Human motion detection and tracking in videos

Guo, Jing

2007

Guo, J. (2007). Human motion detection and tracking in videos. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2642>

<https://doi.org/10.32657/10356/2642>

Nanyang Technological University

Downloaded on 26 Apr 2023 18:38:10 SGT

NANYANG TECHNOLOGICAL UNIVERSITY

Human Motion Detection and Tracking in Videos

A thesis submitted to Nanyang Technological University

by

Guo Jing

in fulfillment of the requirement for the degree of Master of Engineering

Jan 2007

Acknowledgments

I would like to thank all the people that have helped or accompanied me during my study in Nanyang Technological University(NTU), Singapore.

First of all, I would like to express my thankfulness towards my supervisors, Dr Deepu Rajan and Dr Chng Eng Siong, without whom this thesis is impossible. Dr Deepu's carefulness and preciseness have made great impression on me, while Dr Chng's cheerfulness and positive attitude have also inspired me a lot.

I also want to express my sincerest gratitude towards Dr Graham Leedham, who is the Head of Computing Systems at the time of this thesis, for his trust, encouragement and help.

Besides, I want to thank the staffs and members of CeMNet, for providing me an ideal research environment for exploring new ideas.

Special thanks must given to all my friends in Singapore, especially Yuxia, Yiqun and Zhou Bo, for their help in one way or another and all my friends in China, for their endless support and encouragements.

Last but not least, I would like to thank the people I love. Their unconditional support had made me who I was and will become and it is their love that makes my life meaningful and colorful.

Contents

Acknowledgments	i
List of Figures	v
Abstract	viii
1 Introduction	1
1.1 Human Motion Detection and Tracking	1
1.1.1 Motion Detection	1
1.1.2 Tracking	3
1.1.3 Applications	5
1.2 Research Motivation and Objectives	6
1.3 Contributions	8
1.4 Organization of the thesis	11
2 Related Work	12
2.1 Motion Detection	12
2.2 Single Camera Tracking	15
2.2.1 Initialization	16
2.2.2 Representations	17
2.2.3 Tracking over time	19
2.3 Multi-camera tracking	24
2.3.1 Feature Based Approaches	25
2.3.2 Approaches Based on 3D Information	27
2.3.3 Alignment Based Approaches	29

3	Motion Detection	31
3.1	Entropy Based Motion Detection	31
3.1.1	Spatial-Temporal Entropy Image(<i>STEI</i>)	31
3.1.2	Difference-based Spatial Temporal Entropy Image(<i>DSTEI</i>)	35
3.1.3	Experiments and Discussions	37
3.2	Motion Detection With Adaptive Background And Dynamic Thresholds	41
3.2.1	Otsu's Thresholding Method	43
3.2.2	Average Noise Level and Normalized Absolute Difference	44
3.2.3	Motion Detection Based On Dynamic Thresholding	45
3.2.4	Experimental Results and Comparison With Gaussian Mixed Model(<i>GMM</i>) Method	48
3.3	Conclusion	50
4	Single Camera tracking	53
4.1	Mean Shift Object Tracking	53
4.1.1	Mean Shift Tracking Algorithm	54
4.1.2	Experimental Results and Discussion	56
4.2	Tracking by Motion Region Correspondence	59
4.2.1	Process Motion Detection Results	59
4.2.2	Tracking Over Time	60
4.2.3	Experimental Results and Discussion	62
4.3	Difference Image based Affine Tracking	65
4.3.1	Affine based tracking and its limitations	66
4.3.2	Obtaining Difference Image Mask	68
4.3.3	Improved Affine Tracking	71
4.4	Conclusion	75
5	Multi-Camera Tracking	76
5.1	Multi-Camera Tracking	76
5.2	Homography Based Cross Camera Correspondence	77
5.3	Experimental Results	80
5.4	Collaborative Tracking Between Cameras	82
5.5	Conclusion	84

6 Conclusion	85
6.1 Conclusion	85
6.2 Future work	86
References	90
Publications	101

List of Figures

1.1	Human motion detection and tracking	3
3.1	Pixels used to accumulate histogram for pixel location (i,j) at Frame(k) .	32
3.2	Experiment result on PETS 2001 datasets2 (camera2). (a) original image (b) cropped original image where motion occurs (c) cropped <i>STEI</i> image (d) cropped <i>DSTEI</i> image	34
3.3	Experimental results. First row: original image. Second row: <i>DSTEI</i> image. Third row: detected motion region. (a) is the 116 th frame of the <i>hall_monitor</i> sequence. (b) is the 215 th image of a walking sequence. (c) is the 587 th testing frame of the PETS 2001 dataset2 (camera1)	38
3.4	Influence of different α . (a) $\alpha = 0.9$ (b) $\alpha = 0.8$ (c) $\alpha = 0.7$	39
3.5	Experimental results on a running sequence. (a) the 63 ^{ed} frame; (b) <i>DSTEI</i> image of the 63 rd frame; (c) detected motion region of the 63 ^{ed} frame; (d) is the 76 th frame (e) <i>DSTEI</i> image of the 76 th frame (f) detected motion region of the 76 th frame	40
3.6	Experimental results on a indoor walking sequence. (a) the 243 rd frame of the sequence (b) <i>DSTEI</i> image of the 243 rd frame (c) detected motion region of the 243 rd frame.	40
3.7	Motion detection based on Otsu's thresholding method	42
3.8	Computed <i>NAD</i> for a test image sequence	45
3.9	Motion detection of a two-person walking sequence. The five columns are the 125 th , 160 th , 200 th , 260 th and 380 th frame. The first row shows the orig- inal frames, and the second and third row are foreground masks obtained by our method and GMM method.	51

3.10	Motion detection of another two-person walking sequence. The five columns are the 520 th , 600 th , 700 th , 790 th and 890 th frame. The first row shows the original frames, and the second and third row are foreground masks obtained by our method and GMM method.	52
3.11	Motion detection of a running sequence. The five columns are the 18 th , 30 th , 47 th , 75 th and 93 th frame. The first row shows the original frames, and the second and third row are foreground masks obtained by our method and GMM method.	52
4.1	Mean shift tracking on a one-person walking sequence: (a), (b), (c), (d) is the 153 rd , 163 rd , 173 rd , 183 rd frame respectively.	57
4.2	Mean shift tracking on a two-person walking sequence, person 1: (a), (b), (c), (d) is the 100 th , 120 th , 145 th , 155 th frame respectively.	57
4.3	Mean shift tracking on a two-person walking sequence, person 2: (a), (b), (c), (d) is the 100 th , 120 th , 130 th , 150 th frame respectively.	57
4.4	Mean shift tracking on another two-person walking sequence person 2: (a), (b), (c), (d) is the 170 th , 178 th , 190 th , 200 th frame respectively.	58
4.5	Processed motion detection and tracking result of the 135 th frame of a two-person walking sequence (a)foreground mask by motion detection (b)the projection of the foreground mask onto the horizontal axis (c)obtained object candidates shown in the bounding box (d) tracked objects with different bounding box colors	60
4.6	Examples of image frames, detected foreground regions and their projection histograms. (a),(b),(c),(d),(e) is the 130 th , 140 th , 170 th , 182 th , 202 th frame of a image sequence respectively.	61
4.7	Tracking results of a running sequence. The frame numbers of (a) to (h) are 17, 24, 29, 37, 64, 77, 85, 93.	63
4.8	Tracking results of a two-person walking sequence. The frame numbers of (a) to (h) are 65, 96, 115, 140, 160, 180, 205, 230.	63
4.9	Tracking results of the hall monitor sequence. The frame numbers of (a) to (h) are 30, 40, 58, 74, 89, 104, 113, 121.	64
4.10	Face tracking	69

4.11	Affine tracking on a two person walking sequence. (a) track the person walking in from right side; (b) track the person walking in from left side.	69
4.12	Difference image mask	71
4.13	Projection of difference image mask	72
4.14	Improved affine tracking on sequence 1. (a) shows the results obtained by original affine tracking method; (b) shows the results obtained by improved affine tracking method	74
4.15	Improved affine tracking on sequence 2. (a) shows the results obtained by original affine tracking method; (b) shows the results obtained by improved affine tracking method	74
5.1	Sample frames of one person walking ((a) and (b)) and two-person walking ((c) and (d)) taken from two cameras simultaneously	78
5.2	Illustration of homography based cross camera correspondence	80
5.3	Tracking results of two sequences recording the same scene. The frame numbers from (a) to (d) are 155, 170, 190, 216. The top row shows images from camera 1 and the second row shows those from camera 2.	80
5.4	Tracking results of a two-person walking sequence recorded by two cameras. (a1) to (h1) show images taken by camera 1, and (a2) to (h2) show images taken by camera 2. The frame numbers of (a1) to (h1) and (a2) to (h2) are 100, 110, 125, 175, 200, 225, 230, 250.	81
5.5	Sample frames of the composite image sequence. (a) is the 155 th frame taken by camera 1. (b), (c), (d) are the 170 th , 190 th and 216 th frames taken by camera 2.	83
5.6	Sample tracked persons of the composite image sequence. (a) is the 155 th frame taken by camera 1. (b), (c), (d) are the 170 th , 190 th and 216 th frames taken by camera 2.	83

Abstract

Human motion detection and tracking are very important research areas in vision based video processing. This thesis presents our efforts to understand and improve human motion detection and tracking techniques. Since motion detection is important both in the initialization and the tracking process itself, we examined the existing motion detection approaches and proposed our own methods. The first is entropy based motion detection, where the entropy of accumulated difference image is used to detect motion regions. Another simple yet effective motion detection method calculates the optimum threshold for every difference image. Single camera tracking follows motion detection and a mean shift tracking algorithm is firstly implemented. Its advantages and limitations are analyzed and possible improvements are given. We then proposed a single camera tracking method by corresponding detected motion regions. Experimental results demonstrate the effectiveness of our algorithm. A motion detection method based affine tracking method is also presented. Finally, some multi-camera tracking problems are addressed. A homography based method is used for cross camera correspondence. Given multiple cameras, more information is available for the same person. Some preliminary results on selecting the front view of a walking person is shown. Besides, a literature survey on the related areas is also included in the thesis.

Chapter 1

Introduction

1.1 Human Motion Detection and Tracking

Human motion detection and tracking by processing information from videos taken by one or more cameras are very active research areas in computer vision [1][2][3] and are one of the major areas in video analysis of human dynamics [4].

Video analysis of human dynamics is an research area devoted to detecting and tracking people(whole body or individual body parts such as face, hand and finger etc.) by processing video frames taken by one or more cameras, aiming to understand human dynamics or behaviors in different environments.

1.1.1 Motion Detection

Motion detection is a fundamental problem in computer vision, with various applications such as object tracking, human motion analysis and human-computer interaction.

Simply speaking, motion detection is a process of segmenting the motion regions(objects) from the rest of the image in a video sequence. Based on different applications, the target motion objects could be people, vehicles, animals, or military targets etc. We focus in this thesis motion detection of people moving both indoor and outdoor.

It is easy for a human being to find a moving object in a video clip, whether the person has just started moving or stopped for a while before he resumes, whether the

CHAPTER 1. INTRODUCTION

scene behind the person is moving or not, or whether there is multi-person moving or only one person. However, for a computer program, it is hard to detect in video sequences exactly the motion regions(containing moving people or other objects) that's meaningful to a human observer.

The video could have been taken with a static camera(i.e. camera fixed to a specific location when taking the video) or moving camera. We say the moving objects as foreground and the scene behind the objects background. So in this definition, we have video sequences that are with static background or moving background. In the thesis, motion detection with static background is described.

For videos taken outdoor, like surveillance videos, overall change of pixel values can be caused by illumination variations due to sunshine, clouds, shadows, moving trees and even passing birds etc; local change of the background scene may be brought by location changes of objects, removal of objects or addition of new objects(like cars)etc. There are also other possibilities of uncertainty changes brought by moving trees depending on the winds.

For videos taken indoor, like meeting videos, overall change of pixel values can be caused by illumination variation due to light, shadows, reflections etc; local change of the scene may also be brought by location changes of objects, removal of objects or addition of new objects(like moving of chairs, removal of a picture on the wall etc).

In Figure 1.1, an abstracted example is given to illustrate the concept of human motion detection and tracking. Figure 1.1(a) depicts the 3D scene at time t_3 , while (b), (c), (d) show the images taken by a static camera at t_1 , t_2 , t_3 , where t_1 is the earliest time point. More analysis about this figure are given in the following subsection, where human tracking process is discussed.

Figure 1.1: Human motion detection and tracking

1.1.2 Tracking

Tracking is the process of obtaining information about the scene, objects, or the camera, given a sequence of images. Most of the tracking algorithms assume the motion model of the object and based on this model and a template at the first image to quickly locate the objects in the next image.

In the example shown in Figure 1.1, getting the motion information of “A” and “B” at

CHAPTER 1. INTRODUCTION

time t_3 by using information from time t_2 are a process of tracking. In Figure 1.1(c), the identities of the two objects are given, e.g. “A”, “B” and now the two detected regions have some contextual meaning rather than just two collection of image pixels. Giving identities to detected objects can be seen as the process of object recognition, which is usually based on the results of motion detection and(or) tracking. In Figure 1.1(d) a new person “C” joins the scene, where “A” and “B” are still being tracked. The process of getting the locations of “A” and “B” in Figure 1.1(d) based on information from Figure 1.1(c) is the basic of “tracking”. From Figure 1.1 we can see, if we detect the objects at each frame, and find out their identities by using recognition techniques, then we don’t need to do tracking. But why tracking is still so important? There are at least the following reasons:

- Detection and recognition at each frame can be expensive. Nowadays, with increasing computing speed and capacity, raw detection can be done in real time, however, in most applications, motion detection is just a very fundamental step of the whole process, CPU has more things to do [5]. If tracking is employed, some processing cycles can be saved by using the information about objects from previous images.
- Motion detection may not get the exact motion objects. Sometimes due to the changing background(illumination, transposed background objects etc.), the motion detection results are far from perfect. If we rely solely on the motion detection results, the objects may be lost and background region maybe falsely detected as moving objects.

Object detection and tracking are closely interrelated areas. Most of the tracking system starts with motion detection [1] and when the target object is lost, motion detection is needed again to restart and resume the tracking.

1.1.3 Applications

Detecting and tracking human are very important for many computer vision tasks and the various applications mainly fall into the following categories:

- **Visual surveillance.** Modern intelligent surveillance systems aim to replace the traditional passive video surveillance systems with human operators employed to analyze the videos and monitor the site [1][6]. With motion detection and tracking abilities, the surveillance system would be able to analyze the site information and be able to report abnormal events, raise alarm, or compute flux of people at important areas etc. Based on different applications, higher level processing like face recognition to realize automatic access control to special areas or to help police to catch suspects in airport etc.
- **Smart space.** With the emergence of “Pervasive Computing” in the late 1980s [7], which states that computing should be everywhere anytime and should back into the environment without disturbing people, “smart space” [8] as a physical realization of such concept has been a hot research area since then [9]. An ideal smart space should be able to detect and track the person inside, understand the interactions between people and to provide proper service. The fundamental tasks for such smart space are to detect and track the person inside, update status when new person enters and stop tracking the person who has left the space, all of which are not possible without human motion detection and tracking.
- **Human-Computer Interaction.** Computers nowadays have very fast speed and big capacity, however, they don’t “understand” people - how we feel, how we behave, how we communicate with each other. Human motion detection and tracking provide some possibilities for computers to “understand” us by processing (real time

in many applications) videos from one or more cameras attached to computers. Advanced human-computer interface systems may use both speech and vision methods to analyze human motion to allow people to control and command in a high level interaction with computerized systems or multimedia databases. Examples of human detection and tracking systems can be found in [10][11][12].

1.2 Research Motivation and Objectives

Visual information is essential for humans and ‘computer vision’ is the area of research which aims to duplicate human vision by automatically analyzing and understanding images and image sequences obtained by cameras. Therefore, among many different possible sensors, the cameras seem very important.

Tracking is one basic process in a big research category called human motion capture [6], which is the process of capturing body movements of a subject at some resolution (i.e., the target can be a subject’s limbs, faces, as well as the subject as a whole). Human motion capture aims at analyzing human actions by a computer, which include broad research areas of tracking, pose estimation and recognition etc, among which tracking is a very fundamental and critical part.

Vision based human tracking is a very broad research field that may be addressed from various viewpoints: model-based vs non-model-based, 2D approaches vs 3D approaches, sensor modality (visual light, infrared (IR) light, range data, etc.), number of sensors, mobile vs stationary sensors, low level tracking vs tracking for pose estimation or recognition, one person vs multiple persons, whole body tracking vs body part tracking, distributed vs centralized processing, various motion-type assumptions (rigid, nonrigid, elastic), etc. In this thesis, various tracking methods are categorized based on the number of sensors used, i.e. single camera tracking or multi-camera tracking.

CHAPTER 1. INTRODUCTION

There are numerous single-camera tracking algorithms. If we only want to track a single person in a limited space to a certain detail, a single camera is enough. But when we want to track people in a larger space or when there are many people or we want to get detailed information about one person, like the 3d information of the person, a single camera tracking system will not suffice. All single camera tracking methods try to use only 2D information to keep track of a 3D object, and are challenged by occlusion, partial occlusion as well as appearance changes. So some researchers have developed multi-camera tracking systems to better solve these problems.

The advantages of multi-camera tracking over single camera tracking are obvious in the following aspects:

- Larger coverage of the scene.

It is not possible for one camera to provide adequate coverage of the environment because of limited field of view (FOV)[13].

- Minimise the effects of object occlusions

It is desirable to have multiple cameras observing critical areas, to provide robustness against occlusion[14], which is impossible when only one camera is used.

- 3D information of the target object.

Some applications need to know the 3D location of the target object and with multiple cameras, more reliable information can be obtained. In some other situations, the 3D model of the object is necessary, which requires information from multiple views.

- Better view point selection.

Under some circumstances, the best view of people is selected among multiple cameras, and the selected view is used as the input for further processing. In [15],

three cameras are used and the one which captures most of the front face is chosen as the best view.

When tracking in videos, we should be able to track multiple persons and should be able to track them wherever they are and even under severe occlusions, so a multi-camera tracking system is more capable than a single camera tracking system. This leads to our research objective: build a robust multi-camera tracking system on the basis of good human motion detection and single camera tracking methods.

1.3 Contributions

In this thesis, two motion detection methods and two single camera tracking methods are proposed and elaborated. Besides, Gaussian Mixed Model based motion detection method [16] and mean-shift based tracking algorithm [17] are implemented and the corresponding experimental results are presented and compared with those of our methods. Based on our proposed motion detection and single camera tracking methods, a homography based multi-camera tracking method are used to acquire useful information from videos.

Before going into details of our proposed methods, we first introduce the concepts used both in the following and through the thesis to make a clear understanding.

- **Frame** I is used to denote one frame from a video sequence and I_t is used to denote the frame at time t or the t_{th} frame in the sequence.
- **Pixel value at location** (x, y) is represented by $I(x, y)$, which is a vector of length 1 if the image is a gray or binary image, a vector of length 3 if the image is a color image.

CHAPTER 1. INTRODUCTION

- **Background image** B is used frequently referring to the image from a video sequence with no moving objects. The background image can be fixed or can be updated with time denoted as B_t .
- **Absolute difference image** D : we will compare a frame I_t with a reference image to get an absolute difference image D_t . The reference image could be the background image B_t , then $D_t = |I_t - B_t|$; or the reference image could be the frame before I_t , then $D_t = |I_t - I_{t-1}|$ and it is called consecutive difference image alternatively.
- **Foreground or foreground region** denotes the region where motion occurs, which is also the target of motion detection process. In this thesis, our interested foreground regions are those regions occupied by moving persons.
- **Background or background region** refers to the image region which is static comparing with foreground, for example the room one person is walking in. Foreground and background together form the whole image at the time of motion detection.

Based on the above notations, the main contributions of the thesis can be summarized as follows:

- **Entropy based motion detection.** In this motion detection method, the original information theory item entropy is employed to denote the level of motion in different locations of the video frame.

We first calculate the absolute difference image between consecutive video frames, which is used to form spatial temporal pixel blocks. At each pixel location, a spatial temporal block centered at the pixel and lasts for several consecutive frames is formed and its entropy is calculated accordingly. After that a spatial-temporal

entropy image is formed by combining the entropy at each point. It is found that the entropy is high at the motion region and low where there is no motion, so entropy image can be used to represent the extent of motion in different pixel locations of video sequences. Thus by thresholding the entropy image, a binary image is obtained comprising the motion region (foreground region) and background region.

- **Otsu's thresholding[18] based motion detection.** This is a quite simple yet fast and effective motion detection method, in which Otsu's thresholding method is used to find the optimum threshold of a difference image.

A background image where there is no motion is formed by some training images and then at each frame an absolute difference image is formed by comparing with the background image. If RGB color images are used, the threshold for each color channel is calculated using Otsu's thresholding method and combined together to form a final binary foreground mask, where "1" denotes foreground and "0" denotes background.

- **Tracking by corresponding motion detection results.** This tracking method employs motion detection results into the tracking process to be able to automatically initialize and resume the tracking process after occlusion happens. Motion detection results are processed at each frame to obtain object candidates. The location of the candidate objects are obtained by projecting the detected motion region vertically and horizontally to the image axis. Once the locations of candidate objects are done, parameters of each candidate object for example the height, weight, and color will be recorded and are used to remove the spurious candidate. Tracking over time is realized by establishing correspondence between existing objects in the previous frame and object candidates in the current frame. Adding and deleting objects are also realized. we use color histogram and location information

to characterize detected motion regions, which makes the detected regions more distinctive from one another than only using location information and this also makes the tracking more robust to occlusion.

- **Improved affine tracking.** Some researchers see tracking as an iterative optimization process in each frame to find the optimum transformation parameter that minimize the distance between the transformed region and the object model. The most often used transformation are translations and affine warps [19][20]. Using affine tracking as example, we combine difference image between consecutive image frames into the optimization process, i.e the difference image provides several starting points of gradient decent to search for the object, rather than only using one starting point to find the candidate region. In this way our tracker have more discriminating power when occlusion happens without resorting to sampling methods such as particle filter [21].

1.4 Organization of the thesis

In this chapter, the motivation of motion detection and tracking in videos is given.

Related work is discussed in Chapter 2. Algorithms in the area of motion detection, single camera tracking and multi-camera tracking are reviewed.

In Chapter 3, two motion detection methods are proposed and compared with several other motion detection methods.

Three single camera tracking algorithms are described in Chapter 4. The mean shift based tracking algorithm is firstly implemented and a simple yet effective tracking method is proposed. After that, an improved affine tracking method is given.

In Chapter 5, a multi-camera tracking framework is built for further research. The basic framework is described and preliminary results are presented.

Chapter 6 concludes the thesis.

Chapter 2

Related Work

In this chapter, various motion detection and human tracking methods are reviewed. Motion detection tries to extract the motion objects from image sequences. The motion objects can be seen as the foreground, and the scene without objects can be seen as background. From this point of view, motion detection is the process that tries to extract moving foreground objects from the background. In many tracking systems, motion detection is used to initialize a tracker or used as part of the tracking process.

After motion detection, tracking algorithms are discussed. Many tracking methods have been proposed and can be classified based on different criteria. In this chapter the tracking algorithms are categorized into two classes based on the number of sensors used: single camera tracking and multi-camera tracking. In 2.2 and 2.3, single camera tracking and multi-camera tracking algorithms are reviewed respectively.

2.1 Motion Detection

Human motion detection is a fundamental research area in computer vision. Many vision applications require segmenting the motion region out of the scene, which is usually called motion detection. Motion detection is an important part of many computer vision tasks like human tracking, pose estimation and face recognition.

CHAPTER 2. RELATED WORK

Since motion is a temporal event, most motion detection methods use temporal information from adjacent images or a much longer image sequence [22][23][24][25]. The most popular motion detection method is frame subtraction, i.e. a current frame is compared pixel-wise with a reference image. If a pixel value is above a preset value, it is assumed to be brought by motion. Using a static camera to observe a scene is quite common in a smart room application or a surveillance system [22][26][12]. The static scene is often referred to as background and the moving object is referred to as foreground.

Many motion detection methods have been extensively investigated. Both [27] and [23] gave a good discussion of the research methods. One direct method is to use temporal difference. The absolute difference at each pixel between two or three consecutive frames is calculated and a threshold is applied to get the difference image. In [23], a three frame difference algorithm was used. Their three-frame differencing rule suggested that a pixel was legitimately moving if its intensity had changed significantly between both the current image and the last frame, and the current image and the next-to-last frame. This method was simple to implement and it could adapt fast to background changes, but it was not so effective to get the whole region of the moving object due to the following reason: since absolute difference was used, the difference image may include both the pixels which were previously background but now covered by foreground and pixels that were previously foreground but became uncovered background; on the other hand, if the motion was not big between frames, the inner part of the moving object cannot be detected. In [28], a connected component analysis was used to cluster the difference image into motion regions to facilitate further processing. In [29], besides using motion information from two consecutive frames(frame difference as in the paper), they also constructed and maintained an up-to-date background model from the video sequence and compared each frame with the background(background difference as in the paper). Frame difference and background difference were combined together to detect motion more precisely.

CHAPTER 2. RELATED WORK

Using statistical background model is a more popular method to do motion detection [22][16][30]. A simple background model can be the average image over some training period. Motion can be detected by thresholding the difference between the mean background model and the current image. Instead of using a threshold, in [31] the pixel mean and variance of the R,G, and B channel were stored for each pixel as background model and were updated recursively. A current pixel was compared to the model, if in either channel the distance between current pixel and the mean value of background model was greater than 3 times the standard deviation, the pixel was set to foreground. Otherwise it was set to background. Some researchers claim that the median value was more robust than the mean value [32]. Cucchiara et al [30] modelled the background using median function; they report that the median function had proven effective while at the same time of less computational cost than using complex statistics like mixed gaussian model. Cheung and Kamath [33] also reported similar results.

Using mean or median value as background model is effective when the scene is uni-model [12], but when the scene is more complex, for example there are moving leaves or waves, a uni-model representation is not sufficient. A Gaussian mixture model was proposed for background subtraction in [25]. In [16], Stauffer and Grimson model the recent history of each pixel as a mixture of K Gaussian distributions(usually, K is from 3 to 5). Each Gaussian distribution had a weight and was ordered according to the ratio of its weight to variance. Background model was chosen according to these ordered distributions. Pixel values that did not fit the background distributions were considered foreground. Online K-means approximation was used to update the model. In [34], Gaussian mixture model method was further elaborated.

Unlike previous methods that used parametric Gaussian models, Elgammal et al. [35][24] used a recent sample of pixels over a period to form a non-parametric kernel density estimation of the pixel density function. The non-parametric methods estimated density

function without making assumptions about its explicit form. The pixels in the incoming frame was compared to the distribution and was declared as foreground if it was unlikely to come from this distribution. The pixel sample was updated continuously to adapt to changes in the scene.

Optical flow is another cue to do motion detection. Optical flow methods can be used to detect independently moving objects even in the presence of camera motion. However, most flow computation methods were computationally complex and very sensitive to noise, and cannot be applied to video streams in real-time without specialized hardware [23][36].

Besides the above mentioned methods, there are some other approaches. In [27], a one step Wiener prediction filter was used to predict pixel intensity based on a recent history of values. The prediction coefficients were recomputed each frame from the sample covariance to achieve adaptivity. Linear prediction using Kalman filter was used in [37]. Edges had also been used to form a background model. One of the merit was that edges are not sensitive to illumination changes. Since this method was only possible to detect foreground edges, a combination of color and edge information was used in [38][39]. There are still many aspects about motion detection. For example, both [23] and [39] adopted a hierarchical approach, processes were done at pixel level, region level and frame level.

2.2 Single Camera Tracking

The area of vision based human tracking is a well established research field. In Moeslund and Granum's survey paper [6], tracking was defined as establishing coherent relations of the subject and/or limbs between frames, while in [40] tracking was seen as the processing of measurement obtained from a target in order to maintain an estimate of its current state. Tracking is a very broad research field that may be addressed from various

viewpoints, such as tracking a single person or multiple persons, the number of cameras used etc. In this section, four common aspects of single camera tracking are addressed.

First, the initialization of a tracking system, which mainly concerns camera calibration, the threshold values needed in tracking, and model initialization etc. Second, many tracking algorithms start with the figure-ground segmentation problem, i.e., segmenting the human figure from the rest of the image. Some of the motion detection methods are reviewed in section 2.1. Third, a representation of the target object needs to be built to facilitate tracking. Fourth, how the objects are to be tracked from frame to frame should be defined.

2.2.1 Initialization

Initialization covers the actions needed to ensure a tracking system commences its operation with a correct interpretation of the current scene. Some of the initialization may be performed off line, such as the camera calibration, while other parts are preferably included as the first phase of operation, such as selecting the object to be tracked in the first frame.

In some tracking systems, the parameters of the camera need to be known. These can be obtained through off line camera calibration, and for a stationary camera setup, occasional re-calibration will suffice. An online camera calibration procedure was proposed in the work by Azarbayejani et al [41]. So if the camera setup regularly changes, this method may be preferred. However, almost all other systems implemented off line camera calibration [42]. Some calibration software is also available [43] [44][45].

Initialization to adapt the scene characteristics mainly relates to the appearance assumptions and the segmentation methods (described in the previous section) using them. In tracking systems where figure-ground segmentation or motion detection were employed, a possible off line initialization was often carried out to find out the thresholds and capture reference images which would be used during processing.

Some model based tracking system requires to build an object model in initialization period. In Toyama and Black's tracking algorithm [46], an exemplar of target object was generated in the learning period, which was later used in the test sequences.

2.2.2 Representations

The target object have been described in many different forms to facilitate tracking. Some common ones are line-based, region-based, probabilistic models and 2D human models.

The silhouette is a popular object representation due to its simplicity. It can be obtained from the segmentation results. In [47], Haritaoglu et al proposed a shape analysis algorithm that determines whether a person was carrying an object and segmented the object from the person so that it could be tracked, e.g., during an exchange of objects between two people. The method combined periodic motion estimation with static symmetry analysis of the silhouette of a person in each frame of the sequence. In [22], silhouette based representation was also employed. For each tracked person, after displacing the silhouette of the object from the previous frame by the median-based estimate, a binary edge correlation was performed between the current and previous silhouette edge profiles, which would result in the final alignment of silhouettes. While effective, one possible problem with contour representation was that, in very complex scenes, such as those containing people or natural objects, contour features were unreliable and difficult to find and use.

2D blob representation was employed in *Pfinder* algorithm [12]. In this method, feature vectors at each pixel were formed by adding spatial coordinates to the spectral (or textural) components of the imagery. These were then clustered so that image properties such as color and spatial similarity were combined to form coherent connected regions, or "blobs". 2D blob representation was a probabilistic method where clusters of 2D points

CHAPTER 2. RELATED WORK

were represented by 2D spatial means and covariance matrices. A Gaussian model was used to represent the spatial statistics of the region.

Among region based representation methods, object template is one approach that strictly preserve region structure. A template region is kept as the model of the target object, and typically the correspondence between the object template and any hypothesized region is established by minimizing some metric in the feature space to obtain the global transformation at the pixel level (or sparse feature level), for example [19]. Such approaches perform well for tracking rigid bodies, but this method may have problems when dealing with human body, since human movements are extremely non-rigid, sometimes it is not possible to find the transformation parameters between the template region and the correct human region.

Comparing with template based representation, histogram is a very popular probabilistic object representation, which totally ignore the region structure. Both [17] and [48] employed color histogram as object characterization in their tracking algorithm. Histogram based approaches have great flexibility to track deformable and non-rigid objects as well as being robust to partial occlusion, but they can lose the tracked region to another region with similar feature distribution.

To combine the structure-keeping merit of template and the probabilistic essence of histogram, a joint spatial-feature object representation has been employed recently by some researchers [49][50]. Distributions in the joint domain, estimated by kernel density estimation technique, was used to represent the object. An input image of n pixels was represented as a collection of d -dimensional vectors where the vectors contain both the location of the pixel and the features at the pixel.

An interesting approach was proposed in [51] using a dynamic model, involving Kalman filters and hidden Markov models. Pseudo-2D Hidden Markov Models (P2DHMMs) were used for capturing the shape of a person within an image and the output of

CHAPTER 2. RELATED WORK

P2DHMMs was used by Kalman-filtering algorithm to track the person. The P2DHMM was trained with static images, each showing a person within complex environment. In this way, while the HMM parameters of the system were learned successfully, several of the P2DHMM states would be assigned to the background, and other states would be assigned to the body regions of the person. Then the trained HMM could be used to find persons in the image.

The object representation methods mentioned above did not use explicitly 2D model. In some tracking systems, the kinematic models was employed which studied the human body in terms of the degrees of freedom it exhibits. In [52], the authors used both kinematic and geometric models, where optical flow features were used to track human arm and torso. Stick figure models aim to understand how the human body moves. In [53], the human body structure was represented by a stick-figure model with 6 joints. In their paper [54], Ramanan and Forsyth modelled a 2D view of the human body as a puppet of colored, textured rectangles. A nine segment tree model was used, consisting of the torso plus the left/right upper/lower arms/legs. In each frame, they searched for candidate body segments and grouped them to form a sequence of likely puppet configurations.

2.2.3 Tracking over time

Most of the tracking algorithms are similarity based tracking, i.e. they try to find out the most similar region(or whatever is used to represent the target object) in the consecutive frames compared with an object model or a model updated along with the time. The essence is how to find the most similar region in the next frame. So from this point of view, the tracking problem reduces to a searching method which can find the most similar region to an object model in some images, which satisfy some requirement like accuracy and speed (because most of the tracking tasks require realtime tracking up to a

CHAPTER 2. RELATED WORK

certain accuracy). The tracking problem can be classified into two loose classes: tracking as an iterative optimization problem and tracking as a Bayesian inference problem.

(i) Tracking as an iterative optimization problem

In [50], region tracking was formalized as a search over the transformation parameter space for the parameter that maximizes the similarity of the transformed region and the model region at each frame. Tracking can be viewed as a sequence of iterative optimization problems: for each image frame, the task is to find the optimum solution to minimize some object function, which represents the distance between the object being tracked in the last frame and the hypothesized regions in the new frame.

Different distance measures between the target representation in two frames can be used: sum of squared distance (SSD) [19], Bhattacharyya coefficients [17] [55], and KL(Kullback-Leibler) distance [50].

Once we have the objective function depicting the distance between the object model and the candidate hypothesis, the problem becomes how to find the parameters that minimizing the objective function most efficiently. Different iterative optimization techniques have been employed by researchers: gradient based optimization [50], trust-region optimization technique [56] and mean shift search method [17][55].

The mean-shift algorithm is a nonparametric statistical method for finding the nearest mode of a point sample distribution [57]. Comaniciu et al. has adopted it as an efficient technique for appearance-based blob tracking [17]. In the blob tracking scenario, sample pixels were given different weights, where the sample weights were chosen such that pixels on the foreground blob were with high weights, and pixels in the background were with low weights. The mean-shift algorithm would specify

CHAPTER 2. RELATED WORK

how to combine the sample weights in a local neighborhood with a set of kernel weights to produce an offset that would track the centroid of the blob in the image, which was done iteratively until a local mode was reached. In [55], the problem of selecting the scale of the mean-shift kernel was addressed. The kernel scale was a crucial parameter for the mean-shift algorithm, since it would determine the size of the window within which sample weights were examined. If the chosen kernel size was too large, the tracking window would contain many background pixels as well as the foreground object pixels. On the other hand, a too small kernel size would lead to poor object localization. In [55], a well-developed theory of feature scale selection due to Lindeberg [58] was combined with the mean-shift algorithm, resulting in a method that would represent and track blobs as the modes of a differential function in scale space.

In [49], a Bayesian approach was employed to real-time object tracking using non-parametric density estimation. The target model and candidates were represented by probability densities in the joint spatial-feature domain. The new location and appearance of the target were jointly derived by computing the maximum likelihood estimate of the parameter vector that characterized the transformation from the candidate to the model. A gradient-based iterative method was used to find the parameter vector.

In a recent paper [56], Liu and Chen proposed to use trust region optimization to solve the tracking problem. The tracking process was viewed as a sequence of iterative optimization problems: for each image frame, the task was to find an optimal solution that would best describe the status of a target object. The authors claimed that most of the iterative optimization techniques used in tracking were line-search in that the iterates were restricted to some iteration-dependent directions, e.g., the gradient. The gradient based optimization method and mean-shift technique

CHAPTER 2. RELATED WORK

were indeed line search methods. Instead, the authors proposed to use trust-region methods for the efficiency and reliability. A trust-region method derived its iterates by solving the corresponding optimization problem in a bounded region iteratively. In this way, there would be more options to select the iterates. In fact, line-search methods could be considered special cases of trust-region methods [59]. The efficiency of the trust-region method was demonstrated by making comparisons with a mean-shift tracker as well as carrying out a variety of experiments in different scenarios.

(ii) Tracking as a Bayesian inference problem

Methods based on Bayesian framework have been playing an important role in tracking, e.g. [60][61]. Tracking methods in this category usually use the measurement of an object from current image and tracking results from the previous image to maintain an estimate of the objects in the current image. Usually a state representation of the object is employed, as in [61]. The object state to be estimated typically consists of kinematic components (position, velocity, etc.) and other components (signal strength, feature information, etc.). In the following, how tracking can be viewed as a Bayesian inference problem is illustrated [61].

Let X_t denotes the state to be estimated, and $\vec{Z}_t$ the observations up to time t : $\{Z_0, Z_1, \dots, Z_t\}$. Using Bayes theorem, $p(X_t|\vec{Z}_t) = \frac{p(X_t) \cdot p(\vec{Z}_t|X_t)}{\int p(X_t) \cdot p(\vec{Z}_t|X_t) dX_t}$, which is the posterior distribution of X_t , and is what Bayesian inference attempts to estimate. Assume we have obtained $p(X_t|\vec{Z}_t)$, then tracking is solved, i.e. tracking can be formulated as Bayesian Inference problem. On the other hand, as tracking over time, $p(X_t|\vec{Z}_t)$ is propagated from $p(X_{t-1}|\vec{Z}_{t-1})$, tracking is also a density propagation problem.

In reality, instead of obtaining the posterior density itself, a Bayesian inference task may focus on estimating some properties of the density, such as moments, quantiles,

CHAPTER 2. RELATED WORK

highest posterior density regions, etc. All these quantities can be expressed in terms of posterior expectations of functions of $f(X_t)$. The posterior expectation of a function $f(X_t)$ is $E[f(X_t|Z_t)] = \frac{\int f(X_t)p(X_t)p(Z_t|X_t)dX_t}{\int p(X_t)p(Z_t|X_t)dX_t}$. In most applications, it is very hard to use analytic or numerical evaluation method to estimate this equation. An important alternative technique is Monte Carlo integration. The basic Monte Carlo method approximates a definite integral by uniformly sampling from the domain of integration, and averaging the function values at the samples. But there are limitations to the basic Monte Carlo method, and some methods are proposed to handle these limitations, including importance sampling, rejection sampling, Gibbs sampling etc. In [61], a Sequential Importance Sampling(SIS) method is employed to estimate the object state in the tracking sequences.

As analyzed above, object tracking at the most fundamental level can be traced to recursive Bayesian Estimation. From this point of view, the Kalman filter, extended Kalman filter, and recently the particle filter, which are widely used in the tracking field, all belong to Bayesian based tracking techniques. Kalman filter uses linear-Gaussian approximation which generally leads to computationally efficient algorithms, but may fail for certain demanding, highly nonlinear applications. Particle filters offer an attractive alternative for such problems as explicit linearization is avoided.

The Kalman filter was introduced by R. Kalman in 1960 in the control theory [62], and has been widely used in tracking area to predict the object's state in current image [63]. It is an algorithm which makes optimal use of imprecise data(observation) on a linear system with Gaussian errors and the previous state to continuously update the best estimate of the system's current state. For linear system and white Gaussian errors, Kalman filter is the "best" estimate based on all previous measurements. However, in practical tracking system, the object state dynamics are

probably non-linear. To solve this problem, extended Kalman filter or EKF was employed [64]. The EKF approach is to apply the standard Kalman filter (for linear systems) to nonlinear systems with additive white noise by continually updating a linearization around the previous state estimate, starting with an initial guess. In other words, only a linear Taylor approximation of the system function at the previous state estimate and that of the observation function at the corresponding predicted position are considered. This approach gives a simple and efficient algorithm to handle a nonlinear model. However, convergence to a reasonable estimate may not be obtained if the initial guess is poor or if the disturbances are so large that the linearization is inadequate to describe the system.

In their paper [60], Isard and Blake introduced the well-known CONDENSATION algorithm to track contours in clutter via factored sampling. The main idea was to pinpoint the inappropriateness of the Gaussian state density assumption for tracking in clutter where multiple competing observations existed. It was based on sampling the posterior distribution estimated in the previous frame and propagating these samples to form the posterior for the current frame.

Comparing with Kalman filter, particle filtering is a powerful Bayesian methodology for sequential inference in nonlinear non-Gaussian state-space systems, which has been employed in tracking area [65][21]. Rather than approximate the models to facilitate inference, the particle filter approximates object's state distribution using a set of samples. There are also some extended particle filter techniques, such as smoothing particle filter [66] and kernel particle filter [67].

2.3 Multi-camera tracking

As described in [13], tracking is essentially a correspondence problem. In single camera tracking, correspondences need to be established between the target objects seen in the

current image and those in the previous image. On the other hand, multi-camera tracking is a correspondence problem between tracks of objects seen from different viewpoints at the same time instant.

Numerous single-camera tracking algorithms have been developed. Comparing with single-camera tracking, multi-camera tracking is a relatively new problem in computer vision area, but it has gained increasing interests in recent years. Papers on this topic can be categorized by several standards, such as the features used, the correspondence strategy, and whether cameras are calibrated or not etc. We classify the papers addressing multi-camera tracking problems into the following three categories.

2.3.1 Feature Based Approaches

The simplest feature that we can think of to get correspondence between multi-cameras is color. Since objects seen from different cameras are probably of different sizes, geometric features such as height, object size and the region template cannot be used directly to get correspondences. Color histograms are efficient features in the sense that they are robust to partial occlusions, are rotation and scale invariant and computationally efficient. In [15], tracking in each camera was realized by combining color distributions and particle filters. Three main views of head regions were used to create three characteristic histograms for target model, i.e. three histograms of the front, side and back view of the head region were stored as the target model. Color histogram of hypothesized region in each view was compared with the target model using *Bhattacharyya coefficient*. Information is shared between different cameras to choose the best view.

When used as a feature to match detected regions from different cameras, especially to get the correspondence of the whole human body, color is not a reliable feature. Lighting variations may cause the same object seen from different views to have quite different colors. Different cameras can also have different intrinsic parameters and photometric

CHAPTER 2. RELATED WORK

properties like contrast, color balance, etc. Because of the unreliability of color feature correspondence, many researchers have proposed approaches to make it more robust by statistical modelings and through augmentation by other features, such as the height of a person.

In [68], the color distributions at different heights of the person were used to create a color model for the person. The distance between ground plane to the top of the person was divided into regions of equal length and a color model was developed for each region. To remove change of intensity levels across cameras, normalized color, i.e. the ratios $r/(r + g + b)$ and $g/(r + g + b)$ were used. An intensity value was added as the third variable to differentiate colors that were very different in intensity but similar in chromaticity. Non-parametric kernel density estimation method was used to estimate the color distribution of a person at different heights. These color models were used to firstly get a rough correspondence between different detected regions across different cameras, then corresponding regions were back projected in 3D space in a manner that yielded 3D points guaranteed to lie inside the objects. All the cameras were calibrated.

In many cases, more than one feature is used to generate correspondence relationship across cameras. In [69], Cai and Aggarwal built a system that started with tracking from a single camera and switched to another camera which would provide a better view when the system predicted that the active camera would no longer have a good view of the subject of interest. Features used for tracking included location, intensity, and geometric features such as the height of the person. Multivariate normal distributions were employed to model class-conditional densities of these features. The intensity features used were taken from the centerline of the upper body in each projection to reduce the difference between perspectives. The points from the centerline were matched, spatially and temporally, using Bayesian classification schemes.

The features used in [70] were grouped into geometry-based modalities and recognition-based modalities; the former included epipolar geometry, homography, and landmark

CHAPTER 2. RELATED WORK

modalities, while the latter comprised of apparent height and color modalities. The correspondence was performed statistically in a Bayesian Network approach.

In [71], motion information and image features (body height, cloth color, etc.) were integrated together in a Kalman filter framework and were used to get correspondence across multi-camera. The cameras used were all calibrated so that the motion observed in 2D image were used to project back to estimate the 3D state.

Homography was used in [72] for registering stationary cameras on top of a known ground plane. This method continuously tracked moving objects across two or more uncalibrated cameras. The motion and appearance of the moving objects were modelled simultaneously to do tracking. The motion models were obtained using a Kalman Filter process, which predicted the position of the moving object in both 2D and 3D. A *Joint Probability Data Association Filter* (JPDAF) was used to integrate multiple cues (motion and appearance) and multiple views to track a large number of moving people with partial and total occlusions.

Dockstader and Tekalp [73] proposed a distributed, real-time computing platform for improving feature-based tracking. The cameras used were all calibrated. A state-based 2D predictor-corrector filtering technique was employed for each camera view, and the output of each view was passed to a central processor that integrated and filtered the data using the Bayesian belief network(BNN) and 3D Kalman filter, respectively.

Aside from the fact that feature based matching methods are relatively simple, the feature may become unreliable when occlusions happen or when the same object seen from different views have great disparities.

2.3.2 Approaches Based on 3D Information

If camera calibration and the 3D environment model are known, cross-correspondence problem can be established by projecting the location of each 3D object in the world

CHAPTER 2. RELATED WORK

coordinate system, and establishing equivalence between objects that project to the same location.

In [68] and [74], Mittal and Davis adopted a region based stereo algorithm that was capable of finding 3D points inside an object, if the regions belonging to the object in two views were given. For a matched pair of regions, the mid-points of the regions along the epipolar line were projected back to get 3D points. These 3D points were then projected to the ground plane to get the 2D position of points on the ground plane. The motivation of taking the mid-points of the matched region was explained by a proof that in the case of orthographic projection and a convex object, the points of intersection of the rays through the mid-points of corresponding segments was the only point that can be guaranteed to lie inside the object. Results from different camera pairs were combined together to estimate the presence probability of an object at ground plane points using kernel estimation technique.

In [71] and [73], camera calibration information was used to get the ground plane coordinates of the tracked objects and these ground plane coordinates were used as one of the features to get correspondence across cameras.

The authors of [75] adopted a similar approach where each camera was calibrated, and the world was a known ground plane. So the location of any object bounding-box in any camera could be found in 3D coordinates in the world. Correspondence between different views was established by linking views with similar projected 3D location.

Krumm et al [76] used stereo cameras and combined information from 2 stereo cameras in 3D space. In each camera, they performed background subtraction and detected human-shaped blobs in 3D space. Two stereo modules connected to the two stereo cameras would report the 2D ground plane locations of its person-shaped blobs to a tracking program on a third PC. The person tracker would produce a final report on the locations and identities of people in the room using knowledge of the two cameras' relative

CHAPTER 2. RELATED WORK

locations, fields of view, and heuristics on the movements of people. Color histograms were created and maintained for each person to disambiguate people when they were close together.

Multi-camera tracking may take the advantage of 3D information in controlled environments, like smart room or football stadiums, however, it is difficult or sometimes impossible to have calibrated cameras and accurate environment maps in more general surveillance scenario. And if the cameras move, they must be re-calibrated again.

2.3.3 Alignment Based Approaches

Alignment-based approaches rely on recovering the geometric transformation between the cameras. Here, the correspondence between tracks is not explicitly resolved, but if the transformation is recovered accurately, the tracks of the same object will overlap each other when aligned by the computed transformation.

Lee et al. [77] employed an approach that used trajectory information for alignment. The motion trajectories in different cameras were randomly matched against one another and plane homographies were computed for each match. The correct homography was the one that was statistically most frequent. Finer alignment was achieved through global frame alignment. After the alignment, the 3D relative positions and orientations of the cameras were fully recovered. Simultaneously the problem of automatic external calibration of multiple cameras in an extended scene could also be solved. The method of trajectory alignment, however, was expensive, for a large number of possible alignments need to be computed.

In [78], Caspi and Irani presented an approach for both spatial alignment and temporal synchronization (temporal alignment) between two different video sequences of the same dynamic scene, recorded by stationary un-calibrated video cameras. They have considered the problem from a frame alignment point of view, extending the spatial image

CHAPTER 2. RELATED WORK

alignment methods to incorporate time information also. After the complete alignment of camera sequences both in time and space, the same object in different cameras would map to the same location. As was the case with spatial alignment, this could only be done when disparity between cameras was small.

In [13], an approach of tracking moving objects in an environment covered by multiple un-calibrated cameras with overlapping fields of view was addressed. It assumed that the tracks of objects were available in each camera, i.e. tracking was done in each camera, so the multi-camera tracking problem reduced to the correspondence between the tracks of the same object. Since the cameras had overlapped views, the authors claimed that the ideal place to establish this correspondence would be the instant when an new object was seen, because all subsequent points in that trajectory were automatically corresponded. Their tracking system computed the edges of the footprint (i.e. field of view lines (FOV lines)) of a camera as seen in other cameras, and these lines were used to establish correspondence between trajectories. A method was proposed which would automatically recover these FOV lines by observing motion in the environment.

In [79] and [80], approaches were employed that tried to establish time correspondences between non-overlapping camera views. The motion of the object, and the predicted time taken by the object to move from one camera to another were used to establish correspondence. Javed et al. [80] used the observations of people or vehicles moving through non-overlapping cameras and jointly modelled object velocities, inter-camera travel times and locations in a Parzen windows frame work to automatically learn the inter-camera correspondence probabilities. A MAP approach was used to establish correspondence.

Chapter 3

Motion Detection

In this chapter, two motion detection methods are presented. In the first method, entropy of accumulated difference image is used to detect motion regions and one paper about this has been accepted [81]. The second method is based on dynamical thresholding of the difference image and the basic idea is included in [82].

3.1 Entropy Based Motion Detection

Entropy is a concept from information theory and has been widely used in image segmentation [83][84][85][86]. Ma and Zhang [87] proposed a motion detection method based on entropy. The motion detection method proposed in this section is motivated by their work which is explained in section 3.1.1.

3.1.1 Spatial-Temporal Entropy Image(STEI)

Entropy is a measure of uncertainty of a random variable. Let X be a discrete random variable and its probability mass function be $p(x) = Pr\{X = x\}$, $x \in X$. Then the entropy for random variable X is $E(X) = -\sum_{x \in X} p(x) \log p(x)$.

In image sequences, the pixel value at a particular location could change from frame to frame due to the following reasons: (i) the camera channel noise and noises brought by flickering of light; (ii) there are objects in motion which make the pixel value change

CHAPTER 3. MOTION DETECTION

from background to object or from object to background. In [87], the change of pixel value from frame to frame is considered as the state transition of pixel, e.g. in a 256 level gray image, each pixel has 256 states. Along with time, the pixel's state would change from one to another. Pixel state's change brought by noises would be in a small range, but those brought by motion will be large. So the diversity of state at each pixel can be used to characterize the intensity of motion at its position.

With above assumption, the probability distribution of each pixel's state is observed along temporal axis. For image sequences, a temporal histogram is used to present state distribution in a sliding window. The probability density function (*pdf*) of pixel's state is obtained by histogram normalization. In [87], *YUV* color images are used to generate histograms. To simplify illustration, we use gray-level images here. A rectangular spatial window called accumulating window is used to reflect the relationship between a pixel and its neighborhood. Fig. 3.1 shows the pixels used to form the spatial-temporal histogram, i.e. $w \times w \times L$ pixels are accumulated to form the histogram of pixel(i, j), denoted by $H_{i,j,q}$, where q denotes the bins of the histogram, the total number of bins is Q , then all the components of the histogram is: $\{H_{i,j,1}, \dots, H_{i,j,Q}\}$.

Figure 3.1: Pixels used to accumulate histogram for pixel location (i,j) at Frame(k)

Once the histogram is obtained, the corresponding probability density function(*pdf*)

CHAPTER 3. MOTION DETECTION

for each pixel can be computed by

$$P_{i,j,q} = \frac{H_{i,j,q}}{N} \quad (\text{Eq. 3.1})$$

where N is the total number of pixels in the histogram bins and $\sum_{q=1}^Q P_{i,j,q} = 1$. Once the *pdf* of the pixel is known, the state diversity level of this pixel is calculated using the definition of entropy

$$E_{i,j} = - \sum_{q=1}^Q P_{i,j,q} \log(P_{i,j,q}) \quad (\text{Eq. 3.2})$$

where $E_{i,j}$ is called spatial-temporal entropy of pixel (i, j) . $E_{i,j}$ is quantized into 256 gray levels to form an energy image, named as spatial temporal entropy image (*STEI*). In *STEI*, the lighter the pixel is, the higher its energy is, and the more intensive its motion is.

Although it seems that *STEI* can characterize motion quite well, the following questions arises: while the frame to frame state transition of pixels brought by motion can cause high entropy, will the state transition of pixels in the spatial accumulating window also bring high entropy? This question arises since a spatial window of size 3×3 is used. When motion occurs, the spatial-temporal histogram and accordingly the *pdf* of the pixel will spread wider, which results in high entropy. But since a spatial accumulating window is used, not only motion can change the pixel's distribution in histogram, the spatial structure around the pixel will also affect the shape of the histogram. Especially when the pixel is an edge pixel, its histogram would also spread for a wide range, causing the entropy to become as high as those caused by motion. To verify this hypothesis, an experiment is done as follows.

Fig. 3.2(a) is the 664th testing frame in PETS 2001 datasets2 (camera2). In the experiment, the spatial accumulating window size is set to 3×3 , the spatial temporal duration is chosen as $L = 5$. First, testing frames from 660 to 664 in PETS 2001 datasets2 (camera2) are read into memory; then they are changed to gray images and

CHAPTER 3. MOTION DETECTION

quantized into $Q = 20$ gray levels. Histograms are formed by accumulating pixels in these images, pixel's *pdf* and entropy are then calculated using equation (Eq. 3.1) and (Eq. 3.2). Fig. 3.2(c) shows the cropped *STEI* image.

Figure 3.2: Experiment result on PETS 2001 datasets2 (camera2). (a) original image (b) cropped original image where motion occurs (c) cropped *STEI* image (d) cropped *DSTEI* image

We can see from Fig. 3.2(c) that pixels at edges get high entropy, i.e. both motion and spatial diversity can cause high entropy and they are hard to differentiate, which can lead to false detection of motion.

To solve this problem, the spatial influence must be removed. One way to remove spatial influence is to calculate entropy based on difference image, i.e. the pixel's histogram is formed by accumulating pixels in difference images. This method is explained in section 3.1.2.

3.1.2 Difference-based Spatial Temporal Entropy Image(*DSTEI*)

From the analysis of section 3.1.1, we know that *STEI* image cannot differentiate the high entropy caused by motion and spatial structure of the image. A method based on difference image is proposed, i.e. forming histogram by accumulating pixels in difference images. The entropy images formed this way is denoted as Difference-based Spatial Temporal Entropy Image (*DSTEI*).

Images in real world are often degraded by random noise, which may occur during image capture, transmission or processing, and may be dependent on or independent of image content. The two common type of image noises are white noise and salt and pepper noise. White noise occurs in all recorded images to a certain extent and this kind of noise is due to the discrete nature of radiation, i.e. the imaging system is recording images by counting photons. In many practical applications, this noise can be modelled with an independent, additive model, where the noise has a zero-mean Gaussian distribution described by its standard deviation. Salt and pepper noise is another common form of noise, which is caused by errors in the data transmission and is highly dependent on the quality of data transmission. The corrupted pixels may be set to the maximum value, or are set to zero and so on. In many computer vision based applications, researchers assume images noises to be Gaussian distributed [35].

If image noises are Gaussian distributed, pixels where no motion occurs would follow zero-mean Gaussian distributions in several difference images; When motion occurs around the pixel, pixels used to form the histogram would have higher value in difference images. For this case, the histogram for this pixel would distribute in a wide range, causing entropy to be much higher than those where no motion occurs. So entropy obtained this way can denote the intensity of motion.

The following steps are used to detect motion object based on *DSTEI*:

CHAPTER 3. MOTION DETECTION

- (i) Calculate difference images. *RGB* color images are first converted to 256 gray level images, denoted by $F(k)$, where k is the frame number. The difference image $D(k)$ is calculated by (Eq. 3.3) in which $|\cdot|$ denote absolute value, and $\Phi(\cdot)$ is the quantization function. In all our experiments, $\Phi(\cdot)$ quantizes the 256 gray levels into $Q = 20$ gray levels.

$$D(k) = \Phi(|F(k) - F(k-1)|) \quad (\text{Eq. 3.3})$$

- (ii) Histogram accumulation. If we follow [87] to calculate entropy image for a particular frame, then multiple difference images are needed to obtain the *DSTEI*: to get *DSTEI* for frame $F(k)$, we need to accumulate histograms from $D(k-L+1)$ to $D(k)$ where L is the number of frames for accumulation. Similarly, to get *DSTEI* for frame $F(k+1)$, we need to accumulate histograms from $D(k-L+2)$ to $D(k+1)$. Then to do real-time motion detection, L histogram accumulating processes will be needed, which is computationally expensive. To solve this, histogram $H_{i,j}(q)$ for pixel (i, j) is updated online using simple recursive updates. Specifically, the first L frames, i.e. from $D(1)$ to $D(L)$ are used to accumulate histogram for $D(L)$ using

$$H_{i,j,q}(L) = \frac{1}{L} \sum_{k=1}^L h_{i,j,q}(k) \quad (\text{Eq. 3.4})$$

and histograms for the subsequent frames are updated by

$$H_{i,j,q}(k+1) = \alpha H_{i,j,q}(k) + (1-\alpha)h_{i,j,q}(k+1) \quad (\text{Eq. 3.5})$$

where the constant α is set empirically to control the influence of history frames. Because recursive update function (Eq. 3.5) is used, α strongly influences the effect of update while L 's effect gradually weaken at later frames. Section 3.1.3 has a discussion of the influence of different α .

- (iii) Obtain *DSTEI*. Once we have the accumulated histogram of each pixel at the k^{th} frame, the *pdf* for pixel(i, j) is formed by normalizing the histogram using

$$P_{i,j,q}(k) = \Gamma(H_{i,j,q}(k)) \quad (\text{Eq. 3.6})$$

where $\Gamma(\cdot)$ is the normalization function such that the pixel number in each bin of $H_{i,j,k}$ is divided by the total number of pixels in the histogram. Since $D(k)$ is quantized into $Q = 20$ gray levels, the normalized histogram, i.e. $P_{i,j,k}$ has 20 discrete values. After *pdf* for each pixel is obtained, entropy is calculated using

$$E_{i,j}(k) = - \sum_{q=1}^Q P_{i,j,q}(k) \log[P_{i,j,q}(k)] \quad (\text{Eq. 3.7})$$

Entropy is quantized into 256 gray level to generate *DSTEI* image.

- (iv) Motion object localization. After obtaining *DSTEI* image, many methods can be used to derive the motion region. One way is to find the peak in the *DSTEI* image and then apply region growing method to locate motion object. One simple method is simply to binarize the *DSTEI* image by a threshold T . Entropy based thresholding method can be used to find the optimum T , like method used in [86]. In this paper, a simple thresholding method is used.

The example in Fig. 3.2(d) shows the effectiveness of motion detection based on *DSTEI* where spatial influence is removed and only motion regions cause high entropy.

3.1.3 Experiments and Discussions

Our proposed method is tested on several image sequences: the *hall_monitor* sequence, a walking sequence, and the PETS 2001 dataset2 (camera1) testing frames, shown in Figure 3.3 (a), (b) and (c), respectively. In the experiments, a 3×3 spatial window is chosen and α is set to 0.8. The *DSTEI* images, shown in the second row of Figure 3.3,

CHAPTER 3. MOTION DETECTION

have some noise components which are subsequently removed by thresholding to obtain the foreground moving regions as shown in the last row of the figure. Note that in Figure 3.3 (c), the leaves of the tree are moving, but the proposed algorithm is able to remove its effect after thresholding. We compare our method with that proposed in [87] and show the result in Figure 3.2(d) where the *DSTEI* image has captured the foreground moving regions quite well.

Figure 3.3: Experimental results. First row: original image. Second row: *DSTEI* image. Third row: detected motion region. (a) is the 116th frame of the *hall_monitor* sequence. (b) is the 215th image of a walking sequence. (c) is the 587th testing frame of the PETS 2001 dataset2 (camera1)

Some experiments are also done to test the influence of different α (in Eq. 3.5). In Figure 3.4 (a), (b) and (c), L is set to 5, and α is 0.9, 0.8, 0.7. We can see that larger α causes longer tail in *DSTEI* image. Although different α will influence the tail of

CHAPTER 3. MOTION DETECTION

DSTEI image, they will not bring much difference to the motion detection results and these tails can also be used to indicate the trajectory of motion.

Figure 3.4: Influence of different α . (a) $\alpha = 0.9$ (b) $\alpha = 0.8$ (c) $\alpha = 0.7$

From Figure 3.3 we can see that good results are obtained using proposed motion detection method. But there are also cases when the results are not good. Figure 3.5 and Figure 3.6 show two such cases.

Figure 3.5(a) is the 63rd frame of the running sequence where the person has just crossed the road and run back. In this frame and several previous frame, the person is moving slowly compared with when he is running. Figure 3.5(d) is the 76th frame of the sequence and the his speed is faster than in frame 63rd. Comparing the two *DSTEI* images(Figure 3.5(b), Figure 3.5(e)), we can see that when the person is moving faster, the *DSTEI* image will have longer tail. When apply thresholding to the *DSTEI* image, the detected motion region is larger than the actually motion object(Figure 3.5(f)), i.e. it is hard to get accurate motion detection results when the object is moving fast. However, the tail information may be used as one cue to identify object's motion speed and direction.

Figure 3.6(a) is the 241st frame of a indoor walking sequence where the person is relatively big. Since the *DSTEI* image can be seen as the accumulation of the difference image, it can be deduced that in *DSTEI* image the boundary will be very salient comparing with the inner part of the object. This is also shown in Figure 3.6(c), i.e. when

CHAPTER 3. MOTION DETECTION

Figure 3.5: Experimental results on a running sequence. (a) the 63rd frame; (b) *DSTEI* image of the 63rd frame; (c) detected motion region of the 63rd frame; (d) is the 76th frame (e) *DSTEI* image of the 76th frame (f) detected motion region of the 76th frame

the motion object is big, the proposed motion detection method may fail to detect the inner part of the object.

Figure 3.6: Experimental results on a indoor walking sequence. (a) the 243rd frame of the sequence (b) *DSTEI* image of the 243rd frame (c) detected motion region of the 243rd frame.

From the above experiment results, we can see that the proposed entropy based motion detection method can obtain satisfactory results. In our method, difference images are used for histogram accumulation and noises are assumed to follow Gaussian distributions. However this assumption will be violated in some cases such as when there are heavy shadows or there is a sudden illumination change, then false detection will occur. Future

work may be done to solve this problem and make our motion detection method more robust. Another problem with the motion detection method is that it is not very efficient, as histogram is needed for each pixel in the image. A possible solution is that we do edge detection on the difference image and only compute histogram for pixels on the edges.

3.2 Motion Detection With Adaptive Background And Dynamic Thresholds

In section 3.1, we have proposed an entropy based motion detection method. However, the algorithm has a limitation in that the performance degrades as the size of the moving region increases. To better initialize and support people tracker, another simple yet effective motion detection method is proposed [82].

The basic idea is that we use a dynamic threshold to binarize the difference image between a background image and a new coming image. We keep a static background image which is only updated periodically, and compare the incoming frame with it to obtain the absolute difference image. If through a preliminary judgement that there are moving object in the scene, Otsu's threshoding method [18] is used to find the threshold to binarize the difference image, i.e. pixels are classified as background or foreground. The threshold for the R,G and B color channels are found respectively and used to get the final foreground mask. Besides, an update strategy is employed to make the algorithm more robust. The overall motion detection flow chart is shown in Figure 3.7, each part of which will be illustrated in the remainder of this section. The left side and the right side of Figure 3.7 depict two integrated part of the motion detection process. The right part consists of the steps to compare a current frame with the background frame and detect the motion region, if there is any. The left part of the figure is process to periodically update background and average noise level to make the motion detection algorithm more robust.

CHAPTER 3. MOTION DETECTION

Figure 3.7: Motion detection based on Otsu's thresholding method

N and n are preset numbers to form a history image sequence. N and n would be different if the sequences are different. Suppose we want to form a history image sequence of 20 frames ($H = 20$) by keeping 1 out of 5 ($n = 5$) consecutive frames, and the current frame number is t , so only when $t/5(N)$ is natural numbers, this particular frame will be put into the history image sequence. That is to say, frame 1, 5, 10, ..., $5 * N$, will be kept in the history image sequence.

3.2.1 Otsu's Thresholding Method

Otsu's method is an image segmentation method that tries to divide the image gray level into two classes. The method is based on a very simple idea: find the threshold that minimizes the weighted within-class variance. Since this method operates directly on the gray level histogram, it is fast once the histogram is computed.

Let the normalized histogram be denoted by $P(l)$, where l is the gray level and $l = 1 : L$ (L is the total number of gray levels). A grey value t needs to be found to divide the histogram into two classes: $l = 1 : t$ and $l = t + 1 : L$.

The weighted within-class variance is calculated by

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t) \quad (\text{Eq. 3.8})$$

Where the class probabilities $q_1(t)$ and $q_2(t)$ are estimated as

$$q_1(t) = \sum_{l=1}^t P(l) \quad (\text{Eq. 3.9})$$

$$q_2(t) = \sum_{l=t+1}^L P(l) \quad (\text{Eq. 3.10})$$

and the class means $\mu_1(t)$ and $\mu_2(t)$ are given by

$$\mu_1(t) = \sum_{l=1}^t \frac{lP(l)}{q_1(t)} \quad (\text{Eq. 3.11})$$

$$\mu_2(t) = \sum_{l=t+1}^L \frac{lP(l)}{q_2(t)} \quad (\text{Eq. 3.12})$$

Finally, the individual class variances are calculated as

$$\sigma_1^2(t) = \sum_{l=1}^t [l - \mu_1(t)]^2 \frac{P(l)}{q_1(t)} \quad (\text{Eq. 3.13})$$

$$\sigma_2^2(t) = \sum_{l=t+1}^L [l - \mu_2(t)]^2 \frac{P(l)}{q_2(t)} \quad (\text{Eq. 3.14})$$

We could have just run through the full range of t values and pick the one that minimizes Eq. 3.8. But it turns out that minimizing the within-class variance is the same as maximizing the between-class variance, which can be calculated recursively, permitting a much faster calculation.

When using Otsu's method to do image segmentation, we assume that the histogram of the image is bimodal, which may not be true for all images. But in our motion detection method, we use Otsu's method to find the optimum threshold of a difference image. When there is a motion object in a frame, the histogram of difference image between the frame and a background is bimodal, because the pixel values in the difference image should fall into two classes: one class is formed by pixels in motion region, and the other class is formed by pixels outside the motion region.

3.2.2 Average Noise Level and Normalized Absolute Difference

Suppose we now have a background image B and incoming image $I(t)$. The normalized absolute difference (NAD) of a frame at time t is represented by

$$d(t) = \frac{1}{3N} \sum_{c=r,g,b} |B_c - I_c(t)| \quad (\text{Eq. 3.15})$$

where N is the number of pixels in the image, $c = r, g, b$ are the 3 color channels and $I_c(t)$ is the color channel c of image at time t .

In an ideal case when there is no motion $I(t) = B$ and $d(t)$ will become 0. However there is always noise in the images and so in this case NAD is larger than 0. If we calculate the averaged NAD using some images where there are no motion objects, then NAD can be used to denote the average noise level.

CHAPTER 3. MOTION DETECTION

Figure 3.8: Computed NAD for a test image sequence

Figure 3.8 shows the computed NAD values for a test image sequence where a person moves into the scene at frame 35 and gradually walks away.

From the above analysis, we can see that NAD can be used as a reference to judge whether there is moving object or not in the scene, for example we can use the dashed line in Figure 3.8 as a threshold to judge whether there is motion or not.

3.2.3 Motion Detection Based On Dynamic Thresholding

The motion detection method has the following steps:

- Learn initial background model and image noise level.

Some training images are used to build a background model and estimate the image noise level. We need to build a background model and estimate the image noise level for each video to be tested. In our experiments, both indoor and outdoor video sequences have been used.

A pixel wise median filter over time is applied to the training images to obtain a background image. Using this method we can obtain a fairly good background model even if there are moving foreground objects in the sequence, as long as

CHAPTER 3. MOTION DETECTION

the foreground object won't stay static at a particular place for more than half of the training images. We use B to denote the background image. The above conditions specified for using median filter over some training images to obtain the background images may not be satisfied at all times. At those conditions, new technique is needed to detect whether some object has stayed too long and become part of the background. In our experiment, we assume that the conditions of using median filter are satisfied.

When forming the median background image, the variance for each color of each pixel σ^2 is calculated. The pixels are classified as stationary if pixel value in each color channel is within 2 times the standard deviation from the median value. (Assume image noises are Gaussian distributed, this means that each pixel in the image is the sum of the true pixel value and a random, Gaussian distributed noise value. The true pixel value would be the same in the background locations (no moving foreground.)). When adding the Gaussian distributed noises, the pixel values will also follow Gaussian distribution, with mean as the mean value (approximately median value). As 95% of the pixel values will fall within 2 standard deviations of their median value, we believe that this is good enough to select true background pixel and to remove the noises.) Only pixels that are marked stationary are processed to get the NAD , which in fact denotes the noise level. Using Eq. 3.15 upon difference values of stationary pixels in the training frames, we obtain a average noise level d_{mu} . The standard deviation is also computed: d_{sigma} . A threshold can now be formed to denote whether there is moving object in the scene.

- Motion detection.

The motion detection steps are shown in Figure 3.7 in the right rectangle and illustrated in the following:

CHAPTER 3. MOTION DETECTION

- (i) Obtain absolute difference image and compute the NAD .

The incoming frame $I(t)$ is compared with the background image B and the normalized sum of absolute difference $d(t)$ is calculated using Eq. 3.15.

- (ii) Compare the NAD with the average noise level d_mu to decide whether there is moving object in the scene or not.

Frame $I(t)$ is declared as having moving object if Eq. 3.16 holds and the frame will be further processed. Otherwise, a new loop for the next frame will begin.

$$d(t) < (d_mu + 2.5 * d_sigma) \quad (\text{Eq. 3.16})$$

- (iii) Thresholding the difference image to obtain the foreground.

If there is moving object in the scene, Otsu's thresholding method is used to find the threshold for the difference image at each color channel, i.e. $T_r = Otsu(D_r)$, $T_g = Otsu(D_g)$, $T_b = Otsu(D_b)$, where D_r , D_g , D_b are the difference image for r, g, b color channel respectively. The thresholds are then used to form 3 foreground masks: f_r , f_g , f_b . Logical 'OR' are used to form the final foreground mask: $f = f_r \vee f_g \vee f_b$. The detected motion area can then be obtained by using logical 'AND' between the f and original image $I(t)$.

- Update background and average noise level.

The background image and average noise level can be updated periodically to make the motion detection algorithm more robust. This part is denoted in the left rectangle of Figure 3.7.

We choose to form a history image sequence H by keeping 1 out of every n consecutive images, i.e. if frame number t satisfies $t = N * n$ where N denotes the natural number, this frame is saved to H . Once the the number of frames in H reaches a

preset number K , the background image and average noise level are updated using the same method as in the initialization part. Then H is reset to zero and we start to accumulate images for the next update.

3.2.4 Experimental Results and Comparison With Gaussian Mixed Model(GMM) Method

The proposed motion detection method is tested on several image sequences. Gaussian Mixed Model based background subtraction method [16] is also implemented for comparison. Before presenting the results, the GMM method will be firstly explained.

In [16], the values of a particular pixel over time is considered as a “pixel process”, which is a time series of pixel values, e.g. scalars for gray values or vectors for color images. Assume at time t , the history of a particular pixel at (x_0, y_0) is know as

$$\{X_1, ..., X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\} \quad (\text{Eq. 3.17})$$

where I is the image sequence.

The recent history of each pixel shown in Eq. 3.17 is then modelled with a mixture of K Gaussian distributions. The probability of observing the current pixel value is

$$P(X_t) = \sum_{i=1}^K [\omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t})] \quad (\text{Eq. 3.18})$$

where K is the number of Gaussian distributions, $\omega_{i,t}$ is an estimate of the weight of the i^{th} Gaussian at time t , $\mu_{i,t}$ is the mean value of the i^{th} Gaussian, $\Sigma_{i,t}$ is the covariance matrix and η is a Gaussian probability density function

$$\eta(X_t, \mu_{i,t}, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)} \quad (\text{Eq. 3.19})$$

where n is the length of $\mu_{i,t}$. Currently, the value of K is usually from 3 to 5. Besides, the covariance matrix is assumed to be of the form: $\Sigma_{i,t} = \sigma_k^2 * I$, assuming that red,

CHAPTER 3. MOTION DETECTION

green and blue pixel values are independent and having the same variances so as to avoid the costly matrix inversion at the expense of some accuracy.

The K Gaussians are ordered by the value of ω/σ , which increases both when a distribution gains more weight and when the variance decreases. Then the first B distributions are chosen as the background model, where

$$B = \operatorname{argmin}_b \left(\sum_{k=1}^b \omega_k > T \right) \quad (\text{Eq. 3.20})$$

where T is a threshold of the minimum portion of the data that the background should account for.

With the existing mixed Gaussian models, each new pixel value X_t is checked against the K Gaussian distributions until a match is found. If the a pixel value is within 2.5 standard deviations of a distribution, then a match is found. If the pixel value matches one of the B background distributions, then the pixel is classified as background pixels, otherwise foreground pixels. If none of the K distributions match the current pixel value, the distribution with least ω/σ is replaced with a new distribution with the current value as its mean value, an initially high variance and low prior weight.

The prior weights of the K distributions at time t , $\omega_{k,t}$, are also adjusted as

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \quad (\text{Eq. 3.21})$$

where α is the learning rate and $M_{k,t}$ is 1 for the matched model and 0 for all the other models. The weights are renormalized after this update.

The μ and σ parameters for unmatched distributions remain the same, while the parameters of the matched distribution are updated as follows

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho X_t \quad (\text{Eq. 3.22})$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \quad (\text{Eq. 3.23})$$

where

$$\rho = \alpha\eta(X_t|\mu_k, \sigma_k) \quad (\text{Eq. 3.24})$$

The justification of the equations can be found in [16].

To compare our motion detection method with GMM method, the method is implemented. $K = 3$ Gaussian models are used for each pixel and the distance from mean is set as 2.5 times the standard deviation. The obtained results are shown in the following.

The results on a two-person walking sequence, four-person walking sequence and a running sequence are shown in Figure 3.9, Figure 3.10 and Figure 3.11. We can see that our method can achieve similar motion detection results as *GMM* method, which is probably the most popular motion detection method. However, in *GMM* method each pixel is modelled using several gaussian models which require high computational cost, while our method is a histogram based method and can be computed very efficiently.

One potential problem is that if the person size is quite small relative to the whole frame, his motion may not bring much difference to the *NAD*, and accordingly this frame may be falsely classified as having no motion object. One way to solve the problem is to divide the image into several blocks(for example 4), and compute the *NAD* individually for this block. If any of the obtained *NAD* indicate motion object, then this frame will be further processed to find the optimum threshold.

3.3 Conclusion

Two motion detection methods are proposed in this chapter. An entropy based motion detection method is firstly introduced, where the entropy of the difference image is used for motion detection. In contrast to traditional methods, no background model is needed, and the recursive estimation of histogram makes this method suitable for real-time motion detection. However, as analyzed in section 3.1.3, this method is not so suitable for indoor

CHAPTER 3. MOTION DETECTION

Figure 3.9: Motion detection of a two-person walking sequence. The five columns are the 125^{th} , 160^{th} , 200^{th} , 260^{th} and 380^{th} frame. The first row shows the original frames, and the second and third row are foreground masks obtained by our method and GMM method.

applications since in indoor images persons' sizes are usually large relative to the whole image size.

To solve this problem, a simple yet effective motion detection method is proposed. Several image sequences are tested using both our method and the famous *GMM* method, which shows that our method can achieve similar results.

CHAPTER 3. MOTION DETECTION

Figure 3.10: Motion detection of another two-person walking sequence. The five columns are the 520th, 600th, 700th, 790th and 890th frame. The first row shows the original frames, and the second and third row are foreground masks obtained by our method and GMM method.

Figure 3.11: Motion detection of a running sequence. The five columns are the 18th, 30th, 47th, 75th and 93th frame. The first row shows the original frames, and the second and third row are foreground masks obtained by our method and GMM method.

Chapter 4

Single Camera tracking

Most multi-camera tracking systems rely on the results of single camera tracking, so before coming to multi-camera tracking, our research was firstly focused on single camera tracking algorithms. In the first part of this chapter, the mean shift tracking algorithm is implemented, experimental results are given and its limitations are also analyzed. In the second part of this chapter, a tracking method by corresponding detected motion regions is proposed, which is also included in [88]. After that, an improved affine tracking method is proposed by employing difference image information into the affine tracking process [89].

4.1 Mean Shift Object Tracking

The mean shift algorithm is a non-parametric technique that climbs the gradient of a probability distribution to find the nearest mode of the point data distribution[90]. This algorithm has been adopted as an efficient technique for object tracking[91][17][55]. In the tracking scenario, sample points are distributed regularly along the image pixel grid, with a pixel's value being the sample weight $w(a)$ of the point at that location. The sample weights are chosen such that pixels on the candidate object have high weights, and pixels in the background have low weights. In the mean shift algorithm, the sample weights $w(a)$ in a specific window are combined with a set of kernel weights $K(a)$ to

CHAPTER 4. SINGLE CAMERA TRACKING

produce an offset(i.e. mean shift vector) that tracks the centroid of the object in the image.

The basic mean shift procedure is simple and can be expressed in the following steps:

1. Choose the initial location of the search window.
2. Choose the search window size.
3. Compute the mean location, i.e. centroid, of the sample points in the search window.

4. Center the search window at the mean location computed in step 3.

5. Repeat steps 3 and 4 until convergence (or until the number of iterations reaches a preset threshold).

In [17], mean shift is employed to derive the object candidate that is most similar to a given object model while predicting the object location in the next frame. The algorithm is explained in the following.

4.1.1 Mean Shift Tracking Algorithm

The object is represented by its color histogram. Let $\{x_i^*\}_{i=1,\dots,n}$ denotes the pixel locations of the object model centered at 0. Here the generic pixel coordinates are normalized between 0 and 1 according to the center pixel. We define a function $b(x)$, which returns the color bin index $(1, \dots, m)$ of pixel at location x , where m is the total number of bins.

The probability of the color u in the target model is given by

$$q_u = C \sum_{b(x_i)=u} k(\|x_i\|^2) \quad (\text{Eq. 4.1})$$

where $k(\cdot)$ is a convex and monotonic decreasing kernel profile function(a special class of radially symmetric kernels) which assigns smaller weights to the locations that are further from the center of the target and C is the normalization factor $C = \frac{1}{\sum_{i=1}^n k(\|x_i\|^2)}$,

to make sure that $\sum_{u=1}^m q(u) = 1$.

CHAPTER 4. SINGLE CAMERA TRACKING

Object representation become more robust after applying this weighting strategy since pixels near the object centroid are more reliable than the peripheral ones, which are often affected by the occlusions or background.

Then the object model is given as:

$$\vec{q} = (q_1, \dots, q_m) \quad (\text{Eq. 4.2})$$

When object model is given, we want to find the most similar region in a new frame.

Suppose $\{x_i\}_{i=1,\dots,n}$ denote the pixel locations in a candidate region centered at pixel y . Using the same kernel k and a kernel bandwidth h , the probability of color u in the candidate region is obtained by

$$p_u = C_h \sum_{b(x_i)=u} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right) \quad (\text{Eq. 4.3})$$

Accordingly the candidate model is represented by

$$\vec{p}(y) = (p_1(y), \dots, p_m(y)) \quad (\text{Eq. 4.4})$$

The task of the target location estimation problem is formulated as the derivation of the estimate that maximizes the Bayes error associated with the object model and candidate distributions. For this purpose, the discrete sample estimate of the Bhattacharya coefficient is utilized for its near optimality given by its relationship to the Bayes error and the metric structure it imposes. Given two histograms by Eq. 4.2 and Eq. 4.4, the Bhattacharya coefficient and the distance between them are defined as

$$\rho(y) = \rho[p_u(y), q_u] = \sum_{u=1}^m \sqrt{p_u(y) q_u} \quad (\text{Eq. 4.5})$$

$$d(y) = \sqrt{1 - \rho[p_u(y), q_u]} \quad (\text{Eq. 4.6})$$

CHAPTER 4. SINGLE CAMERA TRACKING

Now our aim is to minimize the above distance function of y in the neighborhood of a given location y_0 by exploiting the mean shift iterations. Using Taylor expansion around the value $p_u(y_0)$, the Bhattacharya coefficient $\rho[p_u(y), q_u]$ is approximated as

$$\rho[p_u(y), q_u] \approx \frac{1}{2} \sum_{u=1}^m \sqrt{p_u(y_0) q_u} + \frac{C_h}{2} \sum_{i=1}^n \omega_i k\left(\left\|\frac{y - x_i}{h}\right\|^2\right) \quad (\text{Eq. 4.7})$$

where

$$\omega_i = \sum_{u=1}^m \sqrt{\frac{q_u}{p_u(y_0)}} \quad (\text{Eq. 4.8})$$

The first term being independent of y , the second term in the above equation has to be maximized to minimize the distance. This term corresponds to the density estimate using kernel profile k at location y with the data weights ω_i . Given the target distribution $\vec{q} = (q_1, \dots, q_m)$ and the estimated target location y_0 in the previous frame, the target location y in the current frame can be estimated by maximizing Eq. 4.6. The maximization can be achieved efficiently based on mean shift iterations in the following steps:

1. Compute the histogram $\vec{p}(y_0) = (p_1(y_0), \dots, p_m(y_0))$ in the current frame, calculate $\rho[\vec{p}(y_0), \vec{q}] = \sum_{u=1}^m \sqrt{p_u(y_0) q_u}$.
2. Compute the weights $\omega_i, i = 1, \dots, n$ according to Eq. 4.8.
3. Derive the new location y_1 by mean-shift:

$$y_1 = \frac{\sum_{i=1}^n x_i \omega_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^n \omega_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)} \quad (\text{Eq. 4.9})$$

where $g(x) = -k'(x)$. Update $\vec{p}(y_1) = (p_1(y_1), \dots, p_m(y_1))$ and $\rho[\vec{p}(y_1), \vec{q}] = \sum_{u=1}^m \sqrt{p_u(y_1) q_u}$.

4. If $\|y_1 - y_0\| < e$, stop, else set $y_0 \leftarrow y_1$ and go to step 1.

4.1.2 Experimental Results and Discussion

The mean shift tracking algorithm has been tested on several image sequences taken in indoor environment. The Epanechnikov profile has been used for computing histogram,

CHAPTER 4. SINGLE CAMERA TRACKING

i.e. the mean shift iterations were computed with the uniform profile. The object histogram has been derived in RGB space with $8*8*8$ bins. In all the experiments we use fixed scale, i.e. the same bounding box is used for the same person.

Tracking results are presented in Figure 4.1, Figure 4.2, Figure 4.3 and Figure 4.4

Figure 4.1: Mean shift tracking on a one-person walking sequence: (a), (b), (c), (d) is the 153^{rd} , 163^{rd} , 173^{rd} , 183^{rd} frame respectively.

Figure 4.2: Mean shift tracking on a two-person walking sequence, person 1: (a), (b), (c), (d) is the 100^{th} , 120^{th} , 145^{th} , 155^{th} frame respectively.

Figure 4.3: Mean shift tracking on a two-person walking sequence, person 2: (a), (b), (c), (d) is the 100^{th} , 120^{th} , 130^{th} , 150^{th} frame respectively.

Figure 4.1 presents some images from a one-person indoor walking sequence. (a) to (d) shows the process that the person walking through the scene, from the results we can see that the person can be tracked very well.

CHAPTER 4. SINGLE CAMERA TRACKING

Figure 4.4: Mean shift tracking on another two-person walking sequence person 2: (a), (b), (c), (d) is the 170th, 178th, 190th, 200th frame respectively.

Figure 4.2 and Figure 4.3 show the results of a two-person indoor walking sequence. In Figure 4.2, the person moving towards left is tracked. We can see that he can be tracked correctly during and after occlusion. When the person moving towards right is tracked, as shown in Figure 4.3, although during occlusion he is almost totally occluded by the other person, the mean shift tracker managed to track him correctly. However, in another two person walking sequence the person being occluded can no longer being tracked correctly, as shown in Figure 4.4. Some reasons are as follows. First, the histogram used to characterize a person should be a good representation for sufficient discrimination among different persons and also between persons and the background. In Figure 4.3 the two persons wear clothes with quite distinct colors, i.e. red and black, and their histograms are enough to differentiate them. But in Figure 4.4, the two person's clothes have quite similar color, so their histograms are not so distinctive. When one person is occluded by the other, the tracker attaches to the wrong person. Second, mean shift tracking algorithm proposed in [17] is a deterministic algorithm in that the tracker searches for the local maximum of the object appearance model. Sometimes when occlusions happen, it is quite possible that the tracker falls into a local minimum. Since the starting point for mean shift iteration is the object location in the previous frame, it is very hard for the tracker to recover from the failure, as in Figure 4.4.

Hence, the mean shift tracker searches the image space deterministically and might fall into a local minimum. This is a problem to over come when building an efficient

tracking system.

4.2 Tracking by Motion Region Correspondence

The mean shift tracking algorithm described in section 4.1 has a problem when occlusion happens, and it cannot initialize automatically. In this section, a simple tracking method is proposed trying to overcome these problems. In this method, motion detection results are processed at each frame to obtain object candidates. Tracking over time is realized by establishing correspondences between existing objects in the previous frame and object candidates in the current frame. Adding and deleting objects are also realized. The tracking method is similar to the method used in [92], but they only use location proximity and region size to get correspondence; so when the object becomes occluded or stationary, wrong correspondence may occur. Instead, we use color histogram to characterize detected motion regions, which makes the detected regions more distinctive from one another than only using location information. This also makes the tracking more robust to occlusion.

4.2.1 Process Motion Detection Results

Motion detection results using algorithm described in section 3.2 are further processed to obtain object candidates, each of which has the following attributes: centroid, bounding box and color histogram. These object candidates are used to get correspondences between existing objects.

For regions to become object candidates, they must be close to each other and their projection on to the horizontal axis must have some overlap and their total area must be larger than a threshold T_{area} . In our experiments, T_{area} is set to 400 pixels.

The object candidates are obtained by processing the vertical projection histogram of the foreground mask, i.e. the projection of foreground mask onto the horizontal axis. Figure 4.5 shows a foreground mask, its projection and detected object candidate.

Figure 4.5: Processed motion detection and tracking result of the 135th frame of a two-person walking sequence (a)foreground mask by motion detection (b)the projection of the foreground mask onto the horizontal axis (c)obtained object candidates shown in the bounding box (d) tracked objects with different bounding box colors

From the vertical projection histogram, shown in Figure 4.5(b), we can obtain the horizontal coordinates range of a particular foreground region. Then the vertical slice of the difference image defined by the obtained horizontal coordinates is projected to the vertical axis to get the vertical range of the region. Then the bounding box of the region is known as in Figure 4.5(d).

When two persons occlude each other, the detected foreground regions will merge into one, and only one region is detected as shown in Figure 4.6 (c) and (d).

In W4 tracking system [22], they analyze the vertical projection histogram of the silhouettes to determine whether the foreground region contains multiple people. This is possible when the multiple people are not so close to each other, like in Figure 4.6 (b) and (c), however when they are too close to each other, or one is totally occluded by another, this method can not distinguish between single and multiple people, like in Figure 4.6 (d).

Here, we use this method only to get the rough estimate of the detected foreground. The correspondence and occlusion problems are solved in the tracking part.

4.2.2 Tracking Over Time

Object tracking is realized by establishing relationships among detected motion regions. In the first frame, new objects are initialized at each detected region. The regions'

CHAPTER 4. SINGLE CAMERA TRACKING

Figure 4.6: Examples of image frames, detected foreground regions and their projection histograms. (a),(b),(c),(d),(e) is the 130th, 140th, 170th, 182th, 202th frame of a image sequence respectively.

attributes are passed to the objects. An object has the following attributes: 2D coordinates of centroid C , bounding box B , the color histogram H , and characteristic color c (different bounding box colors are used to represent different objects).

When a new frame comes in, object candidates are obtained by processing the motion detection results. Suppose in frame $t - 1$ there are p existing objects, and in frame t there are q object candidates. The goal is to establish correspondence between the p objects and the q candidate objects. By comparing them, object addition and deletion can also be handled.

The correspondences between existing objects and object candidates are established by comparing their color histograms. The similarity of two histograms are determined using Bhattacharya coefficient as used in[17]. (Histogram intersection, a simple but widely used histogram comparison method, has also been used, but comparison results show that Bhattacharya coefficient can better depict the similarity of two histograms.) If the similarity of an object and object candidate is above a preset threshold, we say

they are actually the same object and the object attributes are updated. According to our experiments, the histogram similarity of the same objects are larger than 0.9, which has been chosen in our experiments as the preset value. The objects are tracked over time in this way.

There are two cases when correspondences have not been found between all the objects in frame $t - 1$ and all the object candidates in frame t :

1. One or more existing objects cannot find correspondence. This maybe brought by occlusion or object exit from camera's field of view. In this case, the object is shifted from object list to history object list. A history object list is kept for a period before we delete it, since the object may be occluded by others or there is some mistake in the motion detection step, if we delete it right after a correspondence is not found, when the object appears again, a new object tracker will be assigned to it wrongly. In our experiments, 100 frames are kept before we delete the candidate object that is left un-matched.

2. One or more existing object candidates cannot find correct correspondence in the existing objects. In this case new object is initialized at the non-corresponded object candidate and a new color is assigned to it. An example of non-correspondence happens when one person is occluded by another. The two objects will merge together and be detected as one object candidate. So the two existing objects cannot find correct correspondence and are moved to history object list. When occlusion ends, the two persons are detected separately again. By comparing the histograms, the two object candidates should correspond to the two history objects and such the tracks are resumed after occlusion.

4.2.3 Experimental Results and Discussion

The tracking method has been tested on some image sequences and results are shown in Figure 4.7, Figure 4.8 and Figure 4.9.

CHAPTER 4. SINGLE CAMERA TRACKING

Figure 4.7: Tracking results of a running sequence. The frame numbers of (a) to (h) are 17, 24, 29, 37, 64, 77, 85, 93.

Figure 4.8: Tracking results of a two-person walking sequence. The frame numbers of (a) to (h) are 65, 96, 115, 140, 160, 180, 205, 230.

CHAPTER 4. SINGLE CAMERA TRACKING

Figure 4.9: Tracking results of the hall monitor sequence. The frame numbers of (a) to (h) are 30, 40, 58, 74, 89, 104, 113, 121.

In Figure 4.7 sample tracking results of a outdoor running sequence are shown. The person starts running from left to right, then turn and runs back. Good tracking results are obtained as shown in Figure 4.7.

The tracking results of a two person walking sequence are shown in Figure 4.8. In the sequence, one person walks from left towards right and the other walks from right to left. In (a) and (b) the two persons walk towards each other, and red and green bounding box are used to differentiate them. In (c) and (d) they meet each other, and the person walking towards right is occluded by the other person. A blue bounding box is used to denote the occlusion. After walking away from each other, the two person are continued tracked, with the correct color of bounding box, as shown in (e) and (f). In (g), one person is walking out the camera's field of view and in (h), the other person is tracked continuously.

Tracking results of the hall monitor sequence is presented in Figure 4.9. There are two persons coming into views at different time. From the results, we can that the two persons can be tracked very well.

Through the results, we can see that the persons can be tracked very well. However, there are some problems when persons are in occlusion. As shown above, two persons will be detected as one, and a new bounding box is assigned to the detected motion region.

One possible improvement is trying to split persons in occlusion into single ones when processing the motion detection results. The other is not to try to correspond existing objects to detected regions directly but use the detected region as a search region for finding the optimum object correspondence.

4.3 Difference Image based Affine Tracking

One category of research papers see tracking as an iteratively optimization process in each frame to find the optimum transformation parameter that minimize the distance between the transformed region and the object model. The most often used transformation are translations and affine warps. In this section, using affine tracking as example, we combine difference image between consecutive image frames into the optimization process, i.e the difference image provides several starting points of gradient decent to search for the object, rather than only using one starting point to find the candidate region. The improved method has also been written in one paper [89].

Binarized absolute difference image between consecutive frames provides more starting points to search the candidate region in the new frame. The image region that gives the smallest difference to object model is chosen as the optimum image region. In this way, we can overcome the clumsiness of optimization based searching and can recover more easily when occlusion happens.

To obtain difference image, otsu based method [18] is adopted to threshold the absolute difference image between consecutive frames without manually choosing the threshold.

We use a gray image template to represent target object, although more complex object model can be employed. Affine transformation is used to represent the object deformations between object model and object in new frames.

4.3.1 Affine based tracking and its limitations

In this section we will outline the affine based tracking algorithm, following [20][93][94] and its limitations will be explained after that.

Assume the object model is a template image $T(\mathbf{x})$, where $\mathbf{x} = (x, y)$ indicate the pixel location. Tracking is defined as to find a region in image $I(\mathbf{x})$ that is closest to the template. Let $\mathbf{u}(\mathbf{x}, \mathbf{p})$ denote the image transformation that will transform the image region in image $I(\mathbf{x})$ to the template frame, where $\mathbf{p}$ is the transformation parameters. If it is for translation, then $\mathbf{u}(\mathbf{x}, \mathbf{p}) = \begin{pmatrix} x + p_1 \\ y + p_2 \end{pmatrix}$. If the object being tracked involves more complicated motion, then affine warps maybe considered:

$$\begin{aligned} \mathbf{u}(\mathbf{x}, \mathbf{p}) &= \begin{pmatrix} p_1 \cdot x + p_3 \cdot y + p_5 \\ p_2 \cdot x + p_4 \cdot y + p_6 \end{pmatrix} \\ &= \begin{pmatrix} p_1 & p_3 & p_5 \\ p_2 & p_4 & p_6 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \end{aligned} \quad (\text{Eq. 4.10})$$

where $\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6)$ is the affine transformation parameter vector.

The goal of tracking is to find the parameter $\mathbf{p}$ that minimizes the error:

$$E(\mathbf{p}) = \sum_{\mathbf{x}} \rho(I(\mathbf{u}(\mathbf{x}, \mathbf{p})), T(\mathbf{x})) \quad (\text{Eq. 4.11})$$

where ρ is some error norm.

We adopt the traditional L^2 error norm in this paper, i.e

$$E(\mathbf{p}) = \sum_{\mathbf{x}} (I(\mathbf{u}(\mathbf{x}, \mathbf{p})) - T(\mathbf{x}))^2 \quad (\text{Eq. 4.12})$$

The minimization of the sum of squared error in Equation (Eq. 4.12) is performed with respect to $\mathbf{p}$ over all the pixels in template image, i.e. image I is warped back to

CHAPTER 4. SINGLE CAMERA TRACKING

the template coordinate. Assume that a current estimate of $\mathbf{p}$ is known and the object transformation between consecutive frames are small, we can solve $\mathbf{p}$ by finding the increments $\Delta\mathbf{p}$ iteratively, i.e. by minimizing the approximation function:

$$E(\mathbf{p} + \Delta\mathbf{p}) = \sum_{\mathbf{x}} (I(\mathbf{u}(\mathbf{x}, \mathbf{p} + \Delta\mathbf{p})) - T(\mathbf{x}))^2 \quad (\text{Eq. 4.13})$$

with respect to $\Delta\mathbf{p}$. The parameter is updated as follows:

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p} \quad (\text{Eq. 4.14})$$

The expression in Equation (Eq. 4.13) can first be linearized by performing a first order Taylor expansion and solved using a gradient descent method [94]. We continue (Eq. 4.13) and (Eq. 4.14) until reaching the convergence point, for example when the norm of $\Delta\mathbf{p}$ is below a threshold or the number of iterations reaches a maximum.

When new frames come in, the recovered parameter $\mathbf{p}_{pre}$ in the previous frame is set as the starting parameter, i.e. $\mathbf{p}$ in Equation (Eq. 4.13) to start the iteration. Once the parameter $\mathbf{p}$ is recovered, the region that is most similar to the object model is found, thus tracking is realized and can be maintained this way.

The steps of affine tracking is as follows:

- (i) Initialize the transformation parameter $\mathbf{p}$ of the template in the current frame with $\mathbf{p}_{pre}$, which is the estimated parameter in the previous frame.
- (ii) Obtain $\Delta\mathbf{p}$ by minimizing equation (Eq. 4.13)
- (iii) Update $\mathbf{p}$ by equation (Eq. 4.14)
- (iv) Continue step 2 and step 3 until the performance requirement is reached. In our experiments, we set a maximum iteration number to stop the iteration.

From the affine tracking algorithm above, we can see that it is a deterministic method in that it always try to find the most similar region from one starting point. When the object is occluded, the tracker cannot find the optimum region because of the occlusion. However, the tracker will iterate and try to find the most similar region and continue tracking in the new frame based on the the previous tracking result. So it is quite easy for the tracker to drift away from the target object when occlusion happens.

The tracking algorithm is tested on several video sequences to verify the above analysis.

Figure 4.10 (a) and (b) show some tracking results of two sequences in the Dudek Sequence from the University of Toronto, which first appeared in [95]. We can see that the face can be tracked properly though his movement brings significant changes.

Figure 4.11 shows several frames of a two person walking sequence. In 4.11(a), the person walking in from the right side is correctly tracked. In(b), the person walking in from the left side is correctly tracked before occlusion. However, when occlusion happens, the tracker wasn't able to track the target person and drifts to the person in front. After occlusion, the tracker continues following the wrong person as the estimated parameters from the previous frame is used as the starting point of search. Besides, the wrong person is more similar to the target model of the background.

In section 4.3.3, we will combine motion information into the tracking process to improve the tracking algorithm. The motion information is obtained through difference image between consecutive frames, which will be explained in the following section.

4.3.2 Obtaining Difference Image Mask

We create difference image mask to indicate the motion between consecutive frames. The mask is obtained by thresholding the absolute difference image between consecutive frames. Otsu thresholding method [18] is employed to determine the optimum threshold for each difference image, avoiding manually setting a fixed threshold for all images.

CHAPTER 4. SINGLE CAMERA TRACKING

Figure 4.10: Face tracking

The basic idea is that we use a dynamic threshold to binarize the difference image. Otsu's thresholding method [18] is used to find the threshold to binarize the difference image, i.e. pixels are classified as difference or no difference. The threshold for the R,G and B color channels are found respectively and used to get the final difference mask, i.e. motion mask of consecutive frames.

Figure 4.11: Affine tracking on a two person walking sequence. (a) track the person walking in from right side; (b) track the person walking in from left side.

CHAPTER 4. SINGLE CAMERA TRACKING

Otsu's method is an image segmentation method that tries to divide the image gray level into two classes. The method is based on a very simple idea: find the threshold that minimizes the weighted within-class variance. Since this method operates directly on the gray level histogram, it is fast once the histogram is computed.

Let the normalized histogram be denoted by $P(l)$, where l is the gray level and $l = 1 : L$ (L is the total number of gray levels). Otsu's method finds a grey value t that divides the histogram into two classes: $l = 1 : t$ and $l = t + 1 : L$, maximizing the between-class variance. More detailed deduce can be found in [18]

When using Otsu's method to do image segmentation, it is assumed that the histogram of the image is bimodal, which may not be true for all images. But we use Otsu's method to find the optimum threshold of a difference image, the histogram of which is bimodal when motion object exists, i.e. pixel values brought by noises and those brought by motion between consecutive frames.

It is possible that no motion happens in one frame, and the difference image is only brought by noises. In this case, most of the pixels in the difference image mask will be '1'. A ratio r is set to identify this case, and 0.8 was used in our experiments. If the percentage of total pixels with value '1' in the difference exceeds r , then it means that there is no significant motion in that frame.

Several steps for creating the difference image mask are as follows:

- (i) Obtain absolute difference image.

The incoming frame $I(t)$ is compared with the previous image $I(t - 1)$ to obtain absolute difference image $D(t)$.

- (ii) Thresholding the difference image.

Otsu's thresholding method is used to find the threshold for the difference image at each color channel, i.e. $T_r = \text{Otsu}(D_r)$, $T_g = \text{Otsu}(D_g)$, $T_b = \text{Otsu}(D_b)$, where D_r ,

CHAPTER 4. SINGLE CAMERA TRACKING

D_r, D_g, D_b are the difference image for r, g, b color channel respectively. The thresholds are then used to form 3 masks: f_r, f_g, f_b , in which '1' indicate difference and '0' means no difference happens.

- (iii) Logical 'OR' are used to form the final difference image mask: $f = f_r \vee f_g \vee f_b$.
- (iv) Check the difference mask. If the percentage of total pixels with value '1' in the difference exceed r , that means no motion happens in this frame. Set all pixels in mask to '0'.

Example difference image masks are shown in Figure 4.12. The first row shows the original frames and the second row shows obtained difference image masks.

Figure 4.12: Difference image mask

4.3.3 Improved Affine Tracking

In this section, we improve the affine tracking algorithm introduced in section 4.3.1 by combining the motion information from the difference image.

Candidate motion areas are obtained by processing the vertical projection histogram of the difference image mask, i.e. the projection of difference image mask onto the horizontal axis. Figure 4.13 (a) shows a foreground mask, its projection and the possible moving region in the original frame.

CHAPTER 4. SINGLE CAMERA TRACKING

From the vertical projection histogram, shown in Figure 4.13 (a), we can obtain the horizontal coordinates range of a particular foreground region. Then the vertical slice of the difference image defined by the obtained horizontal coordinates is projected to the vertical axis to get the vertical range of the region. Then a bounding box is used to describe a region where motion exists.

Figure 4.13: Projection of difference image mask

When motion is small, one candidate motion area may split into several, or when occlusion happens, several different motion areas may merge into one, as shown in Figure 4.13 (b). However, we only use these areas to obtain the starting point of the iteration to find the most similar region, so even if there are some errors of the starting points, the optimization process will be able to converge to the correct region.

The affine parameter p_5 , p_6 in Equation (Eq. 4.10) indicate the translation of x and y between the current image region and the template. Now we have the motion areas by processing the difference image mask, we use the location of the motion areas as different p_5 and p_6 , i.e. different starting parameter $\mathbf{p}$ can be obtained.

CHAPTER 4. SINGLE CAMERA TRACKING

The steps of improved affine tracking are as follows:

- (i) Initialize the transformation parameter $\mathbf{p}$ of the template in the current frame.

First, let $\mathbf{p} \leftarrow \mathbf{p}_{pre}$, where $\mathbf{p}_{pre}$ is the estimated parameter in the previous frame,

Then, initialize the transformation parameter $\mathbf{p}_2, \dots, \mathbf{p}_k$, where k is the number of candidate motion areas. $\mathbf{p}_2, \dots, \mathbf{p}_k$ have different p_5 and p_6 from $\mathbf{p}_1$, which are given by the candidate motion areas obtained from the difference image mask.

- (ii) Obtain the final $\mathbf{p}_1, \dots, \mathbf{p}_k$ by performing the minimization process respectively, i.e iteratively minimizing equation Eq. 4.13 and updating parameter $\mathbf{p}$ using equation Eq. 4.14. Record final errors between the template and the transformed region Equation Eq. 4.11 in the image for each parameter as $E_1, \dots, E_k$.
- (iii) Find the minimum of $E_1, \dots, E_k$, suppose it is E_m . Choose the corresponding parameter $\mathbf{p}_m$ as the optimum transformation parameter.

The tracking steps mentioned above are a brute force way in that we use several starting points in each frame, i.e. several searching processes are conducted in each frame. Numbers of the searching times in each frame can be controlled by other parameters, for example we can use the trajectory of the tracked objects to predict when there will be an occlusion and only use information from difference image from the time occlusion happens till the objects are correctly tracked again after occlusion. If there is no motion and the object stay static in one frame, then there is no need to do searching in this frame.

The improved tracking method is tested on several image sequences and some tracking results are shown in Figure 4.14 and Figure 4.15, where (a) shows the original affine based tracking results and (b) shows the improved affine based tracking results. In Figure 4.14 (a), the tracker drifts to the wrong person when occlusion happens and follows the wrong

CHAPTER 4. SINGLE CAMERA TRACKING

person after occlusion. While in Figure 4.14 (b), as we employed motion information from difference image, the tracker managed to follow the correct person after occlusion, though it attaches to the wrong person when the target person is just occluded. Figure 4.15 shows the tracking results on another two person walking sequence where the improved affine tracker also shows good tracking performance.

Figure 4.14: Improved affine tracking on sequence 1. (a) shows the results obtained by original affine tracking method; (b) shows the results obtained by improved affine tracking method

Figure 4.15: Improved affine tracking on sequence 2. (a) shows the results obtained by original affine tracking method; (b) shows the results obtained by improved affine tracking method

4.4 Conclusion

In this chapter, three tracking methods are presented, i.e. mean shift tracking algorithm, tracking by motion region correspondence method and difference image based improved affine tracking method. Their disadvantages are pointed out and possible improvements are also given.

The mean shift algorithm is an efficient technique for tracking 2D object through image sequences and our experiments on one person walking sequence show good results. However, the mean shift tracker searches the image space deterministically and when occlusion happens, the tracker might drift away. Sampling techniques such as particle filter may be combined together to improve the tracking performance.

The tracking method by corresponding detected motion regions is proposed in the second part of this chapter. This method works fairly well and no fixed scale(for example, bounding box of the same size) is needed, the person is tracked as his actual size. One problem is that when people are in occlusion, they will be detected as one connected region instead of single person. The histogram based representation is another possible problem when two tracked persons have very similar histograms. Future work is required to solve these problems.

After that, an improved affine tracking method is also introduced by employing motion information from consecutive difference images. Based on the difference image, more starting points of the affine search are included and the tracker would find the optimum region from different locations. Thus, the tracker would be able to return to the correct track even if it temperately drifts away. Employing difference image into tracking is effective with sequences taken by fixed camera, however, when the background is changing, it is hard to differentiate the motion of object and the background only based on difference images. How to make use of the motion information when with moving background is a question needs to be further investigated.

Chapter 5

Multi-Camera Tracking

In the beginning of this chapter, the motivation for multi-camera tracking is reviewed. Then the correspondence problem in multi-camera tracking is presented and some preliminary multi-camera tracking results are given.

5.1 Multi-Camera Tracking

Human tracking is a well established research field [6] and has been used in surveillance, smart room applications etc. A surveillance system takes care of monitoring an area in real time and tracking is needed to provide a semantic description of the activities taking place in the interest area. In smart room applications, tracking is needed to keep track of the persons in the room and to provide information for high level processing like expression analysis. Different issues must be faced to obtain interest information at different levels, for example different algorithms are required when tracking the whole human body or when tracking the fine details of faces.

Traditionally, tracking techniques have been developed within single camera systems. Various single camera tracking approaches have been presented and good performances have been obtained. However, single camera systems are affected by different problems related to the systems' limited field of view, their unavailability of 3D information and their low performances during occlusion.

Multi-camera systems provide an alternative to overcome the above mentioned challenges. With multiple cameras, larger coverage of the interest area is available, and 3D information can be obtained. Besides, we have the flexibility to focus one camera on a region of interest to get detailed information, while still knowing the global status of the area through other cameras. Thus, with a multi-camera system, tracking can be better resolved, satisfying different requirements.

Recently, multi-camera tracking has gained increasing interests. In the following of this chapter, several issues in multi-camera tracking are discussed and our preliminary experimental results are presented.

5.2 Homography Based Cross Camera Correspondence

Tracking in multi-camera systems can be viewed as a problem of cross camera correspondence [13], i.e. establishing correspondence between objects seen from multiple views. Establishing correspondence helps in obtaining more information about the tracked person, since another view of the person is then available.

Several approaches have been proposed to solve cross camera correspondence problem.

In [68], a region based method is employed, where they identify people in multiple views based on characteristics of human body regions. A rough correspondence is obtained between different detected regions across different cameras based on their color models. Then corresponding regions are back projected in 3D space in a manner that yields 3D points guaranteed to lie inside the objects. To use back projection in 3D space, all the cameras need to be calibrated.

An approach that does not rely on calibrated cameras is proposed in [13]. It assumes that tracking is done in each camera, and the tracks of the same object is matched at

CHAPTER 5. MULTI-CAMERA TRACKING

the instance the person enters the camera's field of view. All subsequent points in the object's trajectory are automatically corresponded.

Homography based methods have also been employed [77]. Homography specifies the correspondence of points on the same plain (usually the floor people walk on) seen from different views. If we assume that we can detect the foot point of the person in one camera, then homography can be used to find the corresponding point in another camera and the cross camera correspondence is realized in this way. However when the feet of the person are not detected correctly, the corresponding points recovered by homography may be wrong.

Figure 5.1: Sample frames of one person walking ((a) and (b)) and two-person walking ((c) and (d)) taken from two cameras simultaneously

We employ homography based camera correspondence in our experiments and show experimental results on one-person and two-person walking sequences in a room. Since single camera tracking results are available using the algorithm described in Chapter 4, it is sufficient to use homography to get correspondence at the instance the person enters one camera's field of view.

In [96], a homography is defined as an invertible mapping of points and lines on the projective plane. Assume we have a pair of corresponding points on a plane seen from 2 cameras, i.e. $p_1 = \begin{pmatrix} x \\ y \\ w \end{pmatrix}$ and $p_2 = \begin{pmatrix} x' \\ y' \\ w' \end{pmatrix}$ and a transformation matrix $M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}$. The relation between p_1 and p_2 can be written as:

$$p_1 = rMp_2 \quad (\text{Eq. 5.1})$$

where r is an arbitrary nonzero constant. In homogenous coordinates, homographies and points are defined up to a nonzero scalar r , i.e. there are 8 degrees of freedom for homography. If we set $w = 1$. Eq. 5.1 can be written as

$$p_1 = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} p_2 \quad (\text{Eq. 5.2})$$

where H is the 3×3 homography matrix with 8 degrees of freedom.

There are three equations in Eq. 5.2, but only two of them are linearly independent because of the scale factor. So every point correspondence (p_1, p_2) gives two equations and at least four point correspondences are needed to compute a homography.

Since no marked landmarks on the floor are available, feature corresponding points are found manually. As shown in Figure 5.1, the centroids of the rectangles near the person's feet are used as the corresponding points. Homography matrix is recovered by the normalized direct linear transformation(*DLT*) algorithm given by Hartley and Zisserman [96].

After homography matrix is recovered, the cross camera correspondence can be solved. Suppose there is one person on the ground plane(floor), and a bounding box is used to represent it, as shown in Figure 5.2. Let P be the middle-bottom point on the bounding box and p_1, p_2 be the two corresponding image points in camera 1 and camera 2. Set image 1 as the reference image, and p'_2 is the mapping of p_2 from image 2 to image 1 by the recovered homography matrix H , i.e. $p'_2 = H * p_2$. Since errors exist both in the detected points and homography matrix H , it is very unlikely that p_1 and p'_2 superimpose on each other. Let d be the Euclidian distance between p_1 and p'_2 . If d is less than a threshold, p_1 and p_2 are decided to be representing the same 3D point P , hence correspondence is established.

CHAPTER 5. MULTI-CAMERA TRACKING

Figure 5.2: Illustration of homography based cross camera correspondence

5.3 Experimental Results

Figure 5.3: Tracking results of two sequences recording the same scene. The frame numbers from (a) to (d) are 155, 170, 190, 216. The top row shows images from camera 1 and the second row shows those from camera 2.

In section 5.2, homography is employed to solve the cross camera correspondence problem. Thus the persons tracked in two views can be related to each other. Some of the tracking results are shown in Figure 5.3 and Figure 5.4.

Images of the same scene taken by two cameras at different view points are presented in Figure 5.3. First row shows images from camera 1, and second row shows images from camera 2. In the 155th frame, the person appears in the view filed of camera 2 and in

CHAPTER 5. MULTI-CAMERA TRACKING

the 216th frame, the person walks out from camera 1's field of view. Tracking is maintained smoothly in both cameras using the single camera tracking algorithm described in Chapter 4.

Figure 5.4: Tracking results of a two-person walking sequence recorded by two cameras. (a1) to (h1) show images taken by camera 1, and (a2) to (h2) show images taken by camera 2. The frame numbers of (a1) to (h1) and (a2) to (h2) are 100, 110, 125, 175, 200, 225, 230, 250.

The tracking results of a two-person walking scene taken by two cameras are shown in Figure 5.4. In the sequence, one person walks in from left side (person A) and the

other person (person B) walks in from the right side. They walk to each other, shake hands and leave.

Figure 5.4 (a1) to (h1) show images taken by camera 1. In (a1), only person A is in camera 1's view and a red bounding box is used to denote the tracking result. In (b1) person B appears, and a green bounding box is assigned to him. They walk toward each other and the occlusion is shown in (d1). When the two persons are in occlusion, there is only one object candidate from the motion detection step and a blue bounding box is used to denote the occlusion. After they walk away from each other, the two persons are tracked again correctly, as in (e1), (f1). Person A is walking out of camera's view in (f1) and person B is continuously tracked as shown in (g1).

Figure 5.4 (a2) to (h2) show images taken by camera 2. In (a2) and (b2), only person B is in camera 2's view and a green bounding box is used to denote the tracking result. In (c2), person A appears and a red bounding box is assigned to him. Occlusion is shown in (d2) and the two persons are tracked correctly after they separate, as shown in (e2). In (f2), person B is walking out of camera 2's field of view and person A is continued to be tracked in (f2) and (g2).

In Figure 5.4 (b1), person B is entering camera 1's field of view from the right side. This person has already been tracked in camera 2 and a green bounding box is assigned to him. Using homography matrix, we know that the new person in camera 1 is actually the person that has been tracked in camera 2, so a bounding box of the same color is given to the new person in camera 1. It is the same when person A enters camera 2's field of view from the left side while he has already been tracked in camera 1.

5.4 Collaborative Tracking Between Cameras

After the cross camera correspondence problem is solved, more information is available for the same person. So instead of just doing tracking in each view, some collaboration

CHAPTER 5. MULTI-CAMERA TRACKING

can be done between the two cameras.

One way of collaboration is to choose a better view of the tracked person. From the two image sequences shown in Figure 5.3, a composite image sequence can be made by selecting the front view from the two image sequences at the same time instant. The result frames are like those in Figure 5.5. We can also use the tracked region to composite a video sequence for the person as shown in Figure 5.6. Thus a profile can be build for each person.

Figure 5.5: Sample frames of the composite image sequence. (a) is the 155th frame taken by camera 1. (b), (c), (d) are the 170th, 190th and 216th frames taken by camera 2.

Figure 5.6: Sample tracked persons of the composite image sequence. (a) is the 155th frame taken by camera 1. (b), (c), (d) are the 170th, 190th and 216th frames taken by camera 2.

Another way of collaboration between multiple cameras is needed when one person is occluded in one view while it is visible in another view. Based on the homography between two views, the location of the occluded person can be robustly estimated by projecting back the location of the person in the visible view. In [97], similar homography based

occlusion reasoning is used and tracking results of a outdoor sequence taken by 2 cameras are given.

For the two camera images shown in Figure 5.4, the occlude person is not visible in either view, so homography based method cannot be used to estimate the location of the occluded person.

5.5 Conclusion

Cross camera correspondence is an important issue in multi-camera system. In the section 5.2, several approaches are first reviewed, and a homography based cross camera correspondence method is employed in our experiments. Some experimental results are given in section 5.3.

With multiple cameras, if the cross camera correspondence is solved, more information can be obtained for the target person, i.e. we have images from more than one view. Thus, some collaboration can be done during multi-camera tracking. Selecting the best view and occlusion handling are two aspects where collaboration can be done.

Chapter 6

Conclusion

6.1 Conclusion

This thesis summarizes our efforts to understand and improve human motion detection and tracking techniques. The research work is mainly focused on the following aspects:

(i) Motion detection

Two motion detection methods are proposed. The first is the entropy based motion detection method. This is a novel motion detection method in that the entropy of the difference image is used for motion detection and no background model is needed. Another simple yet effective motion detection method is proposed which is suitable for realtime applications. In this method, an optimum threshold is calculated in each frame to binarize the difference image. Experimental results demonstrate its effectiveness.

(ii) Single camera tracking

First, we implemented the mean shift tracking algorithm proposed in [17]. It is an efficient tracking method, however, since the mean shift tracker searches the image space deterministically, the tracker might drift to a local minimum. Then a tracking method by corresponding detected motion regions is proposed. Though

CHAPTER 6. CONCLUSION

simple, this method tracks the people fairly well. But the tracking algorithm may fail if two tracked persons have very similar histograms, since color histogram is used to represent them. More distinctive person model needs to be built. Improvement is also required to better deal with occlusion problems. After that, an improved affine tracking method is presented. We combine motion detection results into the affine tracking process, i.e. more starting points of the affine optimization are added by processing the motion detection results. In this way, the improved affine tracking method is much more robust, especially when there is occlusion.

(iii) Multi-camera tracking

Some preliminary multi-camera tracking results are shown. Cross camera correspondence is first established employing homography. Given images from multiple cameras, more information is available for a tracked person, so some collaboration between multiple single camera tracking processes can be done. An example is given, selecting the front view of a walking person.

6.2 Future work

Although research on human motion detection and tracking have been carried out by many distinguished researchers, more efforts should be devoted to the area to conquer the existing problems. Because of the time limit for a master student, though given lots of guidance, lots of work can be done to make the technology more applicable to the real world. These include robust tracking of human across multiple cameras, multi-modal information fusion for efficient tracking and quick calibration of multi-camera setup etc.

(i) Robust human detection and tracking with occlusions in videos.

Occlusions in natural scenes produce problems in many human tracking and detection algorithms. Therefore robust algorithms capable of handling occlusions are

CHAPTER 6. CONCLUSION

needed. This is especially a major problem if higher level analysis are done based on the motion detection and tracking results. Take gesture recognition for example, which will usually employ motion detection and tracking results as input. Assume one person's hand is partially blocked by another person when the two are communicating with each other, which is quite common in daily life. If the motion detection and tracking are not be able to detect or predict the location of the occluded hand, the information given to gesture recognition system is not complete and the system may fail.

(ii) Robust whole body multi-camera tracking

From related literatures, we notice that the fundamental problem of multi-camera tracking is how to establish correspondences of objects in different views. The motion correspondence problem in single camera tracking is extended to multiple cameras. We wish to explore new methods to establish such correspondences. One possibility is to model the trajectories of moving person as a time series and to determine similarities of the time series parameters to ascertain tracks belonging to the same object. Whether this can be done in an un-calibrated camera setup needs further investigation.

We have shown preliminary results in multi-camera tracking where homography is used to establish correspondence. However, the constraint in using homography is that a plane on which the target object is located should be available, thus more robust methods are needed. Contour based approaches, e.g. level sets, which have been recently used for single camera tracking, maybe a possible direction. Another possibility is to explore if the mean shift tracking algorithm can be adapted to multi-camera tracking in the sense that data from more than one view is used to search the mode for a particular data distribution.

CHAPTER 6. CONCLUSION

(iii) Multi-camera face tracking for best view selection.

In certain applications like a smart meeting room, it may be more meaningful to track the face of the person who is speaking. The ultimate objective might be to extract features for facial expression analysis. In order to track the face, we need to first locate it. Although there are face identification methods that use skin color, they are prone to errors due to change in lighting conditions. Some structural features that enable identification of face regions in a sequence maybe explored and incorporated into the tracking algorithm. Moreover, if a face is visible in more than one view, it is desirable that the face is tracked in a view that shows the maximum area, e.g. a frontal view is more preferable to track than a side view. We have illustrated the results of such view selection for whole body in Figure 5.5 and Figure 5.6, where the view is selected manually. To automate this process is a preferred choice for practical use.

(iv) Multi-modal information fusion

In this thesis, only vision based motion detection and tracking methods are discussed. With easily availability of data from different sensors like microphone arrays, it is worthwhile to use information from multi-sources for the task of tracking speaking persons. Fusion of visual and audio information to achieve such tracking is a natural extension of the vision based research. For example, if the environment is not reverberative, the location of the audio source can be estimated through Direction of Arrival(DoA) algorithm [98]. Then the visual process of identifying the speaker and tracking him can be limited to the area identified by the audio array. Thus the problem becomes the detection and tracking of the speaker using additional audio information. Implicit in this problem is the issue of camera calibration. If the location of the speaker is identified by the DoA algorithm and the

CHAPTER 6. CONCLUSION

camera array needs to search for a face region around this region, it implies that the cameras are calibrated. This leads to the next topic for future work.

(v) Quick self-calibration of a multi-camera set up.

As mentioned above, multi-modal information fusion would need calibrated cameras. Some multi-camera calibration techniques have already been proposed [99][100] and a multi-camera self-calibration tool box by Svoboda et al. is available on their website [101]. In their method, Svoboda et al. propose a automatic calibration method which yields complete camera projection models and only requires the user to wave the laser pointer throughout the working volume to get point correspondences. Though this is a very powerful multi-camera calibration method, it requires that all the cameras are fixed and once the camera moves, the systems must be re-calibrated again. So the calibration will be a problem if not all cameras are static. Online calibrating of multiple cameras, i.e. obtaining cameras parameters while one or more of them are not fixed are desirable.

References

- [1] W. Hu, T. Tan, L. Wang, and S. Maybank, “A survey on visual surveillance of object motion and behaviors,” *IEEE Transactions on Systems, Man, and Cybernetics. PART C: Applications and Reviews*, vol. 34, pp. 903–908, August 2004.
- [2] S. T. Birchfield and S. Rangarajan, “Spatiograms versus histograms for region-based tracking,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.
- [3] A. Elgammal, “Learning to track: Conceptual manifold map for closed-form tracking,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2005.
- [4] J. J. Wang and S. Singh, “Video analysis of human dynamics - a survey,” *Real-Time Imaging*, vol. 9, no. 5, pp. 321–346, 2003.
- [5] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach, Chapter 19*. Prentice Hall, 2003.
- [6] T. B. Moeslund and E. Granum, “A survey of computer vision-based human motion capture,” *Computer Vision and Image Understanding(CVIU)*, vol. 81, no. 3, pp. 231–268, 2001.
- [7] M. Weiser, R. Gold, and J. S. Brown., “The origins of ubiquitous computing research at PARC in the late 1980s,” *IBM Systems Journal*, vol. 38, no. 4, pp. 693–696, 1999.
- [8] W. Mark, “Turning pervasive computing into mediated spaces,” *IBM Systems Journal*, vol. 38, no. 4, pp. 677–692, 1999.

REFERENCES

- [9] A. Waibel, T. Schultz, M. Bett, R. Malkin, I. Rogina, R. Stiefelhagen, and J. Yang, "SMaRT: the smart meeting room task at ISL," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, pp. IV–752–755, April 2003.
- [10] S. A. Drab and N. M. Artner, "Motion detection as interaction technique for games & applications on mobile devices," in *Pervasive Mobile Interaction Devices (PERMID 2005) Workshop at the Pervasive 2005*, May 2005.
- [11] V. Vezhnevets, "Face and facial feature tracking for natural human-computer interface," in *Graphicon 2002*, 16-21 Sep 2002.
- [12] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfindex: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 19, pp. 780–785, July 1997.
- [13] S. Khan and M. Shah, "Consistent labeling of tracked objects in multiple cameras with overlapping fields of view," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 25, Oct 2003.
- [14] J. Black and T. Ellis, "Multi camera image tracking," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, July 2001.
- [15] K. Nummiaro, E. Koller-Meier, T. Svoboda, D. Roth, and L. V. Gool, "Color-based object tracking in multi-camera environments," in *25th Pattern Recognition Symposium, DAGM 2003*, pp. 591–599, September 2003.
- [16] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 246–252, 1999.
- [17] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, (Hilton Head Island, South Carolina), pp. 142–149, 2000.

REFERENCES

- [18] N. Otsu, "A threshold selection method from grey-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, pp. 62–66, January 1979.
- [19] G. Hager and P. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 20, pp. 1025–1039, October 1998.
- [20] M. Black and A. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," in *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 329–342, 1996.
- [21] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, Feb. 2002.
- [22] I. Haritaoglu, D. Harwood, and L. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, no. 8, 2000.
- [23] R. Collins, A. Lipton, and T. Kanade, "A system for video surveillance and monitoring," in *Proceedings of American Nuclear Society (ANS) Eighth International Topical Meeting on Robotics and Remote Systems*, pp. 25–29, April 1999.
- [24] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," in *Proceedings of the IEEE*, pp. 1151–1163, July 2002.
- [25] N. Friedman and S. Russell., "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 175–181, 1997.
- [26] F. Porikli and O. Tuzel, "Human body tracking by adaptive background models and mean-shift analysis," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, March 2003.

REFERENCES

- [27] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 255–261, 1999.
- [28] A. Lipton, H. Fujiyoshi, and R. Patil, "Moving target classification and tracking from real-time video," in *IEEE Workshop on Applications of Computer Vision*, pp. 8–14, 1998.
- [29] S. Y. Chien, S. Y. Ma, and L. G. Chen., "Efficient moving object segmentation algorithm using background registration technique," *IEEE Transactions on Circuits and Systems for video technology*, vol. 12, pp. 780–785, July 2002.
- [30] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 25, October 2003.
- [31] S. J. McKenna, S. Jabri, Z. Duric, and A. Rosenfeld, "Tracking groups of people," *Computer Vision and Image Understanding (CVIU)*, no. 80, pp. 42–56, 2000.
- [32] Y. Yang and M. Levine, "The background primal sketch: an approach for tracking moving objects," *Machine Vision and Applications*, vol. 5, pp. 17–34, 1992.
- [33] S.-C. S. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Proceedings of SPIE, Visual Communications and Image Processing 2004*, vol. 5308, pp. 881–892, January 2004.
- [34] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proceedings of 2nd European Workshop on Advanced Video Based Surveillance Systems*, vol. 5308, 2001.
- [35] A. Elgammal, D. Harwood, and L. Davis., "Non-parametric model for background subtraction," in *IEEE ICCV Frame-rate workshop*, Sept 1999.
- [36] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, pp. 585–601, 2003.

REFERENCES

- [37] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell, "Toward robust automatic traffic scene analysis in real-time," in *Proceedings of IEEE International Conference on Pattern Recognition (ICPR)*, pp. 126–131, 1994.
- [38] S. Jabri, Z. Duric, H. Wechsler, and A. Rosenfeld, "Detection and location of people in video images using adaptive fusion of color and edge information," in *Proceedings of IEEE International Conference on Pattern Recognition (ICPR)*, pp. 627–630, 3-7 Sept. 2000.
- [39] O. Javed, K. Shafique, and M. Shah, "A hierarchical approach to robust background subtraction using color and gradient information," in *Proceedings of Workshop on Motion and Video Computing (MOTION'02)*, pp. 22–27, 5-6 Dec. 2002.
- [40] Y. Bar-Shalom and T. E. Fortmann, *Tracking and data association*. Academic press, INC. Harcpirt Brace Jovanovich, Publishers, 1986.
- [41] K. Toyama and A. Blake, "Real-time 3-d tracking of the human body," in *Proceedings of IMAGE'COM 96*, May 1996.
- [42] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, pp. 1330–1334, Oct 2000.
- [43] Z. Zhang. Microsoft Easy Camera Calibration Tool: <http://research.microsoft.com/~zhang/calib/>.
- [44] R. Willson. Tsai Camera Calibration Software: http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/_rgw/www/TsaiCode.html.
- [45] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab: <http://www.vision.caltech.edu/bouguetj/calib-doc/>.
- [46] K. Toyama and A. Blake, "Probabilistic tracking in a metric space," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 591–599, July 07 - July 14 2001.

REFERENCES

- [47] I. Haritaoglu, R. Cutler, D. Harwood, and L. S. Davis, “Backpack: Detection of people carrying objects using silhouettes,” *Computer Vision and Image Understanding (CVIU)*, vol. 81, pp. 385–397(13), March 2001.
- [48] Z. Zivkovic and B. Kröse, “An EM-like algorithm for color-histogram-based object tracking,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2004.
- [49] D. Comaniciu, “Bayesian kernel tracking,” in *Proceedings of Annual Conference of the German Society for Pattern Recognition (DAGM02)*, pp. 438–445, 2002.
- [50] A. Elgammal, R. Duraiswami, and L. Davis, “Probabilistic tracking in joint feature-spatial spaces,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. I: 781–788, September 16-18 2003.
- [51] G. Rigoll, S. Eickeler, and S. Mller, “Person tracking in real-world scenarios using statistical methods,” in *Proceedings of IEEE Conference on Automatic Face and Gesture Recognition*, pp. 342–347, March 2000.
- [52] M. Yamamoto and K. Koshikawa, “Human motion analysis based on a robot arm model,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 664–665, 1991.
- [53] Y. Guo, G. Xu, and S. Tsuji, “Tracking human body motion based on a stick figure model,” *Journal of Visual Communication and Image Representation (JVCIR)*, vol. 5, pp. 1–9, 1994.
- [54] D. Ramanan and D. A. Forsyth, “Finding and tracking people from the bottom up,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 467–474, June 18-20 2003.
- [55] R. Collins, “Mean-shift blob tracking through scale space,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 234–240, June 18-20 2003.

REFERENCES

- [56] T.-L. Liu and H.-T. Chen, “Real-time tracking using trust-region methods,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, pp. 397–402, Mar 2004.
- [57] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, pp. 603–619, May 2002.
- [58] T. Lindeberg, “Feature detection with automatic scale selection,” *International Journal of Computer Vision (IJCV)*, vol. 30, pp. 79–116, November 1998.
- [59] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*. MPS/SIAM Series on Optimization, SIAM, Philadelphia, August 2000.
- [60] M. Isard and A. Blake, “Contour tracking by stochastic propagation of conditional density,” in *Proceedings of European Conference on Computer Vision (ECCV)*, pp. 343–356, 1996.
- [61] B. Li and R. Chellappa, “A generic approach to simultaneous tracking and verification in video,” *IEEE Transactions on Image Processing*, vol. 11, pp. 530–544, May 2002.
- [62] R. E. Kalman, “A new approach in linear filtering and prediction problems,” *Transactions of the American Society of Mechanical Engineers [ASME], Journal of Basic Engineering*, vol. 82, pp. 35–45, March 1960.
- [63] A. Utsumi, *Tracking Multiple persons from multiple camera images*, ch. 2, pp. 7–42. Analyzing video sequences of multiple humans, Kluwer Academic Publishers, 2002.
- [64] J. Strom, T. Jebara, S. Basu, and A. Pentland, “Real time tracking and modeling of faces: an EKF-based analysis by synthesis approach,” in *IEEE International Workshop on Modelling People*, vol. 2, pp. 55–61, Sept 20 1999.
- [65] C. Hue, J.-P. L. Cadre, and P. Prez, “A particle filter to track multiple objects,” in *IEEE Workshop on Multi-Object Tracking (WOMOT’01)*, pp. 61–68, July 08 - August 08 2001.

REFERENCES

- [66] Y.-Q. Liu, Y. Shen, and Z.-Y. Liu, “A new smoothing particle filter for tracking a maneuvering target,” in *Proceedings of International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 1004 – 1008, July 2-5 2003.
- [67] C. Chang and R. Ansari, “Kernel particle filter: iterative sampling for efficient visual tracking,” in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, vol. 2, pp. 977–980, Sept 14-17 2003.
- [68] A. Mittal and L. Davis, “M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene,” *International Journal of Computer Vision (IJCV)*, vol. 51, pp. 189–203, February 2003.
- [69] Q. Cai and J. Aggarwal, “Tracking human motion in structured environments using a distributed-camera system,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 21, pp. 1241–1247, November 1999.
- [70] T.-H. Chang and S. Gong, “Tracking multiple people with a multi-camera system,” in *IEEE Workshop on Multi-Object Tracking (WOMOT’01)*, July 08 - August 08 2001.
- [71] A. Utsumi, H. Mori, J. Ohya, and M. Yachida, “Multiple-camera-based human tracking using non-synchronous observations,” in *Proceedings of Fourth Asian Conference on Computer Vision*, pp. 1034–1039, 2000.
- [72] J. Kang, I. Cohen, and G. Medioni, “Tracking people in crowded scenes across multiple cameras,” in *Proceedings of Asian Conference on Computer Vision*, January 2004.
- [73] S. L. Dockstader and A. M. Tekalp, “Multiple camera fusion for multi-object tracking,” in *IEEE Workshop on Multi-Object Tracking (WOMOT’01)*, July 08 - August 08 2001.
- [74] A. Mittal and L. Davis, “Unified multi-camera detection and tracking using region-matching,” in *IEEE Workshop on Multi-Object Tracking (WOMOT’01)*, July 08 - August 08 2001.

REFERENCES

- [75] P. H. Kelly, A. Katkere, D. Y. Kuramura, S. Moezzi, and S. Chatterjee, "An architecture for multiple perspective interactive video," in *Proceedings of the third ACM International Conference on Multimedia*, pp. 201–212, ACM Press, 1995.
- [76] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer, "Multi-camera multi-person tracking for easy living," in *3rd IEEE International Workshop on Visual Surveillance*, July 2000.
- [77] L. Lee, R. Romano, and G. Stein, "Monitoring activities from multiple video streams: Establishing a common coordinate frame," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, August 2000.
- [78] Y. Caspi and M. Irani, "A step towards sequence-to-sequence alignment," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2, pp. 758–767, June 2000.
- [79] V. Kettner and R. Zabih, "Bayesian multi-camera surveillance," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2, pp. 2253–2259, June 23 - 25 1999.
- [80] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking across multiple cameras with disjoint views," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 952–957, 2003.
- [81] J. Guo, E. S. Chng, and R. Deepu, "Foreground motion detection by difference-based spatial temporal entropy image," in *Proceedings of IEEE Region 10 Conference (TENCON)*, vol. 1, pp. 379–382, Chiang Mai, Thailand, 21-24 Nov 2004.
- [82] J. Guo, R. Deepu, and E. S. Chng, "Motion detection with adaptive background and dynamic thresholds," in *Proceedings of Fifth International Conference on Information, Communication and Signal Processing (ICICS)*, pp. 41–45, Bangkok, Thailand, 6-9 Dec 2005.
- [83] W.-B. Tao, J.-W. Tian, and J. Liu, "Image segmentation by three-level thresholding based on maximum fuzzy entropy and genetic algorithm," *Pattern Recognition Letters*, vol. 24, August.

REFERENCES

- [84] T. Pun, “A new method for gray-level picture thresholding using the entropy of the histogram,” *Signal Processing*, vol. 2, August.
- [85] H. Zhu, Z. Fu, and Z. Li, “A new image thresholding method based on relative entropy,” in *Proceedings of IEEE International Conference on Communications, Circuits and Systems and West Sino Expositions*, Aug. 2002.
- [86] C. Leung and F. Lam, “Maximum gray-scale image entropy segmetation,” in *International workshop on Multimedia Data Storage, Retrial, Integration and Applications*, Jan. 2000.
- [87] Y.-F. Ma and H.-J. Zhang, “Detecting motion object by spatial-temporal entropy,” in *Proceedings of IEEE International Conference on Multimedia and Expo(ICME) 2001*, Aug. 2001.
- [88] J. Guo, R. Deepu, and E. S. Chng, “Motion detection and human tracking for smart room applications,” *WSEAS Transactions on Siganal Processing*, vol. 2, pp. 903–908, June 2006.
- [89] J. Guo, R. Deepu, and E. S. Chng, “Employing difference image in affine tracking,” in *Proceedings of The Eighth IASTED International Conference on Signal and Image Processing*, Honolulu, Hawaii, USA, 14-16 Aug 2006.
- [90] Y. Cheng, “Mean shift, mode seeking, and clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 17, pp. 790–799, August 1995.
- [91] G. R. Bradski., “Computer vision face tracking for use in a perceptual user interface,” in *IEEE Workshop on Applications of Computer Vision*, 1998.
- [92] S. Khan, O. Javed, and M. Shah, “Tracking in uncalibrated cameras with overlapping field of view,” in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance(PETS)*, Dec. 2001.
- [93] G. Hager and P. Belhumeur, “Efficient region tracking with parametric models of geometry and illumination,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 20, no. 10, pp. 1025–1039, 1998.

REFERENCES

- [94] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.
- [95] D. F. A. Jepson and T. El-Maraghi, “Robust online appearance models for visual tracking,” in *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 415–422, 2001.
- [96] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.
- [97] Z. Yue, S. Zhou, and R. Chellappa, “Robust two-camera tracking using homography,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, pp. 1–4, May 2004.
- [98] Z. Shan and T.-S. P. Yum, “A conjugate augmented approach to direction-of-arrival estimation,” *IEEE Transaction on Signal Processing*, vol. 53, pp. 4104–4109, Nov 2005.
- [99] I. Kitahara, H. Saito, S. Akimichi, T. Onno, Y. Ohta, and T. Kanade, “Large-scale virtualized reality,” *Computer Vision and Pattern Recognition, Technical Sketches*, June 2001.
- [100] L. Lee, R. Romano, and G. Stein, “Monitoring activities from multiple video streams: establishing a common coordinate frame,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, pp. 758–767, August 2000.
- [101] T. Svoboda, D. Martinec, T. Pajdla, J.-Y. Bouguet, T. Werner, and O. Chum. Multi-Camera Self-Calibration: <http://cmp.felk.cvut.cz/~svoboda/SelfCal/>.

Publications

- [1] J. Guo, E. S. Chng, and R. Deepu, "Foreground motion detection by difference-based spatial temporal entropy image," in *Proceedings of IEEE Region 10 Conference (TENCON)*, Vol.1, pp.379-382, Chiang Mai, Thailand, 21-24 Nov 2004.
- [2] J. Guo, R. Deepu, and E. S. Chng, "Motion detection with adaptive background and dynamic thresholds," in *Proceedings of Fifth International Conference on Information, Communication and Signal Processing (ICICS)*, pp.41-45, Bangkok, Thailand, 6-9 Dec 2005.
- [3] J. Guo, R. Deepu, and E. S. Chng, "Employing difference image in affine tracking," in *Proceedings of The Eighth IASTED International Conference on Signal and Image Processing*, Honolulu, Hawaii, USA, 14-16 Aug 2006.
- [4] J. Guo, R. Deepu, and E. S. Chng, "Motion Detection and Human Tracking for Smart Room Applications," *WSEAS Transactions on Signal Processing*, Vol.2, No.6, pp. 903-908, June 2006.