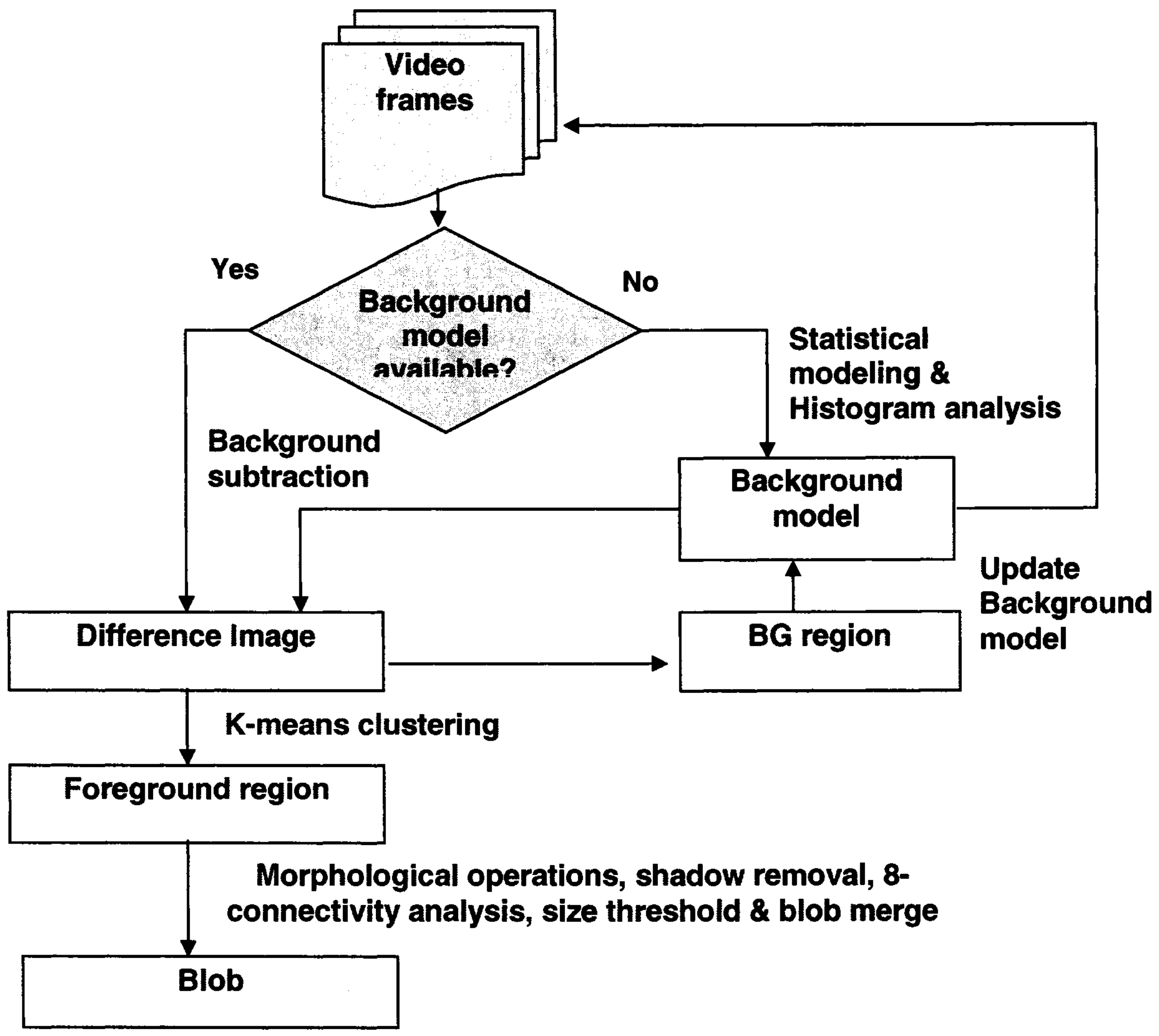

Video frames
Background model available?
Yes
No
Statistical modeling & Histogram analysis
Background subtraction
Background model
Update Background model
Difference Image
BG region
K-means clustering
Foreground region
Morphological operations, shadow removal, 8-connectivity analysis, size threshold & blob merge
Blob

Figure 4.7: Flow chart of foreground region segmentation

4.2 Tracking Multiple People

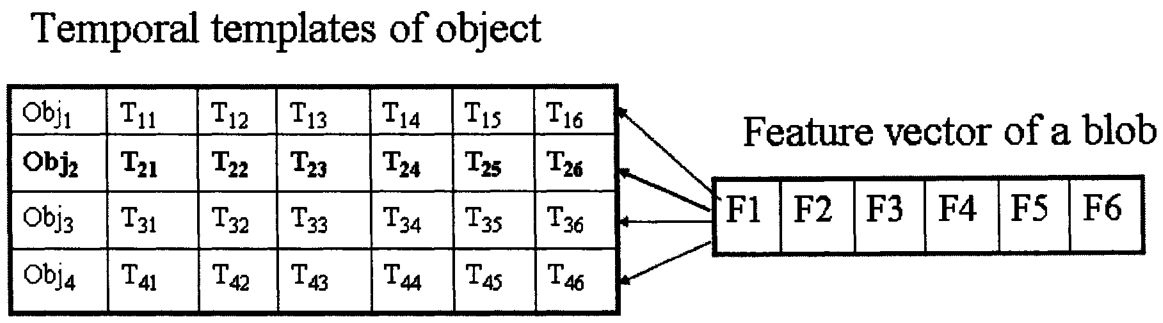
As mentioned in chapter 3, a feature-based tracking system requires feature selection, feature extraction, feature matching, maintenance of total objects and occlusion handlings. In the following sub-sections, each of these stages is described in respect to our feature-based tracking system.

4.2.1 Feature Selection, Feature Extraction and Feature Matching

Feature is a persistent characteristic of an object. We have considered average RGB color, size, blob bounding box and motion vector to describe each blob. This description of blob or features is represented by a feature-vector. The ultimate size of our feature-vector was 10 (i.e. 1 for size, 2 for motion vector in x and y direction, 3 for red, green and blue color component and 4 for x, y, width and height of the blob bounding box). Here we would like to clarify that, HSV color space has been used only for foreground region segmentation and shadow removal. For feature extraction and histogram analysis, we have used RGB color space. Although HSV color space can separate color component from intensity, RGB color space can represent the distribution of color of a blob (i.e. dress of a person) more accurately. For example, whenever we say that a person is wearing green shirt, then we need to know how much green is that (light or deep green). Only green color is not sufficient to distinguish a person from other. In HSV color space, average value of H component can not serve this purpose, that is why we have found in our experiments that the average value of H is close to zero for people wearing dark color dress. In contrast, RGB color space can represent the distribution of color of a person quite well.

In a tracking system, a combination of spatial and temporal information can increase tracking accuracy. The spatial information says where a person is in the current scene. The temporal information describes where a person was in the previous frame and where it is right now. In our feature vector, the blob bounding box contains the spatial information, whereas motion vector contain the temporal information. The features we have used are the most popular features in the literature and suggested by many researchers [Zhou01, Xu04, Ivanovic04] as most stable features. We have calculated the feature vector of a blob in every frame to get the most recent information of a blob.

After extracting the feature-vector, we need to compare the feature-vector with the temporal templates of objects to find current location of an object. Temporal templates are list of feature vectors of all the people currently being tracked. A temporal template is maintained for each person and it is updated in every frame.



Temporal templates of object
Obj1 T11 T12 T13 T14 T15 T16
Obj2 T21 T22 T23 T24 T25 T26
Obj3 T31 T32 T33 T34 T35 T36
Obj4 T41 T42 T43 T44 T45 T46
Feature vector of a blob
F1 F2 F3 F4 F5 F6

Figure 4.8: Matching a feature vector with temporal templates of objects

When a feature vector best match with a temporal template of a person object, then that blob is considered as the new position of that person and the template is updated with the new information. Matching can be done exhaustively or selectively with templates in the predicted region. We have used a heuristic to reduce required number of matching. We have assumed that there is a limit of maximum distance a person can travel [Intille97]. We have only matched the feature vector of a blob with those templates that are within that limit. The limit is updated dynamically from motion vector of objects. The initial limit we have assumed was 100 pixels. Here, how to match and how to find best match are issues to be solved. We have used a score matrix S with dimension NxM to store the result of matching between objects and candidate blobs (Assume there N number of temporal templates and M number of blobs in the current frame). For matching feature vector with a temporal template, we have used a combination of Euclidean distance [Xu04] and Pearson correlation coefficient. Euclidean distance measures the amount of difference between two vectors, whereas correlation coefficient calculates the measure of similarity. We have proposed to introduce correlation coefficient in feature matching for two reasons; First of all, it increases the range of score values that helps to better find best matched blob using a threshold. The second advantage is, it consider both the distance and similarity measure at the same time. Equation 4.1 has been used for calculating measure of similarity between a feature vector and a temporal template.

$$SIM(F_i,F_j) = w_1C(F_i,F_j) + w_2(1 - D(F_i,F_j)) \qquad \ldots 4.1$$

$$C(F_i,F_j) = \frac{1}{d}\sum_{k=1}^{d}\frac{(F_{ik} - \overline{F}_i)(F_{jk} - \overline{F}_j)}{\sigma_{F_i}\sigma_{F_j}} \qquad \ldots 4.2$$

$$D(F_i,F_j) = \sqrt{\sum_{k=1}^{d}(F_{ik} - F_{jk})^2} \qquad \ldots 4.3$$

Here, d is the dimension of the feature-vector, $\overline{F}_i$ is the mean of a feature-vector and σ_{F_i} is the standard deviation calculated by $\sigma_{F_i} = \sqrt{\frac{\sum_{k=1}^{d}(F_{ik} - \overline{F}_i)^2}{d}}$. Euclidean distance and correlation can be given different weight factor during comparison by w_1 and w_2 and $\sum_{i=1}^{2} w_i = 1$. The measure of similarity between template i and feature-vector j is stored in S_{ij}. The value of correlation coefficient varies in between -1.0 to 1.0. The advantage of correlation coefficient is that the value of correlation does not depend upon the units of measurement of the vectors. A variation of values in feature-vector with different dimension and units is observed in our feature-vector. Another strength of correlation is that, a value more than 0.8 indicates strong correlation and a value less than 0.5 indicate weak correlation.

In case of Euclidean distance, the feature vectors are normalized before finding difference to eliminate the scaling problem. Each element in the vector is divided by the maximum of the corresponding element in two vectors. It ensures that the value of Euclidean distance is in between 0 and 1.0. The lower the value, the higher is the similarity.

In our experiment, we have found that color histogram of a blob is a strong feature and does not change much due to variation of rotation and scaling [Swain91]. So, we have calculated color histogram of blobs in RGB color space and used them during matching. The histograms are matching by correlation method shown in equation 4.4.

$$Hist(H_i, H_j) = \frac{\sum_{k=1}^{b} H_i(k).H_j(k)}{\sqrt{\sum_{k=1}^{b}(H_i(k))^2 \sum_{k=1}^{b}(H_j(k))^2}} \qquad \ldots 4.4$$

Here H_i is the histogram of template i and H_j is the histogram of blob j. b is the total number of histogram bins. We have used widely used value [McKenna00] for number of bins (16 in this case) for each color channels, so total of 16^3 bins are there in each histogram.

The final equation for calculating measure of similarity between a feature-vector and a temporal template is shown in equation 4.5:

$$SIM(F_i, F_j) = w_1 C(F_i, F_j) + w_2(1 - D(F_i, F_j)) + w_3 Hist(H_i, H_j) \qquad \ldots 4.5$$

Here, three measurements can be given three different weight factors by w_1, w_2 and w_3, where $\sum_{i=1}^{3} w_i = 1$. Right now we have given equal emphasis to all of them, but it might happen in some cases that one measurement should be given less emphasis than others (i.e. color is not clearly visible, so histogram comparison is not significant in that case). The value of similarity varies in between -1.0 to 1.0. We can find the best match by checking the value in score matrix. Table 4.4 shows some typical value in a score matrix. In this example, there are 4 temporal templates of objects and 4 blobs in the current scene. The value in row 2 and column 1 (i.e. 0.71582 in this case) shows the measure of similarity between object 2 and blob 1.

Table 4.4: An example of Score matrix

Blob / Template	1	2	3	4
1	0.68127	0.25125	0.746725	0.83946
2	0.71582	0.746725	■	0.86354
3	0.45912	■	0.548117	0.61512
4	0.58245	0.606269	0.717308	■

If we assume that there is only one candidate blob for every object, then the location (column) of maximum score in a row indicates the best matched blob for an object. If this score is the highest score in that column, then a best match is found. Otherwise, this blob has best match with some other template. When no best match is found for a template, then that template might be lost or occluded and further processing is required to track that object. According to table 4.4, the template 1 is either lost or occluded, because it does not have any best match candidate blob. Whenever a blob did not find any best match template of object, then that blob might be a new object or occluded object. To distinguish a new object from occluded object, we have checked whether the blob bounding box overlap with any bounding box of other object. If it does not overlap, then it is considered as a new object in the scene; otherwise that object is located somewhere in the occluded blob. According to table 4.4, template 2 best matches with blob 3, template 3 best matches with blob 2 and template 4 best matches with blob 4. In some cases, we have found in our experimentas that an occluded blob might get best match with an object, but having lower score value (< 0.5). To avoid this situation, we have used a threshold in our matching criteria. A best matched value should be more than a threshold. Surprisingly we have found that the score value for a matched blob with an object is more than 0.9 most of the time. When an object found its best matched blob, then the location of the blob indicate current location of that object and the temporal template of that object is updated with the new values in the feature-vector.

4.2.2 Maintenance of Total Objects

From the previous discussion, it is obvious that the following five possible scenarios might occur during tracking [Collins00].

1. Continuation of an object
2. An object can disappear
3. New object can appear
4. Two or more objects can merge
5. An object can separate into several objects

Out of these, step 4 requires the most consideration and further processing. A continuation of an object occurs when a perfect match is found. To find a temporal consistency of an object, we have maintained an age and lost counter in our data structure for each template [Xu04, McKenna00]. Sometime a cluster of noisy pixels form a blob. The interesting thing is that, they disappear from the scene after sometime. This kind of noise is mostly found due to poor digitization of color by camera (i.e. cheap web cam).

We have solved this problem by a heuristic. Whenever we have found a continuation of an object, we have increased the age counter. If an object is successfully tracked for N frames, then the object is considered as legitimate and tracking result is shown for that object [McKenna00]. The value of N varies from 3 to 15 frames, depending on situations. Similarly, whenever an object is lost, the lost counter was increased. An object is lost when it goes out of the scene or there is no perfect match in the score matrix. When an object is lost for M frames, then the temporal template of that object is deleted from the list [Haritaoglu01]. The value of M depends on type of application. If we want to recognize an object for a long time, even though it goes out of the scene, then a higher value should be used for M (i.e. 10000 frames). If we have limitation of memory and the remembering of an object is not an important task, then we can use some smaller value for M. In our tracking system, the value of M was from 20 to 50 frames. Whenever an object reappears from occlusion, its feature vector match with one of the temporal templates and tracking of that object continues successfully afterwards.

4.2.3 Occlusion Handling

When two or more people cross each together, then one of them occludes the others and forms a merged blob. Tracking during occlusion is the most difficult task for a tracking system. Occlusion handling requires detection of occlusion and finding the location of a person in the occluded blob. Occlusion detection can be done by using motion information. We can predict the location of an object (i.e. blob bounding box) in the next frame by using motion information. We can predict whether two objects are going to be occluded in the next frame by comparing their predicted blob bounding box [Xu02]. If the predicted blob bounding box overlap, then they are considered as occluded object in the next frame and occlusion handling routine is applied to find their current position. For occlusion handling, we have used the histogram backprojection technique proposed by Swain *et al.* [Swain91]. They originally used this technique for object localization and segmentation in the scene [Huang99]. Later on, Xu *et al.* [Xu05] used this approach and introduced Spatial-Depth-Affinity matrix for segmenting object from occluded blob during occlusion. We have used the original idea proposed by Swain *et al.* [Swain91] to find location of an object in the occluded blob.

Histogram backprojection use the histogram of an object to find its location. As we calculate the histogram of each blob in every frame for feature matching, histogram backprojection does not require any additional calculation.

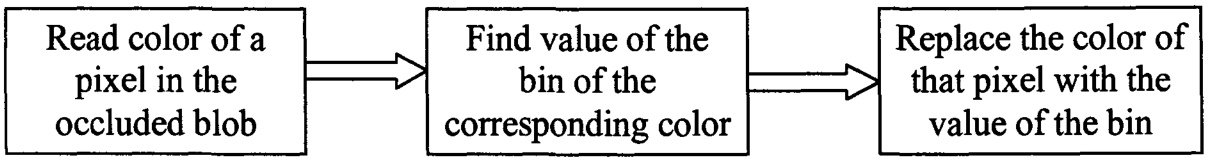

Read color of a pixel in the occluded blob
Find value of the bin of the corresponding color
Replace the color of that pixel with the value of the bin

Figure 4.9: Histogram backprojection

The main idea of histogram backprojection is that, histogram of a person (i.e. mostly his dress) represents distribution of his color. It can also be considered as an appearance model of a person [Mckenna00]. If two persons having different colors, then their color distributions will be different. If we check the value of a pixel in the occluded blob in both the histograms and if that pixel belongs to the object we are looking for, then the histogram of the target object best represents that color and it will have higher value than the other histogram in the corresponding histogram bin. If we replace the value of that pixel with the value of the histogram bin of the corresponding color, then that pixel will get some higher value. If the pixel was chosen from non-target object, then the histogram bin will contain lower value and the pixel's value will be replaced with that lower value. At the end, all the pixels in the occluded will be divided into two groups, one having higher value and the other having lower value. The higher value region corresponds to the location of target object in the occluded blob. Table 4.5 shows the algorithm of histogram backprojection proposed by Swain *et al.* [Swain91]. The step of operations in histogram backprojection is explained in figure 4.9 and 4.10 with an example.

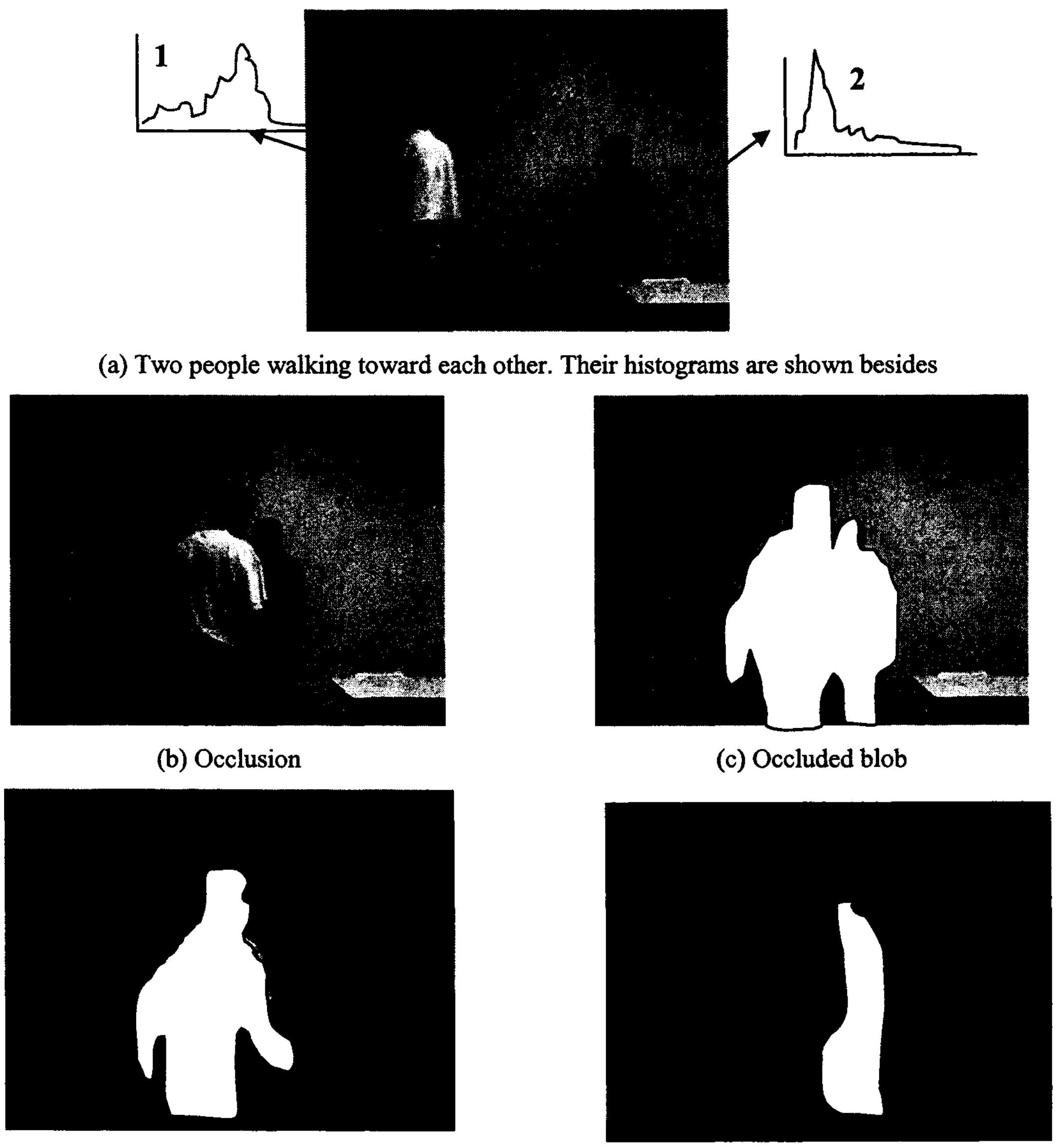
Table 4.5: Histogram Backprojection

```
Input: I and M are color histogram of image and model (target)
Output: Location of target model in the image
1. For each histogram bin j do
   $R_j = \min\left(\frac{M_j}{I_j}, 1\right)$
2. For each x, y do
   $b_{x,y} = R_{h(c_{x,y})}$
3. $b = D^r * b$ (convolution operation)
4. $(x_t, y_t) = \mathrm{loc}(\max_{x\,y} b_{x,y})$
```

Here R_j is a ratio histogram, (x, y) is the pixel coordinate in the model and (x_t, y_t) is the expected location of the target or model in the image, given the model appears in the image. D^r is a disk of radius r and calculated by equation 4.6:

$$D^r_{x,y} = \begin{cases} 1 & if \sqrt{x^2 + y^2} < r \\ 0 & otherwise \end{cases} \qquad \ldots 4.6$$

After we replace all the pixel values with the value in the histogram bin, we will find some connected components in the resultant image. In real-life, a person does not wear a single color dress. So, those pixels of the target object that does not match with histogram will get lower or zero value and the resultant image contains several connected components. If we find the largest connected component, then that component probably belongs to the target object and the location of that target object is updated with this new location. While tracking during occlusion, the objects temporal template is not updated except its position. We have assumed that the object's features do not change a lot during occlusion. Figure 4.10 shows an example of tracking during occlusion using histogram backprojection.



(a) Two people walking toward each other. Their histograms are shown besides

(b) Occlusion

(c) Occluded blob

(d) Result of histogram backprojection in occluded blob for target one

(f) Result of histogram backprojection in occluded blob for target two

Figure 4.10: Example of histogram backprojection to segment merged blob

There are some problems of histogram backprojection and we need to apply some additional technique to solve them. First of all, when an object is totally occluded, then histogram backprojection will not return any connected component. In this case, we have used blind tracking. In blind tracking, an object's position is estimated by its motion vector. In reality, this situation occurs for a very short period of time, because the object reappears from total occlusion after several frames.

Another problem of backprojection is that, when two people have similar color, then backprojection cannot segment the occluded blob into individuals, rather they segment the whole blob as a candidate region for the target. In this case, the center of the entire occluded blob is considered the new location of target object. Under this situation, the object will be tracked incorrectly for a short period of time, but their predicted position will be very close to the actual position and this inaccuracy will not affect significantly on overall accuracy of intelligent surveillance system, i.e. detecting stalking behaviour from tracking. Figure 4.11 shows our proposed framework for feature-based tracking of multiple people.

4.3 Conclusion

In this chapter, we have discussed different approaches for segmentation and tracking of multiple people. The background has been modeled by statistical method and histogram analysis for bootstrapping in dynamic scene. The segmentation has been done by clustering technique. For tracking, we have used feature-vector and histogram of blob. Feature matching and occlusion handling is critical for success of a tracking system. We have proposed a new feature matching technique for tracking system. The advantage of our equation is the dynamicity and robustness to differentiate between good and bad matching. In general, we have proposed a framework for tracking which is robust and faster.

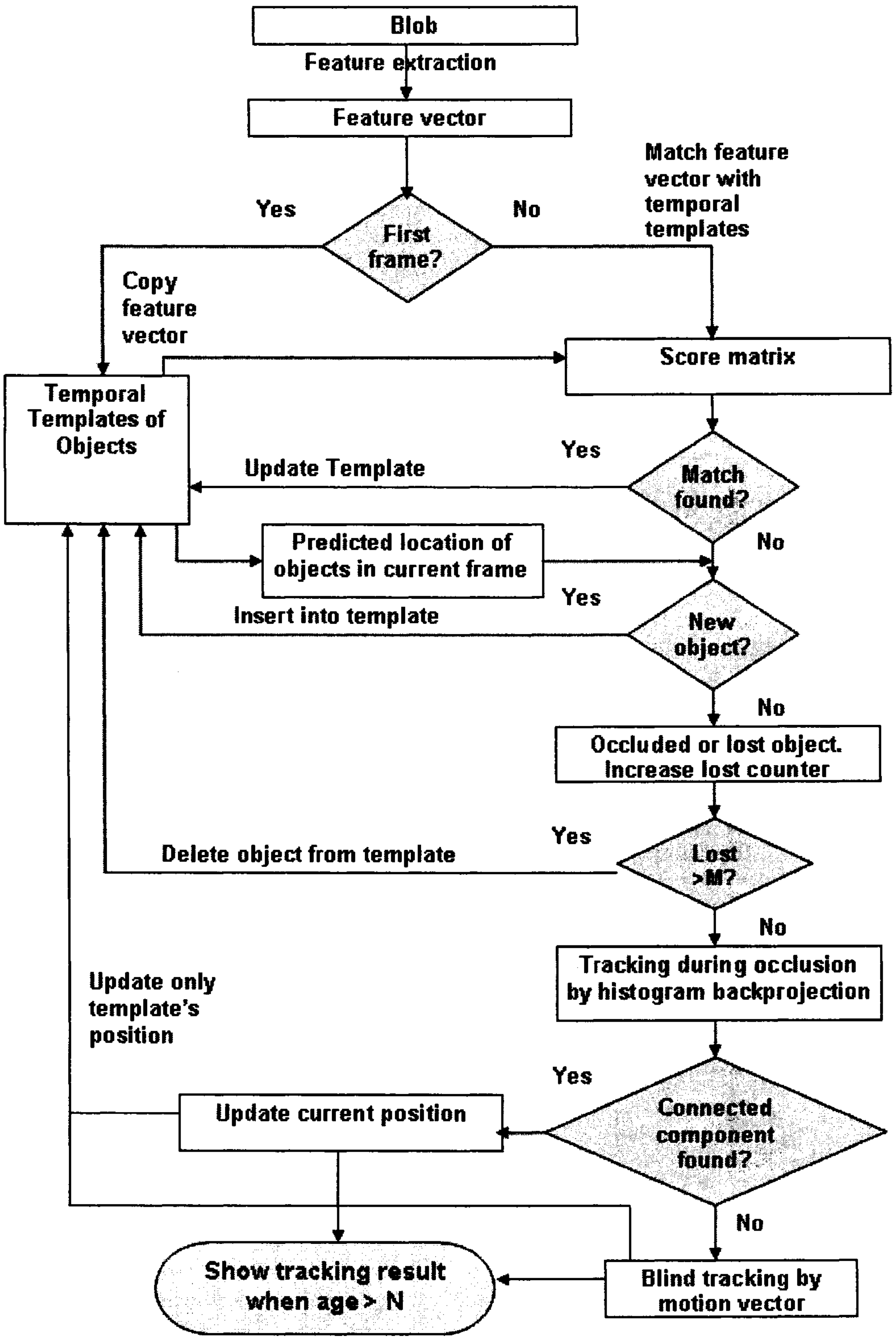
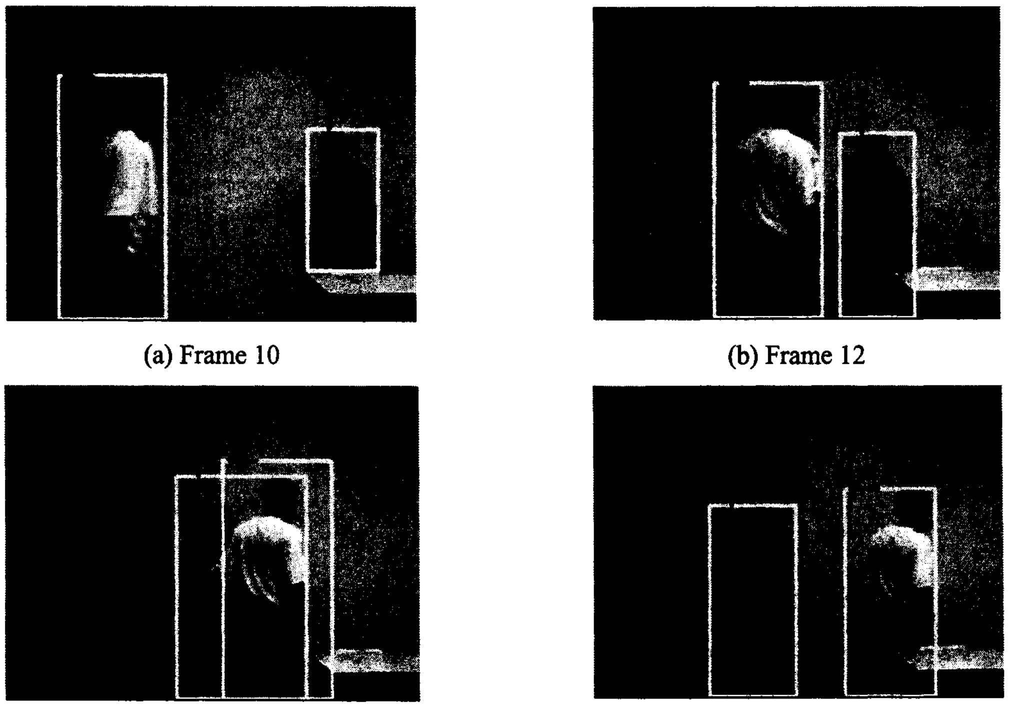

Blob
Feature extraction
Feature vector
Match feature vector with temporal templates
Yes
No
First frame?
Copy feature vector
Score matrix
Temporal Templates of Objects
Yes
Update Template
Match found?
No
Predicted location of objects in current frame
Yes
Insert into template
New object?
No
Occluded or lost object. Increase lost counter
Yes
Delete object from template
Lost >M?
No
Tracking during occlusion by histogram backprojection
Update only template's position
Yes
Update current position
Connected component found?
No
Show tracking result when age > N
Blind tracking by motion vector

Figure 4.11: Flow chart of feature-based tracking approach

CHAPTER 5
EXPERIMENTAL RESULT

We have implemented our tracking system using Visual C++ 6.0 and OpenCV [opencv]. We have evaluated our tracking system on two types of image sequences captured by a digital camera; they are images captured in burst mode and videos. In both cases, the resolution of images was 320x240 pixels.

The burst mode of a digital camera allows capturing 16 images within a short period of time. We have captured 4 of these image sequences in a classroom where two people are walking towards each other and one person is occluding the other one. One person is coming out from a partial occlusion by a background object (i.e. table). We have performed our initial experiment on these image sequences and achieved very good tracking accuracy. Figure 5.1 shows a result of tracking two people in a classroom.



(a) Frame 10

(b) Frame 12

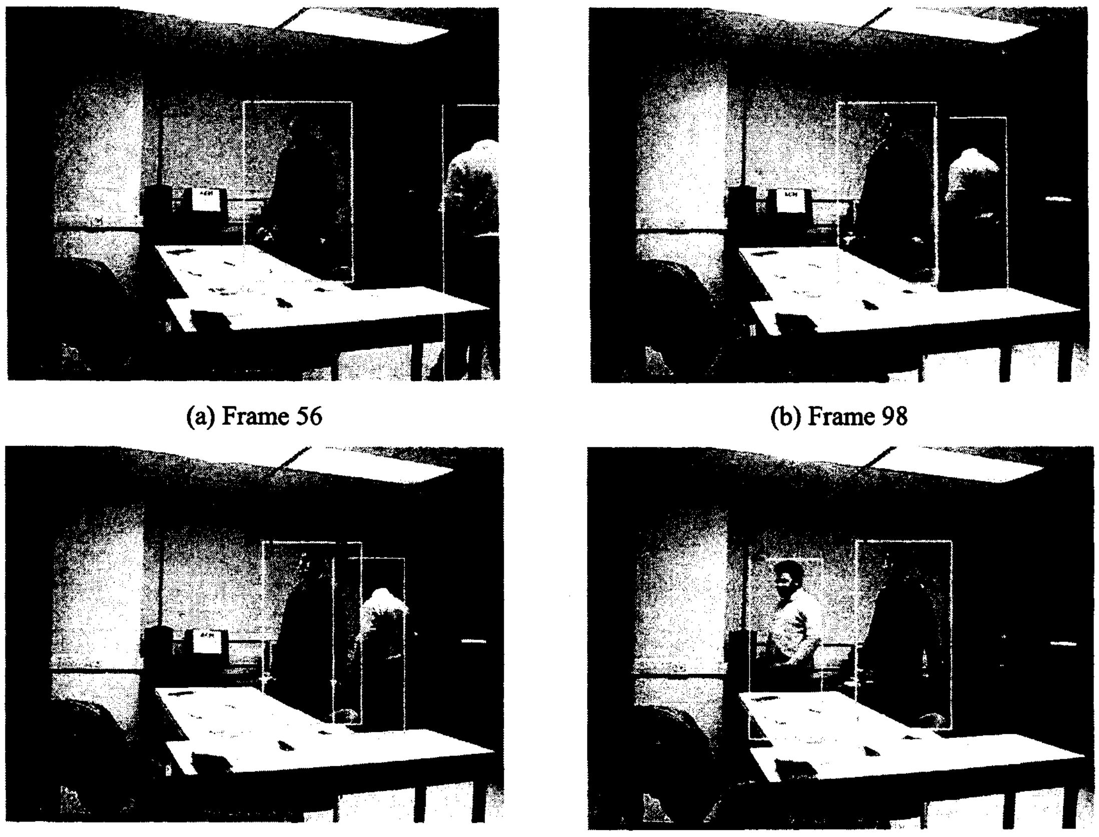
(c) Frame 14

(d) Frame 16

Figure 5.1: Result of tracking two people in a classroom

We would like to mention that the tracking result is shown with the blob bounding box and the assigned label (at the top of each blob bounding box) to each person in the original sequence of images. The label is assigned at the beginning of tracking and those labels are correctly kept even during occlusion. When they come out of the occlusion, they get the same label they had before occlusion.

Next, we have experimented with video. In this case, we have captured video of two places having different lighting conditions. The first place is a student lab and the second place is a large classroom. Each video was converted to an image sequence by a frame grabbing software. In both places, three people are moving and occluding each other. The tracking result is shown in figure 5.2 and 5.3.

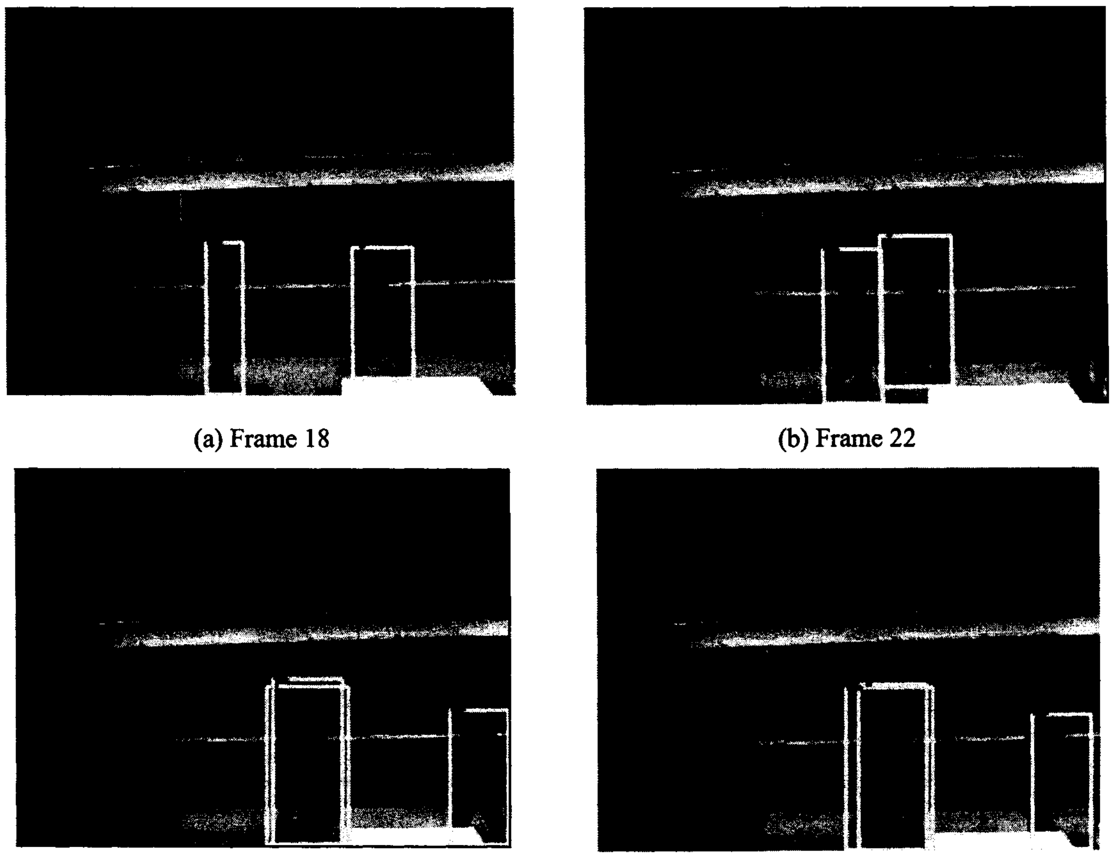


(a) Frame 56 (b) Frame 98

(c) Frame 100 (d) Frame 158

Figure 5.2: Tracking three people in a student lab

Figure 5.2 shows the result of tracking with a video having around 250 frames. In this figure, it is observed that the environment is more complex than figure 5.1 in terms of number of background objects. In this case, our tracking system performs with same accuracy and efficiency.


(a) Frame 18 (b) Frame 22

(c) Frame 24 (d) Frame 25

Figure 5.3: Tracking three people in a big classroom

Figure 5.3 shows the result of tracking in a big classroom. Here, the situation is more difficult than the previous situations, because of less amount of light in the environment and similarity of color of people. Our system successfully handled the occlusion problem in this case and tracked three people in the scene.

5.1 Analysis and Discussion

There are several parameters in our tracking system that affects the tracking result, such as minimum blob size, minimum track length, maximum lost frame etc. In case of image sequences captured in burst mode, the quality of image is better than quality of image sequences in video. Consequently, those image sequences have less segmentation error and people are better detected. In that case, minimum blob size is tuned to some higher value. In case of videos, the segmentation error is much higher and more split of blobs are observed. It requires a smaller value for minimum size of blob to consider all the split part of a person.

In case of images captured in burst mode, number of noisy pixels is much less than videos. So, the value of minimum track length was smaller to start tracking a person earlier. In case of videos, the noisy pixels cluster together, form a blob and appear in the scene for longer period. A higher value for minimum track length is required to avoid tracking those noisy blobs.

Another important parameter in our tracking system is the maximum lost frame. It was assigned a lower for videos, because more noisy blobs appear in video and they should be removed as soon as possible. If the maximum lost frame is higher, they remain in the memory for a longer period of time and they can make the system running out of memory.

One important observation we have found in the tracking result is that, sometime a person does not get the label in sequential order. It happens because when a person is assigned a label, there might be some noisy blobs in the memory and they are increasing their age counter. This phenomenon causes to assign labels to newly detected person in some random order. The labelling module always assign label based on the best available label staring from 1. When a noisy blob with lower value for label is deleted from the blob list and a new blob appears at that time, then it gets a lower value for label.

From our experiments, we have observed that our tracking system can process 10-15 frames per seconds. The speed varies depending on number of people in the scene and the number of occlusions. Of course, this speed is for 320x240 pixels images. For large size images, such as 640x480 pixels images, the system has to process 4 times more pixels and takes longer period of time to process each frame.

CHAPTER 6
CONCLUSION

Video surveillance and monitoring human activity is becoming an important research area in Computer Vision. The growing research in video-based human activity recognition is due to development in compression techniques in multimedia (i.e. video), lower hardware cost for camera and computer accessories, increased processing capability of computer and importance of security surveillance. This thesis is mostly motivated by numerous applications of automatic video surveillance in near future. In this thesis, we have analyzed different components of intelligent video surveillance and their challenges in real world. In particular, we have investigated in details a feature-based approach of tracking multiple people in video. From our investigation, we have suggested a robust framework for feature-based tracking of multiple people. Evaluation of a tracking system requires developing a complete system with some pre-processing steps. Consequently, we have also investigated background modeling and foreground region segmentation techniques.

Our tracking system has some advantages over other tracking systems in the literature. It is free from assumptions about structure of a person. The visibility of a part of human body (i.e. head or leg region) is not required. Moreover, we have showed that score matrix can be used to detect occlusion. In particular, our contributions in this thesis are:

1. We have proposed a combination of Euclidean distance and Pearson correlation coefficient for feature matching and histogram comparison.
2. Our approach of tracking is simple, which eventually resulted a fast tracking system.

We have observed some errors in our experimental result. Those errors are mostly due to segmentation error. Accuracy of tracking largely depends on accuracy of foreground region segmentation. Future work can be done to find more accurate foreground region by better modeling of background. Specially, some of the background issues (i.e. rapid variation of lighting condition, waving trees etc.) are still an active research topic and need to be solved in near future.

Some future extension of our work can be multi-camera surveillance and tracking. In that case, a mechanism for collaboration between cameras is required. Also, we have assumed that the camera is stationary. Future work can be done for tracking using moving cameras. An immediate extension of our tracking system can be detecting stalking behaviour of a person by analyzing his trajectory in the scene. Most tracking systems in the literature assume that people appear in the scene separately, which is not true all the time. When a person appears in the scene in a group, then use of appearance model for tracking does not work. Future investigation is required to detect a person within a group on initial appearance in the scene.

In our tracking system we have made extensive use of color, but color is not clearly distinguishable from a distance (i.e. black vs deep blue). This limitation is another reason for failure of a tracking system in some cases. Specially, when a person occludes other person having almost similar color, histogram backprojection fails in that case to separate two people. More work is required for accurate measurement of color or to find an alternative of color (i.e. texture) in those situations. In future, we will investigate more to solve the occlusion problem by splitting a person into sub-blobs and tracking by sub-blob matching. Human vision system has a superior capability than computer vision system, because human use his intelligence and experience for interpreting information from visual cues. In future, a vision system should be integrated with an AI or knowledge representation system for more accurate tracking.

REFERENCES

[Adams94] Adams, R.; Bischof, L.; "Seeded region growing", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 16(6), pp. 641 - 647, June 1994

[Aggarwal99] Aggarwal, J.K.; Cai, Q.; "Human motion analysis: A review", *Journal of Computer Vision and Image Understanding*, Volume 73(3), pp. 428-440, March 1999

[Anderson85] Anderson, C.; Burt, P.; Van der Wal, G.; "Change detection and tracking using pyramid transformation techniques", *Proceedings of SPIE - Intelligent Robots and Computer Vision,*, Volume 579, pp. 72–78, 1985

[Barron94] Barron, J.; Fleet, D.; Beauchemin, S.; "Performance of optical flow techniques", *International Journal of Computer Vision*, Volume 12(1), pp. 42–77, 1994

[Beymer97] Beymer, D.; McLauchlan, P.; Coifman, B.; Malik, J.; "A real-time computer vision system for vehicle tracking and traffic surveillance", *IEEE Conference on Computer Vision and Pattern Recognition*, June 1997

[Beymer99] Beymer, D.; Konolige, K.; "Real-Time Tracking of Multiple People Using Continuous Detection", *In Proceeding of International Conference on Computer Vision*, 1999

[Broek04] Broek, E. L. van den; Rikxoort, E. M. van; "Evaluation of color representation for texture analysis", *Proceedings of the Sixteenth Belgium-Netherlands Artificial Intelligence Conference (BNAIC)*, pp. 35-42, October 21-22, 2004 Netherlands - Groningen

[Bunyak05] Bunyak, F.; Ersoy, I.; Subramanya, S.R.; "A Multi-Hypothesis Approach for Salient Object Tracking in Visual Surveillance", *IEEE International Conference Image Processing*, Volume 2, pp. 446 – 449, 11-14 Sept. 2005

[Chellappa04] Chellappa, R.; Chowdhury, A. Roy and Zhou, S.; "Human Identification

Using Gait and Face", *The Electrical Engineering Handbook (3rd Ed), CRC Press*, 2004

[Chen04] Chen, B.; Lei, Y.; "Indoor and outdoor people detection and shadow suppression by exploiting HSV color information", *4th International Conference on Computer and Information Technology*, pp. 137 - 142, 14-16 Sept 2004

[Collins00] Collins, R. T.;Lipton, A. J.; Kanade, T.; Fujiyoshi, H.; Duggins, D.; Tsin, Y.; Tolliver, D.; Enomoto, N. and Hasegawa, O.; "A system for video surveillance and monitoring", *Technical Report CMU-RI-TR-00-12*, 2000 CMU

[Collins00a] Collins, R. T.; Lipton, A. J.; Kanade, T.; "Introduction to the special section on video surveillance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, pp. 745-746, August 2000

[Comaniciu00] Comaniciu, D.; Ramesh, V.; Meer, P.; "Real-time tracking of non-rigid objects using mean shift", *IEEE Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 142 – 149, 13-15 June 2000

[Cucchiara05] Cucchiara, R.; Grana, C.; Prati, A.; Vezzani, R.; "Computer vision system for in-house video surveillance", *IEE Proceedings - Vision, Image and Signal Processing*, Volume 152(2), pp. 242 – 249, 8 April 2005

[Cui95] Cui, Y.; Swets, D. and Weng, J.; "Learning-Based Hand Sign Recognition Using Shoslif-m", *Proc. Int'l Conf. Computer Vision*, pp. 631-636, 1995

[Dever02] Dever, J.; da Vitoria Lobo, N.; Shah, M.; "Automatic Visual Recognition of Armed Robbery", *Proceedings 16th International Conference on Pattern Recognition*, Volume 1, pp. 451 - 455, 11-15 Aug. 2002

[Dirks03] Dirks, W.; Yona, G.; "A comprehensive study of the notion of functional link between genes based on microarray data, promoter signals, protein-protein interactions and pathway analysis", *Technical Report*, 2003

[Efros03] Efros, A.A.; Berg, A.C.; Mori, G.; Malik, J.; "Recognizing action at a distance", *Proc. 9th IEEE International Conference on Computer Vision*, pp. 726-733, 13-16 Oct. 2003

[Faugeras93] Faugeras, O.; "Three dimensional computer vision - a geometric viewpoint", *Artificial Intelligence, M.I.T. Press, Cambridge, MA*, 1993

[Ford98] Ford, A.; Roberts, A.; "Colour Space Conversions", *http://www.poynton.com/PDFs/coloureq.pdf*, August 11, 1998

[Fujiyoshi98] Fujiyoshi, H.; Lipton, A.J.; "Real-time human motion analysis by image skeletonization", *Fourth IEEE Workshop on Applications of Computer Vision*, pp. 15 - 21, 19-21 Oct. 1998

[Gao04] Gao, Jiang; Collins, R.T.; Hauptmann, A.G.; Wactlar, H.D.; "Articulated Motion Modeling for Activity Analysis", *Conference on Computer Vision and Pattern Recognition Workshop*, pp. 20, 27-02 June 2004

[Gavrila99] Gavrila, D.M.; "The Visual Analysis of Human Movement: A Survey", *Computer Vision and Image Understanding*, Volume 73(1), pp.82-98, 1999

[Haritaoglu00] Haritaoglu, I.; Harwood, D.; Davis, L.S.; "W4: Real-time surveillance of people and their activities", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, pp. 809–830, August 2000

[Haritaoglu01] Haritaoglu, I.; Flickner, M.; "Detection and tracking of shopping groups in Stores", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 1, pp. I-431 - I-438, 2001

[Haritaoglu98] Haritaoglu, I.; Harwood, D.; Davis, L.S.; "W4: Who? When? Where? What? - A Real Time System for Detecting and Tracking People", *Proc. International Conference on Face and Gesture Recognition*, pp. 222–227, 14-16 April 1998

[Haritaoglu99] Haritaoglu, I.; Harwood, D.; Davis, L. S.; "Hydra: multiple people detection and tracking using silhouettes", *International Conference on Image Analysis and Processing*, pp. 280 – 285, 27-29 Sept. 1999

[Horprasert99] Horprasert, T.; Harwood, D.; Davis, L.; "A statistical approach for real-time robust background subtraction and shadow detection", *Proc of ICCV'99 FRAME-RATE Workshop*, 1999

[Hu04] Hu, W.; Tan, T.; Wang, L.; Maybank, S.; "A survey on visual surveillance of object motion and behaviors", *IEEE Transactions on Systems, Man and Cybernetics, Part C*, Volume 34(3) Aug. 2004

[Huang05] Huang, Y.; Essa, M.; "Tracking multiple objects through occlusions", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 1051-1058, 20-25 June 2005

[Huang99] Huang, J.; Kimar, S. R.; Mitra, M.; Zhu, W. -J.; "Spatial colour indexing and applications", *International Journal of Computer Vision*, Volume 35(3), 1999

[Indupalli06] Indupalli, S.; Ali, M. A.; Boufama, B.; "A Novel Clustering-Based Method for Adaptive Background Segmentation", *First International Workshop on Video Processing for Security in Proceedings of Third Canadian Conference on Computer and Robot Vision*, June 7-9, 2006 Quebec City, Canada

[Intille97] Intille, S.S.; Davis, J.W.; Bobick, A.F.; "Real-time closed-world tracking", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 697 – 703, 17-19 June 1997

[Isard98] Isard, M.; Blake, A.; "CONDENSATION—Conditional Density Propagation for Visual Tracking", *International Journal of Computer Vision*, Volume 29(1), pp. 5-28, 1998

[Ivanov98] Ivanov, Y.; Bobick, A.; Liu, J.; "Fast lighting independent background subtraction", *Proceedings in IEEE Workshop on Visual Surveillance*, pp.49 - 55, 2 Jan. 1998

[Ivanovic04] Ivanovic, A.; Huang, T.S.; "A probabilistic framework for segmentation and tracking of multiple non rigid objects for video surveillance", *International Conference on Image Processing*, Volume 1, pp. 353 – 356, 24-27 Oct. 2004

[Jepson03] Jepson, A. D.; Fleet, D. J.; and El-Maraghi, T. F.; "Robust online appearance models for visual tracking", *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 25, pp. 1296– 1311, 2003

[Khalaf01] Khalaf, R. Y.; Intille, S. S.; "Improving multiple people tracking using temporal consistency", *Massachusetts Institute of Technology, Cambridge, MA, MIT Dept. of Architecture House_n Project Technical Report*, 2001

[Lipton98] Lipton, A.J.; Fujiyoshi, H.; Patil, R.S.; "Moving target classification and tracking from real-time video", *Fourth IEEE Workshop on Applications of Computer Vision*, pp.8 - 14, 19-21 Oct. 1998

[McKenna00] McKenna, S. J.; Jabri, S.; Duric, Z.; Rosenfeld, A.; Wechsler, H.; "Tracking groups of people", *Computer Vision and Image Understanding, 80*, pp. 42-56, 2000

[McKenna99] McKenna, S. J.; Raja, Y.; Gong, S.; "Tracking colour objects using adaptive mixture models", *Image and Vision Computing*, Volume 17, pp. 225-231, 1999

[Moeslund01] Moeslund, T.B.;Granum, E.; "A survey of computer vision-based human motion capture", *Computer Vision and Image Understanding*, Volume 81(3), pp. 231-268, 2001

[Niu04] Niu, W; Long, J; Han, D. and Wang, F.; "Human Activity Detection and Recognition for Video Surveillance", *Proceedings of the IEEE Multimedia and Expo Conference*, 2004

[O'Connell02] O' Connell, D.J.; "Dense matching and image segmentation using projective geometry", *MSc. Thesis, School of Computer Science, University of Windsor*, 2002

[opencv] "Intel Open Source Computer Vision Library", *http://www.intel.com/technology/computing/opencv/index.htm*

[Paragios99] Paragios, N.; Deriche, R.; "Geodesic Active Regions for Motion Estimation and Tracking", *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, Volume 1, pp. 688 – 694, 20-27 Sept. 1999

[Peterfreund99] Peterfreund, N.; "Robust Tracking of Position and Velocity With Kalman Snakes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 21(6), June 1999

[Polana94] Polana, R.; Nelson, R.; "Low level recognition of human motion (or how to get your man without finding his body parts)", *IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, 1994

[Radke05] Radke, R.J.; Andra, S.; Al-Kofahi, O.; Roysam, B.; "Image change detection algorithms: a systematic survey", *IEEE Transactions on Image Processing*, Volume 14(3), pp. 294 - 307, March 2005

[Ridder95] Ridder, C.; Munkelt, O.; and Kirchner, H.; "Adaptive background estimation and foreground detection using Kalman-filtering", *Proc. Int. Conf. Recent Advances in Mechatronics*, pp. 193–199, 1995

[Sattar05] Sattar, J.; "A Visual Servoing System for an Amphibious Legged Robot", *Masters Thesis, School of Computer Science, McGill University*, August 2005

[Starner95] Starner, T. E. and Pentland, A.; "Visual recognition of american sign language using hidden markov models", *Proc. Intl. Workshop on Automatic Face- and Gesture Recognition*, 1995

[Stauffer00] Stauffer, C.; Grimson, W. E. L.; "Learning patterns of activity using real-time tracking", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, pp. 747-757, August 2000

[Stauffer03] Stauffer, C.; "Estimating tracking sources and sinks", *Proc of 2nd IEEE Workshop on Event Mining (in conjunction with CVPR'2003),* June 2003 Madison, Wisconsin

[Stauffer99] Stauffer, C.; Grimson, W. E. L.; "Adaptive background mixture models for real-time tracking", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 246-252, Jun 1999

[Swain91] Swain, M.; Ballard, D.; "Color indexing", *International Journal of Computer Vision*, Volume 7(1), pp. 11–32, 1991

[Toyama99] Toyama, K.; Krumm, J.; Brumitt, B.; Meyers, B.; "Wallflower: Principles and practice of background maintenance", *International Conference on Computer Vision*, pp. 255–261, 1999

[vsam] http://www.cs.cmu.edu/~vsam/Web2000/index.html

[Wang03] Wang, J. J. and Singh, S.; "Video analysis of human dynamics - a survey", *Real-Time Imaging*, Volume 9(5), pp. 321-346, 2003

[Web1] http://www.cs.otago.ac.nz/research/vision/Research/OpticalFlow/opticalflow.html

[Welch95] Welch, G.; Bishop. G.; "An introduction to the Kalman filter", *Technical Report TR 95-041, University of North Carolina, Department of Computer*, 1995

[Wiki1] http://en.wikipedia.org/wiki/HSL_color_space

[Wiki2] http://en.wikipedia.org/wiki/YUV

[Wiki3] http://en.wikipedia.org/wiki/Computer_vision

[Wiki4] http://en.wikipedia.org/wiki/Closed-circuit_television

[Wiki5] http://en.wikipedia.org/wiki/Kalman_filter

[Wren97] Wren, C.; Azarbayejani, A.; Darrell, T.; Pentland A.; "Pfinder: Real-time tracking of the human body", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 19(7), pp. 780–785, July 1997

[Xu02] Xu, M.; Ellis, T.; "Partial observation vs. blind tracking through occlusion", *British Machine Vision Conference*, pp. 777-786, Sept. 2002 Cardiff,

[Xu04] Xu, L.; Landabaso, J. L.; Lei, B.; "Segmentation and tracking of multiple moving objects for intelligent video analysis", *BT Technology Journal*, Volume 22(3), July 2004

[Xu05] Xu, L.; Puig, P.; "A hybrid blob- and appearance-based framework for multi-object tracking through complex occlusions", *Proceedings 2nd Joint IEEE International Workshop on VS-PETS*, 15-16 October, 2005 Beijing,

[Yazdi02] Yazdi, M; Zaccarin, A.; "Semantic object segmentation of 3D scenes using color and shape compatibility", *In Proceedings of 6th World Multiconference on Systemics, Cybernetics and Informatics*, July 2002 Orlando, FL, USA,

[Zhao04] Zhao, T.; Nevatia, R.; "Tracking multiple humans in complex situations", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 26(9), pp. 1208 – 1221, Sept. 2004

[Zhou01] Zhou, Q.; Aggarwal, J. K.; "Tracking and classifying moving objects from video", *Proceedings of 2nd IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 9 December 2001 Kauai, Hawaii, USA,

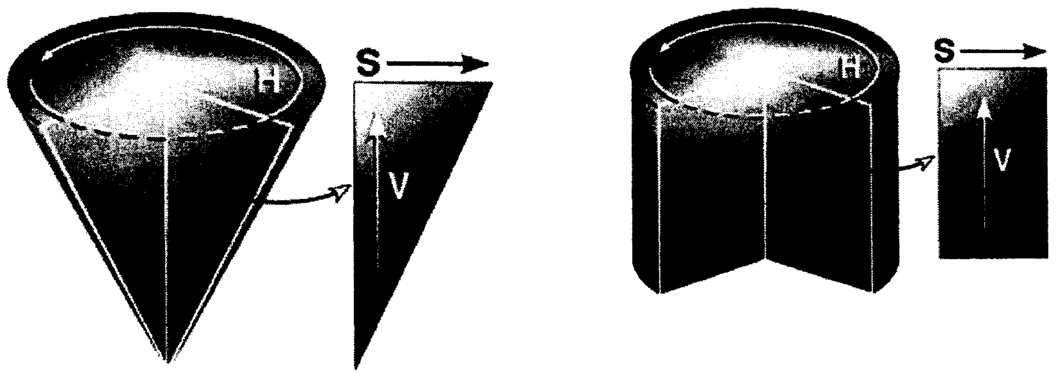
APPENDIX I

1. HSV/HIS Color spaces

In case of HSV color space, H and S are used to model color distribution. Hue corresponds to '**colour**', while S corresponds to '**vividness**' or '**purity**' [McKenna99]. If *R*, *G*, and *B* are between 0.0 and 1.0 and *MAX* and *MIN* equal the maximum and minimum of the (*R*, *G*, *B*) values, then following equations are used to calculae H, S and V:

$$H = \begin{cases} 60 \times \frac{G-B}{MAX-MIN} + 0, & \text{if } MAX = R \\ & \text{and } G >= B \\ 60 \times \frac{G-B}{MAX-MIN} + 360, & \text{if } MAX = R \\ & \text{and } G < B \\ 60 \times \frac{B-R}{MAX-MIN} + 120, & \text{if } MAX = G \\ 60 \times \frac{R-G}{MAX-MIN} + 240, & \text{if } MAX = B \end{cases}$$

$$S = \frac{MAX - MIN}{MAX} \qquad V = MAX$$



(a) Cone (b) Cylinder

Figure A.1: Different visualizations of HSV color space

2. Morphological operations

Morphology is a technique used extensively in image processing for changing shape of objects. The value of each pixel in the output image is determined by comparing the value of the corresponding pixel (in the input image) with its neighbors. By choosing the size and shape of the neighborhood, we can construct a morphological operation that is sensitive to specific shapes in the input image.

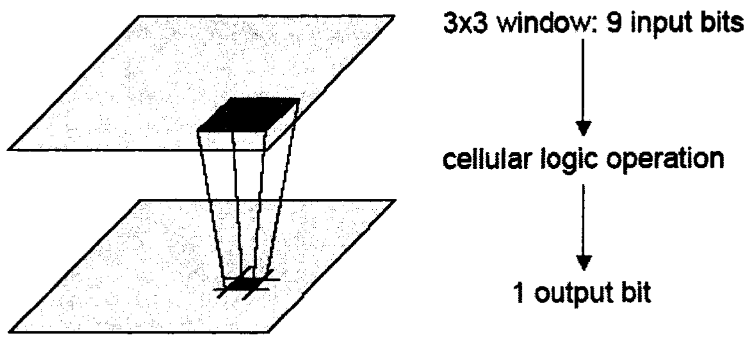


Figure A.2: Morphological operation

There are three primary morphological functions: *erosion*, *dilation*, and *hit-or-miss*. Others are special cases of these primary operations or are cascaded applications of them. For example, opening is erosion followed by dilation and closing is dilation followed by erosion. Morphological operations can be performed on binary images (value of a pixel is 0 or 1) or greyscale images (pixel values varies from 0 to 255).

In morphological operations, the neighbourhood (connectivities) is described by structuring element. It is a square grid, denoted by N_4 or N_8 and symmetric around the origin.

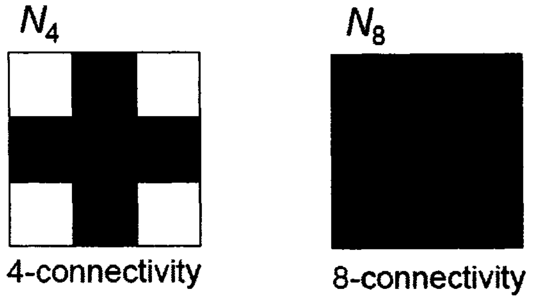


Figure A.3: Structuring elements

Dilation

In dilation, the value of the output pixel is the *maximum* value of all the pixels in the input pixel's neighbourhood. In other words, it expands each object pixel with its N_4 (N_8) neighbours (Add 4 (8) neighbour pixels in the set). For binary image, if any neighbour of a pixel has value 1, the pixel is set to 1. If A is a binary image, B is a structuring element, the dilation is denoted by $A \oplus B$.

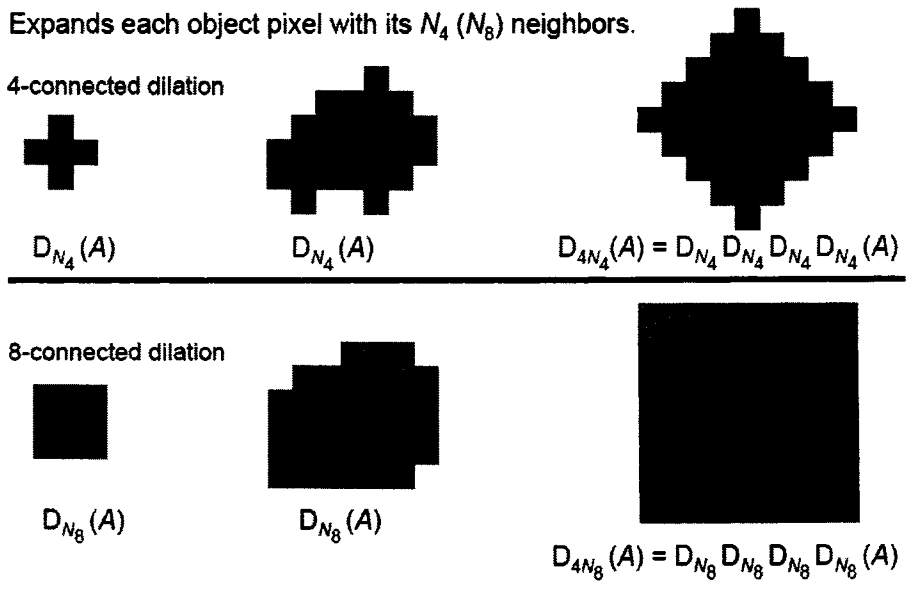

Expands each object pixel with its N_4 (N_8) neighbors.
4-connected dilation
$D_{N_4}(A)$
$D_{N_4}(A)$
$D_{4N_4}(A) = D_{N_4}D_{N_4}D_{N_4}D_{N_4}(A)$
8-connected dilation
$D_{N_8}(A)$
$D_{N_8}(A)$
$D_{4N_8}(A) = D_{N_8}D_{N_8}D_{N_8}D_{N_8}(A)$

Figure A.4: Dilation with 4 and 8 connectivity structuring element

Erosion

In erosion, the value of the output pixel is the *minimum* value of all the pixels in the input pixel's neighbourhood. In other words, it removes each object pixel having a neighbour in the background. It yields all pixels whose N_4 (N_8) neighbours are all object pixels. For binary image, if any neighbour of a pixel has value 0, the pixel is set to 0. If A is a binary image, B is a structuring element, the erosion is denoted by $A \Theta B$.

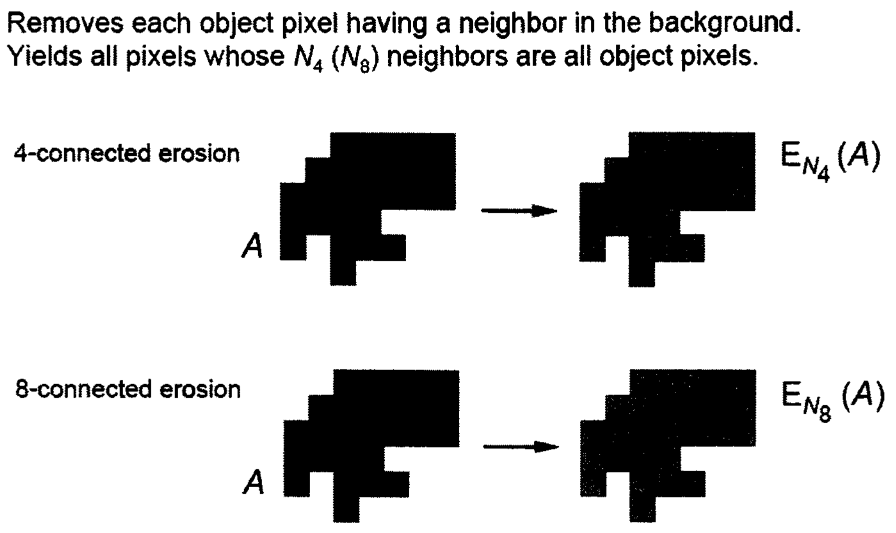

Removes each object pixel having a neighbor in the background.
Yields all pixels whose N_4 (N_8) neighbors are all object pixels.

Figure A.5: Erosion with 4 and 8 connectivity structuring element

Duality: erosion vs dilation

There is a duality between erosion and dilation. Erosion followed by dilation or dilation followed by erosion retains the original shape of object, but this process can remove noise or can connect two disconnected shape in the image. For example, erosion followed by dilation can remove noise like these:

Due to duality of erosion and dilation, N number of erosion followed by N number of dilation can delete large noise, but retain almost the original shape of foreground objects.

Conditional Dilation

Conditional dilation can be used to avoid connecting disconnected components. Conditional dilation essentially involves dilation of an image followed by intersection with some other "condition" image.

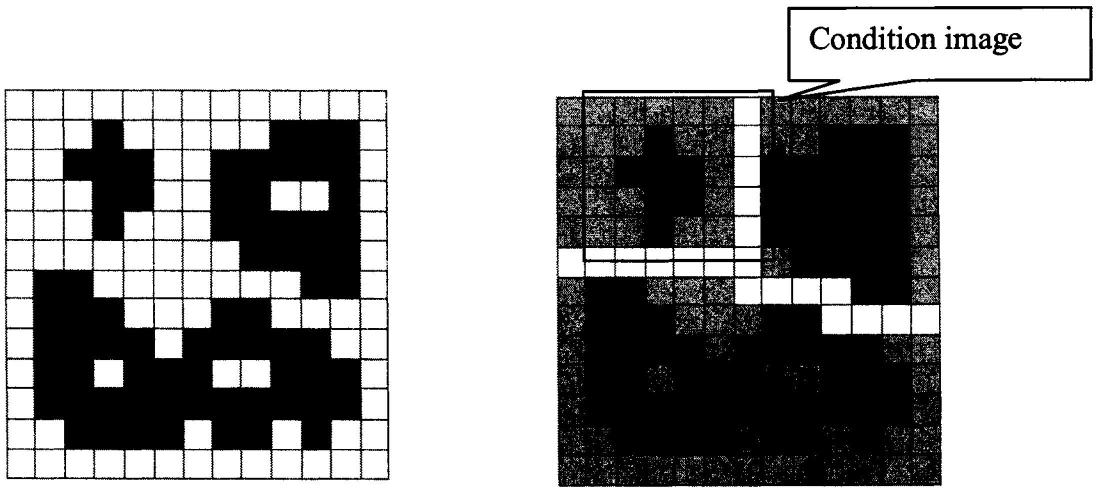


Figure A.6: Conditional dilation

In other words, new pixels are added to the original image if the pixel is in the dilation of the original, and the pixel is in the condition image. In this way, the condition image acts like a mask on the dilation.

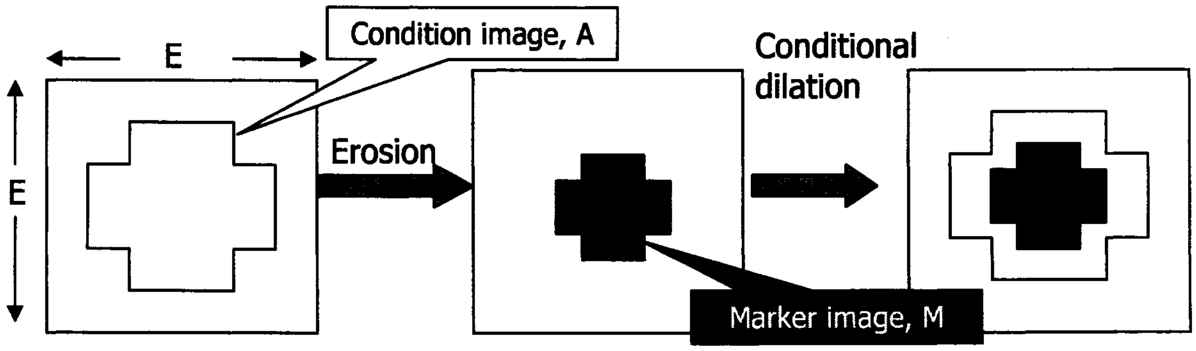


Figure A.7: Conditional dilation with mask image

M $\subseteq$ A is called the marker having the value of pixel is "1". The conditioned dilation is defined as $D^1(M,A) = (M \oplus B) \cap A$. Sometime we need to do dilation with the condition image as the original image; otherwise it may grow outside the original image.

3. Kalman filter

Kalman filter is a recursive filter that can predict the value of a dynamic system [Wiki5]. It updates the model continuously from measurement. It is an efficient technique for prediction, because it can predict the state of a system even in the case of faulty measurement. It can also find a good estimation of a state in the past by interpolation or smoothing.

Kalman filter assumes linearity in dynamic system and use a Gaussian noise. A vector of real numbers is used to represent the state of the system [Wiki5]. A simple version of Kalman filter is 1D Kalman Filter, which is described below.

Suppose we want to estimate an unknown state, *s*, at time *t* (s_t). If the estimate is based on the measurement, *m*, at time *t* (m_t), then estimate s_t is calculated by:

$$s_t = s_t^- + k_t(m_t - hs_t^-)$$

Where, $s_t^- = as_{t-1}$ and $m_t = hs_t + w$

Here s_t^- is an initial estimator and k_t is called kalman gain. *a* and *h* are rate and *w* is a random variable (Gaussain noise). Kalman gain can change with time.

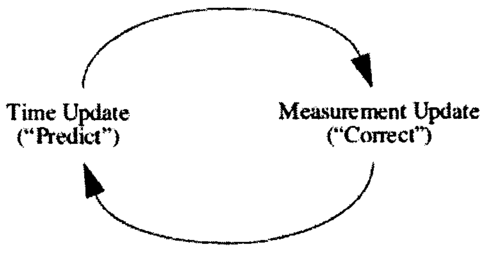


Figure A.8: Mechanism of operations in Kalman filter

VITA AUCTORIS

Mohammad Ahsan Ali was born in 1976 in the district of Kushtia, People's Republic of Bangladesh. He earned his B.Sc. Engineering in Computer Science and Engineering in 2001 from Bangladesh University of Engineering and Technology, Dhaka. Mohammad Ahsan Ali is currently a candidate for the Master's degree under the supervision of Dr. B. Boufama in the School of Computer Science at the University of Windsor, Ontario, Canada and expecting to graduate in Summer 2006.