Numéro d'ordre : 2011- Année 2011

Université Lumière Lyon 2
Laboratoire d'InfoRmatique en Image et Systèmes d'information
École Doctorale Informatique et Mathématiques de Lyon

Thèse de l'Université de Lyon

Présentée en vue d'obtenir le grade de Docteur,
spécialité Informatique

par

Atif ILYAS

Object Tracking and Re-identification in Multi-Camera Environments

---

Thèse soutenue le 17 Juin 2011 devant le jury composé de :

Rapporteur	Philippe Joly	Professeur, Université Paul Sabatier de Toulouse
Rapporteur	Thierry Chateau	MCF-HDR, Université Blaise Pascal de Clermont-Ferrand
Examinateur	Alain Trémeau	Professeur, Université Jean Monnet de Saint Etienne
Directeur	Serge Miguet	Professeur, Université Lumière Lyon 2
Co-directrice	Mihaela Scuturici	Maître de Conférences, Université Lumière Lyon 2

Laboratoire d'InfoRmatique en Image et Systèmes d'information
UMR 5205 CNRS - Université Lumière Lyon 2 - Bât. C
69676, Bron cedex - France
Tel: +33 (0) 4 78 77 43 77 - Fax: +33 (0)4 78 77 23 38

The dedication goes here.

Abstract

The video surveillance domain shows very strong growth in recent years. But the proliferation of cameras in public or private spaces makes it extremely difficult for human operators to analyze the data produced by these systems. Many techniques for automatic analysis of the video have been proposed by researchers, and begin to be commercially available. But most of these systems consider the cameras independently one of each other. The objective of this thesis is to address the wide area surveillance, covered by multiple non-overlapping field of view cameras. One of the problems we are interested in is the objects re-identification: when an object appears in the field of a camera, the system should decide whether this object has already been observed and monitored by one camera system or it is a new object. We want to perform this task without any a priori knowledge of the cameras position relative to each other.
In the literature, many algorithms exist for moving objects tracking in a video. These algorithms are sufficient to detect object trajectories and to verify that objects have a coherent motion. But these algorithms are not sufficiently robust to object occlusions, intersections, merges and splits. This drawback of current algorithms is problematic, since they form the building blocks of a multi-camera environment. Therefore, the first part of this thesis is to improve the segmentation and object tracking algorithms.
At first, we propose an improvement to the foreground/background segmentation algorithms based on codebook. We also propose an evaluation methodology to objectively compare segmentation techniques, based on the analysis of the precision and recall of algorithms. Based on a test set derived from public databases, we show the good behavior of our modified algorithm.
A second contribution of this thesis concerns the development of a robust and compact descriptor for moving object tracking in videos. We propose a simple 1-D appearance model, called the Vertical Feature (VF), independent of the view angle and of the apparent size of objects. This descriptor provides a good compromise between very compact color models, that lose all the spatial information of tracked object's color, and traditional appearance models, too expensive for deformable objects. We associate a motion model of tracked objects and our descriptor, and show the superiority of a combined model approach on traditional tracking approaches, based on the mean shift or on Kalman filter.
A descriptor is associated with each object tracked by a camera. Multi-camera tracking, we presents a variability of these descriptors, due to changes in lighting conditions, and also due to the technical characteristics of the cameras, which can differ from one model

to the other. We are therefore interested in the problem of the cameras color calibration in order to make similar the descriptors of a same object, seen by different cameras in the system. Existing approaches estimate the Brightness Transfer Functions (BTF) by measuring the response of each camera using known objects. We compare methods based on the Mean BTF (MBTF) and on Cumulative BTF (CBTF) of their color histograms, and show the weaknesses of these approaches when some colors are not enough represented in the objects used for calibration. We propose an alternative (MCBTF) algorithm and we show its superiority over existing methods.
Finally, systematic experiments are conducted on the objects re-identification problem in a multi-camera environment, which allows validating all of our proposed algorithms.

Keywords: foreground-background segmentation, object recognition, object tracking, multi-camera environment, color calibration, object re-identification, evaluation techniques.

Resumé

Le domaine de la vidéosurveillance a connu une très forte expansion ces dernières années. Mais la multiplication des caméras installées dans des espaces publics ou privés, rend de plus en plus difficile l'exploitation par des opérateurs humains des masses de données produites par ces systèmes. De nombreuses techniques d'analyse automatique de la vidéo ont été étudiées du point de vue de la recherche, et commencent à être commercialisées dans des solutions industrielles, pour assister les opérateurs de télésurveillance. Mais la plupart de ces systèmes considèrent les caméras d'une manière indépendante les unes des autres. L'objectif de cette thèse est de permettre d'appréhender la surveillance de zones étendues, couvertes par des caméras multiples, à champs non-recouvrants. L'un des problèmes auxquels nous nous sommes intéressés est celui de la ré-identification d'objets : lorsqu'un objet apparaît dans le champ d'une caméra, il s'agit de déterminer si cet objet a déjà été observé et suivi par l'une des caméras du système. Nous souhaitons effectuer cette tâche sans aucune connaissance a priori du positionnement des caméras les unes par rapport aux autres.
Il existe dans la littérature beaucoup d'algorithmes permettant le suivi des objets en mouvement dans une vidéo. Ces algorithmes sont suffisants pour détecter des fragments de la trajectoire et vérifier que les objets ont un mouvement cohérent. Par contre, ces algorithmes ne sont pas suffisamment robustes aux occultations, aux intersections, aux fusions et aux séparations. Cette insuffisance des algorithmes actuels pose problème, dans la mesure où ils forment les briques de base d'un suivi multi-caméras. Une première partie du travail de thèse a été donc de perfectionner les algorithmes de segmentation et de suivi de façon à les rendre plus robustes.
Dans un premier temps, nous avons donc proposé une amélioration aux algorithmes de segmentation premier plan/arrière plan basés sur les dictionnaires (codebooks). Nous avons proposé une méthodologie d'évaluation afin de comparer de la manière la plus objective possible, plusieurs techniques de segmentation basées sur l'analyse de la précision et du rappel des algorithmes. En nous basant sur un jeu d'essai issu de bases de données publiques, nous montrons le bon comportement de notre algorithme modifié.
Une deuxième contribution de la thèse concerne l'élaboration d'un descripteur robuste et compact pour le suivi des objets mobiles dans les vidéos. Nous proposons un modèle d'apparence simplifié, appelé caractéristique verticale (VF pour Vertical Feature), indépendant de l'angle de vue et de la taille apparente des objets. Ce descripteur offre un bon compromis entre les modèles colorimétriques très compacts, mais qui perdent

toute l'organisation spatiale des couleurs des objets suivis, et les modèles d'apparence traditionnels, peu adaptés à la description d'objets déformables. Nous associons à ce descripteur un modèle de mouvement des objets suivis, et montrons la supériorité d'une approche combinant ces deux outils aux approches traditionnelles de suivi, basées sur le mean shift ou sur le filtre de Kalman.
Chaque objet suivi par une caméra peut ainsi être associé à un descripteur. Dans le cadre du suivi multi-caméras, nous sommes confrontés à une certaine variabilité de ces descripteurs, en raison des changements des conditions d'éclairage, mais également en raison des caractéristiques techniques des caméras, qui peuvent être différentes d'un modèle à l'autre. Nous nous sommes donc intéressés au problème de l'étalonnage des couleurs acquises par les caméras, qui visent à rendre identiques les descripteurs d'un même objet observé par les différentes caméras du système. Les approches existantes estiment les fonctions de transfert de luminosité (BTF pour Brightness Tranfert Function) en mesurant la réponse donnée par chaque caméra à des objets connus. Nous comparons les méthodes basées sur une moyenne (MBTF) ou sur un cumul (CBTF) des histogrammes de couleur, et montrons les faiblesses de ces approches lorsque certaines couleurs sont trop peu représentées dans les objets servant à l'étalonnage. Nous proposons une alternative (MCBTF) dont nous montrons la supériorité par rapport aux méthodes existantes.
Enfin, des expérimentations systématiques sont menées sur le problème de la ré-identification d'objets dans un environnement multi-caméras, qui permettent de valider l'ensemble de nos propositions.

Mots clés: segmentation premier plan/arrière plan, reconnaissance d'objet, suivi d'objet, environnement multi-caméras, étalonnage des couleurs, ré-identification d'objet, techniques d'évaluation.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

ARMA : Auto-Regressive Moving Average

BBF : Best Bin First

BBN : Bayesian Belief Network

BTF : Brightness Transfer Function

CB : CodeBook

CBTF : Cumulative Brightness Transfer Function

CCM : Cross-Correlation Matrix

CCTV : Closed Circuit Television

FA : False Alarm

FN : False Negative

FOV : Field of View

FP : False Positive

FPR : False Positive Rate

HOG : Histogram of Oriented Gradient

ICA : Independent Component Analysis

ICH : Inverted Cumulative Histogram

JC :Jaccard Coefficient

LBP : Local Binary Pattern

MBTF : Mean Brightness Transfer Function

MCB : Modified CodeBook

MCBTF : Modified Cumulative Brightness Transfer Function

MHI : Motion History Image

MOG : Mixture of Gaussians

PCA : Principal Component Analysis

PCC : Percentage of Correct Classification

PDF : Probability Density Function

PR : Precision

QVGA : Quarter Video Graphics Array

RE : Recall

ROC : Receiver Operating Characteristic

SIFT : Scale Invariant Feature Transform

SIS : Sequential Importance Sampling

SMC : Sequential Monte Carlo

SURF : Speed Up Robust Features

SVM : Support Vector Machine

SWLH : Spatially Weighted LBP Histogram

TN : True Negative

TP : True Positive

TPR : True Positive Rate

TR : Tracking Rate

VF : Vertical Feature

WCC : Without Color Calibration

Chapter 1

Introduction

1.1 Context and Issues

This work has been done in the context of computer vision, with applications to video surveillance. According to Wikipedia [1], surveillance means "the monitoring of the behavior, activities, or other changing information". Basic video surveillance equipments are composed of television systems in which video signals are transmitted to a specific place, from one or more cameras to a set of monitors. These systems are called Closed Circuit TeleVision (CCTV). Generally, they are used for security purposes.
A human operator cannot actively monitor a large number of video cameras. After hours of concentration, the operator does not pay attention to everything that happens on the screens. Problems can also occur, especially in the context of wide-area surveillance, when unexpected events happen simultaneously in front of several cameras and the attention of the operator is focused on a single monitor. Based on these requirements, automated systems are in place, using computer vision algorithms.
In general, the term "video surveillance" raises ethical problems. Users are not disposed to accept a video surveillance system that can, according to some opinions, affect their privacy. In the mean time, these systems can be very useful in specific cases: they can raise alarms in case an unexpected event occurs or, after the occurrence of an abnormal event, identify its causes.
In the context of a posteriori (off-line) finding of unexpected event causes, the purpose of this thesis is to propose solutions for non redundant storage of information about objects that pass in front of cameras in a wide area video surveillance system (figure 1.1).

[1] http://en.wikipedia.org/wiki/Surveillance

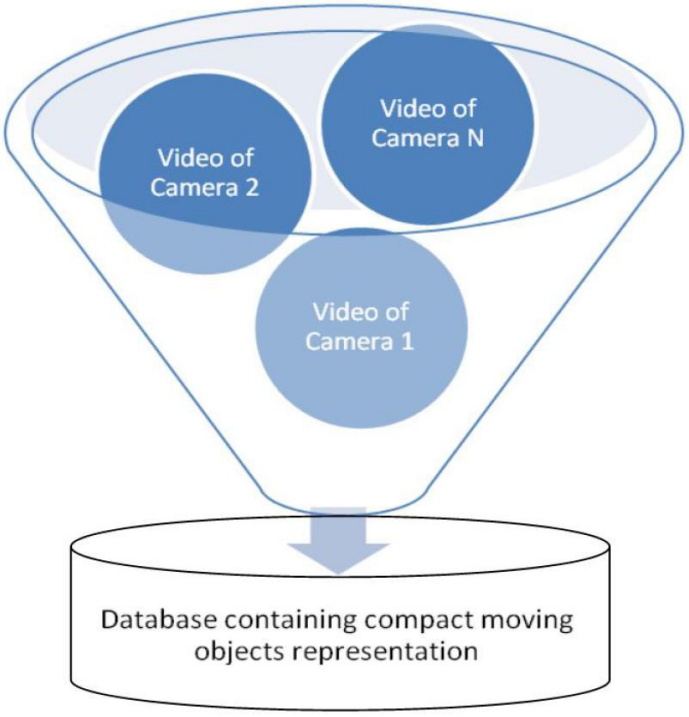

Figure 1.1: Layout of the central database containing moving object informations.

We consider an environment composed of several fixed cameras with non-overlapping field of view. We suppose that the position of cameras can be modified or new cameras can be added at any time. Therefore our system will not use any geometric information about the camera network. Automatic discovery of relative cameras positions requires a learning step that has to be rebuilt after every (dis)placement of any camera in the system. Network architecture will therefore be centralized. Video data is transmitted to a central server which will process and store in the database information about moving objects in a compact form. With these assumptions, several proposals were made, in various stages of video data processing (figure 1.2).

In order to perform moving object detection, representation and tracking, we would like to re-use and to improve the performance of object tracking using a single camera. It is difficult to get better object tracking results in a multi camera environment without optimizing the performance for a single camera. The issues we address are the following:

Object Detection: Image segmentation into objects and background is the first but most important step for object tracking. Objects detection from background is affected by: local and global luminosity variation, object's color similarity with background, poor video quality, moving background like tree leafs, object's shadow, etc.

Object Appearance: Object appearance may be very different depending on the view-

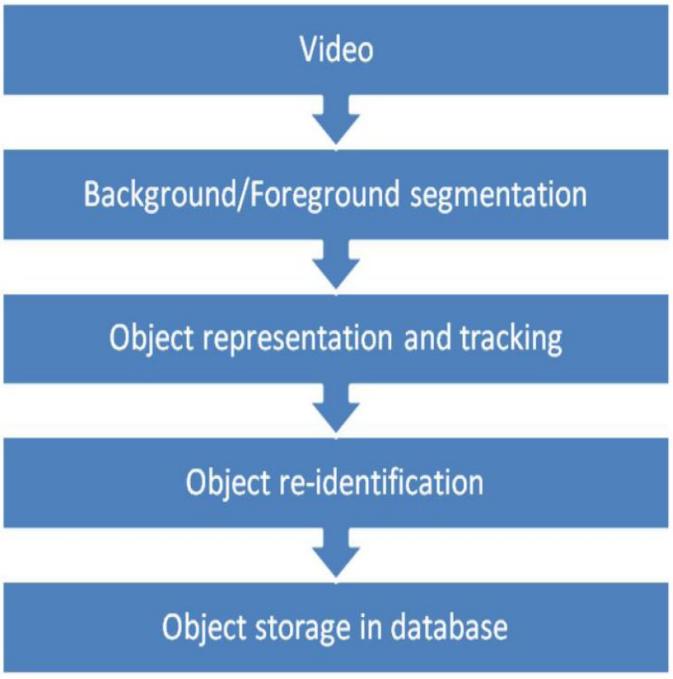

Figure 1.2: Video data processing steps in automatic video surveillance systems.

ing conditions. The most important cause is object shape variation due to the individual motions of body parts. Similarly, the object's view angle and its distances to the camera is a common reason of false object matching. An object appearance is very different when observed from its front, back and side views. Similarly, the object's appearance close to a camera might be significantly different from the one obtained at a greater distance. Occlusion can also cause problems because of hiding certain parts of the object that will change its appearance.

Object Re-identification: when an object appears in the field of view of a camera, the system should decide whether this object has already been observed or if it is a new object. To achieve this, the representation of objects should be invariant to position and distance from camera. Object's apparent size and view angles might be very different from one camera to another.

Multi Camera Environment: Multi-camera environments give benefit of monitoring large areas. But it also increases challenges of object tracking due to object color appearance which may be very different in a multi-camera environment due either to different camera models or to the type of installation (indoor, outdoor or mixed environment).

Real Time Performance: In many object tracking applications, it is required to track the object's position in real time. The object tracking is a complex task, consisting of many basic units like object detection, object recognition, object tracking, occlusion detection, objects data summarization, etc. Complex and computationally extensive algorithms for

each step make it difficult to achieve real time performance. Simple algorithms are unable to give satisfactory performance. There is always a compromise between real time performance, system cost and object tracking precision.

1.2 Thesis Objective

In the presented context, the first objective is to improve the solutions related to non-rigid objects detection and tracking for single camera systems. We need a better background representation allowing the detection of moving objects, and robust and compact descriptors for object tracking. We consider developing a system having the ability to track the objects under variation of brightness, object size, object view angle in camera's Field of View (FOV). Similarly, we should be able to recognize an object which exits and then re-enters in the camera's FOV.
The second objective of the thesis is to extend single camera algorithms to a multi-camera environment, in order to make object tracking more effective for large area automatic visual surveillance. The algorithm for multi-camera systems should have the ability to re-identify objects when they exit from one camera's FOV and re-enters in the FOV of the same or of another camera possibly different illumination conditions. Therefore an inter-camera color calibration will be needed.

1.3 Principal Contributions and Organization of Thesis

The main contributions of our work are the following:

1. Robust foreground-background segmentation algorithm under challenging situations like the variation of light intensities, small and large number of objects in a scene, object stopping its motion, minimizing the object shadows problem.

2. 1-D appearance model called Vertical Feature (VF), which represents a compact descriptor for moving object tracking in videos. It is independent of the view angle and of the apparent size of objects

3. Robust object tracking and re-identification algorithm.

4. Multi-camera color calibration method improving object re-identification in a multi-camera environment.

The thesis is organized as follows:
Chapter 2: presents a state of the art of object detection in videos, object tracking,

multi-camera object re-identification and inter-camera color calibration. Sometimes object detection is also referred as foreground-background segmentation. Poor segmentation results lead to false object matching hence the need to find the best performing algorithms. In our research work, we concentrate on human tracking. We discuss the existing techniques, dividing them into main classes based on their inherent properties. Object tracking in multi-camera is a challenging task and object appearance may be very different in multi-camera environments. Therefore we discuss the existing techniques of inter-camera color calibration to increase the object re-identification performance in multi-camera environment.

Chapter 3: In this chapter, we propose some modifications in original codebook method of foreground-background segmentation. We called the proposed method Modified CodeBook (MCB). We compare our algorithm's performance with Mixture of Gaussians (MOG) and original CodeBook (CB) method. We also discuss existing foreground-background segmentation evaluation techniques and we suggest a method for segmentation evaluation. Results shows that the proposed modification of the original codebook improves foreground-background detection performance.

Chapter 4: This chapter deals with object tracking and re-identification for single camera systems. We present a 1-D appearance model for object recognition, called vertical feature (VF). We combine VF with object motion parameters for object tracking. We also present an object-object occlusion detection method. We compare our algorithm with motion based and appearance based models. The results show the superiority of the combined approach.

Chapter 5: In this chapter, we discuss the importance of camera's color calibration in multi camera environments. 1-D appearance model uses object spatio-color information. If object color appearance is very different in multiple cameras then it becomes the reason for false object recognition. We discuss the camera color calibration methods for overlapping and non-overlapping FOV cameras environments. We explain our method of multi-camera color calibration in the case of non-overlapping camera environments. This technique uses cumulative histogram matching in order to calculate the Brightness Transfer Function (BTF). This function is used to project one camera's color information to another camera to minimize the color variation between cameras. We also compare the color calibration techniques for overlapping and non-overlapping FOV camera environments.

Chapter 6: In this chapter, we combine the algorithms proposed in chapters 3, 4 and 5 to develop a multi-camera object tracking system. We discuss the object re-identification performance in non-overlapping multi-camera environments. The results in this chap-

ters show that the proposed object tracking algorithm has appreciable performances in multi-camera environments. We compare the object tracking performances in multi-camera environments with and without calibration of camera's colors. The results show that object re-identification performance is significantly improved with color calibration. The proposed color calibration algorithm presented in chapter 5, outperform existing techniques.

Chapter 7: The final chapter of this thesis, concludes our research contribution and shows the advantages and the limitations of each step of our tracking system. We present some future works and discuss some of the possible applications of this research work.

Contents

Chapter 2

State of the Art in Video Analysis

Analyzing image sequences to detect and determine temporal events is often known as video analysis. According to wikipedia [1], "video analysis is used in a wide range of domains including entertainment, health care, automotive, transport, home automation, safety and security. Motion detection in videos is one of the simpler forms where motion is detected with regard to a fixed background scene. More advanced functionalities include object tracking and object motion estimation". In our research work, we present the algorithms of object detection, tracking and motion estimation in a single camera and multi-camera environments.

Object detection and tracking is used in video surveillance systems to monitor objects activities. The figure 2.1, shows the general layout of the video surveillance system. Cameras are installed in various regions for monitoring the objects activities. The videos are stored in a database. These camera's videos are displayed on CCTVs. To help human operator automatic visual surveillance systems are also integrated. These systems can analyze and summarize the object activities (e.g object position) in camera's videos. In the first part of the automatic system, objects are extracted from background. The object recognition and tracking algorithms are applied to get the object's positions in the cameras. These algorithms help the system to re-identify an object if it re-enters in the same or another camera's field of view (FOV). The video summarization methods can also extract objects trajectories. If an object/human is doing some unauthorized activity. For example, if an object enters in a restricted area then the system generates an alarm for the human operator. Similarly, human operator can also generate a query for some specific object's activity or can check the activity summary of some specific day (for example weekend).

[1] http://en.wikipedia.org/wiki/VideoContentAnalysis (access date: 18th April, 2011)

In this chapter, we present the existing methods used in video surveillance systems.

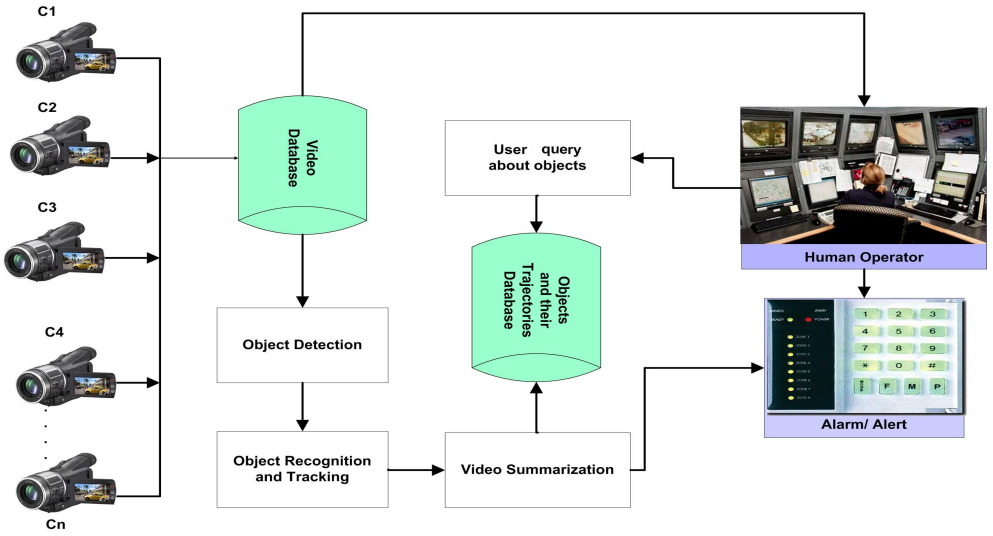

Figure 2.1: Video surveillance system general layout.

The performance of a video surveillance system depends on following parts: object detection, object recognition, classification, tracking, re-identification and color calibration in multi-camera environment. In this chapter, we provide a review to understanding next chapters. Interested readers may refer to additional references for further reading. We discuss existing techniques of object detection in section 2.1. We present existing algorithms of object recognition and tracking in a single and multi camera environment in section 2.2. We discuss the state of the art of object re-identification and color calibration in multi camera environment in the section 2.3 and 2.4 respectively.

2.1 Object Detection in Videos

Object detection is the first and most important step of moving object tracking. These techniques can be divided into several categories. Our goal of segmentation is to isolate moving objects from stationary and non stationary background. After isolating the moving objects from background, we perform moving objects tracking in a single or multi-camera environments.

Moving objects extraction from background is an important task. Results of object extraction depend upon the variation of local or global light intensities, objects shadow, background and foreground regular or irregular movement. [Wang and Suter, 2007] describes, that a good object detection technique should have the following properties :

- accurate in shape detection (i.e., the model should be able to ignore shadow, highlight, etc.);

- reliable in different light conditions (such as a light switched on/off, gradual illumination changes) and to the movement of background objects (e.g., if a background object is moved, that object should not be labeled as a foreground object);
- flexible to different scenarios (including both indoor and outdoor scenes);
- robust to different models of the background (i.e., a time series of observation at a background pixel can be either uni-modal or multi-modal distributed) and robust in the training stage even if foreground objects exist in all training examples;
- accurate despite camouflage (e.g., if a foreground object has a similar color to the background) and foreground aperture (if a homogeneously colored object moves, many of the interior pixels of the object may not be detected as moving);
- efficient in computation.

We classify object detection techniques into three major categories: without background modeling, with background modeling and combined approach.

2.1.1 Object Detection Without Background Modeling

This section describes the simplest and fundamental approaches. These techniques were frequently used due to their simplicity and computational efficiency. Image thresholding, temporal gradient and spatio-temporal gradient are commonly used techniques from this class. This segmentation class use only current and previous video frames. These techniques can isolate moving objects from the stationary background only. The 1st group, assumes object's colors are different from the background. Object can be extracted from the background by using image color thresholding techniques. 2nd group of these techniques assumes that background is stationary and objects are moving in the scene (temporal gradient and optical flow). Some researchers combine both groups to get better results. In this section, we will discuss some existing methods of image histogram thresholding, entropy, temporal gradient and optical flow based techniques to isolate foreground from background.

Object Detection Using Image Thresholding: Image thresholding is the simplest object detection method. In this technique, it is assumed that objects and background have different colors. [Ritter and Wilson, 2000] explain image thresholding method to classify pixel as object or background. They said, each pixel can be classified as an object or background pixel. If a pixel color value is within a given threshold color value then assign the binary value 1 to it else consider it as a background pixel and assign value 0 to

it. Instead of a global threshold value, adoptive, local or dynamic thresholding are frequently used (see [Shapiro and Stockman, 2002]). Histogram of image helps to find the thresholding value to separate object from the background. The normalized histogram is the approximation of probability density function for a certain gray/color level value to occur [Petrou and Bosdogianni, 2010]. They discuss single, multi, global and optimal image thresholding techniques using image histograms.

The [Ridler and Calvard, 1978] algorithm uses an iterative clustering approach. An initial threshold value is estimated by mean image intensity. Pixels above and below the threshold are assigned to the white and black labels respectively. The threshold is iteratively re-estimated from the mean of the two classes means.

[Otsu, 1979] and [Tsai, 1985] algorithms are based on discriminant analysis and use the zeroth and the first order cumulative moments of the histogram for calculating the thresholding value. The [Rosin, 2001] algorithm fits a straight line from the peak of the intensity histogram to the last non-empty bin. The point of maximum deviation between the line and the histogram curve will usually be located at a corner which is selected as the threshold value.

In most of real scenarios, objects and background are sharing many common colors, which make it difficult to select optimal threshold value to isolate object from background. [Su and Amer, 2006] focused on two types of thresholding categories (estimation of the image regions scatter changes due to color spatial location), and proposed a non-parametric algorithm to calculate the global threshold. This method is slower than traditional approaches (Poisson, Euler) but improves object detection. The negative aspect is that a single threshold is calculated for the entire image. Image thresholding is a useful method for objects detection, when objects and background's color are very different.

Temporal Gradient: Temporal gradient or frame differencing is a one of the simplest and basic method to detect moving objects from stationary background. In this method, difference of two consecutive frames are taken. Only those pixels which have significant movement (above than a defined threshold) are marked as object pixels and remaining pixels are classified as background pixels.

[Leung and Yang, 1987] assume that the objects are moving continuously because objects can not be detected if they stop in image scene. More detail on temporal gradient is explained by [Jain and Nagel, 1979]. [Foresti et al., 2005] use derivative model technique for motion detection in multi camera environment. They use both static and moving cameras in surveillance system. Adaptive threshold technique is used to isolate objects from background. [Rosin and Ioannidis, 2003] compare different thresholding

techniques for temporal change detection. They also propose some evaluation method for object detection and thresholding. Figure 2.2 shows object detection results, which they use for comparison of different techniques. They find in their experiments that the technique proposed by [Rosin, 2001] gives better performance than other techniques.

[Verbeke and Vincent, 2007] accumulate last ten frames and use a frame differenc-

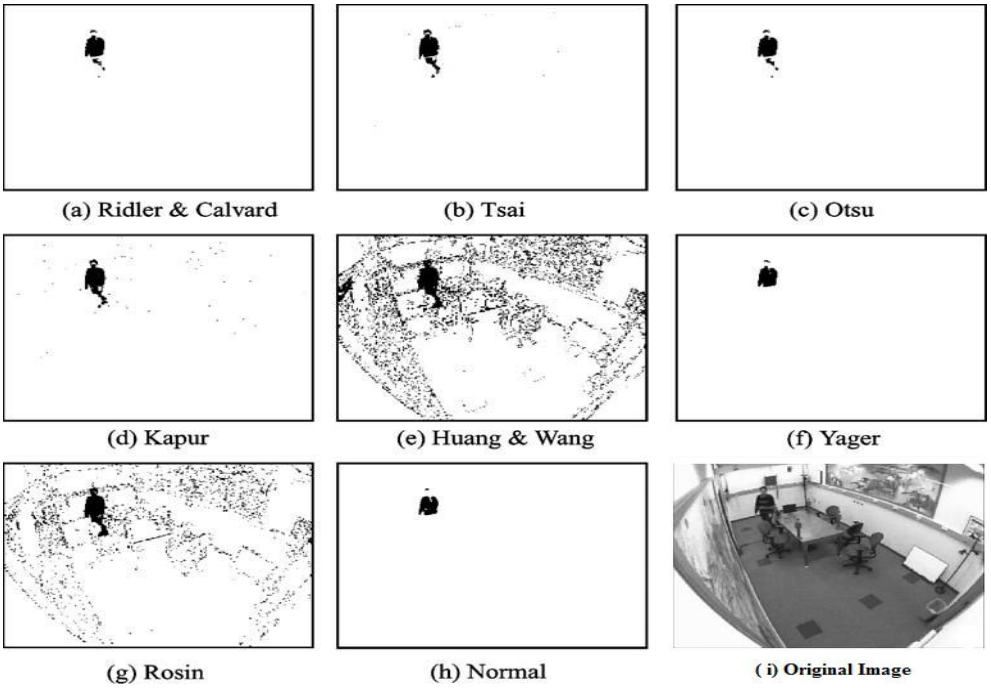

Figure 2.2: Results of thresholding the difference image from frame 218 of the intelligent room sequence with various algorithms [Rosin and Ioannidis, 2003]

ing technique to find the region where the motion has taken places. They use Principal Component Analysis (PCA) technique to reduce the data dimensions. Their technique is better than simple frame difference techniques, as they accumulate previous ten frames for calculating motion region in the image. But the method fails if the object stops its motion. It is also sensitive to changes in light intensity, shadow, sensor noise. Like other frame differencing techniques, it also fails to isolate moving background from foreground pixels. [Bradski and Davis, 2002] present an algorithm for object motion detection and measure the object motion in a scene using timed Motion History Image (tMHI). This representation can be used to determine the object current pose and measure the motions induced by the object in a video scene. The MHI generates a 2D template image for each action. The MHI approach relies on template matching and thus can detect occurrences of a previously learned action.

Spatio-Temporal Gradient: Spatio-temporal gradient is a method to detect moving objects from stationary background. This method use spatial gradient of the current image

and temporal gradient of the current and previous frames. The objects motion in the image plane is called optical flow. [Horn, 1986] defined the optical flow as the apparent motion of the brightness pattern in a spatial domain.

There are two articles [Barron et al., 1994] and [Liu et al., 1998] which evaluate performance of many optical flow techniques including two state of the art techniques [Horn et al., 1981] and [Lucas and Kanade, 1981]. [Barron et al., 1994] discuss nine optical flow algorithms and compare them according to their precision but they do not calculate the complexity of algorithms. [Liu et al., 1998] fills this gap by measuring the precision and time calculation of these optical flow algorithms. Whatever method from this class is chosen, the calculation of optic flow is computationally extensive. Figure 2.3 shows the accuracy of different optical flow techniques and their execution time. This graph is useful for selecting the optimal optical flow technique according to the need of precision and real time performance. Figure 2.3 also shows that each of the optical flow technique is computational extensive but the algorithms published in [Liu et al., 1995] and [Camus, 1995] are reasonably fast and have good precision.

Optical flow based motion segmentation has some benefits. It can detect discontinuities

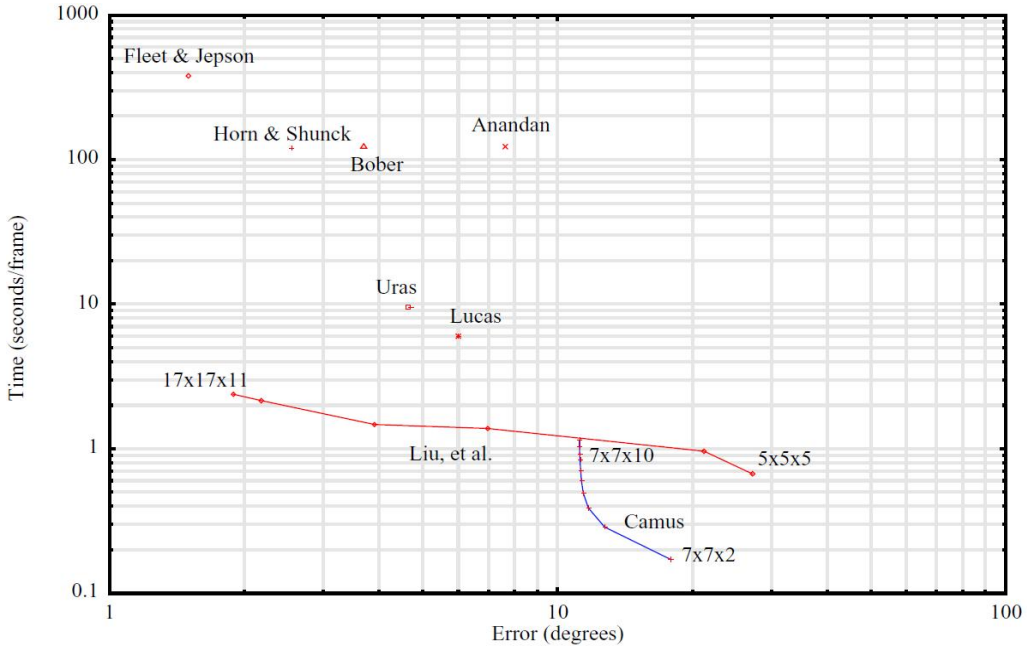

Figure 2.3: 2-D performance diagram for various optical flow algorithms [Liu et al., 1998]

in the optical flow which helps in segmenting images into regions that correspond to different objects [Horn and Rhunck, 1993]. The optical flow has a good performance for rigid object detection like cars, air-plane, etc., due to their uniform optical flow. Optical flow can give non-uniform optical flow for non-rigid objects due to individual

movement of body parts [Mori et al., 1994]. For example, the detection of a walking person, where one leg is moving in the walking direction, and the other leg is still at the same position produce non-homogeneous optical flow. These conditions make it non-practical for applying optical flow techniques for non-rigid objects [Broggi et al., 2000]. In addition, the non-rigid movement of pedestrians can cause noisy results, as optical flow algorithms tend to fails in regions where there are multiple motions, occlusion, and non-rigidly moving areas [Niyogi and Adelson, 1994].
The other method which use spatio-temporal gradient to detect objects is image entropy method. Entropy is a disorder measurement associated to a system. In our case, pixel intensity variation compared to its neighborhood pixels during certain period of time is called entropy. [Ma and Zhang, 2001] propose a method based on space-time histograms to calculate entropy. The moving areas are those where spatio-temporal entropy of the sequence reaches a maximum value. Unlike foregoing techniques, temporal dimension is used by a local analysis algorithm through the 2D+T video volume. The [Kapur et al., 1985] algorithm uses the entropy of the image. It considers the thresholding image as two classes of events with each class characterized by a Probability Density Function (PDF). The method then maximizes the sum of the entropy of the two PDF to converge to a single threshold value. [Parker, 1996] implements entropy of the intensity histogram using two fuzzy logic definitions described by [Huang and Wang, 1995] and [Yager, 1979]. Entropy method works better in the situation when object and background consist of uniform colors. If object as well as background have many colors, then image segmentation into foreground and background performance using image entropy technique becomes poor.

2.1.2 Segmentation Using Background Modeling

Background modeling techniques have the ability to segment image into objects and background in a challenging situation like moving background, luminosity variation and also in the condition when an object stops for some frames. Actually, background modeling techniques use image pixel history to model the background. Probabilistic and statistical models are frequently used for this purpose.
In general, background modeling approaches assume that there is a static camera and that image features, such as color intensity or objects edge gradient information differ from the background. In addition, an assumption is often made that illumination condition variations are small and gradual. These techniques generally model the background

with respect to relevant image features. Foreground pixels can then be determined if the corresponding features from an input image significantly differs from those of the background model.

Background Modeling: This class of methods model the background using previous frames history. Every image pixel is matched with its background model. If pixel color value is similar to the background model, then it is considered as a background pixel otherwise it is an object pixel. The probability density of the feature can be described using a parametric representation (single Gaussian distribution). In general, single Gaussian distribution is not sufficient for the background modeling due to the background movement, which might be regular or random. To overcome this problem semi-parametric (Mixture of Gaussians) or non-parametric (kernel density) distributions are used in practical background modeling techniques. Kernel density distributions are more flexible than Mixture of Gaussians (MOG) but require large amount of memory to implement them and are computationally extensive.

The most frequently used technique is the Mixture of Gaussians proposed by [Stauffer et al., 2000]. They avoid the computation complexity by using the same variance for (R, G, B) color channels. The method gives good results but it suffers from the shadow problem and it is sensitive to the variation of light intensities. The approach proposed by [Kaewtrakulpong and Bowden, 2001] is strongly inspired from [Stauffer et al., 2000]. However, the authors considered that [Stauffer et al., 2000] suffers from slow learning at the beginning, especially in busy environments. By re-investigating the update mechanism, the authors propose different equations at different phases. [Thome and Miguet, 2005] use MOG and combine it with shadow removal technique found in [Salvador et al., 2001]. Their results are better than the results claimed in [Stauffer et al., 2000]. [Dickinson et al., 2003] model the background by an adaptive MOG in color and space and they claim better results than traditional MOG. Despite of MOG popularity, there are a number of well documented limitations to the per-pixel MOG model. Variations which are reoccurring in a scattered and irregular or where one mode dominates, are still not well represented.

[Elgammal et al., 2000] present a kernel-based density estimation method and showed it was effective in handling situations where the background contains small and repetitive motions such as tree branches and bushes. Since the cost to compute the kernel density at each pixel is very high, several pre-calculated lookup tables are used to reduce the computational burden of the algorithm. Moreover, because the kernel bandwidth is estimated by using the median absolute deviation over samples of consecutive intensity values at the pixel, the bandwidth estimation may be inaccurate if the distribution of the

background samples is multi-modal. Median value is not a true representative of samples having two or more distributions. Similarly [Han et al., 2008] model the background by a sequential density approximation and each time step, densities are estimated and Gaussian component is assigned to each model. The covariance of each component is derived from the Hessian matrix estimated at the mode location. To detect the modes they employ the variable-bandwidth mean shift. However this method is computationally very extensive and it is not suitable for real time computer vision applications.
[Pic et al., 2004] use an adaptive technique for the estimation of the background on the base of learning. The learning rate is calculated after every frame for each pixel. The algorithm procedure for calculating learning rate, make it computation expensive. It fails to provide good results in the presence of fast changes in foreground and background. That is why it is more sensitive to the variation of light intensities. [Gordon et al., 1999] estimate the background by combining the information of color and range/depth using stereo cameras. They show the superiority of combined approach. They claim that classical problems of object shadows detection as a foreground might be minimized. Because, object position and its shadow position are at different location in image scene. However they assume that background is static or its variation is small. They only report testing results of segmentation in indoor environments.
[Kim et al., 2005] perform foreground-background segmentation by using the codebook method. This technique shows good object detection performance and it is also more robust to the problem of shadow and light intensities variation. Figure 2.4 shows objects detection results. It is evident from the figure 2.4 that codebook method proposed by [Kim et al., 2005] shows better segmentation performance than other discussed famous techniques like MOG [Stauffer et al., 2000] and Kernel [Elgammal et al., 2000]. The codebook can tackle the conventional problem of image shadow and it can also take into account the motion of background. But there are some limitations of the codebook method, which we will discuss in detail in chapter 3.

Background Estimation: These techniques use prediction methods to estimate the background. Background pixel's intensity value is estimated using its past history of pixel color value. If current pixel value is similar to the background estimation value then it is considered a background pixel else it is an object pixel. Background pixel estimation for each pixel is updated after every image frame.
The VuMeter method proposed by [Goyat et al., 2006] is a non-parametric model. It is a probabilistic approach to define the image background model using estimation of probability distribution function. A pixel can have two states, background or foreground pixel. Each pixel background is updated using a fixed learning rate.

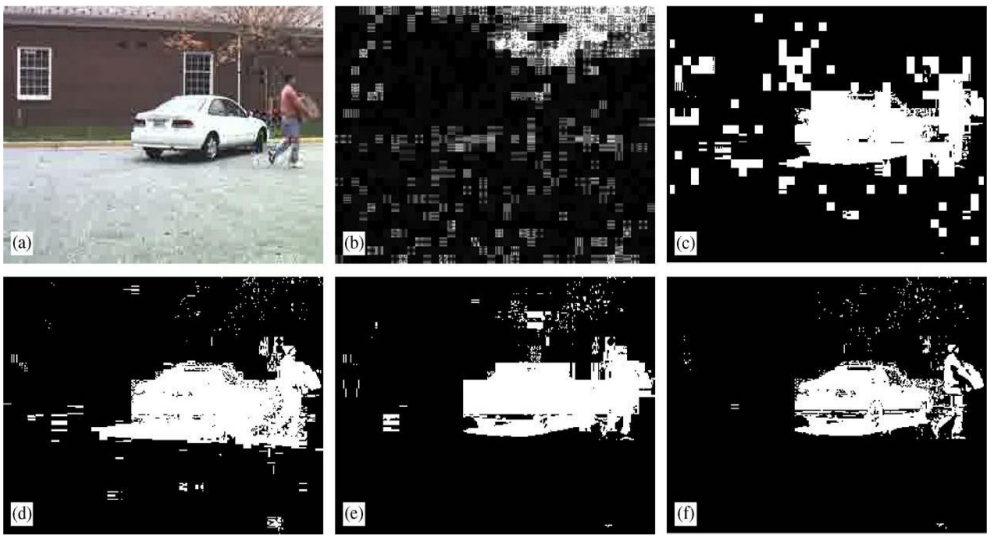

Figure 2.4: Detection results on a compressed video [Kim et al., 2005]: (a) original image, (b) standard deviations, (c) unimodal model in [Horprasert et al., 1999], (d) MOG [Stauffer et al., 2000], (e) Kernel [Elgammal et al., 2000], (f) CB [Kim et al., 2005]

Adaptive filters like Kalman are also used for background estimation and modeling. [Ridder et al., 1995] model each pixel by using a Kalman filter. This method addresses many dynamic background segmentation problems. However, they do not take advantage of inter-pixel correlation and global appearance. Thus, they may fail to extract objects when the color distributions of the foreground and background are similar. [Zhong and Sclaroff, 2003] propose an algorithm that explicitly models the dynamic, textured background via a robust Kalman filter algorithm, which is used for estimating the intrinsic appearance of the dynamic texture. The foreground object regions are then obtained by thresholding the weighting function used in the robust Kalman filter. [Doretto et al., 2003] find that Auto-Regressive Moving Average (ARMA) model proposed by [Soatto et al., 2001], is a first-order linear model but it can capture many dynamic textures.

Background Subtraction: This technique generates the background image using running averaging process of current and n previous frames. The background learning rate might be fixed or adaptively calculated. This background image is subtracted from current image and all the pixels above some threshold value are considered as object pixels. [Horprasert et al., 1999] proposed an algorithms for background subtraction using current and previous frames. They proposed color model which separates the brightness from the chromatic component to remove object shadows. [Lo and Velastin, 2001] and [Cucchiara et al., 2003] used average and median pixel value from current and previous images to develop a background image. Background subtraction method is computationally fast but require more memory to store n previous frames [Piccardi, 2004]. Back-

ground subtraction techniques use one reference image as a background image. These techniques are unable to subtract non-stationary background, e.g movement of tree.
In general, background modeling techniques improves the foreground-background segmentation performance significantly in almost every challenging environment. They have better performance in out-door and indoor environment. Similarly, they have better performance for modeling background movement as they use many model to model the background. This becomes the reason of superiority of background modeling on background subtraction and estimation. But they are still unable to completely remove shadow. The stationary objects are absorbed in the background if they stay for some seconds and the background modeling technique adopts stationary object color in background. Sudden variation of light intensity make background model unstable. Method proposed by [Kim et al., 2005] has better performance in these situations and other possible approach to overcome above discuss problems is using combined approach.

2.1.3 Combined Approach

This object detection class combine the background modeling techniques with other algorithms. Combining spatial or temporal gradient with background modeling technique is the famous example of this class. [Tian et al., 2005] claims that foreground objects are absorbed at different rates at different pixels, causing object fragmentation. Fragmentation problems also arise where foreground objects overlap spatially with background objects of similar color. These types of errors are unavoidable under the assumption of an independent pixel model. Scene images are generated by a set of discrete objects (both background and foreground) such that pixel values generated by the same object exhibit a strong spatial, chromatic, and temporal coherence. Such relationships are not represented by a per-pixel model, but can be used to address the above classification problems, and to produce a more robust segmentation in general.
[Cong et al., 2009] detect the moving objects by combining the MOG background modeling technique with successive frames temporal gradient. [Izadi and Saeedi, 2008] combine spatial gradient with MOG to detect objects. They isolate objects from background and also remove shadow by using filtering and morphological operations. Figure 2.5 illustrates the steps of combining MOG models with intensity gradient. It is evident from the figure that combing the image intensity gradient with background modeling techniques improve its segmentation performance and also helps to remove object shadows. [Izadi and Saeedi, 2008] show good results in different conditions but it is computationally expensive due to combing many steps (see figure 2.5) to get final foreground-

background image. They use several filters (morphological, median, etc) which may create artifacts or remove object parts during filtering and morphological closing operation.

[Javed et al., 2002] also use MOG and intensity gradient to remove shadows and com-

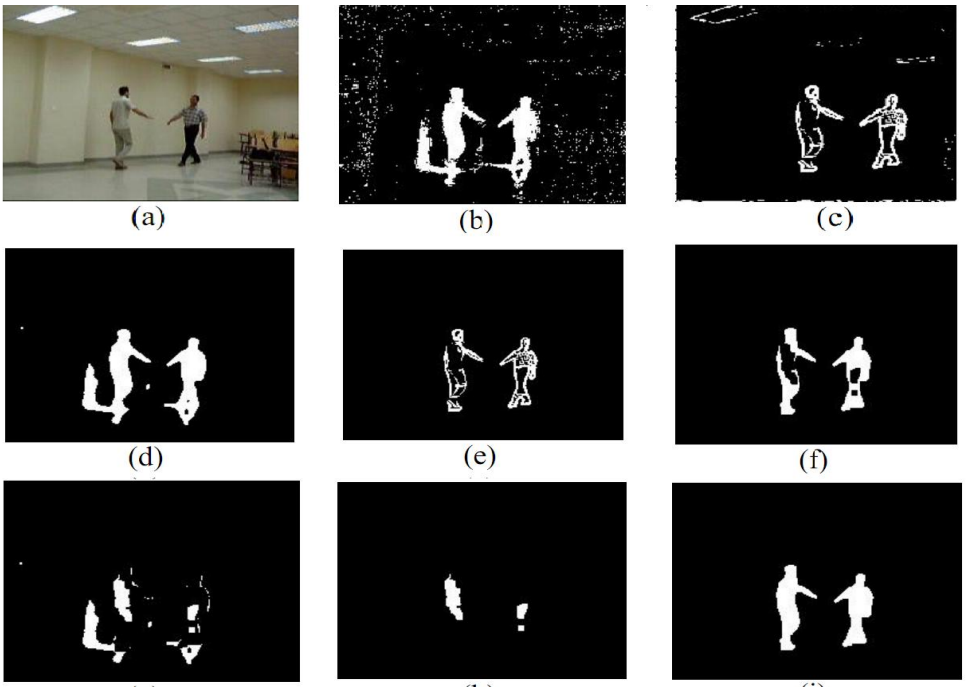

Figure 2.5: The steps (a)-(i) of combining MOG models with intensity gradient ([Izadi and Saeedi, 2008]); (a) a frame of a sequence, (b) segmentation using MOG, (c) intensity Gradient mask, (d) filtration applied to image (b), (e) filtration applied to image (c), (f) morphological close applied to image(e), (g) subtraction of image (f) from image (b), (h) non-shadow regions, (i) resulting image

pensate the variation of light intensities. [Heikkilä and Pietikäinen, 2006] model each pixel by a group of Local Binary Pattern (LBP) histograms computed over a circular region around the pixel. This method deals well with dynamic background as it gathers information over a region rather than a single pixel. However, it has a drawback of inaccuracy of the shape information of the segmentation results due to region based segmentation. Another problem of this method is its slow background learning rate. [Tian and Men, 2009] modify the method of [Heikkilä and Pietikäinen, 2006] by introducing a Spatially Weighted LBP Histogram (SWLH) as a feature vector. SWLH improves shape information accuracy better than [Heikkilä and Pietikäinen, 2006].

[Li et al., 2004] proposed a Bayesian framework that incorporates spectral, spatial, and temporal features to characterize the background appearance at each pixel. They claim that their method can handle both static and dynamic backgrounds. Good performance was obtained on image sequences containing objects in a variety of environments, like

offices, public buildings, subway stations, campuses, parking lots, airports, and sidewalks. Their algorithm's performance decreases significantly if foreground objects are constantly presented in the scenes. Like all other fields, combining multiple algorithms improve the performance of the object detection but make object detection more computational expensive.

2.1.4 Evaluation of Segmentation Algorithms

In this section, we discuss some existing methods, which are used for measuring the quality of foreground-background segmentation algorithms in a more quantitative approach. Segmentation evaluation techniques require ground truth (ideal segmented object). These ground truth images are compared with segmented images from segmentation algorithms to get the segmentation performance. Some interesting methods on the segmentation techniques evaluation based on Receiver Operating Characteristic (ROC) curve are explained in [Chalidabhongse et al., 2003], [Rosin and Ioannidis, 2003], [Davis and Goadrich, 2006] and [Wang et al., 2005].
When comparing segmentation algorithms, ROC analysis is often employed when there are known background and foreground (object) distributions [Gao et al., 2000]. Their ROC curves measure the sensitivity for detecting a particular foreground against a particular background. Their algorithm require considerable experimentation and ground truth evaluation to obtain accurate False Alarm (FA) rates and the Miss Detection (MD) rates.
[Rosin and Ioannidis, 2003] use three parameters: the Percentage of Correct Classification (PCC), Jaccard Coefficient (JC) and Yule coefficient [Sneath and Sokal, 1973] for analyzing image segmentation quality. Percentage correct classification (PCC) coefficient tends to give misleading results when the amount of change is small compared to the overall image. In most of situations, total number of pixels occupied by objects are smaller than total number of image pixels. PCC evaluation technique produce very similar quantitative number due to its method of using true positive, false positive, false negative and true negative. We discuss this issue in section 3.4 of the chapter 3. Jaccard Coefficient (JC) and Yule coefficient [Sneath and Sokal, 1973] give better performance. Jaccard Coefficient and Yule coefficient can discriminate segmentation techniques better because they do not include true negative (TN) in their coefficients (see section 3.4).
Precision (PR) and Recall (RE) are most commonly used parameter for evaluation of the experiments. [Davis and Goadrich, 2006] and [Wang et al., 2005] discuss PR and RE in detail and show the results between the precision and recall using ROC curves. The

performance of foreground-background segmentation can also be evaluated by using the quality factor. The quality factor is the harmonic mean of segmentation precision and recall. Harmonic mean is true representative of those qualities whose are derived from the fractions of quantities. [Davis and Goadrich, 2006] claim that dealing with highly skewed datasets, precision-recall curves give a more informative picture of an algorithm's performance.

All of the above discussed techniques, use ground truth to calculate the ROC curves. This makes large scale evaluation impractical on real life data. Some researches use synthetic videos to evaluate the segmentation algorithms. But the problem is that the synthetic data will probably not faithfully represent the full range of real data. One of the good synthetic video data benchmark with ground truth is available here [Dhome et al., 2010a]. This video data consists of eight video sequences having different challenging environments. These situations includes the light intensity variations, moving and stationary background, random noise, large and small number of vehicles, moving person, etc. The big advantage is that ground truth data is also available with these videos. In figure 2.6, segmentation results of six algorithms are shown. These six algorithm are [Stauffer et al., 2000], [Kaewtrakulpong and Bowden, 2001], [Tuzel et al., 2005], [Chen et al., 2007], [Sigari and Fathy, 2008] and [Goyat et al., 2006]. [Dhome et al., 2010b] also discuss the evaluation methods of image segmentation techniques and best results are obtained by using VuMeter method [Goyat et al., 2006].

[Chalidabhongse et al., 2003] propose segmentation evaluation method, called per-

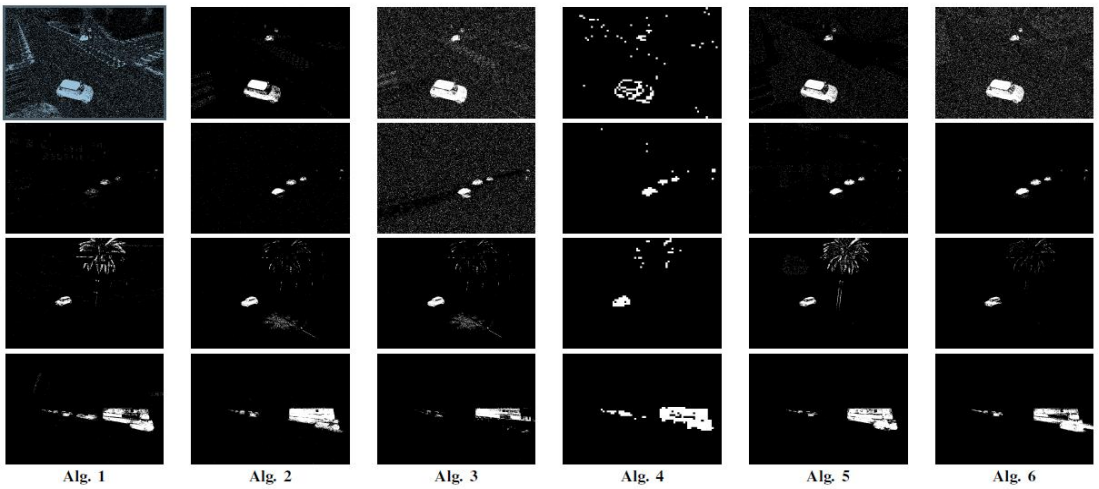

Figure 2.6: Some segmentations computed for the evaluated BSA, illustrating each video sequence [Dhome et al., 2010a]

turbation detection rate (PDR) analysis. This method can measure the sensitivity of a foreground detection algorithm without assuming any knowledge of the actual fore-

ground distribution. It measures the detection of a variable, small difference from the background, obtaining a foreground distribution by assuming that the foreground might have a distribution locally similar in form to the background, but shifted or perturbed. The detection is measured as a function of contrast, the magnitude of the shift or perturbation in uniform random directions in RGB color space. This technique determines the background modeling sensitivity but in real environments there are some other issues which also determine the image segmentation quality. These issues are: sometimes training period on an empty scene is not available. Objects density (few or many objects) also change the performance of many background modeling techniques. The performance of segmentation techniques also changes with periodic or non-periodic movement of background etc. These issues will be discussed in chapter 3.

2.2 Object Tracking

Object tracking is the process of locating an object in the image plane, where it moves around the scene. Although a huge work is done and sophisticated algorithms have been for many years developed, object tracking is still a non trivial problem due to these reasons:

- Loss of information due to 3D world projection into 2D images.
- Noise produced by image sensors and electronics.
- Missing fine details in surveillance videos due to low resolution cameras or high video compression.
- Complex nature of object motion and geometry.
- Nonrigid nature of objects like humans.
- Partial and full object occlusions with other objects and background.
- Changes in illumination conditions.
- Poor real-time processing performance of the most accurate algorithms due to their algorithmic complexities.
- Non homogeneous nature of object's colors in multi cameras.
- Possibility of poor object detection due to image segmentation problem.

- Object appearance may be very different in different view angles in multi camera environments.

There are many good surveys published on object tracking. They discuss several aspects of this research area. We briefly present the most interesting and relevant techniques in next paragraphs.

[Moeslund et al., 2006] present a rich survey report reviewing almost four hundreds published articles on object tracking problem and its rectification approaches in human motion capture including human model initialization, tracking, position estimation and activity recognition. [Moeslund et al., 2006] point out that general models are required to provide robustness for capturing a wide range of human movement. The progress in the object tracking still requires fundamental advances in behavior representation for dynamic scenes, viewpoint invariant relationships for movement and higher level reasoning for interpretation of actions.

[Yilmaz et al., 2006] discuss in details many existing techniques of object tracking in their survey report. They categorize the tracking methods on the basis of the object appearance, shape, color and their motion representations. Figure 2.7 shows some of the possible ways to represent the objects in images. These object features are used for object recognition. They also discuss the important issues related to object tracking, including best possible use of appropriate image features, selection of motion models, and detection of objects.

[Enzweiler and Gavrila, 2009] discuss some of well known object tracking techniques in first part of their survey. In the second part, they compare the performance of wavelet based Ada-Boost cascade [Viola et al., 2005], Histograms of Oriented Gradients with linear SVM [Dalal and Triggs, 2005], neural network using local receptive fields [Wöhler and Anlauf, 1999], and combined shape-texture detection [Gavrila and Munder, 2007]. They claim best results when they combined Histogram of Oriented Gradient (HOG) and linear SVM. This technique is not suitable for real-time multi-object tracking due to the complexity of HOG and large database of objects needs for training.

[Geronimo et al., 2010] propose to deal the different sections of human detection and tracking independently rather than collectively. They divide human tracking algorithms into preprocessing, foreground segmentation, object classification, verification/refinement, tracking and applications. They discuss the various algorithms from each class and explain their advantage and limitations.

There are many possible ways to classify object tracking algorithms into different categories. We prefer to divide them into three classes due to their inherent properties.

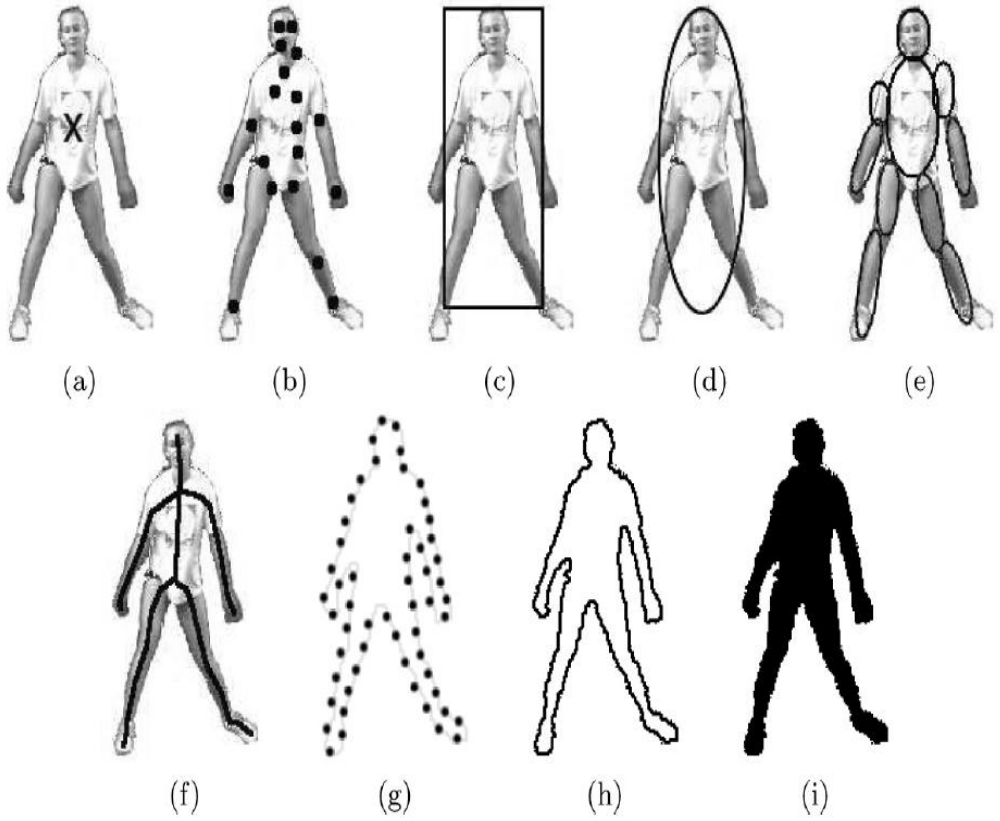

Figure 2.7: Object representations. (a) Centroid, (b) multiple points, (c) rectangular patch, (d) elliptical patch, (e) part-based multiple patches, (f) object skeleton, (g) complete object contour, (h) control points on object contour, (i) object silhouette. [Yilmaz et al., 2006]

These classes are based on object motion, geometrical and appearance models. Sometimes many features from different classes are combined to get better recognition results. In motion based object trackers, a segmented blob is associated to previous frame's object whose motion parameters like position, velocity and acceleration are similar to the previous frame's motion parameters. The Kalman filter, particle filter and optical flow are the most famous in this class. The geometrical models use shape features to identify and track objects. Edge detection, contour matching, moments, area, size, shape are popular object features which are used for object tracking. Appearance based models use object color informations as a key features. There are a large number of features and models which are used for object recognition and tracking. But 1-D and 2-D appearance models are commonly used for object recognition and tracking. Appearance based models are more popular due to their robustness for rigid and non-rigid object recognition. The previous frames objects models are matched with the current frames objects using their color appearance model.

2.2.1 Motion Models

Motion based object trackers use object's geometrical centroid. Object centroid is also known as object's center of gravity. It is the most invariant point of the object. Object centroid is calculated by taking the average of all the object pixel's coordinates (x and y axis). Small variation of object's shape do not affect object centroid. Motion models track the objects taking into account the following aspects:

- How objects move between frames.
- Temporal position and object appearance in successive frames.
- Moving objects velocities is constant or can change smoothly.

Object motion is modeled by using linear or sometimes non-linear filters. Each object is tracked by using motion model. Labels are assigned to objects constantly on the basis of their actual and estimated positions and velocity. If many objects are moving closely to each other, then the object having the closest actual and estimated motion features is considered as a match object.
[Elgammal and Davis, 2001] assume that frame rate of video sequence is high enough so that an object position does not change significantly between frames. Similarly [Cai et al., 1995] also assume that object appearance remains consistent between frames and the velocity of a pedestrian usually changes gradually when the pedestrian desires to stop or start walking [Yasutomi and Mori, 1994], [Cai et al., 1995], and [Xu and Hogg, 1997] suppose that the object has constant velocity. Some of these assumptions can be true regardless of specific scenario parameters, which include camera position and pedestrian flow density. However, others can be invalidated in unconstrained environments. For example, the constant velocity assumption is generally not true for a pedestrian. It may change direction and stop its motion abruptly and unpredictably [Elzein et al., 2003].
Motion-based objects trackers like Kalman filters [Kalman, 1960] are commonly used for their ability to predict the object's next frame position by using the object motion history. R.E. Kalman published his famous paper describing a recursive solution to the discrete-data linear filtering problem. The Kalman filter became popular in the area of autonomous, assisted navigation and tracking. The Kalman filter is the sets of mathematical equations that provides an efficient computational (recursive) prediction. It estimates the state of a process by minimizing the mean of the squared error and predicts the next state by using an estimation process. [Brown and Hwang, 1992] explain in detail about random signal, statistical processes, Kalman filtering and their application. Other good discussions about Kalman filtering and its application to object tracking is

discussed by [Welch and Bishop, 1995] and [Funk and Bishop, 2003] in their technical reports.

[Medeiros et al., 2008] use Kalman filter for distributed embedded wireless camera environments. Each camera estimate object position using a Kalman filter and sends object position to a central base station. [Lee and Ko, 2004] use Kalman filter for object tracking and they detect object-object occlusion using the motion model.

The Sequential Importance Sampling (SIS) algorithm is a Monte Carlo (MC) method that forms the basis for most sequential MC filters developed over the past decades. Sequential Monte Carlo (SMC) methods are also known as Particle filters [Doucet et al., 2000]. Particle filter implementation methodology for object tracking is explained in [Arulampalam et al., 2002]. [Deutscher et al., 2000] introduced the annealed particle filter which combines a deterministic annealing approach with stochastic sampling to reduce the number of samples required. At each time step the particle set is refined through a series of annealing cycles with decreasing temperature to approximate the local maxima in the fitness function. Particle filters [Isard and Blake, 1998] are very popular due to their ability to closely approximate complex real-world multi modal posterior densities using sets of weighted random samples. The key advantage of using particle filter for object tracking is its ability to track an object even if object motion is non-linear in nature and two or more objects are under occlusion.

The principal difficulty with human tracking with particle filters is the exponential growth of particles to correctly estimate objects trajectories [Czyz et al., 2007] and [Martinez-Del-Rincon et al., 2007]. That makes it non suitable for multi-objects tracking systems in real time applications.

[Mikic et al., 2003] present an integrated system for automated recovery of both a human body model and motion, from multiple views image sequences. Model acquisition is based on a hierarchical rule-based approach to body part localization and labeling. Prior knowledge of body part shapes, relative size, and configuration is used to segment the visual-hull. An extended Kalman filter is then used for human motion reconstruction between frames. A voxel labeling procedure is used to allow large inter-frame movements. [Luo and Bhandarkar, 2005] proposed a tracking framework which uses Kalman filter, where the elastic matching algorithm is used to measure the velocity field which is then approximated using B-spline surfaces. [Cheung et al., 2003] first reconstruct a model of the kinematic structure, shape, and appearance of a person and then use this to estimate the 3D movement. Tracking is performed by hierarchically matching the approximate body model to the visual-hull using color matching along the silhouette boundary edge. In general, simple motion features are not sufficient for object tracking

especially if moving objects are humans. Human motion may be random in different frames and motion filters are unable to track the objects during and after occlusion. It normally loses object if it exits from the camera's FOV and re-enters in the scene. Generally, motion features are combined with object geometrical or appearance features for object tracking.

2.2.2 Geometrical Models

Object's geometrical models and features are widely used for object recognition. Object features like moments, edge, skeleton, area, perimeter and object shape models like 2-D or 3-D are commonly used. Many object size, translation and rotation invariant algorithms make it possible to use geometrical properties for object matching and tracking.
Moment is the quantitative measure of shapes from the set of data points. Zero order moment (object area), first order (data mean or center of gravity) and second order (data variance) are used in most of statistical analysis. Objects shapes are matched by using their n moments. Many object size and rotational invariant moments are introduced to overcome object recognition problem. [Hu, 1962] introduces invariant moments which are linear combination of the central moments. Central moment are used to shift origin moments to the mean value. Combine different normalized central moments, create invariant functions to scale and rotation. [Dailianas et al., 1995] detect and track objects using a vector composed of the first three invariant moments of objects. They use Euclidean distance to match objects moments. Similarly [Kadyrov and Petrou, 2001] use six affine distortion invariant descriptors using object moments. These features are invariant to object size, rotation and translation under the assumption that objects are only affinely distorted. [Xu and H.Li, 2008] propose invariant moments in arbitrary dimensions, from 2D, 3D to nD. Observation using 2-D moment invariants has been successfully applied in vision application. [Holm, 1991] extracted closed boundary regions and proposed to represent them by their perimeter, area, compactness, moments, and moment invariants. Moment based methods are sensitive to distortions that affect the "object's center of gravity" like non-uniform illumination and change of object shape if object is non-rigid.
Object's shape, contours or curves are also used for objects recognition. Contours are normally found by using edge detection techniques. [Martelli, 1972], [Rosenfeld and Kak, 1976] and [Cederberg, 1979] find object's shape contour by ordering successive edge points. Boundary scan can also be viewed as a graph formed by linking the edge elements together [Martelli, 1972], [Ashkar and Modestino, 1978] and [Lester et al., 1978].

Sometimes, due to the low quality of surveillance videos, sensors noise and image segmentation problems, object edges are not properly detected.
Chain coding methodologies [Freeman, 1961] and [Sanchez-Cruz and Rodriguez-Dagnino, 2005] are also used to represent object shapes. Chain codes scan the boundary pixels of objects and encode its shape into eight orientations (0, 45, 90, 135, 180, 225, 270, 315). The encoded shape of current object is matched with previously stored objects shape. Object shape variation due to its motion is usually modeled by affine or projective transformation to minimize the variation of object shape. Geometric models are more suitable for representing rigid objects.
Geometrical models and features are unable to give satisfactory results for non-rigid objects due to the following reasons: low video quality, small object size, object shape variation in a video sequence and individual object's parts movement.

2.2.3 Appearance Models

Object appearance information is the most frequently used class for object recognition and tracking. Most of the recent articles published on non-rigid object tracking use appearance information as recognition feature. Object's color is the most important feature for object recognition. Note that shape representations can also be combined with the appearance representations for tracking ([Cootes et al., 2001]). An interesting work on pedestrian detection and tracking is discussed in more details in [Dollar et al., 2009] and [Geronimo et al., 2010]. There are many possible ways to use object appearance for its recognition. Some possible principal classes are probability density functions, 1-D or 2-D appearance models, invariant multi-points of objects, object's templates and multi-view appearance models.
The probability density functions of object appearance features (color, texture) can be computed from the image regions specified by the shape models (interior region of an ellipse or a contour). The probability density function of an object can be parametric, such as Gaussian [Zhu and Yuille, 1996] or a mixture of Gaussians [Parragios and Deriche, 2002]. Similarly, non-parametric kernel based [Elgammal et al., 2003] and histograms [Comaniciu et al., 2000] are also used.
Object histogram was frequently used in the past decade and still popular for object recognition due to its reasonable performance even with the change of object size and rotation. [Comaniciu et al., 2000] use the mean shift technique for object tracking. They use color histogram and the Bhattacharyya distance to find the best match object position. [Krumm et al., 2000] use RGB color channels and they quantize each channel

into four equal-length ranges, giving a 4 x 4 x 4 color cube and a 64-bin color histogram. These quantized histograms are used for multi-camera multi-person tracking. This quantization reduces the effects of spatially varying illumination color and it also significantly reduces object tracking performance. [Wei et al., 2007] give object tracking method which is not automatic and for which training is done off-line. They use selected number of histogram bins based on RGB color space cube, called boosted color bins. They find optimal object trajectory path by using dynamic programming. Similarly [Walder and Lovell, 2002] applied vector quantization (VQ) compression to the image stream and used weighted Euclidean distance between VQ histograms as the measure of image similarity. The main drawback of histogram based object matching techniques is that they completely lose spatial information. This problem is also mentioned by [Birchfield and Rangarajan, 2005]. They show that histogram based object recognition without spatial information reduce its matching performance.

Non parametric techniques such as kernel density estimation are equally popular for object appearance modeling. [Elgammal et al., 2003] divide objects height into three parts: head, torso and bottom. Each part is modeled by a Gaussian kernel density function and they improve the cost to compute the kernel density estimation. Several pre-calculated lookup tables are used to decrease its intensive computations. Division of object height into only three parts is not sufficient to capture the object's vertical color variation. [Mittal and Davis, 2003] use overlapping multi-camera system for human tracking in a cluttered scene. They divide object height into h slices and model each slice with a kernel density estimation explained in [Elgammal et al., 2003]. Using N Gaussians kernels for each of the h slices increases its computational cost extensively.

[Thome and Miguet, 2005] and [Thome et al., 2006] use the technique of human body parts labeling. Object recognition is done by using graph matching theory. It gives good results but performance decreases when dealing with small objects. [Sato and Aggarwal, 2004] discuss many aspects of tracking and interaction. They use objects horizontal size, area, vertical texture, horizontal projection and blob acceleration. The vertical texture in [Sato and Aggarwal, 2004] is a 1-D appearance model. Their 1-D appearance model is neither normalized nor rescaled the object's height and they use geometrical features like horizontal projection and horizontal size for object matching. Therefore, it is less reliable for human tracking as the apparent size can be very different in different frames or in different camera's FOV.

Multi-point based object recognition techniques use many invariant points as object features. These points are used to match current object with the objects present in object's database. Many methods are used for selecting invariant point features. [Lowe,

1999] and [Lowe, 2004] proposes Scale Invariant Feature Transform (SIFT) to detect local features in images. [Bay et al., 2008] present faster algorithm to find objects key points, called as Speed Up Robust Features (SURF). [Juan and Gwun, 2009] discuss the SIFT and SURF algorithms and claim that SIFT and SURF have similar performance but SURF is not stable to rotation and illumination changes. They illustrate that SURF is computationally much faster than SIFT. The drawback is that initially a learning step is necessary and only the learned objects can be identified. Secondly both are computationally expensive and not suitable for real time multiple objects tracking.

The template matching idea is to create a model for an object of interest (called the template, or kernel). This template is matched within an image, or foreground blobs to recognize it. In general, for a given template, position, scale and orientation is used to measure similarity between objects. If the similarity is above a threshold, then a possible matching of the template is reported. Templates are formed using simple geometric shapes or silhouettes [Fieguth and Terzopoulos, 1997]. An advantage of a template is that it carries both spatial and appearance information. Templates, however, only encode the object appearance generated from a single view. Thus, they are only suitable for tracking objects whose poses do not vary considerably during the tracking. [Gavrila and Philomin, 1999] obtain 1100 templates from a pedestrian silhouettes. These 1100 templates, increase computational expenses in pedestrian matching. Especially if each template has to be searched at different positions, orientations and scales. This number of possible combinations grows exponentially with increasing template numbers. It might be difficult to search for the best match with respect to each of the templates [Torre et al., 2005]. Finally, it is noted that although the template hierarchy can capture the variety of object shapes but it can not appropriately handle large shape variations when pedestrians are very close to the camera [Zhao and Thorpe, 2000]. Similarly, it can not perform well for camera environments that differ significantly from those used to create the search templates.

Multi-view appearance models encode different views of an object. One approach to represent the different object views is to generate a subspace from the given views. Subspace approaches, for example, Principal Component Analysis (PCA) and Independent Component Analysis (ICA), have been used for both shape and appearance representation [Mughadam and Pentland, 1997] and [Black and Jepson, 1998].

The object color appearance based particle filters are also used by some researchers [Nummiaro et al., 2003], [Kim and Davis, 2006] and [Czyz et al., 2007]. The benefit of particle filter base object tracking is that they can even track object under occlusion. Real-time, multi-objects tracking becomes much difficult with particle filter due to the

exponential growth of particles to track objects [Czyz et al., 2007] and [Martinez-Del-Rincon et al., 2007].
Cascading of different features is a robust ways to recognize some object. [Alahi et al., 2010] propose fixed cameras (master camera) and moving cameras (slave cameras) based object tracking algorithm. They discuss many recognition features like color histogram, histogram of oriented gradient (HOG), SIFT, SURF and different order of intensity derivatives. They show that cascading of different features increase the object detection and tracking performance but significantly reduces its real time tracking performance. Similarly, [Noceti et al., 2009] combine motion and color histogram based appearance model for object tracking and claim good results.
Recently, many researchers use overlapping view multi-cameras for object tracking under occlusion [Mittal and Davis, 2003], [Kim and Davis, 2006] and [Khan and Shah, 2009]. An overlapping multi-camera environment gives the benefits to get object's multi-views and to match objects under occlusion in another camera's field of views. In spite of overlapping camera's advantages, in most of real world scenarios, it is difficult to install and calibrate overlapped multi-camera surveillance systems (e.g. campuses, railway stations, subways, etc.). [Vazquez et al., 2007] deal with objects occlusion, split and merge in surveillance applications. They propose several rules for occlusion detection and correction based on the variation of the number of objects in previous and current frames. This approach lacks of reliability for instance if occlusion occurs at the same frame where a new object enters in the camera's FOV.
Object appearance based tracking is the most frequently used technique for video surveillance systems. It can give satisfactory results in difficult conditions like low video quality, small/large object sizes, partially view independent. Appearance based algorithms have good object re-identification percentage, even when objects exit from FOV of one camera and enter in the another camera's FOV.

2.3 Object Re-Identification

The task of observing an object in one camera's FOV and recognizing that object again in same camera or another camera's FOV is called object re-identification. An object may enters once or many times in camera's FOV. If algorithms can re-identify an object and store the information when and how many times objects enter and exits camera's FOV, then it is very important for objects activities analysis. Similarly, object re-identification has a significant importance for a large area monitoring scenarios like: airports, university campuses, shopping centers or train stations using multi-camera environments. In

multi-camera environments, object re-identification helps to make consistent record of objects movement when they exits from one camera's FOV and re-enter in the same or another camera's FOV.

Many researchers use overlapping cameras for object tracking. There are several advantages of using overlapping cameras. The objects can be recognized even if they are under occlusion in one or more cameras. Object feature matching and recognition performance is improved as multi-views of objects are available from different cameras. 3-D object shape/detail can be obtained. The large areas surveillance using overlapping cameras is practically difficult due to the installation of a huge number of cameras and then storing and processing huge data. Multi-view matching of same object in overlapping camera is a challenging task. This is not a case in a non-overlapping multi-camera environment. Non-overlapping multi-camera environments are used for large area surveillance system. In this section, we investigate object re-identification in non overlapping multi-cameras.

Figure 2.8 represents the typical example of object tracking in non-overlapping multi-camera environment. In this environment, three cameras are installed: two outdoor and one inside the building. The objects are assigned the same labels when they exit from the one camera's FOV and enter in another camera's FOV. [Javed et al., 2008] use object motion and spatial parameters for object tracking in non-overlapping camera environment. They combine these parameters with object appearance models and show that combining object motion parameters with object appearance models significantly improve object re-identification performance.

Figure 2.9 shows a generic non-overlapping camera environment. In figure 2.9, five

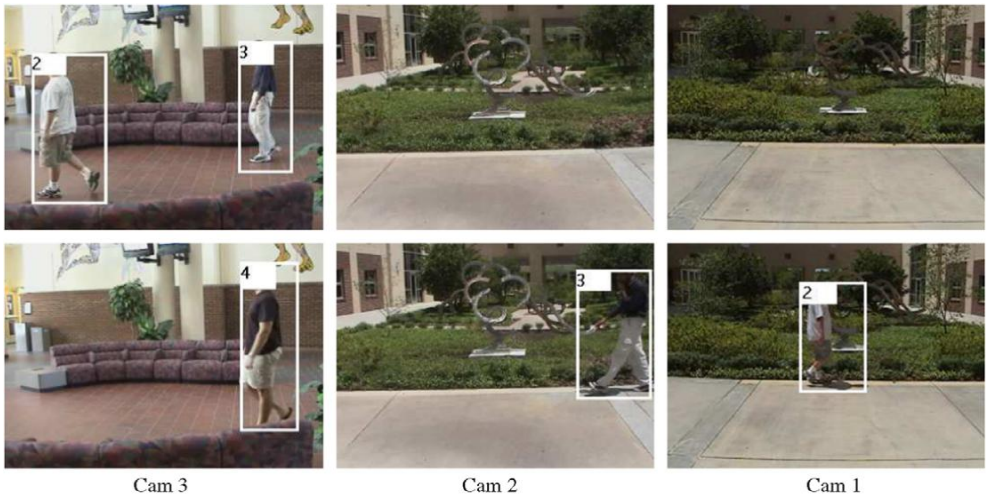

Figure 2.8: Object tracking in multiple non-overlapping cameras environment [Javed et al., 2008]

cameras are installed in a small building for surveillance purposes. This topology makes it possible to track the objects when they enter and exit from some room. In this topology only a small number of entry and exit points are possible. Objects can not enter and exit from the building without passing in the FOV of camera C4. This type of surveillance is only possible in important buildings. In general, for the large areas surveillance, many cameras are installed only on important places. There is a large number of blind regions between the cameras and many possible paths for objects to enter and exit from the region.

The motion features are not useful to re-identify the objects in non-overlapping cam-

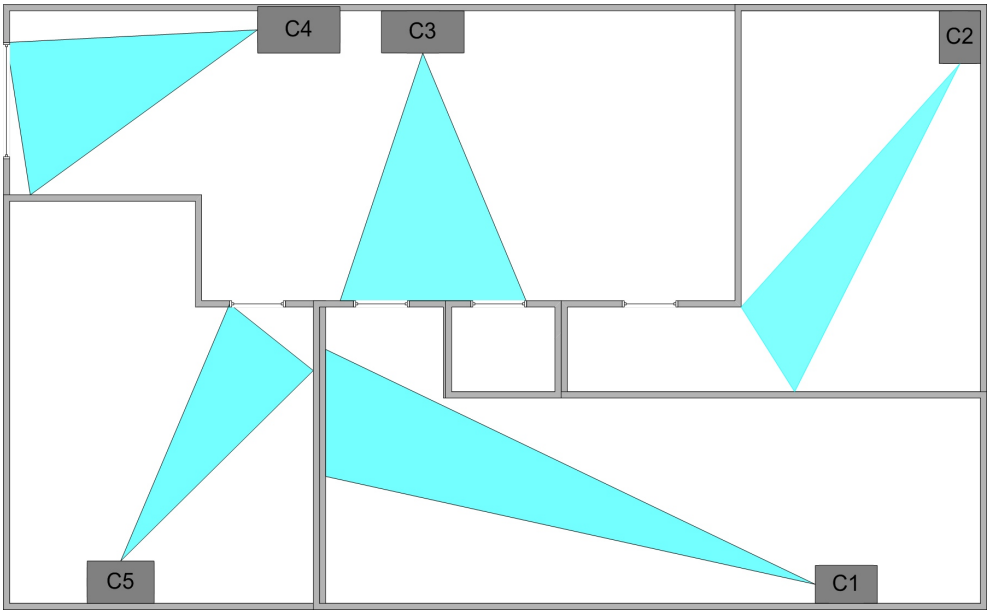

Figure 2.9: Generic topology of non-overlapping multi-camera surveillance system

eras environment. Because, there are many possible entry and exit points and motion features are object position dependent. Objects may be exited from one location and reentered in the scene from another location. Similarly, object geometrical properties are unable to produce good object re-identification results for non-rigid objects due to individual movement of object's parts and large variation of object sizes. There are plenty of single camera based object appearance models that were discussed in section 2.2.3 and that can also be used for object re-identification in a multi-camera environment.

Some researchers use motion features for object re-identification in non-overlapping multi-camera object tracking by combining object motion parameters with spatial information of cameras. [Makris et al., 2004], [Rahimi and Darrell, 2004] and [Zhu et al., 2009] used the information gained from observing location and velocity of objects moving across multiple non-overlapping cameras to determine spatial relationships between

cameras. [Rahimi and Darrell, 2004] assume that object correspondences are known between these cameras. Objects correspondences were not assumed to be known in [Makris et al., 2004] and [Zhu et al., 2009]. They derived a model through learning algorithm which is used to automatically determine the camera's positions and to continue tracking targets across the blind areas of the network. Objects appearance was not used by both methods. [Javed et al., 2005] demonstrate in their paper that appearance modeling improve the spatio-temporal information for robust tracking. Motion and spatial co-ordinate based tracking models suffer if camera's spatial positions are unknown.

Many published articles present object histogram matching techniques like signature based color histograms used in [Park et al., 2006], [Pham et al., 2007] and [Gandhi and Trivedi, 2007]. Recently published articles [Prosser et al., 2008] and [Orazio et al., 2009] also use histogram techniques for object re-identification in non-overlapping camera environments. Some researchers also add some other features in addition to histograms, like [Javed et al., 2008], who combine motion features with object histograms. [Porikli and Divakaran, 2003] use probability based Bayesian Belief Network (BBN) to boost up the performance of histogram based object recognition and re-identification. They match the current object histogram with the histogram stored in database. They calculate similarity between the histograms. If more than one histogram have similarity greater than a given threshold, then select the best object using the highest probability based Bayesian Belief Network (BBN).

[Kettnaker and Zabih, 1999] use a Bayesian formalization to track persons over multiple non-overlapping cameras. The optimal solution is the set of object paths with the highest posterior probability given by the observed data. In their research they assume the uniform motion of the objects and if the object stops for some time then the system is unable to re-identify it in other cameras. Many researchers also find probability based Bayesian Belief Network (BBN) effective for object linking and tracking. [Nillius et al., 2006] proposed a method to resolve multiple hypotheses via Bayesian networks to find the most probable set of paths in an efficient way in a multi-camera environment.

Similarly, some researchers [Lantagne et al., 2003] and [Vacchetti et al., 2004] use object texture characteristics for object identification and tracking. In general, video quality of surveillance cameras is not good enough to extract object texture information correctly. Therefore, this techniques is not suitable for object tracking [Porikli and Divakaran, 2003].

Recently, some work using, key interest points for establishing correspondence between objects are presented. [Arth et al., 2007] use SIFT for cars tracking, [Gheissari et al., 2006] presents results of SIFT for person re-identification. [Arth et al., 2007] reduce ob-

ject key-points using PCA-SIFT [Ke and Sukthankar, 2004] for large scale of cameras. The key point reduction decrease data correspondence between the cameras but object recognition performance is also decreased. [Hamdoun et al., 2008] use modified Speed Up Robust Features (SURF) and matching between descriptors is done by a Best Bin First (BBF) search in a KD-tree [Beis and Lowe, 1997] containing all models. In their experiments, they re-identify one object out of ten previously stored objects in multi-camera environment.

The other possible method to recognize peoples by the way they walk. A particular way or manner of moving is called gait. Some fundamental work on gait and its ability to recognize humans are discussed by [Johansson, 1973] and [Murray, 1967]. Early studies by [Murray, 1967] revealed that gait might be a useful biometric for people identification, a total of 20 feature components including ankle rotation, spatial displacement and vertical tipping of the trunk have been identified to render unique gait signature for every individual. Similarly, [Johansson, 1973] on human motion perception using Moving Light Displays (MLD) have revealed that an observer can recognize different types of human motion based on joint motions. Recently, the use of gait feature for people identification in surveillance applications has attracted researchers [Huang et al., 1999], [Chowdhury et al., 2003], [Carter and Nixon, 1999] and [Roth et al., 2005].

[Bouchrika et al., 2009] present an approach for people tracking and identification between different non-overlapping uncalibrated cameras based on gait analysis. For extraction of gait feature, they derive motion models based on object biological parts data that describe the angular motion of the knee and hip at different states of the gait cycle. A gait cycle is defined as the time interval between successive instances of initial foot-to-floor contact for the same foot [Cunado et al., 2003]. Recognition of human on the based of gait have some limitations. Human way of walking have limited number of pattern. The gait of human actually can identify objects groups and unique object recognition is difficult within a group. All the objects might have similar motion parameters. Similarly, in multi-camera environment, unique gait cycle is difficult to find due to different view angle of objects in cameras.

[Bak et al., 2010] propose a new appearance model based on spatial covariance regions extracted from human body parts. The new spatial pyramid scheme is applied to capture the correlation between human body parts (each part is modeled with 11-D recognition vector) in order to obtain a discriminative human signature. The human body parts using are detected using 15 windows at specific locations around the human silhouette are automatically detected using histograms of oriented gradients (HOG). They trained human body parts detector algorithm by using 10,000 positive and 20,000 negative im-

age samples.
In non-overlapping multi-camera environment, object motion features can be used if camera are spatially calibrated. But camera spatially calibration is a tedious and computationally costly job. Additionally, camera position calibration is needed whenever some camera is installed at new position. Similarly, if an object stops after exiting from one camera's FOV for some time duration and then re-enters in another camera's FOV, then motion based algorithms are unable to recognize it. Object geometrical properties are not effective enough for non-rigid objects due to individual moment of human body. Object appearance is the most important parameter for object re-identification when it exits from the camera's FOV and enters in some another camera. Object's colors might be very different in different cameras due to many reasons. Object appearance models may give bad recognition results if cameras are installed in a very different luminosity conditions. Object recognition performance can be improved by applying inter-camera color calibration. We will discuss previous work on inter-camera calibration in next section and our proposed technique of cameras color calibration in chapter 5.

2.4 Inter-Camera Color Calibration

In the previous section, we found that object appearance models are better option for object tracking in multi-camera environments. This problem decreases the object re-identification performance. The solution of this problem lies in the color calibration of the cameras. The color space transformation of one camera to another camera is called camera color calibration.
The color calibration for object re-identification is an important task. Object appearance in different cameras may be very different even in controlled environment (indoor regions). Many methodologies are used to minimize the color variation in multi-camera environments like normalized color spaces (HSV, YCbCr, Lab), histogram equalization and color calibration.
Histogram equalization was one of the most frequently used technique to minimize the color variation between cameras. Even identical cameras which have the same optical properties and working under the same lighting conditions may not match in their color responses [Bak et al., 2010]. Hence, color normalization procedure has been carried out in order to obtain invariant signature. They use a histogram equalization technique proposed by [D. et al., 2005]. This technique is based on the assumption that the rank ordering of sensor responses is preserved across a change in imaging conditions (lighting or device). Histogram equalization is an image enhancement technique originally

developed for single channel images. Some works on using histogram equalization techniques to compensate the illumination changes are also used in [Cheng and Piccardi, 2006] and [Hamdoun et al., 2008]. Linear histogram equalization technique is not sufficient for illumination modeling. Because there is possibility of illumination variation and might be some camera CCD has better response for red, green or blue color than other colors due to CCD manufacturing process. Secondly, histogram equalization do not take camera color information of other camera. So each camera colors are corrected independently.

[Gilbert et al., 2006] perform inter-camera calibration using updated transformation matrix. However, this method requires thousands of objects to construct an accurate transformation matrix. A similar model was proposed in [Ilie and G.Welch., 2005] without incremental learning. Instead, it requires a hardware calibration phase which is infeasible with camera installations of unknown camera parameters.

The other methodology is a color calibration of cameras by using Brightness Transfer Function (BTF). Actually BTF transforms brightness characteristics of one camera to another camera. In color calibration techniques, one camera is set as a reference camera and the other cameras are calibrated to it by using the statistical properties of the images (cumulative histogram) matching. There are two possible methods for color calibration: color calibration before and after installing the cameras for surveillance.

In the first type of camera color calibration technique, images are taken from the cameras filming the same scene or uniformly illuminated charts. [Mann and Picard, 1995], [Debevec and Malik, 1997] and [Grossberg and Nayar, 2002] use images of a uniformly illuminated color chart of a known reflectance taken under different exposure settings to estimate the parameters of a brightness transfer function. Often, they assume the function is smooth and polynomial. [Porikli and Divakaran, 2003] present a method to match objects in non overlapping camera systems. They compute the BTF for each camera pair before installing them. Figure 2.10, explain their inter-camera color calibration algorithm. They claim that once this mapping is calculated, then object correspondence is reduced to a histogram transformation. But this assumption is not sufficient: if after the color calibration, these cameras are installed in regions having different illuminations conditions than the calibrated color environment, then their performance are significantly reduced.

The second type of camera color calibration technique is done after installing the cameras for video surveillance applications. In most of video surveillance applications, cameras are installed with non-overlapping FOV, in order to monitor large regions. In this situation, no common regions are available. BTF is calculated by passing some objects

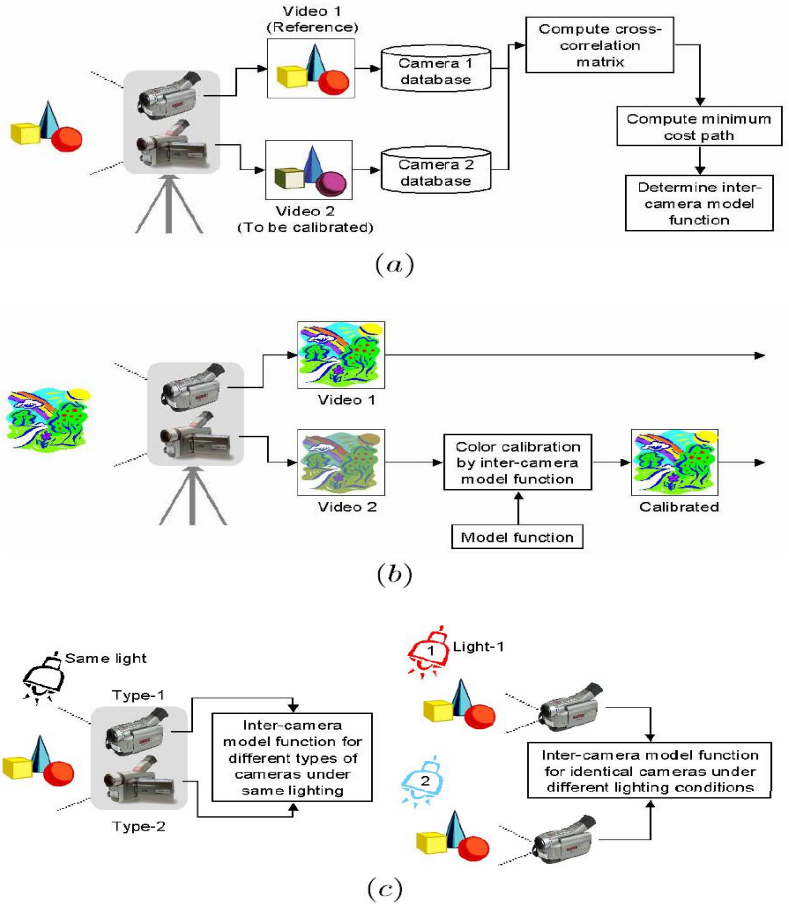

Figure 2.10: (a) A multi-camera setup, which can contain one reference and several uncalibrated cameras, generates camera-wise databases of videos. After obtaining frame-wise histograms and computing the total cross-correlation matrix, a minimum cost path is found by dynamic programming. This path is converted to an inter-camera model function. (b) Using the model function obtained in the previous stage, the output of the second camera is compensated to match its color distribution with the reference camera. (c) Some possible scenarios: single-light different type camera setup, and different-light identical camera setup. [Porikli and Divakaran, 2003]

from one camera to the others cameras. After getting the images of cameras, cumulative histograms are calculated and allow to compute BTF.

[Javed et al., 2008] propose a subspace based color brightness transfer function. They use probabilistic PCA for object matching. They learn color mapping and camera position information in the learning phase. They combine color information with object motion and show the superiority of the combined approach. We do not have the camera installation information and several entry and exit points are possible.

Similarly, two other methods are frequently used for inter-camera color calibration. These methods are Cumulative Brightness Transfer Function (CBTF) and Mean Brightness Transfer Function (MBTF). In both of the methods, inter-camera color mapping is

learned during the training phase. The same objects are allowed to pass in front of these cameras, under the assumption that objects correspondence between the cameras are known. The main difference between the two techniques is that in MBTF, BTF is calculated for each object which passes through a camera pair. MBTF is computed by taking mean value of all BTFs. In CBTF, all of the histograms are accumulated in a single cumulative histogram (one cumulative histogram for each camera) and then BTF is calculated. In chapter 5, we will discuss MBTF and CBTF in detail. Concepts of Cumulative Brightness Transfer Function (CBTF) and Mean Brightness Transfer Function (MBTF) are discussed in detail in [Prosser et al., 2008] and [Orazio et al., 2009]. [Prosser et al., 2008] show in their experiments that CBTF is better than MBTF which seems to be contradicted by the results of [Orazio et al., 2009], who demonstrate the superiority of MBTF over CBTF. In our experiments, we find that results of MBTF and CBTF are not significantly different, if objects colors do not equally cover all the regions of the histogram. For example if there are many objects with dark colors, then inter-camera BTF has less significant information for light colors. These problems suggest to propose some modification for existing inter-camera color calibration techniques CBTF or MBTF.

2.5 Discussion

In this chapter, we present the state of the art of the following fields: object detection in camera's videos, object tracking, object re-identification and inter-camera color calibration. Object detection/ extraction from the image sequence is the first and important part of object tracking. We discuss many object detection techniques from simple to complex and past to recent research work. We can conclude from section 2.1 that background modeling (see section 2.1.2) based object detection techniques has clear advantage over other techniques. The simple and fundamental techniques presented in 2.1.1 are unable to detect objects when they stop in a scene or objects and background share similar colors. The combined approach models (see section 2.1.3) are computationally extensive as they combine the models of section 2.1.1 and 2.1.2. During our experiments, we find, Mixture of Gaussians (MOG) and CodeBook (CB) methods from background modeling are more suitable for object detection. They have reasonable real time performance and they detect objects better. But in some condition, both algorithms are unable to give better results, we discuss this issue in chapter 3.

We divide object tracking (see section 2.2) algorithms on three fundamental groups on the basis of their inherent properties. Objects are recognized and tracked by matching these properties of current and stored objects. Geometrical models (see section 2.2.2)

are unable to recognize non-rigid objects, due to individual movement of persons body parts. Similarly, motion models (see section 2.2.1) are unable to track objects when they exit and re-enter in the camera's FOV. Appearance models (see section 2.2.3) are good choice to track rigid and non rigid objects.

Objective of this research activity is to track non-rigid object (persons) in multi-camera environment. We do not have any information of the scene geometry and the spatial positions of cameras. In the section 2.3, we discussed the various algorithms which are used for object re-identification in multi-camera environment. Some algorithms calculate each camera position during a training time and then use motion models to track and identify objects. Some algorithms use the way of walking objects to identify humans. But humans have limited number of ways to walk. Low video qualities, small object size, camera view angle and individual movement of non rigid motion make geometrical features non practical for non-rigid object re-identification in multi-camera environment. Object appearance models are good choice for object tracking and re-identification in single and multi camera environment. We also found that 1-D and 2-D appearance models are interesting choices for object re-identification.

We come to conclusion that object tracking and re-identification problem in single or multi-camera environment can be effectively addressed by combining appearance and motion models. Motion models are unable to re-identify objects, when they re-enter in the scene but give reasonable tracking performance. Simple appearance models are unable to track object effectively, specially if two or more similar objects are present in the scene. We proposed to track the objects by using the combination of appearance and motion model and when they exit and re-enter in the scene then, use only appearance model to re-identify them.

In the section 2.4, we discussed various techniques which are used to minimize the object appearance difference in multi-camera environment. Some inter-camera color calibration techniques are useful before installing the cameras. But after installing the cameras, scene geometry and illumination conditions, can affect camera's colors values. We find that two techniques MBTF and CBTF have the abilities to calibrate the camera's colors after installing the cameras. In chapter 5, we will discuss the limitations of MBTF and CBTF in details in certain situations.

Contents

Chapter 3

Real Time Foreground-Background Segmentation Using a Modified Codebook Model

Moving objects extraction from image/video sequences is one of the most interesting, well-focused and well addressed but still challenging topic in computer vision. Results of segmentation depend on the variation of local or global light intensities, object's shadow and background changes. Object recognition and tracking algorithms performance depend on object detection quality. In this chapter, we propose an improvement to the foreground/background segmentation algorithms based on codebook. Therefore, a better moving object detection can be performed. We also propose an evaluation methodology to objectively compare segmentation techniques, based on the analysis of the precision and recall of algorithms. Based on a test set derived from a databases, we show the good behavior of our Modified CodeBook (MCB) algorithm.

3.1 Introduction

Video sequences filmed by fixed cameras contain moving objects on a fixed background. In order to perform object tracking, we need to extract moving objects from image sequences. In general, it is assumed that pixels belonging to objects have different color values than background. Background modeling, subtraction and estimations are the widely used techniques to extract objects from background.
Background modeling techniques model the background using previous frames history. Every image pixel is matched with its background model. If pixel color value is similar

to the background model, then it is considered as background model otherwise it is an object pixel. Mixture of Gaussians proposed by [Stauffer et al., 2000] is the largely used technique from this class. Background subtraction technique learns image background through averaging process of current and previous frames. The background learning rate may be fixed or adaptively calculated. This background image is subtracted from current image and all the pixels above some threshold value are considered as object pixels ([Horprasert et al., 1999]). Background estimation techniques use probabilistic model to predict the background pixel color value. This prediction is based on previous frames history of the pixel. If current pixel color value is significantly different from the predicted pixel color value, then current pixel is considered as an object pixel. Background estimation using Kalman filter is the most used technique from this class ([Ridder et al., 1995]).

In the above discussed three classes, background subtraction and background estimation algorithms are unable to perform better in outdoor environments. Global luminosity variation, object's shadow, non-stationary background are some of the major reasons for their failure. Due to these reasons, background modeling techniques are considered as a good choice for their better performance for indoor and outdoor scene.

In our work, we emphasize on moving object extraction from the background filmed by fixed cameras. There are still some situations in which existing background modeling algorithms have limited performance. For instance: objects detection when they stop, global and local luminosity variations, moving backgrounds, object's shadow, object and background color similarity, dynamic changes of background with time, etc.

This chapter is organized as follows: section 3.2, presents in detail two famous background modeling approaches and our proposed modification of the codebook method, originally presented in ([Kim et al., 2005]). In section 3.3, we explain the evaluation techniques to estimate the quality of foreground-background segmentation using ROC curves. In section 3.4, we discuss the performance of Mixture of Gaussians (MOG), codebook and our proposed segmentation technique. In the last section 3.5, we conclude the chapter and discuss the future works.

3.2 Segmentation Techniques

We decided to focus on two approaches: normalized Mixture of Gaussians (MOG) technique described in [Stauffer et al., 2000] and CodeBook (CB) method [Kim et al., 2005]. These methods belong to background modeling technique. Indeed, these two methods are considered the best object detection techniques for real time object tracking. We

applied MOG and CB on different videos having different illumination conditions like sunny day, cloudy environment and large or small number of moving objects. We observed that most of the time detection precision of CB is better than MOG. There are some situations where both methods fail to give satisfying performances:

- When no initial training on empty scene is available.
- When a large number of objects are moving in the scene.
- When object color is similar to the background.

3.2.1 Mixture of Gaussians

[Stauffer et al., 2000] proposed the Mixture of Gaussians (MOG) technique. This technique models a background using N Gaussian distributions based on recent history of each pixel. The probability density function (PDF) of color value of each pixel can be formulated by using the general equation

$$P_r(X_t) = \frac{1}{N}\sum_{i=1}^{N} w_{i,t} \times \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{3.1}$$

where N is the number of Gaussian distributions, $w_{i,t}$ is an estimated weight of each PDF and η is a Gaussian probability density function.

$$\eta(X_t, \mu_{i,t}, \Sigma_{i,t}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_{i,t}|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t-\mu_t)^T \Sigma_{i,t}^{-1}(X_t-\mu_t)} \tag{3.2}$$

In Equation 3.2, $\mu_{i,t}$ is the mean and $\Sigma_{i,t}$ is the covariance matrix of pixel color value. We assume that color channels R, G and B are statistically independent and have different color variance ([Thome and Miguet, 2005]). The current pixel value is matched by using the equation 3.3,

$$||X_t - \mu_{i,t}|| < \kappa\Sigma_{i,t} \tag{3.3}$$

The value of κ is 2 or 3. The equation 3.3 says that if pixel value is between 2 or 3 times of the standard deviation of Gaussian then it is considered as background pixel. The weight $w_{i,t}$ of each Gaussian distribution can be calculated by using the equation bellow:

$$w_{i,t} = (1-\epsilon)w_{i,t-1} + \epsilon(M_{i,t}) \tag{3.4}$$

ϵ is the learning rate. $M_{i,t}$ is 1 for the model which is matched and 0 for the others. For further detail see [Stauffer et al., 2000], [Elgammal et al., 2000] and [Thome and

Miguet, 2005], covering all the information regarding parameter updating, tunning and matching.
Clever tuning of MOG is really important and also technical. If the model updates too quickly, then it absorbs slow moving or stationary objects into background. Similarly, slow update rate takes more time to adjust changes in light intensities.

3.2.2 Codebook

[Kim et al., 2005] propose the CodeBook (CB) method to build a background model. They were inspired by the algorithm presented in [Kohonen, 1988] to build a CB. It is a quantization technique using long scene observation for each pixel. One or several codewords are stored in the codebook for each pixel. The number of codewords for a pixel depend on the background variation. This is the reason that all pixels do not have the same number of codewords. Codebook is an effective and faster way for background modeling.
Each codeword $c_L, L = 1...\chi$ is represented by a RGB vector v_L= $(\bar{R}, \bar{G}, \bar{B})$ and a hextuple $aux_L = \langle \check{I}_L, \hat{I}_L, f_L, \lambda_L, p_L, q_L \rangle$. Where $\check{I}_L$ = min$\{I, \check{I}_L\}$ and $\hat{I}_L$ = max$\{I, \hat{I}_L\}$ are the minimum and the maximum brightness assigned to the codeword respectively. f_L is the frequency or the number of times that codeword is matched. λ_L is the maximum negative run length, meaning the largest time span in which this codeword is not updated/accessed. p_L and q_L are the first and the last access times of the codeword respectively.
The color distortion δ is computed between the current pixel and the codeword using the equation 3.5.

$$colordist(x_t, v_L) = \delta = \sqrt{||x_t||^2 - C_p^2} \tag{3.5}$$

$||x_t||^2$ and C_p^2 are calculated using equations 3.6 and 3.7

$$||x_t||^2 = R^2 + G^2 + B^2 \tag{3.6}$$

C_p^2 is the autocorrelation of R, G and B colors of input pixel and the codeword, normalized by brightness.

$$C_p^2 = ||x_t||^2 \, cos^2\theta = \frac{(R_i R + G_i G + B_i B)^2}{R_i^2 + G_i^2 + B_i^2} \tag{3.7}$$

They also use brightness value, $I = \sqrt{R^2 + G^2 + B^2}$ and its two bounds, $I_{low} = \alpha \hat{I}$ and I_{hi} = min$\left\{\beta \hat{I}, \frac{\check{I}}{\alpha}\right\}$, which are defined during codeword updating.

Initially, background is modeled with CB during training time period using the algo-

Input: Video sequence
Output: moving object extraction from videos

I Initialization: $\chi \leftarrow 0$, $C \leftarrow \phi$ (empty set)

II **for** $t = 1$ *to* τ **do**

(i) x_t = (R, G, B), I = $\sqrt{R^2 + G^2 + B^2}$

(ii) find the codeword c_m in C= $\{c_L | 1 \leq L \leq \chi\}$ matching to x_t using following condition

(a) colordist (x_t, v_L) $\leq \Delta$;

(b) $(I_{low} \leq I \leq I_{hi})$;

(iii) if $C = \Phi$ or there is no match, then create a new codeword in codebook

- $\chi \leftarrow \chi + 1$
- $v_\chi \leftarrow (R, G, B)$;
- $aux_\chi \leftarrow (\langle I, I, 1, 1, t, t \rangle$

(iv) otherwise, update the matched codeword c_m, consisting of

- $v_m \leftarrow (\frac{f_m \bar{R}+R}{f_m+1}, \frac{f_m \bar{G}+G}{f_m+1}, \frac{f_m \bar{B}+B}{f_m+1})$
- $aux_m \leftarrow (\langle min(I, \check{I}_m), max(I, \hat{I}_m), f_m + 1, max(\lambda_m, t - q_m), p_m, t_m \rangle)$

end

III For each codeword c_L, L = 1,...,χ, wrap around λ_L by selecting $\lambda_L \leftarrow$maxλ_L, $(\tau - q_L + p_L + 1)$

Algorithm 1: Object detection using codebook algorithm

rithm 1. In this time period, codewords in the codebook are created or updated using the following criteria.

If $colordist\ (x_t, v_L) \leq \Delta$ and $(I_{low} \leq I \leq I_{hi})$, then matched codeword c_m is updated as:

$v_m \leftarrow (\frac{f_m \bar{R}+R}{f_m+1}, \frac{f_m \bar{G}+G}{f_m+1}, \frac{f_m \bar{B}+B}{f_m+1})$

$aux_m \leftarrow (\langle min(I, \check{I}_m), max(I, \hat{I}_m), f_m + 1, max(\lambda_m, t - q_m), p_m, t_m \rangle)$

Otherwise, create and initialize a new codeword in the codebook. According to our experiments, the values of α between 0.6 and 0.8, and β between 1.15 and 1.35 give better results in most of the situations. The values of α and β closer to to 1, increase CB sensitivity. It better detects moving objects but also detect object's shadows, like MOG and other methods. The number of codewords required for each image pixel can be different. During the codeword matching process with current pixels, only corresponding codewords are used.

Codebook obtained during the training time represent the training image sequence. It

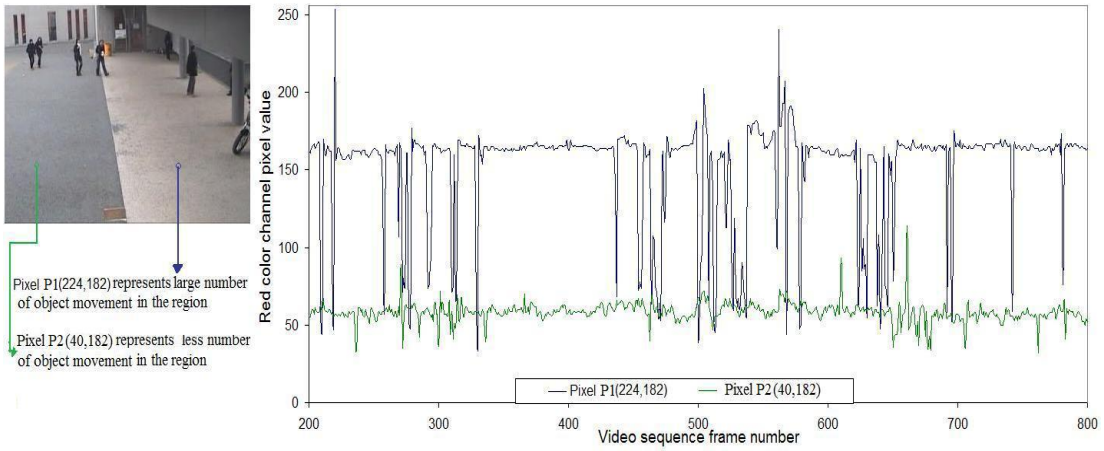

Figure 3.1: Time history of two pixels representing large and small number of objects movement in their areas

may contain objects information also if objects are moving in the scene during training time. The variable λ_L is useful for filtering the codewords which are not updating. It is assumed that codewords representing objects colors has higher value of λ_L. Because codewords representing objects are not updated frequently. The codewords having $\lambda_L \leq \lambda_{th}$ are deleted from codebook or not used in the process of code matching. The optimal value selection λ_{th} is an important task. Authors suggest to use $\lambda_{th} = \frac{\tau}{2}$ is a good choice. τ is representing the duration of training time period. The higher value of λ_{th} becomes the reason of adding many object's pixel color into codebook. Similarly, small value of λ_{th} is unable to model the background movement. We discuss this issue at the end of this subsection. In general, λ_{th} value between 250 and 300 gives better results in many cases. The figure 3.1 shows the time history of two pixels, pixel P_1(224,182) and pixel P_2(40,182). The pixel P_1 is near to right bottom corner of the image. The pixel color variation is showing that there are many objects movement in this region. Actually this pixel is near to the door. That is why many objects are moving towards this direction. The pixel P_2, belongs to the region where a very small number of objects are moving. This is the reason its pixel value does not change frequently. The history of pixel P_1 is also illustrating that its value is changing randomly. The pixel value near to 50 is repeating many times. This become the reason that some codewords representing object colors are updated continuously. This become the reason of poor foreground-background segmentation.

The scene can change after initial training time. For instance, in street surveillance application, cars might enter or leave a parking, etc. If codebook is not adaptive then it will detect false background or foreground pixels due to changes in scene. To avoid this problem [Kim et al., 2005] introduce a cache system. Cache codewords have the

same structure as codewords. After the training period, if an incoming pixel matches a codeword in the codebook, then this codeword is updated. Else, the pixel's information is put in the cache codeword and this pixel is treated as a foreground pixel. The cache codeword staying in cache more than a predefined time period is added into codebook. Although the original codebook is a robust background modeling technique, there are some situations where it fails.

- In winter, peoples commonly use black coat or jacket. If foreground-background segmentation is done using the CB method, it adopts black color as a background for many pixels. That is why many pixels are incorrectly segmented.
- If an object in the scene stops its motion, then it is absorbed in the background.
- Sometimes, CB is unable to absorb slow movement of background, for example tree's leaf movement if wind is blowing with small velocity.

The authors indicate tuning of λ_{th} to overcome these problems. For example, the first problem can be minimized by selecting a small value of λ_{th}. Similarly, one solution for the second and third problem lies in increasing the value of λ_{th}. It is almost impossible to select appropriate value of λ_{th} for all the situations as there is a possibility of occurrence of above discussed three situations simultaneously.

3.2.3 Modified Codebook

In this section, we suggest some changes in the original codebook algorithm. The maximum negative run length λ_L alone is not sufficient for filtering of codewords in the codebook. Similarly, a criterion to move a cache codeword into the codebook if it stays enough time in cache is also insufficient. These parameters are used to delete or add codewords in codebook. In the last paragraph of previous subsection, we showed the situations when codebook is unable to give good results. On the basis of observation and experiments, we come to conclusion that we should also include another parameter into algorithm in order to access, delete, match and add a codeword in the codebook and to move cache codewords into codebook. We add the parameter of frequency for codebook and for cache codewords as follows:

- Only codeword whose parameters of maximum negative run length $\lambda_L \leq \lambda_{th}$ and code access frequency $f_L \geq f_{th}$ are included in the process of matching.
- In the matching process if a new pixel value does not fulfill the criteria of color and brightness then put this pixel into cache and marks it as object pixel.

- If some cache word is staying in cache greater or equal than some reference time t_{ref} and frequency of the cache codeword $f_M^c \geq f_{th}^c$ then, this cache codeword is added into codebook.

The complete algorithm is presented in algorithm 2. In the first part, codebook C, cachebook C^c, numbers of codewords χ in codebook and χ^c in cachebook are initialized to zero. In the second part, background is modeled in a training time τ using the algorithm 1. We have explained this algorithm in previous section. Training process can use any number of frames. But we find that value of $\tau > 200$ frames is sufficient in most situations.

In the 3rd part, each pixel x_t of image sequence is matched with the corresponding codewords in codebook using the conditions (1) and (2) (see algorithm 2). The value of χ depends upon the scene. The pixels presenting stationary background need only 2 or 3 codewords. Similarly, the pixels representing large background movement may use 9 to 10 codewords. If codeword is matched, then update v_m and aux_m. Else initialize a new codeword in the codebook.

After the training time τ, each pixel x_t is matched with the corresponding codewords using the conditions (a), (b), (c) and (d) in the step III (ii). If object is matched then it is updated using the step III (iii). The conditions (a) and (b) help to remove the codewords which are included into codebook due to object movement in the scene during the training time. Similarly, it is also probable, that for instance, some persons having similar color clothes move in the scene, so codewords in codebook might include their colors as a background in the codebook. In figure 3.1, the pixel P_1 value between 45 and 55 is representing persons wearing black coat. The pixel P_1 is showing that the pixel value between 45 and 55 is repeating many times. This becomes the reason of matching object colors frequently in codebook. The pixel's value repeating frequently always produce a small value of λ_L (maximum negative run length). In this condition, λ_L (maximum negative run length) is not sufficient for removing objects pixels from codebook, because many object's clothes have similar colors. Figure 3.1 also illustrate the pixel P_1 value between 45 and 55 during the frames 200 and 500 is repeating with the frequency nearly equal to 10. The object's colors may be removed from background models using the combination of parameters, frequency f_L and λ_L. The codewords belongs to objects have small repetition frequency compared to background. The pixel P_1 value between 160 and 170 (background color value) is repeating during the frames 200 and 500 is repeating more than 150 times. The codewords belong to stationary or slow moving backgrounds are updated more frequently because they have same color value

Input: Video sequence
Output: moving object extraction from video

I Initialization: $\chi \leftarrow 0$, $\chi^c \leftarrow 0$, $C \leftarrow \phi$, $C^c \leftarrow \phi$ (empty set)

II **for** $t = 1$ *to* τ **do**
Construct codebook C using algorithm 1
end

III **for** $t = \tau + 1$ *to* ... **do**

(i) x_t = (R, G, B), I = $\sqrt{R^2 + G^2 + B^2}$

(ii) find the codeword c_m in C= $\{c_L | 1 \leq L \leq \chi\}$ matching to x_t using following condition
(a) $\lambda_L \leq \lambda_{th}$
(b) $f_L \geq f_{th}$,
(c) colordist (x_t, v_L) $\leq \Delta$
(d) $(I_{low} \leq I \leq I_{hi})$

(iii) update the matched codeword c_m, consisting of
- $v_m \leftarrow (\frac{f_m \bar{R}+R}{f_m+1}, \frac{f_m \bar{G}+G}{f_m+1}, \frac{f_m \bar{B}+B}{f_m+1})$
- $aux_m \leftarrow (\langle min(I, \check{I}_m), max(I, \hat{I}_m), f_m + 1, max(\lambda_m, t - q_m), p_m, t_m \rangle)$

(iv) otherwise, find the codeword c_m in C^c= $\{c_L | 1 \leq M \leq \chi^c\}$ matching to x_t based on following condition
(a) $\lambda^c_M \leq \lambda^c_{th}$
(b) colordist (x_t, v^c_M) $\leq \Delta$
(c) $\left(I^c_{low} \leq I \leq I^c_{hi}\right)$

(v) update the matched codeword c^c_m in cache, consisting of
- $v^c_m \leftarrow (\frac{f^c_m \bar{R}+R}{f^c_m+1}, \frac{f^c_m \bar{G}+G}{f^c_m+1}, \frac{f^c_m \bar{B}+B}{f^c_m+1})$
- $aux^c_m \leftarrow (\langle min(I, \check{I}^c{}_m), max(I, \hat{I}^c{}_m), f^c_m + 1, max(\lambda^c_m, t - q^c_m), p^c_m, t_m \rangle)$

(vi) if x_t is not matched in codebook C and cachebook C^c, then
- $\chi^c \leftarrow \chi^c + 1$
- $v^c_\chi \leftarrow (R, G, B)$
- $aux^c_\chi \leftarrow (\langle I, I, 1, 1, t, t \rangle)$

(vii) add cache codeword c^c_M into codebook, satisfying following conditions
- $t_{cw} \geq t_{ref}$
- $f^c_M \geq f^c_{th}$

end

Algorithm 2: Object detection using modified codebook algorithm

most of the time.

Scene may change due to the luminosity variation with time, some object can enter or previously stationary objects can leave the scene, etc. The cache is useful to adopt these changes. If some pixel value is not matched in the codebook then, algorithm try to match the image pixel into cachebook C^c using (a), (b) and (c) of the step III (iv). If it is matched with cachebook codeword c^c_m, then it is updated using step III(v). Else, create a new codeword in cache using III(vi). Please note that, the codewords in the cache which are not updated frequently are deleted to free memory. The codewords, fulfill the criteria III(vii) are added into codewords. The step III(vii) says that the codewords repeating frequently f^c_M is greater than frequency threshold f^c_{th} then after the time t_{cw} greater than reference time t_{ref} put these cacheword into main codebook.

Figure 3.2 presents moving objects segmentation of three video sequences using MOG, CB and MCB. We also present the ideal segmentation is manually labeled images. In figure 3.2.a, MCB gives the best results. CB is unable to detect all the object pixels and MOG detects many background pixels as a moving object. In figure 3.2.(b), sudden change of light intensity occurs. MOG was unable to absorb this quick change, therefore it detects more background pixels as foreground pixels. CB and MCB are more robust for this problem and MCB shows better detection rate than CB. Figure 3.2.(c) shows that MCB correctly detects objects, with better precision than CB and MOG. It is also clear from figure 3.2, that MOG detects shadows, while this problem is not present in the CB and MCB. Once again, MCB shows better objects detection.

If objects having similar color value are frequently moving in the scene, the corresponding pixel values are added into codebook and marked as background. Higher value of f_{th} also marks background pixels as foreground. Similarly, very small ranges of t_{ref} and f^c_{th} combination, add almost all of the stationary foreground objects or moving very slowly into background.

Optimal selection of these parameters values is important. The value f^c_{th} should be greater or equal to f_{th}. According to our experiments, value of f_{th} and f^c_{ref} should be between 10 and 15, value of λ_{th} should be between 220 and 260. t_{ref} should be between 80 and 100 and λ^c_{th} between 30 and 40 show good performance in all situation. We find these values by maximizing the result of MCB and evaluating it through techniques explained in the next section.

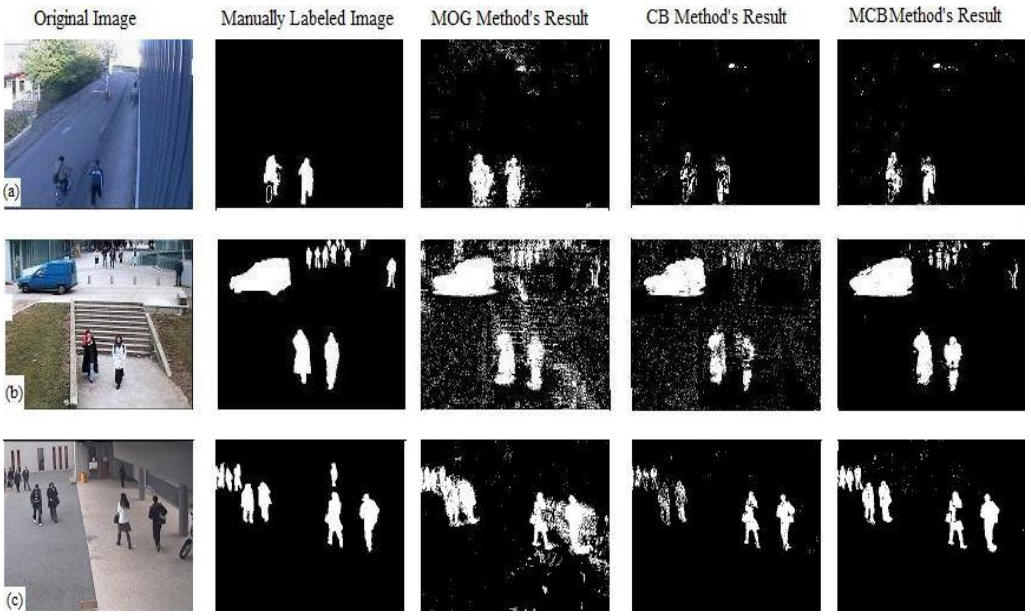

Figure 3.2: Original image, manually labeled image and result of three techniques are shown respectively

3.3 Evaluation of Segmentation Algorithms

In general, results of segmentation techniques are presented by showing some of segmented frames of video with propose algorithm and some standard techniques. But this method is not sufficient for evaluation. Some authors use evaluation techniques which are based on receiver operating characteristic (ROC) to show the comparison between different techniques. Some useful work is available in [Chalidabhongse et al., 2003] and [Davis and Goadrich, 2006]. They use specificity and sensitivity to compare different segmentation techniques. We use precision and recall to plot the graph and give result in the form of a single value. Our method consist of four steps:

1. Select different frames of several challenging videos including high variation in light intensities, large number of moving or stationary objects, sun light, clouds etc. Some of these image frames are shown in figure 3.2

2. Manually label these selected frames. These are used as ideal segmented reference frames (ground truth).

3. Calculate the true positive (TP), false positive (FP), true negative (TN) and false negative (FN) by comparing ground truth with segmented frames using CB, MCB and MOG.

$$TPR = RE = \frac{TP}{TP + FN} \tag{3.8}$$

$$FPR = 1 - \frac{TN}{TN + FP} = \frac{FP}{TN + FP} \tag{3.9}$$

$$PR = \frac{TP}{TP + FP} \tag{3.10}$$

4. We use the previous three steps to form the single value for various method to compare the segmentation techniques results more comprehensively:

   (a) Precision and recall may be combined into a single statistic number F. Which is harmonic mean of these numbers.

   $$F = 2\left(\frac{PR * RE}{PR + RE}\right) \tag{3.11}$$

   (b) We propose a formula, which is a weighted Euclidean distance that can be adapted to the needs of the application:

   $$E = \sum_{j=1}^{m} \sqrt{\gamma \left(FPR\right)^2 + \left(1 - \gamma\right)\left(1 - TPR\right)^2} \tag{3.12}$$

   E is the sum of all the weighted distances from the ideal position to calculated position of TPR and FPR. γ is a weighting coefficient of error from ideal position. Selection of γ parameter is discussed more in detail in section 3.4.

   (c) The segmentation results are also compared using other evaluation techniques describe in [Rosin and Ioannidis, 2003]: the Percentage of Correct Classification (PCC) and Jaccard Coefficient (JC).

   $$PCC = \frac{TP + TN}{TP + FN + FP + TN} \tag{3.13}$$

   $$JC = \frac{TP}{TP + FP + TN} \tag{3.14}$$

3.4 Results

In this section, we present the results of segmentation techniques explained in section 3.2 using the evaluation methods discussed in section 3.3. In general, large and fast changes in light intensities cause instability in all segmentation techniques. Most sensitive segmentation techniques (like MOG) produce large variance.

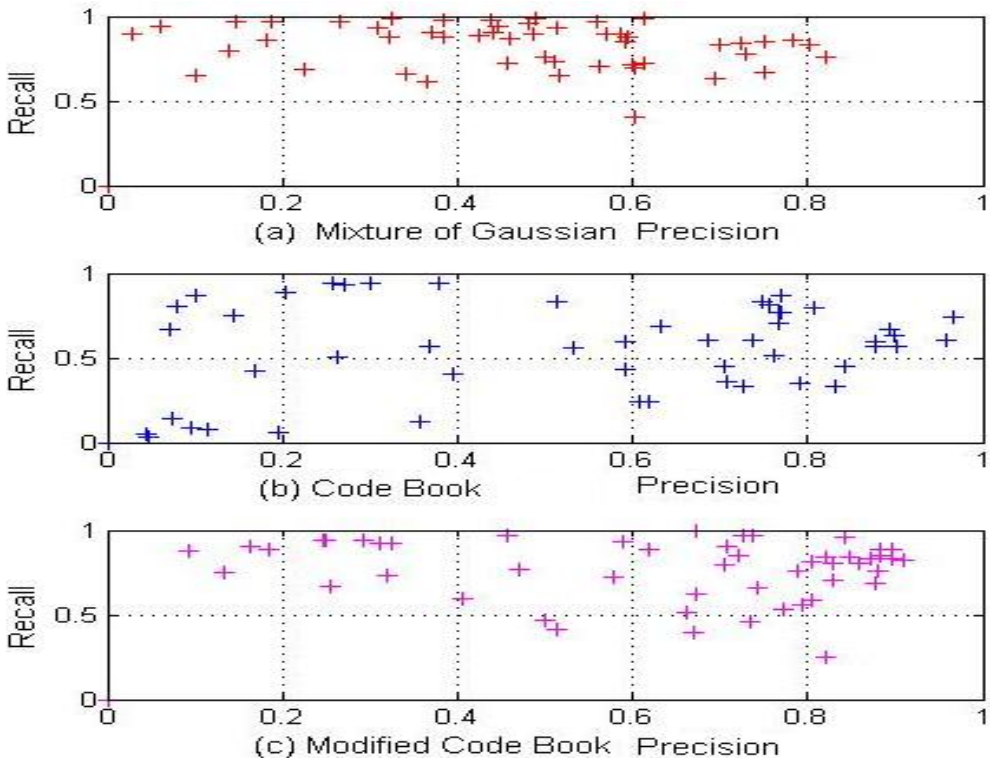

Figure 3.3: Graph between precision and recall of different segmentation techniques

We do not use synthetic data, which is easy to evaluate, but rarely a true indicator of a real scenario. We used 5 videos of duration between 15 and 55 minutes. We select these videos which have different illumination conditions and one to many foreground objects. We do not have an initial training set on an empty scene. For performance measurement, we take 50 frames of indoor and outdoor videos. These frames are selected after the first 200 frames, in order to provide enough time to model the background.
We calculate TPR, FPR, RE and PR for each of the frame. We also calculate precision quality factor F using equation 3.11 and add results of all the frames.
In figure 3.3, we plot Recall (RE) as a function of precision (PR). Each point of the graph

Method	*MOG*	*CB*	*MCB*
Precision quality factor (F)	28.28	24.15	**32.17**
FPR based error factor (E) for γ =0.95	3.02	5.34	**2.87**
Euclidean distance based on specificity	**9.07**	22.40	11.74
Percentage of error coefficient (PCC)	48.04	48.02	**48.95**
Jaccard Coefficient (JC)	20.75	17.56	**25.22**
Segmentation rate (frame/sec)	11.10	12.79	**13.48**

Table 3.1: Comparison of foreground-background segmentation evaluation results

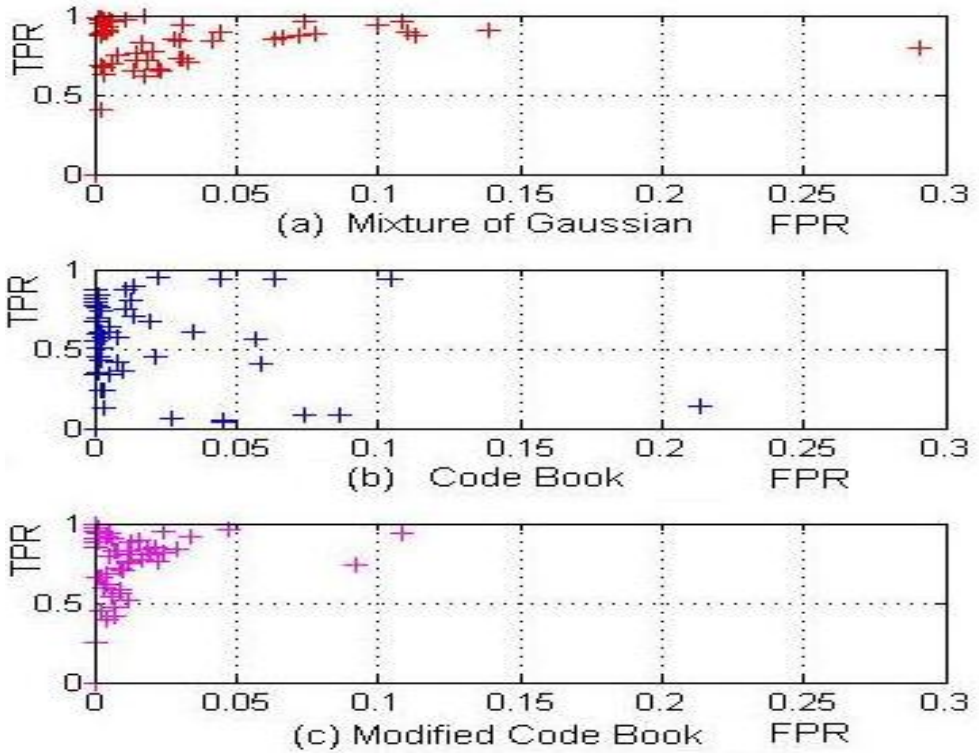

Figure 3.4: Graph between true positive and false positive rate of different segmentation techniques

represents the result for a single frame. The ideal values are PR=1 and RE=1 (the top right corner). MOG has good recall but has a worse precision as compared to CB and MCB. Only 4% of frames have precision greater than 0.8. CB has good precision, about 20%, but very low recall. MCB shows highest value i.e 32% of frames having precision more than 0.8 and a recall comparable to MOG. In table 3.1, value of F for MOG, CB and MCB are 28.28, 24.15 and 32.17. Highest number for MCB verified our claim also.

Our second series of experiments use equation 3.12. We first plot the FPR and TPR for each image frame. TPR is also known as sensitivity and FPR = (1- specificity). The figure (figure 3.4) shows FPR and TPR of image frame. The ideal values are FPR=0 and TPR=1 (the top left corner). In a real life scenario, the quantity to optimize depends upon the application. In monitoring of restricted areas, it is desirable to generate alarm/alert, whenever some object is detected. If we want to minimize the false alarms then FPR should be minimized. If we do not want to miss any foreground object, then FPR can be adjusted in order to maximize TPR. Normally FPR also increases when TPR increases ([Wang et al., 2005] and [Sheshadri and Kandaswamy, 2006]). If we increase the sensitivity of detection, then more noise and light variation will be induced in the segmented scene.

The expected result is always a trade-off between sensitivity and specificity. Some video surveillance applications many false alarms are not tolerated. For example during weekend, maybe we want to observe some regions but we ca not tolerate more than 5% false alarms. In this case we are compromising to ignore some moving objects. From figure 3.4, it is clear that if we do not want more than 5% false alarm then MCB is better than CB and MOG because it have two false positive and it has good sensitivity also. In this case someone can use the higher value of $\gamma = 0.95$ in equation 3.12. The higher value makes equation 3.12 more influenced by FPR. Value of E is higher for the algorithm that has more FPR. In table 3.1 the smallest error, for E, with $\gamma = 0.95$, is obtained by MCB algorithm, followed by MOG and CB. In prohibited or restricted area, we can afford large number of false alarms but we can not afford any object to be missed by the system. In this case, it is better to select MOG because of their sensitivity that is better than CB.
Result of MCB also gets the highest score, 48.95 for PCC and 25.22 for JC, in table 3.1. MOG is more computationally expensive than CB [Kim et al., 2005] and modified CB. The last entry of the table shows that MCB has ability to process more frames/sec compared to CB and MOG. MCB, CB and MOG process 13.48, 12.79 and 11.10 frames/sec respectively on a lower-end laptop having a Core Duo processor 1.86 GHz. Therefore, the detailed discussion in this section, verify our claim that MCB is a better foreground-background segmentation technique than CB and MOG.

3.5 Conclusion

In section 3.4 we discussed results of MOG, CB and MCB. We have included a parameter of frequency in the codebook algorithm for the accessing, deleting, matching and adding codeword in the codebook. Similarly, parameter of frequency helps to move cache codewords repeating frequently into codebook. Comparing MCB with two other techniques CB and MOG shows that MCB is able to produce better results. In short, we can summarize the results as follows:
1. In [Kim et al., 2005], it is claimed that CB works better than MOG and other techniques. In our experiments, this is the case only when few objects with less similar colors with background are present.
2. MCB introduces no new parameter to the original CB. It uses the frequency to improve CB. It does not introduce any additional computational complexity so it is still computationally less expensive than MOG. But it is able to detect moving objects more precisely than codebook and probability based mixture of Gaussians.
3. We evaluate our results in several ways. Visual (qualitative) results can be appreci-

ated on images. ROC analysis plots the performances either in terms of precision versus sensitivity or in the terms of TPR and FPR. At last, two methods for computing a unique quality factor are given. The methodologies indicate that our proposed MCB shows better results than MOG and CB. One can see from the table 3.1 that MCB over performs CB in precision quality, specificity based error factor as well as weighted Euclidean distance. In comparison with MOG, MCB obtains better precision factor, less false alarms and is more robust with variation of light intensities. Moreover, it involves less floating point calculations.

The above discussion conclude that MCB works better than CB in all the conditions. The choice between MCB and MOG depends on the application. If the precision is considered as the most important factor, then MCB is probably the best choice. If a compromise is possible on precision, shadow and false alarms but any small object should not be missed, then MOG can be a better choice. Nevertheless, like CB and MOG, MCB does not handle the case of still objects in a satisfying manner. If foreground object stops for some time, then it will also be included in the background. A more sophisticated algorithm based on object recognition might be useful to overcome this problem.

Contents

Chapter 4

Object Recognition and Tracking Using a Single Camera

Object tracking is the process of locating objects in a video sequence, when they move in the camera's field of view. Object tracking is useful to monitor object activities. We have discussed main classes of object tracking algorithms and their advantages and limitations in section 2.2 of the chapter 2. Our objective is to develop object tracking system, which can work in following challenges:

1. Re-identify a given object when it exits from a camera's FOV and re-enters from a different location.

2. The algorithm should be object apparent size invariant i.e the algorithm should track the object when it moves towards/away the locations of cameras (object apparent size invariant)

3. The algorithm should have the flexibility to be extended to multi-camera environments without losing the real time performance

In the context of above mentioned challenges, we propose a simple 1-D appearance model, called the Vertical Feature (VF), independent of the view angle and of the apparent size of objects. This descriptor provides a good compromise between very compact color models, that lose all the spatial information of tracked object's color, and traditional complex appearance models. We combine a motion model of tracked objects with our 1-D appearance model. We show the superiority of a combined model approach on traditional tracking approaches, based on object appearance or motion model.

4.1 Introduction

In general, object tracking algorithms are combination of various fields of image processing and computer vision. The object tracking performance can be increased by combining several algorithms at the cost of real time performance. In this chapter, we present our algorithm [Ilyas et al., 2010a] for real time object tracking. It is simple to implement and has good performance.

This chapter is organized as follows: section 4.2 deals with object features which are used for object recognition. In section 4.3, we discuss our simple but effective method to detect object occlusion. Section 4.4 presents the algorithm of object position prediction by using Kalman filter. In section 4.5, we integrate the steps discussed in the sections 4.2, 4.3 and 4.4 in a robust object tracking algorithm. We discuss detailed results of our object tracking algorithm and we also compare this algorithm with motion based and appearance based algorithms on standard databases in section 4.6. We combine appearance and motion model in a effective manner to increase the tracking accuracy without losing real time performance of tracking algorithms. We also examine the object re-identification performance in a single camera environment. In the final part of this chapter (section 4.7), we conclude on our object tracking algorithm performance, its limitations and related future work.

Figure 4.1 illustrates the tracking algorithm for a single camera. We use a main database to store all detected objects, their trajectories and object invariant features (1D appearance model) which are required to track an object. Each new object which enters in a camera's FOV is stored once in the database. The cache contains invariant feature VF and motion features (position, velocity) of recent objects present in the camera. In multi-camera environments, there will be one cache for each camera but a single database.

The first step of our tracking algorithm is the image segmentation and object detection. We use a background modeling technique to model and then subtract the background from sequences of images. We use the modified codebook background modeling method [Ilyas et al., 2009] presented in the section 3.2.3. The advantages of this technique are: its robustness, capabilities of object shadow removal and the possibility to control its foreground detection sensitivity. After the image segmentation into background and foreground, we remove small blobs, which are likely representing noise pixels. The remaining blobs are considered as object. Our algorithm starts by finding, if the object is under occlusion or not (the way we detect occlusions is presented in the section 4.3). If an occlusion is detected, then we only perform a Kalman filter prediction for the moving object position (section 4.4). Otherwise, for each object in the frame, the algorithm

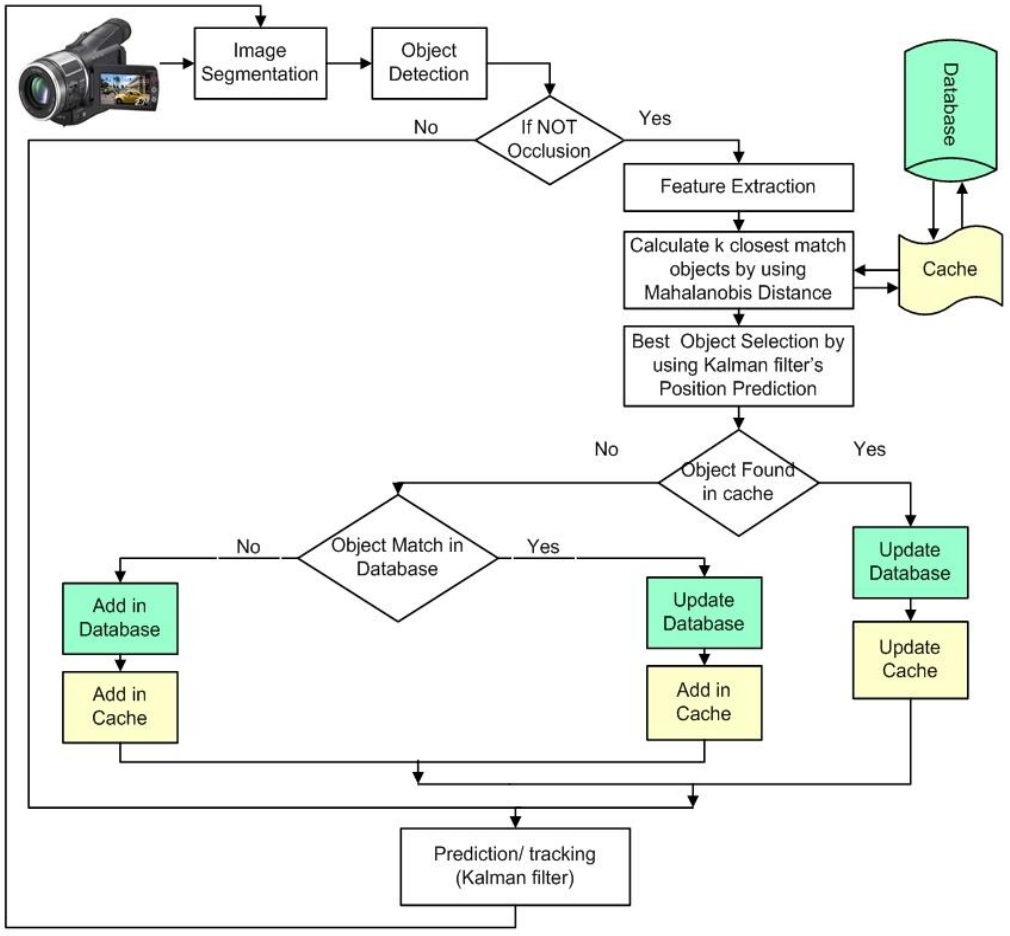

Figure 4.1: Proposed real time multi-object tracking algorithm's flow diagram

extracts several features: the Vertical Feature (VF), the current object position, its area, its predicted position (Kalman filter) and estimated velocities.

The extracted object features combine object's spatial color information and motion features (see section 4.2.1) for object matching. Mahalanobis distance is used to find k similar objects by comparing features of current object and previous frame object's cache. Each of these k objects have also their predicted position in current frame. The current object position is matched with these k object's positions. Select the one having the closest predicted position to the current object position (to a value not exceeding a threshold). If there is a new object, then a Kalman filter model is initialized for this object with initial velocity set to zero. Else, we update the Kalman filters parameters and go to the next frame. We present the details of this process in sections 4.2.2 and 4.5. Our technique is fully automatic and we do not need any off-line training.

4.2 Object Recognition

Object recognition is an essential part of object tracking. In general, more than one object are present in the camera's FOV, then without identification/recognition, we can not decide which blob can be associated with previous frame's object. An object can be recognized due to its appearance, motion or geometrical properties. In this thesis, we work on human (person) tracking.
Human recognition in a scene is a difficult job due to movement of arms, face and legs etc. In video surveillance systems, color information of objects is an important feature. But only color information, for example histogram matching techniques, without spatial information is not sufficient for objects recognition [Birchfield and Rangarajan, 2005]. The other possible solution is to use appearance models. The problem with existing 2-D appearance models is the update procedure which might be difficult when the apparent size of the object change due to perspective or when the shape changes. Our 1-D appearance model keeps partial spatial and color information. Using motion features with 1-D appearance model increases the recognition capabilities even when similar objects are present in the scene.
We consider the situations when an object enters and then exits from camera's FOV and re-enters in the FOV sometime later from another location, or enters in the FOV of some other camera. In this case, the object size, shape and view angle change. We need then, a feature which is invariant with the size, shape and view angle of object. The other requirement of our system is that it should have the ability to track objects in real time; therefore we can not use a complete/complex appearance model like ([Mittal and Davis, 2003], [Hamdoun et al., 2008], [Lowe, 1999] and [Enzweiler and Gavrila, 2009]) because these algorithms are not computationally efficient. In our experiments, we find that a color based 1-D appearance model - named vertical feature (VF), combined with motion features is sufficient to give satisfactory results [Ilyas et al., 2010a].

4.2.1 Object Features

In this section, we will discuss in details about the Vertical Feature (VF) and the motion features (object position and velocity) used for object recognition features. In the section 4.2.1.1, we discuss in details, the object appearance model, its functionality and VF vector variation with time and section 4.2.1.2 presents motion features.

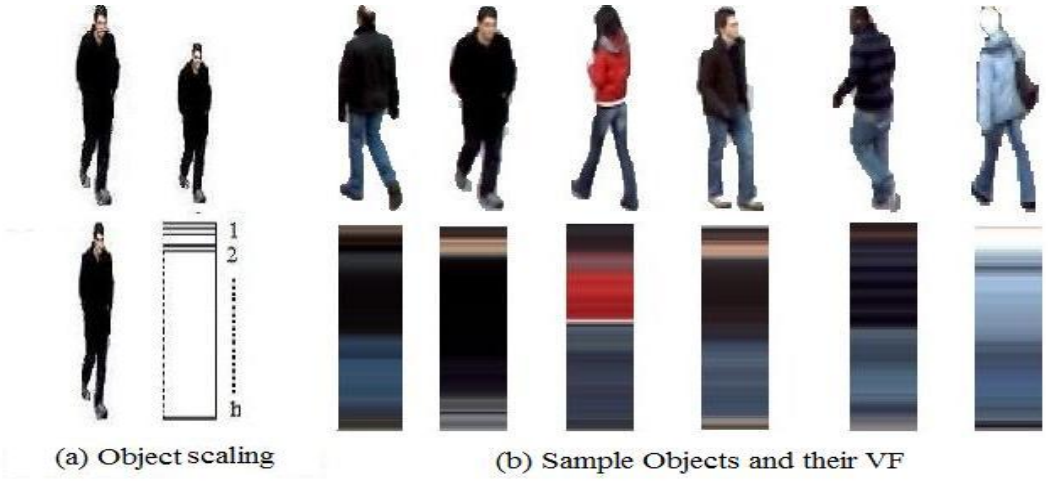

Figure 4.2: Sample objects, their VF representation and scaling

4.2.1.1 Vertical feature

1-D appearance model allows to store the object color and partial spatial information in a very compact form. The idea is to describe each object by a vector, representing its projection on the principal axis (head-foot axis) of the object, which will be considered as a vertical axis in this chapter.

Large or small object are resized or re sampled to the predefined height h (Figure 4.2.a). Objects higher than the predefined height h are re-scaled by using a Gaussian kernel. Objects smaller than h are interpolated to the height h using a bi-cubic interpolation. [Lehmann et al., 1999] discuss various image rescaling techniques and their applications in their survey report including linear, quadratic, cubic, Gaussian, etc. Figure 4.2.b shows some sample objects from different videos and their VF representation. VF preserves some vertical texture information, face and hair color. VF technique is also computationally efficient and economical in terms of memory storage.

In most video surveillance systems, videos are stored using the Quarter Video Graphics Array (QVGA) format (320 x 240). We observed in many videos that object size varies from 100 pixel (when object is near to camera location) to 20 pixels (when object is far from camera location). We decide to fix to the object height to 60 pixel (resulting from a compromise between accuracy and complexity of the descriptor).

After fixing object height to h, each value of y (y=1..h) is represented by the most representative color of the horizontal projection of this specified slice of the object. Method to compute VF is discussed in algorithm 3.

Input: image of object

Output: object vertical feature

fix the height = h; (h=constant)

foreach *segmented object of height* h' **do**
  rescale the object height h' to h
  if $h' < h$ **then**
    use a bi-cubic interpolation
  else
    use Gaussian kernel
  end
  foreach *horizontal slice* y_h; $h = 1 \rightarrow h$ **do**
    find the most representative color in row y_h of rescaled object σ ;
    $VF(y_h) = \xi\left(I(x, y_h)\right)$; $x, y_h \in \sigma$
  end
end

Algorithm 3: The Vertical feature computation algorithm

VF is a vector consisting of h elements for each color channel. The possible choices for $\xi\left(I(x, y_h)\right)$ are mean and median color value of each color channel taken individually, or a particular color value minimizing a given criterion such as the quadratic sum of the distances in RGB color space. Our experiments on object tracking shows that $\xi\left(I(x, y_h)\right)$ mean color values gives the best performance.

Sometimes, we have a risk to lose the true object color if an horizontal slice of the object has two or several very different colors. An object VF is also updated by using short past history of the VF (see equation 4.1). This vertical feature updating helps to make the object VF robust against object appearance change due to its view angle changes in camera's FOV. ζ value between 0.01 and 0.05 gives good results in most of the situations. Larger value of ζ adopt current object color sharply and very smaller value do not absorb the global intensity variation with respect to time.

$$VF^t = (1 - \zeta)VF^{t-1} + \zeta VF \tag{4.1}$$

We also tried to use horizontal texture information, the results are not improved due to the poor quality of surveillance videos as they do not contain adequate level of texture details. Our experiment on texture analysis confirmed [Porikli and Divakaran, 2003] statement, who claimed, "In general, video quality of surveillance cameras are not good enough to extract object texture information, hence this technique is not suitable for object tracking". In our experiment finally, we find that mean color gives the best com-

promise between quality of tracking and computing time. Mean color performance is better on the videos having poor video quality as compared to object median color.

4.2.1.2 Motion features

Object spatial information is also an important parameter for object recognition. Object position $P^i = \left[x^i, y^i\right]$ and velocity $V^i = \left[v_x^i, v_y^i\right]$ are commonly used in a single camera or spatially calibrated multi-cameras as object recognition features. An object position helps to recognize similar objects in a scene. Similarly, velocity component distinguish different objects if they are moving with different velocities.

4.2.2 Feature Matching

In order to match object features, we use the Mahalanobis distance. It differs from euclidean distance by its scale invariance. It also takes into account the objects color covariance in the database. Mahalanobis formula for VF is represented as

$$D\left(VF^1, VF^2\right) = \sqrt{\sum_{y=1}^{h}\sum_{ch=1}^{3}\left(\frac{VF^1(y,ch) - VF^2(y,ch)}{\sigma_{(y,ch)}}\right)^2} \tag{4.2}$$

where $\sigma_{(y,ch)}$ is the color standard deviation of each element of vector VF in the database containing T objects. Number of objects present in the cache are p. We use equation 4.2 and 4.3 to find distances D^i between features of current object O^c and previous frame's objects O^i.

$$D^i(O^c, O^i) = D\left(VF^c, VF^i\right) + D\left(P^c, P^i\right) + D\left(V^c, V^i\right) \tag{4.3}$$

In equation 4.3, $D\left(P^c, P^i\right)$ and $D\left(V^c, V^i\right)$ are also calculated using Mahalanobis distance. where as i= [1...p] and p $\leq$ T. We assume $D^1 \leq D^2 \leq .. \leq D^p$ and we select objects having distances $(D^1, D^2, .., D^k) \leq D_{ref}$. D_{ref} is a similarity threshold. Only objects which are similar in their color appearance are selected.

Figure 4.6 in section 4.6 illustrates that there are some similar objects present in a scene. If we find only one matching object then it may be a false match due to the possibility of a poor image segmentation and change of object view angle. If current object is not present in previous frame then D^f is selected by finding the closest match f of current object c from database of T objects using vertical feature only.

$$f = argmin\left(D\left(VF^c, VF^j\right)\right); j = [1..T] \tag{4.4}$$

The equation 4.4 is useful if an object exits and re-enters in the same or another camera's FOV. In this situation, motion features and Kalman prediction are not used because the object may exit from one side and enters in the camera's FOV from some other side. Similarly, the object's area may be very different when the object re-enters from a different entry point in a camera's FOV. That is why only the Vertical Feature (VF) is used when an object re-enters in a scene as VF is invariant in terms of view angle, apparent size and position.

4.3 Occlusion detection

The performance of object tracking algorithm is affected if the algorithm is unable to handle object-object occlusion. Indeed, a segmented object might correspond to two or more merged objects. Therefore the calculated features do not correspond to a single object and there is a risk to alter several objects's features. To avoid this problem, our algorithm starts by detecting an occlusion. If occlusion is detected then the objects under occlusion are tracked using Kalman estimation only, and object features are not calculated and updated in this situation.
Recently, many researchers used overlapping field of view multi-cameras for object tracking under occlusion [Mittal and Davis, 2003] and [Khan and Shah, 2009]. But in most of real world scenarios, it is difficult to install and calibrate overlapped multi-camera surveillance systems for large area video surveillance (e.g. campuses, railway stations, subways, etc.). Our aim is to track object using single or multiple non-overlapped field of view cameras.
Relevant work on object detection and tracking during an occlusion are explained in [Vazquez et al., 2007] and [Lee and Ko, 2004]. The algorithm ([Lee and Ko, 2004]) detects false occlusions due to non uniform motion of objects. The technique presented in [Vazquez et al., 2007], fails if a new object enters in the camera's FOV and two or more objects start occlusion in the same frame. We propose a simple and efficient algorithm for occlusion detection, based on probabilistic as well as deterministic components. These features are object's predicted position P_p^c by using a Kalman filter and its area A^c. In order to detect occlusions, we verify if:

1. Predicted positions $P_p^{c_1}, P_p^{c_2}, ..., P_p^{c_d}$ of previous frame's objects overlap in a region of current frame.

2. The area A^c of a current object is significantly greater than maximal individual areas of previous frame's objects in that region.

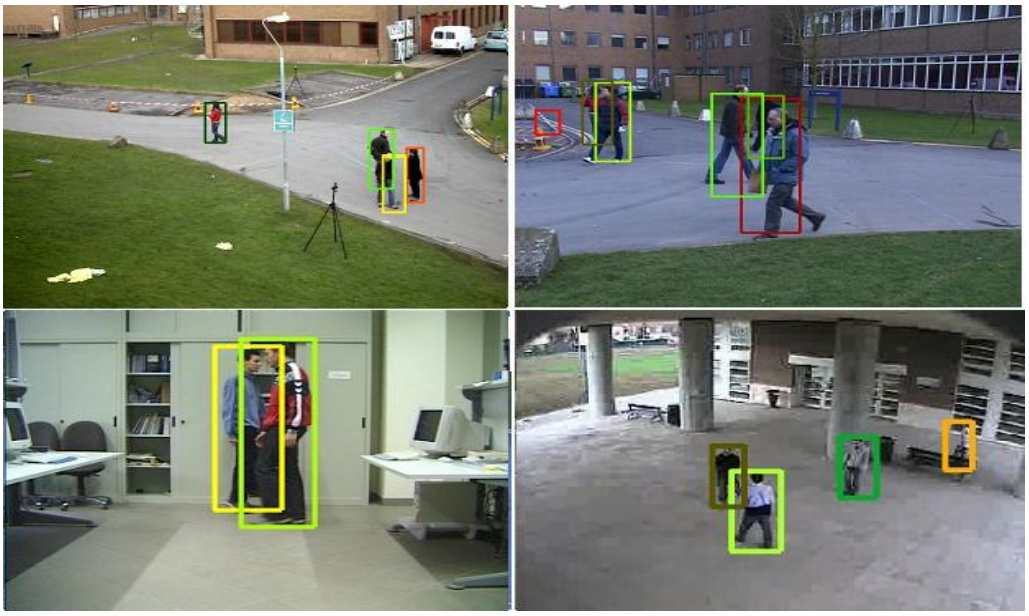

Figure 4.3: Objects Occlusion detection in different video sequences

If conditions 1 and 2 are true, then occlusion is declared. During occlusion, the object's features (vertical feature, area, position and velocity) are neither updated nor stored in the cache. Objects are tracked using their previously estimated velocities and positions. Some sample frames from different videos with objects under occlusion are presented in figure 4.3. The top-right image of figure 4.3 is the most challenging situation: three moving objects are under occlusion and all of them are moving in different directions. In most of the cases, our algorithm is able to successfully detect occlusions of two or more objects (see table 4.2 for more details).

4.4 Motion Model

Object position prediction helps to detect some object-object occlusion. We use the Kalman filter [Kalman, 1960] to estimate the most likely position of an object and to detect object-object occlusion in the next frame. The Kalman filter is an efficient linear and recursive filter. It needs a model to make the relation between input and output data. We will discuss this model in the next subsection.

4.4.1 Kalman Filter Model

The Newtonian equations of motion allow to find an object position in next frame. We model our states as 4-D vector by using an object position (x^i, y^i) and its velocity (v_x^i, v_y^i)

at discrete time t using vector $X_t = \left[x_t^i,\, v_{xt}^i,\, y_t^i,\, v_{yt}^i \right]^t$. There are two sets of equations. The first set consist of the process equations, which are used as an input of the Kalman filter and the second set of equations are called the measurement equations. The detailed discussion about the Kalman filter algorithm and its applications is presented in the technical report [Welch and Bishop, 1995]. Dynamic process equations of the system are described by the following non-linear Newtonian equations:

$$x_{t+1}^i = x_t^i + v_{xt}^i \Delta t + \frac{1}{2} a_{xt}^i \Delta t^2 \tag{4.5}$$

$$v_{x(t+1)}^i = v_{xt}^i + a_{xt}^i \Delta t \tag{4.6}$$

$$y_{t+1}^i = y_t^i + v_{yt}^i \Delta t + \frac{1}{2} a_{yt}^i \Delta t^2 \tag{4.7}$$

$$v_{y(t+1)}^i = v_{yt}^i + a_{yt}^i \Delta t \tag{4.8}$$

Equations 4.5, 4.6, 4.7 and 4.8 can be combined into a matrix form either like equation 4.9 or more descriptive form (equation 4.10) :

$$X_{t+1} = \begin{bmatrix} x_t^i + v_{xt}^i \Delta t + \frac{a_{xt}^i}{2} \Delta t^2 \\ v_{xt}^i + a_{xt}^i \Delta t \\ y_t^i + v_{yt}^i \Delta t + \frac{a_{yt}^i}{2} \Delta t^2 \\ v_{yt}^i + a_{yt}^i \Delta t \end{bmatrix} \tag{4.9}$$

$$\underbrace{\begin{bmatrix} x_{t+1}^i \\ v_{x(t+1)}^i \\ y_{t+1}^i \\ v_{y(t+1)}^i \end{bmatrix}}_{X_{t+1}} = \underbrace{\begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{A_{t+1,t}} \times \underbrace{\begin{bmatrix} x_t^i \\ v_{xt}^i \\ y_t^i \\ v_{yt}^i \end{bmatrix}}_{X_t} + \Delta t \underbrace{\begin{bmatrix} \frac{1}{2} a_{xt}^i \Delta t \\ a_{xt}^i \\ \frac{1}{2} a_{yt}^i \Delta t \\ a_{yt}^i \end{bmatrix}}_{W_t} \tag{4.10}$$

Due to the non-linearity of an object motion, the acceleration $\left(a_{xt}^i, a_{yt}^i \right)$, is modeled by a noise process W_t. Δt is the time difference between two consecutive frames. Similarly $A_{t+1,t}$ is the state transition matrix which links next position to current object's position and velocity. The matrix form of equation 4.9 is:

$$X_{t+1} = A_{t+1,t}\, X_t + W_t \tag{4.11}$$

Similarly, measurement equations are used to measure object estimated position which are used for object prediction in the next frame. The measurement equations can be written as:

$$z^i_{1\,t} = x^i_t + u_{1\,t} \tag{4.12}$$

$$z^i_{2\,t} = y^i_t + u_{2\,t} \tag{4.13}$$

where as $z^i_{1\,t}$ and $z^i_{2\,t}$ are the measurement of x and y object positions. $u_{1\,t}$ and $u_{2\,t}$ are the measurement error. Equations 4.12 and 4.13 can be combined into matrix form

$$\underbrace{\begin{bmatrix} z^i_{1\,t} \\ z^i_{2\,t} \end{bmatrix}}_{Z_t} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{H} \times \underbrace{\begin{bmatrix} x^i_t \\ v^i_{xt} \\ y^i_t \\ v^i_{yt} \end{bmatrix}}_{X_t} + \underbrace{\begin{bmatrix} u_{1\,t} \\ u_{2\,t} \end{bmatrix}}_{U_t} \tag{4.14}$$

The equation 4.14 can be written in a simplified vectorial form as:

$$Z_t = H\,X_t + U_t \tag{4.15}$$

4.4.2 Kalman Algorithm

The Kalman filter estimates a process by using a prediction and then an actual measurement of a process. The filter estimation noise is then minimized to get a better prediction in a next state of process. The Kalman filter equations can be categorized into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and the prior error covariance estimation matrix for the next time step. The measurement update equations are responsible for the feedback i.e. minimizing the error between an object actual and predicted positions.

The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations. The whole process of a Kalman filter is explained in the algorithm 4. The Kalman filter algorithm starts with parameters initialization for each object. The matrices $X_{0,}, P, Q, R, A$ and H are set to their initial predefined value. The first time, object x and y position estimation and prior error covariance matrix also calculated in initialization. In fact, position estimation and the prior error covariance matrix calculation is required in the initialization

Input: Take object current position and velocity
Output: Predict object position for next frame

Initialization: **do**
Initialize matrices $X_{0,}$, P, Q, R, A and H to predefined values
$\hat{\bar{X}}_t = A\hat{X}_{t-1}$
$\bar{P}_t = AP_{t-1}A^T + Q$
end
foreach *frame time t* **do**
  foreach *Kalman filter measurement update equation* **do**
    $K_t = P_tH(HP_tH + R)^{-1}$;
    $\hat{X}_t = \hat{\bar{X}}_t + K_t(Z_t - H_k\hat{\bar{X}}_t)$;
    $P_t = \bar{P}_t - K_tH\bar{P}_t$
  end
  foreach *Kalman filter process update and prediction equations* **do**
    $\hat{\bar{X}}_{t+1} = A\hat{X}_t$;
    $\bar{P}_t = AP_{t-1}A^T + Q$
  end
end

Algorithm 4: Kalman filter computation algorithm

process. After the initialization, the algorithm enters into the second step, which is a measurement correction and update. In the last step of the algorithm, the filter process is updated and the object position is predicted for the next frame.
$\hat{\bar{X}}_{t+1}$ is the estimated/predicted state of the process. $\hat{X}_t$ is the corrected state of the process. P_t is posterior error covariance matrix. $\bar{P}_t = E[\hat{e}_t\hat{e}_t^T]$ is the prior error covariance matrix between estimated and actual state of the process. $Q = E[W_t\, W_t^T]$ is the covariance matrix of noise process and $R = E[U_t\, U_t^T]$ is the covariance matrix of the measurement error.

4.4.3 Kalman Filter Tunning and Results

In the previous section, we discussed the algorithm of the Kalman filter. In this part, we will evaluate the performance of Kalman filter and its parameter selection. In our implementation, parameters P and Q are fixed and all other variables are calculated dynamically.
The parameter $\bar{P}_t$ can be given any value, because it is automatically updated after each frame. The values, which we use in our experiment are shown below:

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}, H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}, \bar{P}_t = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}.$$

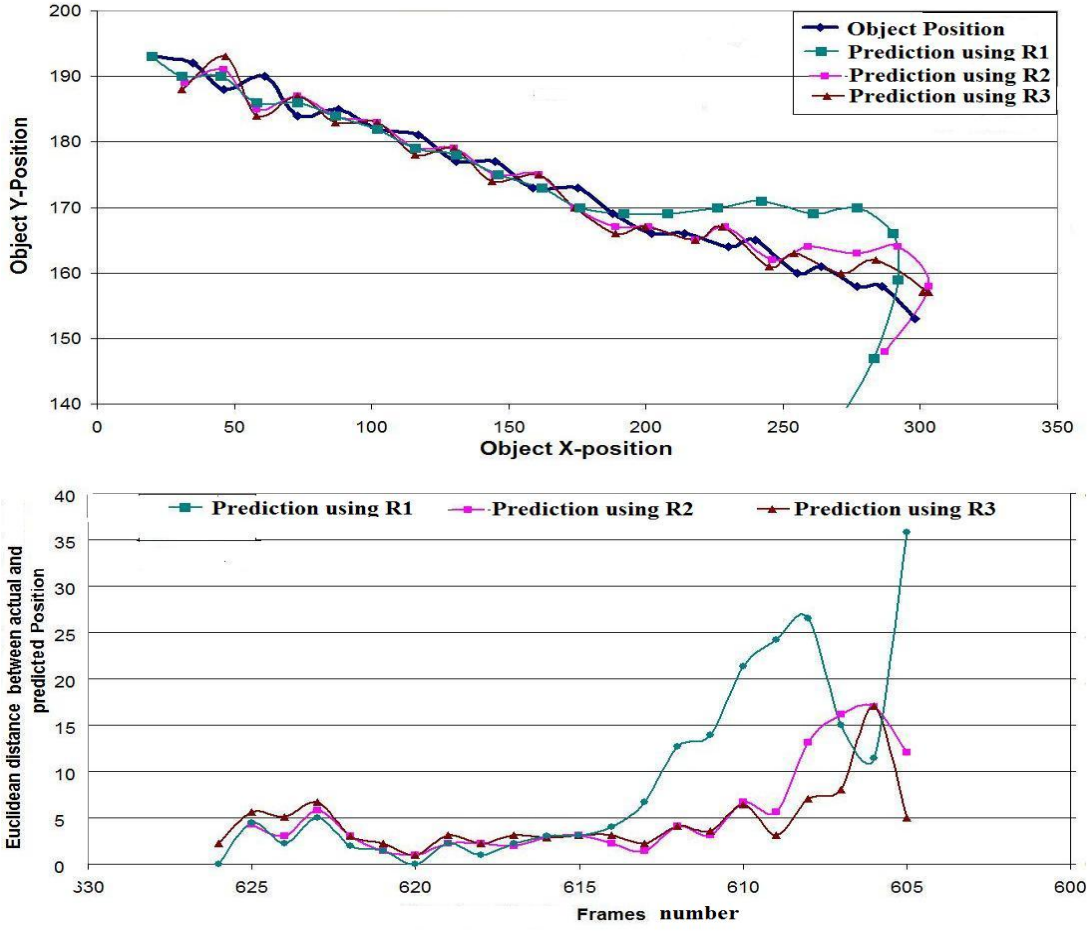

Figure 4.4: Object trajectory and their Kalman filter position projection in a VISOR video sequence. a) shows the object position and Kalman filter position estimation using different matrices (R1, R2 and R3). b) Euclidean distance between object actual position and predicted positions.

We perform the experiments by using three different values of the matrices R. The different matrices which we use are:

$$R_1 = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, R_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } R_3 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}.$$

Figure 4.4.a, shows that the object moves in a zigzag fashion. It is due to the fact that the object's position (center of gravity) changes periodically during the walk, because of the individual movement of legs and arms. In figure 4.4.b, the Euclidean distance between the Kalman filter prediction and the ground truth is less than 10 pixels after 2-3 iterations. In our experiment, we find that R_1 adopt object motion variation very slow and R_3 adopt objects motion variation very quickly. Matrix R is used to calculate Kalman filter gain K. From the algorithm 4, it is evident that higher value of matrix produce small gain and smaller value of matrix R, produce large gain. The Kalman filter

gain K is used to model object motion (see algorithm 4). Kalman filter gain K can also be considered as Kalman filter learning rate. Higher learning rate gives more weight to recent history of object motion pattern and smaller learning rate take more time to model the background. We find in our experiment that R_2 is a suitable option due to a better prediction of human motion pattern and gives better object position prediction.

4.5 Object Tracking Algorithm

In this section, we propose an algorithm, which combine our 1-D appearance model (VF) presented in section 4.2 with motion model discussed in section 4.4 for robust tracking. We maintain a cache (short time memory) that is periodically synchronized with the main database. The current frame object O^c is matched with the previous frames objects in cache using equation 4.3. This equation compares the current object's VF and motion features with previous frames object's features and gives k objects having similar features with current object O^c. We find predicted positions $P_p^1, P_p^2 ... P_p^k$ for these k objects. Find euclidean distance between current object's position P^c and predicted positions $P_p^1, P_p^2 ... P_p^k$. Select object O^d, who has minimum Euclidean distance $De_{min}(O^c, O^d)$ with current object O^c. If $De_{min}(O^c, O^d)$ is less than threshold De_{th}, then object's features (VF, object position, velocity and area) are updated in cache.

If an object was not found in the cache (new object in the frame), we try to match this new object with one of the objects already stored in the database using VF only using equation 4.4. If a matching object is found we update this object's features and add the object in cache. Otherwise, we create a new label for this object in the database and add object's features in cache. Steps involved in this process are explained in the algorithm 5.

In section 6.1 of chapter 6, we discuss in detail to use cache and the database in an optimized manner to minimize object matching time and maximize object matching performance.

$De_{min}(O^c, O^d)$ allows to associate current object to the most probable previous frame's object O^d. De_{th} is the maximum allowed object position error between current object O^c and previous frame object O^d position ($De_{min}(O^c, O^d) < D_{th}$). Figure 4.4.b shows that initially Euclidean distance between object actual and predicted position in video frames is high (between frames number 605 and 610). This Euclidean distance between object actual and predicted positions decrease between 5 and 10 pixels after 4 to 5 frames (after frame number 610). In some cases, principally due to image segmentation errors, the current object position error, between predicted and actual position may increase. De_{th}

value between 20 and 30 pixels is suitable for most of the situations.

Input: Objects from sequences of images
Output: Recognize and track the objects

foreach *current object* O^c *of frame* t **do**
  calculate $D^i(O^c, O^i)$ using equation 4.3 from cache; $1 \leq i \leq p$;
  sort $D^i(O^c, O^i)$ in ascending order ($D^1 \leq D^2 \leq .. \leq D^p$) ;
  find objects $O^1 ... O^k$ having $D(O^c, O^k) < D_{ref}$ from cache; $1 \leq k \leq p$;
  get predicted positions $P_p^1, P_p^2 ... P_p^k$ for these k objects;
  find minimum euclidean distance $De_{min}(O^c, O^d)$ between object's O^c position P^c and predicted positions $P_p^1, P_p^2 ... P_p^k$;
  if $De_{min}(O^c, O^d) < De_{th}$ **then**
    update match object's VF, position, velocity and area in cache ;
  else
    find the closest match object O^f for current object O^c from objects database using equation 4.4;
    if $D(O^c, O^f) < D_{ref}$ **then**
      update object O^f parameters in database and add its parameters in cache ;
    else
      create new object in database ;
      store its parameters in database and add its parameters in cache ;
    end
  end
end

Algorithm 5: Object tracking computation algorithm

4.6 Tracking Results

In this section, we discuss the performance of our proposed object tracking algorithm. We evaluate the results on the well known PETS[1], CAVIAR[2] and VISOR[3] benchmark data set. We compare our algorithm with two of the most used families of methods presented in the literature. The first is based on an object motion estimation ([Kalman, 1960]) and the second is an appearance and color based mean shift algorithm ([Comaniciu et al., 2000]).

Figure 4.5 shows images from three well known databases. The CAVIAR database has a small number of objects present in each frame but objects have much similarities with the background. The video sequences are compressed but their quality is acceptable in general. The object-object occlusions are quite rare. The VISOR database have more

[1] *http://www.cvg.rdg.ac.uk/PETS2009/a.html*
[2] *http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1*
[3] *http://www.openvisor.org/video_categories.asp*

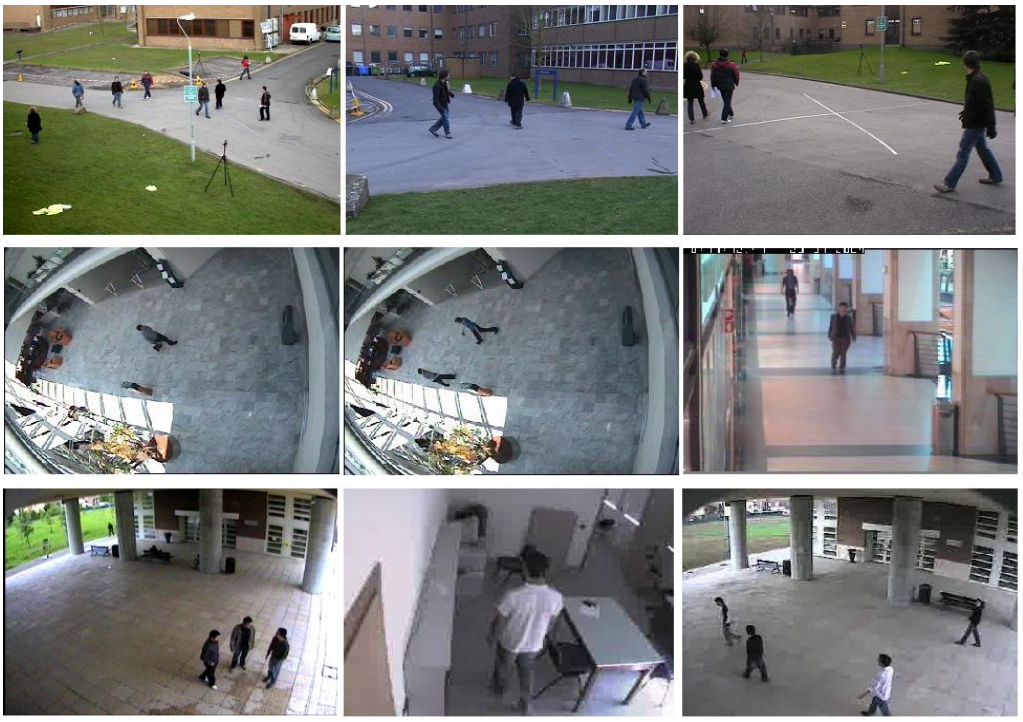

Figure 4.5: Samples Images from database of PETS, Caviar and Visor are shown in row 1, 2 and 3 respectively.

challenging sequences than the CAVIAR database. In some videos (like the right and left bottom images in figure 4.5) similar objects are present and the poor video quality makes it nearly impossible for a human operator to recognize objects correctly. Many entry points are possible and similar objects are present most of the time in the video sequence.

The PETS database has also many challenges. There are many object-object occlusions. Many times severals objects are under occlusions at the same time. The video quality of PETS is good and presents the advantage of having rich color information. Similar to VISOR, PETS also has many entry points in the camera's FOV. Many similar objects enter, exit and re-enter in the camera's FOV. Object's size also changes significantly when the object comes from one to another end of camera's FOV. Some persons deliberately change their direction and velocities in an unpredictable manner.

In short, these databases have many challenges like shadows, occlusions, irregular luminosity, object's shape and size evolution, objects re-entrance and low video quality. Therefore these three databases give a good environment to test object tracking algorithms.

We consider 7410 frames from 15 videos. In each frame, at least one object is present. We compare each object's position in the video with ground truth and distinguish them

Figure 4.6: Real time multi objects recognition and tracking

by drawing a rectangle window around them using different colors. Figure 4.6 illustrates a typical situation, when the algorithm track several persons wearing similar colors clothes and frequent occlusions.

Table 4.1 shows that the Kalman filter gives good results on the CAVIAR dataset due to a small number of object-object occlusions and interactions. Motion-based technique can track the object successfully in this situation. The mean-shift technique is unable to give good results due to object-object and object-background color similarity. The proposed algorithm uses together color, motion and spatial information that is why it supper-seeds both Kalman and mean shift algorithms. VISOR videos have small and large object sizes, similar objects, low and high video quality, uneven luminosity in video frames. The mean shift gives really poor results and Kalman filter fails to track

Dataset	Kalman	Mean shift	Our Algorithm
CAVIAR	93.27%	85.64%	96.72%
VISOR	81.27%	64.70%	89.02%
PETS	73.18%	67.04%	91.35%
Overall	81.57%	70.86%	91.97%
TR (fps)	85.63	7.18	39.32

Table 4.1: Comparison table of different tracking techniques on standard datasets and number of processed frames/sec (fps), when tracking same objects in the images sequence (TR)

Parameter of Recognition	*Result*
Total number of frames in all the videos	7410
Number of persons present in all the frames	16011
Number of persons recognized in all the scenes	14725
Persons not recognized in all the frames	1286
Percentage of successful recognition	91.96%
Correctly re-identification after re-entrance in a scene	56
False re-identification after re-entrance in a scene	9
Percentage of successful re-identification	86.00%
Correctly person recognize after an occlusion	246
False person recognition after an occlusion	16
Percentage of successful recognition after occlusion	94.00%
True person tracking during an occlusion	1700
False person tracking during an occlusion	523

Table 4.2: Detailed result of our proposed human tracking algorithm on the PETS, VISOR and CAVIAR datasets

objects due to occlusions. Kalman filter is unable to distinguish if the same object or if a different object enters the scene as it only uses motion features. Our algorithm gives satisfactory results except for videos having a very low quality.

For the PETS 2009 database, our algorithm is able to track the objects under above defined cases. Our algorithm gets 91.35% on the PETS database which is significantly better than Kalman (73.18%) and mean shift (67.04%) algorithms. In the table 4.1, "overall" give the recognition percentage of total objects present in all the frames of of dataset CAVIAR, VISOR and PETS. For example, table 4.2 shows that we have total 16011 number of persons in all frames and we have correctly recognize 14725. Which gives 91.97% of succes rate. In table 4.1, Kalman has 81.57% and mean shift has 70.86% success rate. The last entry of table 4.1 is the Tracking Rate (TR). We track 15-20 objects in image sequences on a laptop having a Core Duo processor (T2350) with speed 1.86 GHz clock rate, where as video frames size is 320 x 240. Our algorithm is faster than mean shift but slower than Kalman filter because Kalman filter is also integrated in our implementation. The real time and accurate object tracking performance of our algorithm on a low-end computer system shows that our algorithm has potential to apply for multi camera video surveillance systems.

In table 4.2, we only present the detailed results of our algorithm because the Kalman and mean shift algorithm do not have the ability to track the object during occlusions and to recognize an object, when it exits and re-enters the scene. Table 4.2 recalls that

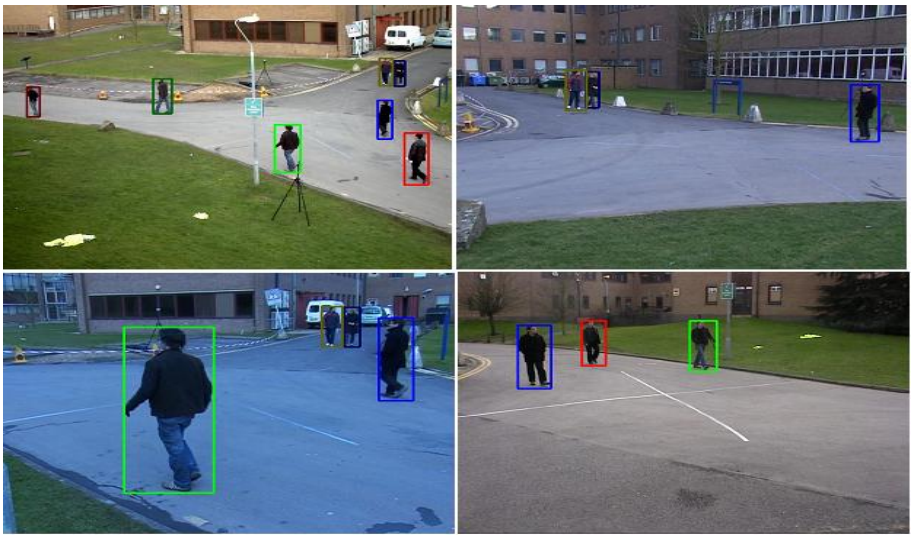

Figure 4.7: Object tracking in a multi-camera environment.

91.97% object tracking rate and 86% object re-identification rate. Our algorithm recognize objects correctly (94%) after occlusion. These successful results illustrate that our proposed algorithm can be easily applied in multi-camera environments.
In the third series of experiments, we applied our object tracking algorithm on PETS multi-camera environments. In this series of experiments, we applied our algorithm on one (top left) camera and find object features for each object. We use these features for object tracking in different cameras by considering each camera as an independent unit. In figure 4.7, identical objects are represented by a bounding rectangle having the same color in different cameras.

There are many possible entry and exit points in each camera's FOV. Objects enter in the scene in a random sequence in each camera from different paths. We observe that object color appearance in multi-camera environment degrade our algorithm performance. We use an object appearance feature (VF) only when an object exits from camera's FOV and re-enters in the same camera or another camera's FOV. If an object's color appearance in one camera is very different from its appearance in another camera, then the object recognition becomes difficult. We will discuss the camera color calibration in next chapter, which improves the object tracking performance in a multi-camera environment.

4.7 Conclusion

We proposed a real time human tracking algorithm using probabilistic and deterministic models to increase the object tracking accuracy. We also proposed a simple 1-D appear-

ance model (VF) and combined it with motion based features for object recognition in a single camera. Results in section 4.6, verified our claim that the proposed algorithm's results are better than motion based or color based models only. We proposed a simple method for object-object occlusion detection which has proven to give satisfactory results. Our algorithm is simple to implement and fast. In general, the algorithm gives satisfactory results, but there are situations in which it fails: for example, when the object's height is smaller than 20 pixels or in the case of very low video quality. Finally, occlusion can not be detected if objects enter in a scene in group.

Next chapter will present an inter-camera color calibration to improve the recognition performance in mixed in-door out-door environments, where significant changes of brightness can be problematic. We will present the extension of our object tracking algorithm to non-overlapping multi-camera environments in chapter 6.

Contents

Chapter 5

Camera Color Calibration for Multi-Camera Environments

An object appearance may be very different in multi-camera systems. There are many possible reasons for it. Some important parameters influencing object's color appearance are the camera gain, the focal length, the aperture size, the illumination conditions and the scene geometry. Cameras can produce different colors, even using the same type of camera [Ilie and G.Welch., 2005]. We are therefore interested in the problem of the cameras color calibration in order to make similar the descriptors of a same object, seen by different cameras in the system. A possible solution to overcome this problem is to calibrate the cameras color space. The camera's color can be calibrated by transforming one camera color information to the color information of another camera. These approaches estimate the Brightness Transfer Functions (BTF) by measuring the response of each camera using known objects or image scene.

The nature of the algorithms used for overlapping or non-overlapping Field of View (FOV) cameras calibration are slightly different from each other. We compare existing methods Cross Correlation Matrix (CCM) and Inverted Cumulative Histogram (ICH) method used for overlapping FOV cameras. Similarly, for non-overlapping FOV cameras, we discuss Mean BTF (MBTF) and on Cumulative BTF (CBTF) of their color histograms, and show the weaknesses of these approaches when some colors are not enough represented in the objects used for calibration. We propose an alternative (MCBTF) algorithm and we show its superiority over existing methods. In the next section, we present general introduction and some related methods of inter-camera color calibration in overlapping and non-overlapping FOV cameras.

Figure 5.1: Human appearance in a non-overlapping camera environment. The columns 1, 2 and 3 are representing the images from the cameras 1, 2 and 3 respectively

5.1 Inter-Camera Color Calibration

Inter-camera color calibration maintains object color appearance similar in a multi-camera environment. Due to non-similar object color appearance in different cameras becomes the reason of false object recognition and tracking. Our proposed 1-D appearance model for object tracking and re-identification uses object's color information. During our experiments, we observed that object recognition performance in a multi-camera environment is poor without camera color calibration. We show the advantages of camera color calibration for object re-identification in the section 6.2 of chapter 6.
Many researches, ([Javed et al., 2008], [Porikli and Divakaran, 2003], [Prosser et al., 2008] and [Orazio et al., 2009]) also emphasize the importance of camera calibration. They show that object tracking performance is significantly improved if camera colors are calibrated.
Figure 5.1 shows some images from non-overlapping camera environment. It is evident from the figure, that a human color appearance in these cameras (C_1, C_2 and C_3) is not same. Particularly, a human appearance in C_2 is a significantly different than the human appearances in C_1 and C_3. If no color calibration technique is applied, then an object could not be re-identified when it exits from the FOV of the camera C_1 or C_3 and enters in the FOV of camera C_2.

In this chapter, we discuss the camera color calibration methods for overlapping and non-overlapping FOV of cameras. We also explain our proposed algorithm ([Ilyas et al., 2010b]) for inter-camera color calibration for non-overlapping camera environment.

This chapter is organized as follows: in section 5.2, we explain two methods: cross-correlation matrix (CCM) and inverted cumulative histogram (ICH) method. These methods are frequently used to calibrate the camera's colors when they are filming the same scene or partially overlapping scenes. The section 5.3 deals with non-overlapping multi-camera calibration method. The BTF methods (5.3.1 and 5.3.2) for non-overlapping camera environment are existing methods. The section 5.3.3 explains the proposed camera calibration technique. In section 5.4, we present the results of color calibration for overlapping and non-overlapping FOV cameras. In the first part, we will evaluate the color calibration performance for overlapping FOV multi-cameras. In the second part, we will discuss the non-overlapping camera's algorithm results. The section 5.5 concludes the chapter and presents some future directions to improve the multi-camera color calibration performance.

5.2 Camera Color Calibration Methods in Overlapping Cameras Environments

In the previous section, we discussed the importance of camera color calibration. There are several methods which are used to adjust cameras colors. In general, one camera is considered as a reference and all the others camera's colors are calibrated according to the reference camera. In these techniques, images from all the cameras are taken, and BTF between reference and other cameras are calculated. This BTF is used to calibrate the cameras colors.

The figure 5.2 shows the general framework of color correction of camera C_j using the color information of camera C_i. The camera C_i's colors are used as reference colors and the other cameras colors are corrected using BTF. The assumption is made that both cameras are filming the same scene or object. In this method, image histograms of camera C_i and C_j are used to calculate BTF. After finding the BTF, each pixel color of camera images C_j is replaced by the other color value using the BTF. We will explain in following sub-sections 5.2.1 and 5.2.1, how to calculate and how to use BTF to calibrate cameras.

In this chapter, for simplicity we will discuss histograms instead of RGB histograms. But we use the same algorithm for calculating camera color calibration for all color channels separately.

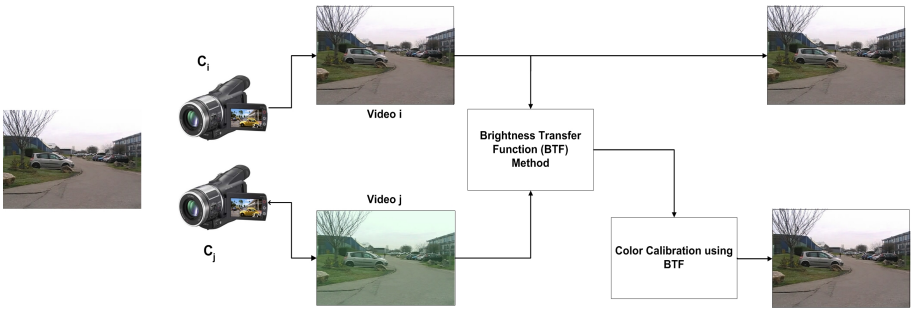

Figure 5.2: Basic block diagram of camera color calibration. The camera C_j colors are corrected using color information of camera C_i

5.2.1 Color Calibration using Cross Correlation Matrix Method

[Porikli, 2003] proposed a color calibration using Cross-Correlation Matrix (CCM) method. They compute inter-camera color compensation functions that transfer the histogram of one camera to another camera. They use a non-parametric method to compute BTF.
They calculate normalized histograms H_i and H_j for cameras C_i and C_j images. H_i and H_j consist of $B_1, ...B_m, ..., B_M$ and $B_1, ...B_n, ..., B_N$ bins respectively. They compute the correlation matrix C_{MN} between these normalized histograms using equation 5.1

$$C_{MN} = \sum_{m=1}^{M} \sum_{n=1}^{N} \left| H_i(B_m) - H_j(B_n) \right| \tag{5.1}$$

The cross-correlation matrix C_{MN} can be represented as:

$$C_{MN} = \begin{bmatrix} c_{11} & c_{12} & ... & c_{1N} \\ c_{21} & c_{22} & ... & c_{2N} \\ . & & . & . \\ c_{M1} & c_{M2} & ... & c_{MN} \end{bmatrix} \tag{5.2}$$

Each element c_{mn} is a positive real number. The minimum-cost path is the curve having the minimum value from the starting point C11 to the end point C_{MN}. This minimum path curve is then projected to matrix diagonal elements which give the cost function $\gamma_{i,j}$. This cost adjusts the camera C_j color.
Figure 5.3 illustrate how cost function is calculated by using image histograms. In the figure, $\varphi = tan^{-1}\left(\frac{M}{N}\right) - tan^{-1}\left(\frac{B_i}{B_j}\right)$ and both histograms have equal bins (M=N), i.e $tan^{-1}\left(\frac{M}{N}\right) = \frac{\pi}{4}$. The magnitude of projection ρ at point μ_ℓ becomes
$\rho = |\mu_\ell| .cos\varphi = \sqrt{B_i^2 + B_j^2} cos\left(\frac{\pi}{4} - tan^{-1}\left(\frac{B_i}{B_j}\right)\right) = \frac{B_i + B_j}{\sqrt{2}}$
The cost-function $\gamma_{i,j}^2$, can be computed by applying Pythagoras theorem on figure 5.3 :
$\gamma_{i,j}^2 = -\rho^2 + (B_i + B_j)^2 = -\frac{1}{2}(B_i + B_j)^2 + B_i^2 + B_j^2$

$$\gamma_{i,j}^2 = -\frac{1}{2}\left(B_i^2 + B_j^2\right) - B_iB_j + B_i^2 + B_j^2 = \frac{1}{2}(B_i - B_j)^2$$

$$\gamma_{i,j} = \frac{1}{\sqrt{2}}(B_i - B_j) \tag{5.3}$$

The equation 5.3 shows that the cost function $\gamma_{i,j}$ is proportional to the difference be-

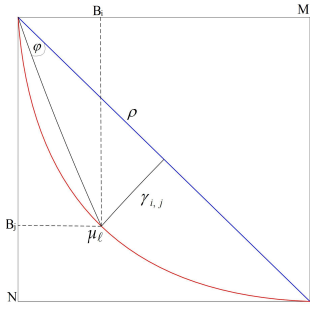

Figure 5.3: The relationship between the minimum cost path and the function $\gamma_{i,j}$

tween match bin B_i of histogram H_i and B_j of histogram H_j. $\gamma_{i,j}$ =0; If both cameras have same colors and similarly $\gamma_{i,j} > 0$; if $B_i > B_j$. In brief, whenever minimum cost path curve is in left side from the diagonal matrix elements, then $\gamma_{i,j}$ is positive. $\gamma_{i,j}$ is negative when curve is in the right side of the diagonal matrix elements because in this case, $B_i < B_j$.

[Porikli and Divakaran, 2003], use the dynamic programming to find the minimum-cost path. The dynamic programming approach is useful to solve sequential or multistage decision problems [Keeney and Raiffa, 1976]. In color calibration problem, dynamic programming is used to find the camera C_i and C_j color alignment. The algorithm of camera calibration is presented in algorithm 6.

The figure 5.4 shows images of the same person from camera C_i and C_j. The camera C_i

Input: Take a single image from camera C_i and C_j
Output: Camera C_j colors are calibrated

Compute normalized histogram H_i and H_j
Compute the correlation matrix as:
$C_{MM} = \sum_{m=1}^{M}\sum_{n=1}^{M}\left|H_i(B_m) - H_j(B_n)\right|$
Find the minimum cost path from $C_{11} \rightarrow C_{MM}$ using dynamic programming
Compute the model function $\gamma_{ij} = \frac{1}{\sqrt{2}}(B_i - B_j)$
Correct C_j image color using equation written bellow:
$Im_j^c(i,j) = Im_j(i,j) + \gamma_{i,j}$

Algorithm 6: Inter camera color calibration using cross correlation matrix method

image and its histogram (only green channel) are shown in the figure 5.4.a and 5.4.b respectively. The camera C_j image (figure 5.4.c) is passed through complex color variation.

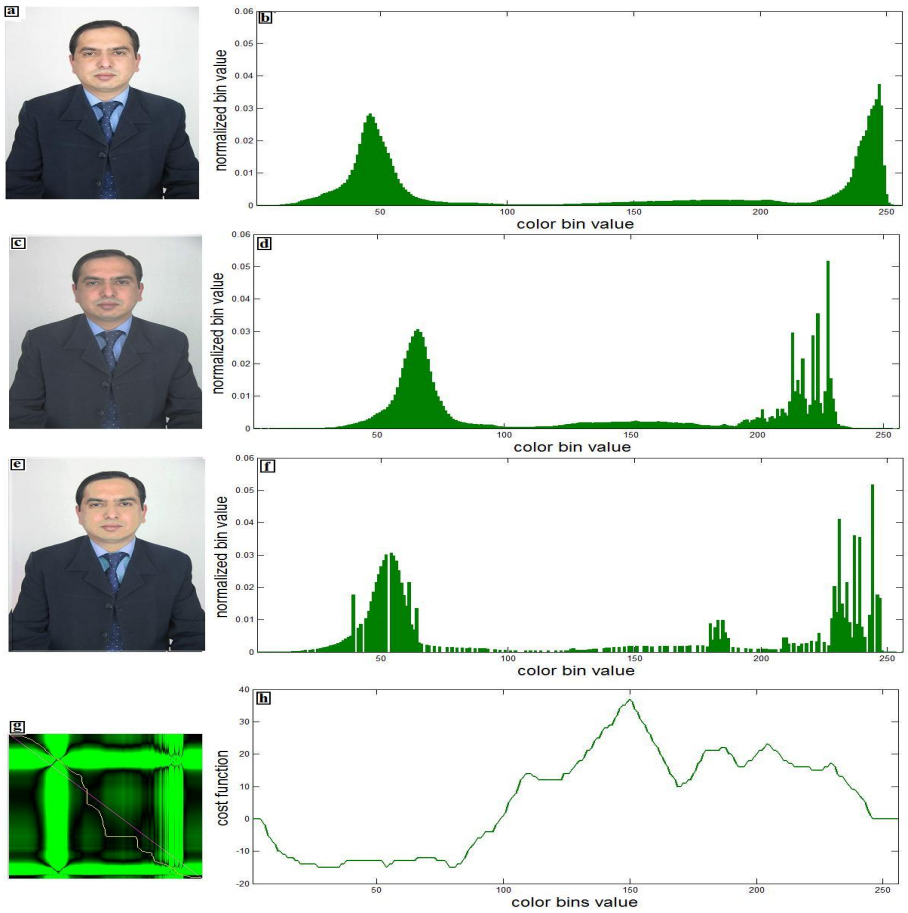

Figure 5.4: (a) and (c) are camera C_i, C_j images and image (e) is after color calibration of camera C_j. (b), (d) and (f) are showing the histogram of images (a), (c) and (e) respectively. (g) illustrate the minimum-cost path from first to last matrix element and (h) shows the cost function $\gamma_{i,j}$

Left half of image (d) histogram is shifted toward right and right half is shifted towards left. But right half histogram shape is completely different than histogram shape presented in image (b). The image (e) represents the color calibrated image of camera C_j using cost function shown in image (h). The figure 5.4.f, shows the histogram of color calibrated image (e). The figure 5.4.b and 5.4.f illustrate that these histogram distributions are similar but not completely identical. Which shows that camera C_j colors are calibrated with camera C_i but color mapping is not perfect.

The figure 5.4.h shows the cost function which is used to correct the image (c). For example, green channel pixel having value 150 has a cost function value 38. Each pixel of image (c) having green channel value 150 is replaced by 188 for color correction. The same procedure is applied on the others color channels. The figure 5.4.g represents the minimum cost path for green color channel between the C_i and C_j cameras normalized histograms. The cost path curve is initially in the right side of diagonal and then in the

left side of diagonal. This is the reason, initially cost function $\gamma_{i,j}$ is negative and after the middle region, it becomes positive.

5.2.2 Color Calibration using Cumulative Histogram Method

In the previous section 5.2.1, we find that image color calibration using CCM method gives good results. But this technique is unable to map one camera color information to another camera completely. During the literature survey, we found that mapping of one camera image cumulative histogram to another camera's cumulative histogram also gives camera color calibration. This method is called camera color calibration using Inverted Cumulative Histogram (ICH) technique.
In color calibration using ICH method; normalized histograms H_i and H_j for a single image of cameras C_i and C_j images are computed. Compute cumulative histograms $\hat{H}_i$ and $\hat{H}_j$ using equation 5.4 as:

$$\hat{H}\left(B_m\right) = \sum_{k=1}^{m} I(B_k) \tag{5.4}$$

where B_1, ...B_m, ...,B_M are brightness values of histograms H_i and H_j bins. The cumulative histogram $\hat{H}_i$ is considered as as a reference histogram. Find the minimum distance between the cumulative histogram bins of $\hat{H}_i$ and $\hat{H}_j$.
The method to find a best matching between two histogram's bins is called BTF. $f_{i,j}(B_m)$ for bin B_m can be expressed in mathematical form as:

$$f_{i,j}\left(B_m\right) = \hat{H}_j^{-1}\left(\hat{H}_i\left(B_m\right)\right) \tag{5.5}$$

The method to find BTF mapping function $f_{i,j}(B_m)$ is explained bellow:

- Find the best match between the bin B_m of $\hat{H}_i$ with all the bins of histogram $\hat{H}_j$.
- If bin B_m of $\hat{H}_i$ is matched with the B_n of $\hat{H}_j$.
- Then $\hat{H}_i(B_{m+1})$ can only match with $\hat{H}_j(B_w)$: where w $\geq$ n.

The ICH color mapping algorithm is presented in algorithm 7. We test the algorithm 7 on the same cameras C_i and C_j images of the section 5.2.1. The comparison between figure 5.5.b and 5.4.f shows that ICH method maps the camera C_i colors to camera C_j significantly better than CCM method. The figure 5.5.g shows a BTF between the camera C_i and C_j. For example, camera C_j image's green color channel bin having value 130 is replaced with the value 150. Similarly, same procedure is adopted for other bins.

Input: Take a single image from camera C_i and C_j
Output: Camera C_j colors are calibrated

compute the normalized histogram H_i and H_j
find the cumulative histogram $\hat{H}_i$ and $\hat{H}_j$ using equation 5.4
use equation 5.5 to find BTF mapping function between cameras C_i and C_j

Algorithm 7: Camera color calibration algorithm using cumulated histogram method

We observed in our experiments that ICH method transforms one camera C_i color space information to another's camera C_j color space effectively. ICH algorithm is computationally faster than CCM algorithm because ICH technique requires 1-D data matching and CCM uses 2-D dynamic programming computationally complex algorithm. More detail on the comparison of CCM and ICH methods are presented in the results section 5.4.

5.3 Camera Color Calibration in Non-Overlapping Camera Environment

In the previous section, we discussed two camera color calibration techniques. The cameras are filming the same or partially overlapping scene. In this section, we will discuss the camera color calibration techniques for non-overlapping FOV cameras. The technique discussed in the section 5.2.2, can be used after some modification for non-overlapping cameras. The possible solution is to use those objects which move from one camera C_i to another camera C_j.

There are two methods commonly used for camera color calibration in a non-overlapping environment. These methods are: Mean Brightness Transfer Function (MBTF) and Cumulative Brightness Transfer Function (CBTF). BTF between two cameras is calculated during a training phase. During this phase, we assume that we know the objects correspondence from C_i to C_j. Let us assume object O enters in the camera C_i's FOV and after exiting Ci, the same object enters in the camera C_j's FOV. Calculate normalized histogram for the object O for the two cameras and compute the BTF $f_{i,j}$ from the two cumulative histograms of the object.

Figure 5.6 shows the objects, we use for calculating BTF in a training phase. The first row of the figure shows objects present in camera JVC. The second and third rows are representing objects present in camera Fuji and Sony respectively. Figure 5.6 illustrate that humans appearance in these cameras are different. Especially human appearance in Fuji camera is significantly different from others cameras.

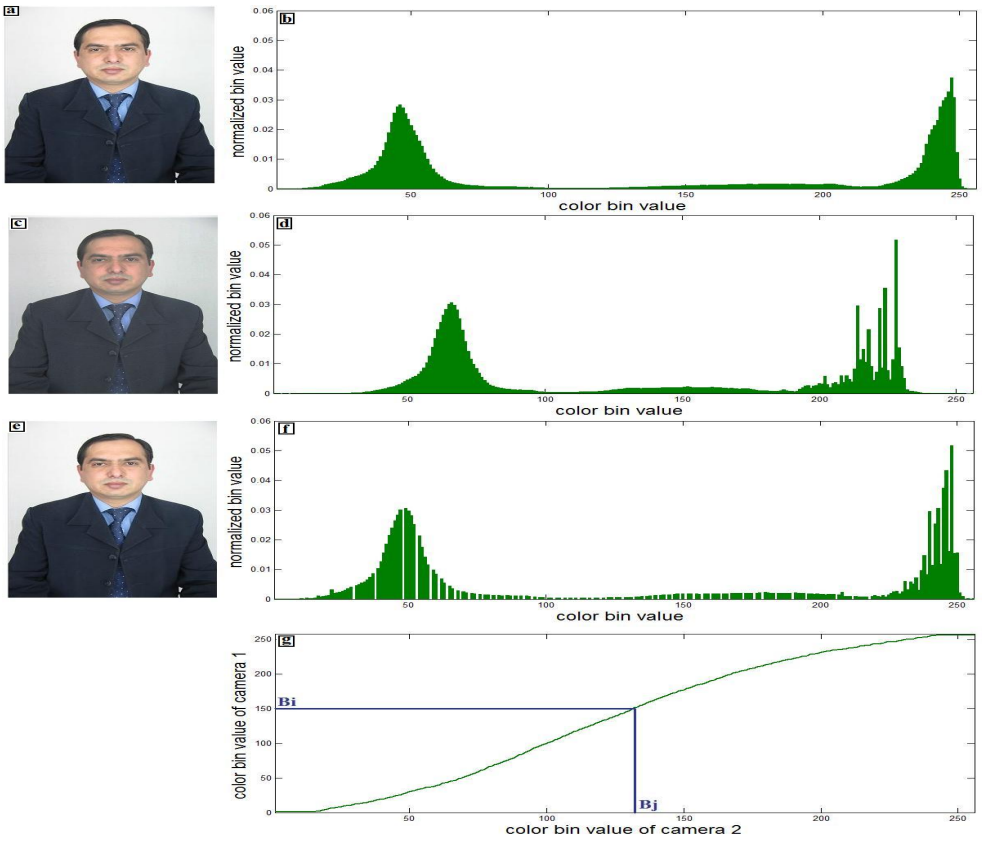

Figure 5.5: (a) and (c) are camera C_i, C_j images and image (e) is after color calibration of camera C_j. (b), (d) and (f) are showing the histogram of images (a), (c) and (e) respectively. (g) represents the BTF between the cameras C_i and C_j.

In the following sub-section, MBTF algorithm is discussed in section 5.3.1 and section 5.3.2 deals with CBTF. In the section 5.3.3, we discuss our proposed algorithm [Ilyas et al., 2010b].

5.3.1 Mean Brightness Transfer Function

Mean Brightness Transfer Function (MBTF) calculates BTF ($f_{i,j}$) between two cameras C_i and C_j during a training period. After calculating BTF for all objects, the mean BTF is calculated from these BTF by finding mean of all BTF curves $f_{i,j}$. The MBTF algorithm is presented in algorithm 8. The figure 5.7 shows that each object present in camera pair C_i and C_j calculate one BTF. The BTF curves present in figure 5.7 are very different. The dark blue and thick line in the graph is the resultant MBTF. It is evident from the figure, that in the lower half part of the curve, most BTF curves overlapped to each other but the upper half curve they are well separated. This is the reason that MBTF function curve is a true representative in the lower half curve. In the upper half, it is not a true representative of any BTF curve.

Figure 5.6: Some objects present in three Cameras C_1 (JVC) , C_2 (Fuji) and C_3 (Sony) are shown in row1, row2 and row3 respectively.

Input: Take k objects from camera C_i and C_j images
Output: Camera C_j colors are calibrated

foreach *object in camera* C_i *and* C_j **do**
  Compute normalized histogram H_i and H_j of same object in camera pair $C_i - C_j$
  Find the cumulative histogram $\hat{H}_i$ and $\hat{H}_j$ using equation 5.4
  Find MBTF $f_{i,j}$ using equation 5.5
end
Calculate MBTF $\overline{f_{i,j}}$ using all k BTF $f_{i,j}$

Algorithm 8: Inter camera color calibration using MBTF method

In general, one object has only a limited number of colors. Each BTF curve gives true responses for the histogram regions where objects colors are more representative. During the MBTF process, the histogram regions having small contribution, lose information due to the averaging process.

5.3.2 Cumulative Brightness Transfer Function

MBTF loses inter-camera color information which are not present in the majority of the objects. This increases the problem of object re-identification when the object exits from

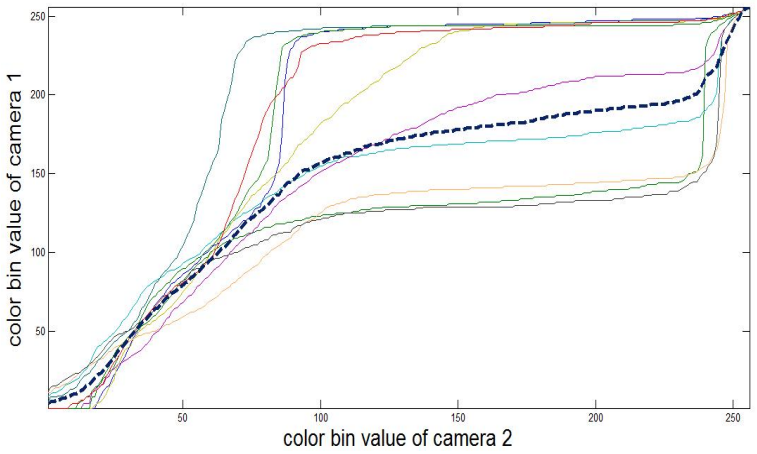

Figure 5.7: The BTF for each object present in camera pair C_i and C_j are plotted. Thick Blue line is the MBTF of all BTF curves.

one camera and enters in the FOV of another camera. If BTF is not properly calculated, then the object color mapping from one camera to the other one might be incorrect. This becomes the reason of false object identification. To overcome this problem, Cumulative Brightness Transfer Function (CBTF) is proposed by Prosser et al. [2008]. The CBTF algorithm is explained in algorithm 9.

They find the histogram of all the objects which are presented in a training time in camera pair $C_i - C_j$. Then accumulate all the histograms information in one histogram. They find one cumulative histogram for each camera.

Input: Take k objects from camera C_i and C_j images
Output: Camera C_j colors are calibrated

Compute the histograms $h_i^1..h_i^k$ and $h_j^1..h_j^k$ for k objects in camera pair C_i-C_j
Accumulate the brightness values for the cameras C_i and C_j histograms $h_i^1..h_i^k$ and $h_j^1..h_j^k$ and then, compute the cumulative histograms $\hat{H}_i$ and $\hat{H}_j$ as:
$\hat{H}(B_m) = \sum_{k=1}^{m} \sum_{l=1}^{k} h^l(B_k)$
Normalize the cumulative histogram $\hat{H}_i$ and $\hat{H}_j$ by the total numbers of object's pixels in training set for camera C_i and C_j
foreach *bin* B_m *of* $\hat{H}_i$ **do**
  find the best match from all the bins of histogram $\hat{H}_j$ using equation 5.5
end

Algorithm 9: Inter camera color calibration using CBTF method

5.3.3 Modified Cumulative Brightness Transfer Function

CBTF technique explained in 5.3.2, claims that it solves the problem of MBTF which lose color information that are not present in the majority of objects. In fact from the CBTF

algorithm, it is evident that they accumulate all the histograms for the camera C_i and C_j respectively and then normalize the cumulative histograms for each camera. In many cases, objects histogram does not have contributions to all the histogram bins. That is why if we take the sum of bin Bm of all histograms and normalize its value by the total number of pixels in the training set, then some information is lost.
The figure 5.8, is illustrating the objects histograms present in the JVC camera. It is evident from the figure that there are some regions in histograms having one or two histograms has contribution. That is why in the process of finding mean histogram $\overline{h_i}$ and $\overline{h_j}$, we only use those histograms having some contribution in that region. We set minimum number of value threshold Θ. The steps of our MCBTF algorithms are:

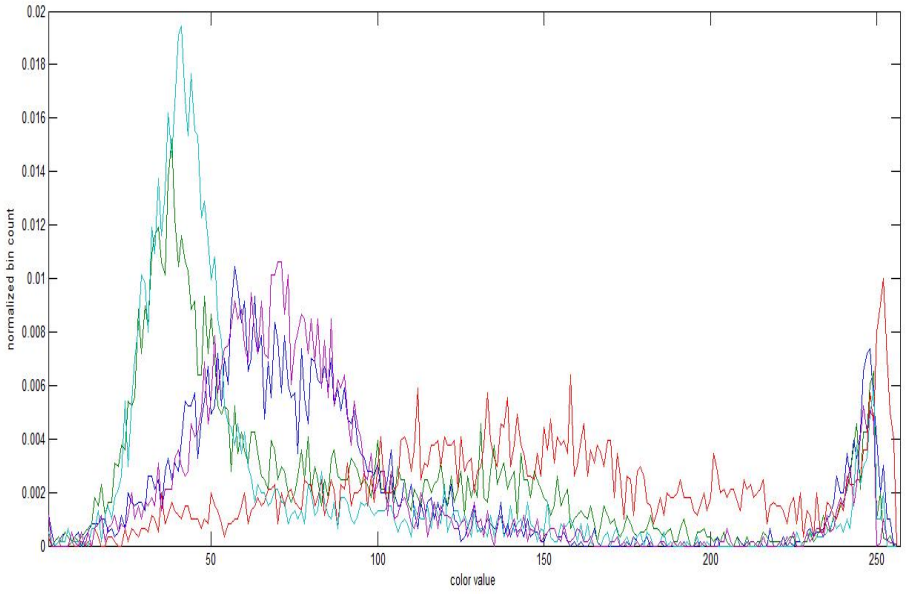

Figure 5.8: Histogram of the objects present in the camera JVC.

1. Compute the histograms $h_i^1..h_i^k$ and $h_j^1..h_j^k$ for k objects for camera pair C^i, C^j

2. Find the mean value of bin B_m of only those histograms that have a significant contribution to B_m as follows:

   (a) Find the normalized mean histogram $\overline{h_i}$ and $\overline{h_j}$ for camera C^i and C^j using equation 5.6:

   $$K_i^m = \{v \in I; h_i^v(B_m) > \Theta\}\,; \tag{5.6}$$

   $I = [1..k]$ is the total set of histogram indices and K_i^m is the subset of I such that the selected histograms have a significant contribution to B_m. $m = [0..255]$ are total number of bins. Θ is basically minimum value of bin B_m for being

considered on its significance. The threshold value of Θ is selected in a training period. The experiments show that a typical value of Θ between 3 and 5 pixels gives satisfactory results in most situations. The number of elements of K_i^m will be noted as $\#K_i^m$

(b) The average value of bin B_m can then be computed as:

$$\overline{h_i}(B_m) = \frac{\sum_{k \in K_i^m} h_i^k(B_m)}{\#K_i^m} \tag{5.7}$$

3. Find the cumulative histograms $\hat{H}_i$ and $\hat{H}_j$ by using equation 5.4

4. The BTF between two cameras is calculated using equation 5.5.

In brief, we can represent the MCBTF algorithm as:

Input: Take k objects from camera C_i and C_j images
Output: Camera C_j colors are calibrated

Compute the histograms $h_i^1..h_i^k$ and $h_j^1..h_j^k$ for camera pair C^i, C^j for k objects
Find the normalized mean histogram $\overline{h_i}$ and $\overline{h_j}$ for camera C_i and C_j using equations 5.6 and 5.7
Compute the cumulative histogram $\hat{H}_i$ and $\hat{H}_j$ using equation 5.4
Calculate the BTF using the equation 5.5

Algorithm 10: Inter camera color calibration using MCBTF method

5.4 Results

In this section, we discuss the performance of inter-camera color calibration methods. In the first part, we discuss the results of two color calibration techniques: ICH and CCM. In the second part, we will discuss the camera color calibration methods MBTF, CBTF and MCBTF in non-overlapping camera environment.
We do the experiment for camera calibration using ICH and CCM methods on different illumination conditions. For example with the illumination changes, color variation, contrast changes and their combinations. The figure 5.9 represent the case when both camera have different colors, contrast and luminosity. The camera C_i and C_j images are shown in figure 5.9.a and 5.9.c. The 5.9.e and 5.9.g represent the color calibration of 5.9.c using ICH and CCM. The figure 5.9.b, 5.9.d , 5.9.f and 5.9.h show histograms of an original image (a), distorted image (c) and color calibrated images (e) and (g) respectively.

Figure 5.9 shows that CCM improves the camera C_j results but it is unable to significantly calibrate the camera color in complex scenario (combination of variation color,

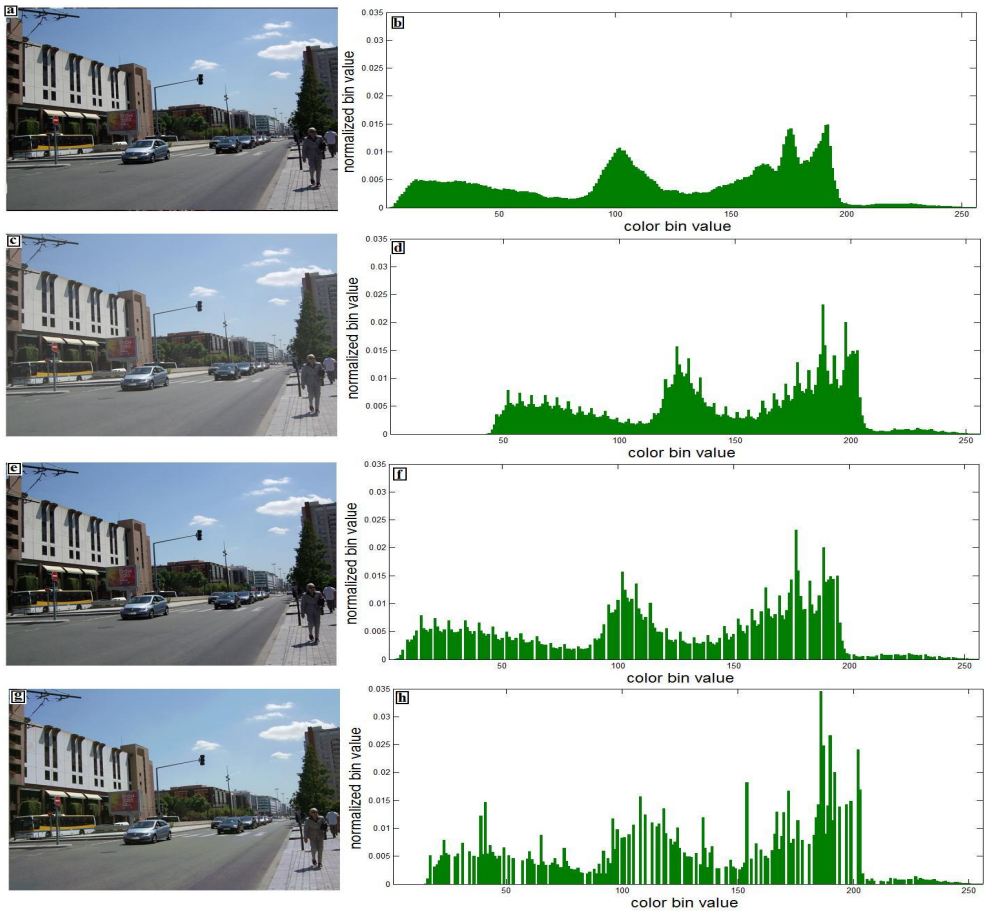

Figure 5.9: (a), (c), (e) and (g) are images of camera C_i, C_j and color calibration of image (c) using ICH and CCM respectively. (b), (d), (f) and (h) are showing the histogram of images (a), (c), (e) and (g) respectively.

luminosity, contrast and gamma correction). ICH clearly calibrate the camera C_j image colors better than CCM.

We find that ICH method has better camera color calibration performance, when they are filming same scene (before installing the cameras). The color calibration after installing the overlapping FOV multi-cameras is possible, using common image regions which are filmed by both cameras to calculate BTF.

In the second series of experiments, we used non-overlapping FOV cameras. We do not have common regions, which might be used to calculate BTF for inter-camera color calibration. We used moving objects for inter-camera color calibration.

We installed three cameras of different models (Fuji, Sony and JVC) with non-overlapping FOV. We set camera C_1 (JVC) as a reference and calibrate others cameras C_2 (Fuji) and C_3 (Sony) by finding their BTF $f_{2,1}$ and $f_{3,1}$. We compute MCBTF, MBTF and CBTF using the three methodologies explained in section 5.3. In the training phase, we use 10 images of each object for five objects present in the sequences for each camera. These 10

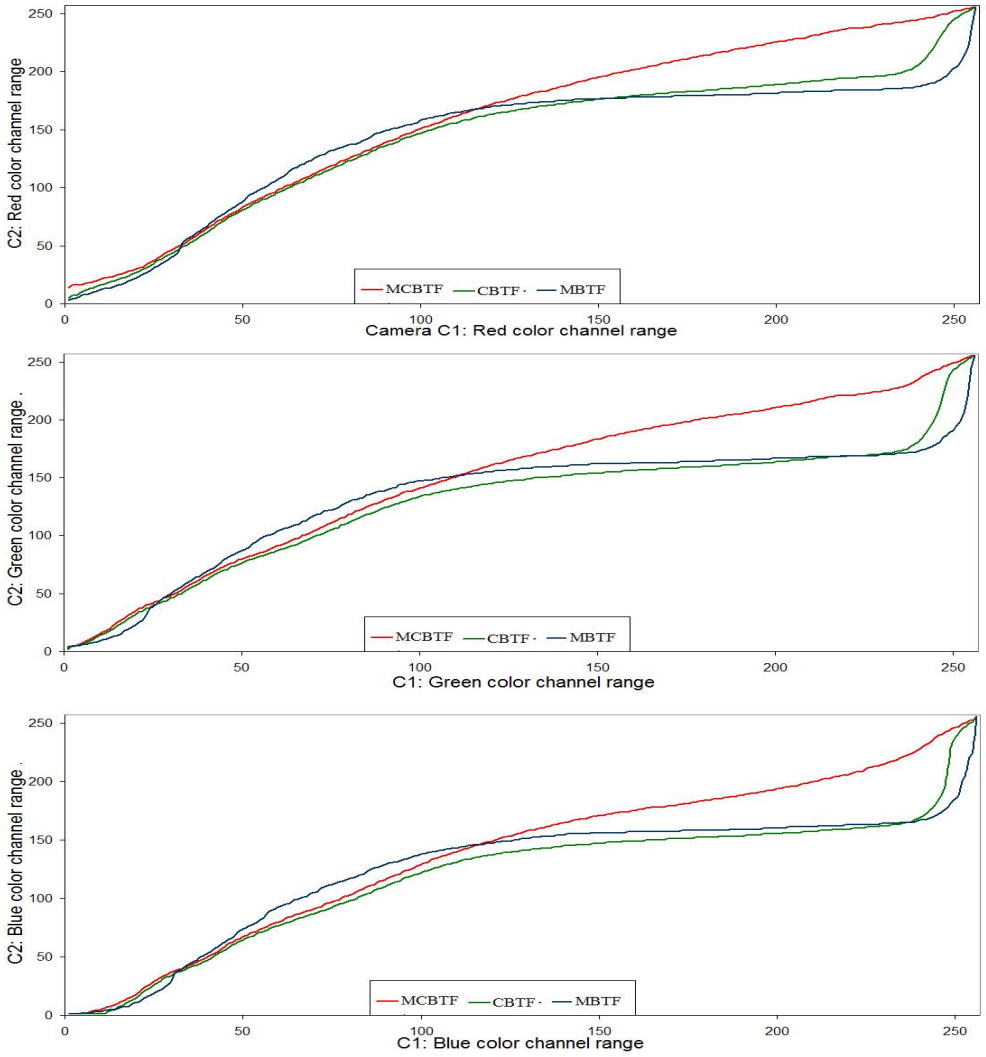

Figure 5.10: The BTF curve between two non-overlapping FOV cameras by using MBTF, CBTF and MCBTF are shown.

images are selected from different location of scene and varies view angles of objects in camera's FOV. This methods helps to calculate better BTF of camera pairs $C_1 - C_2$ and $C_1 - C_3$. Illumination conditions and colors are different for these cameras. The figure 5.10 shows the BTF of the $C_1 - C_2$ pair of cameras for R, G and B channels. The figure 5.10 shows that our MCBTF works better and it maps low-intensity pixels to higher intensities pixels after intensity value 100 for camera C_2 which has low-light intensity. This is the point where our modification works better than CBTF and MBTF. In our experiments, the low light intensity in camera C_2 FOV, shift most colors values to lower bins. Similarly, histogram of every object do not have contribution to all the bins. If we take the same method of CBTF, then BTF for pixels having intensities higher than 100 are mapped to lower values - which decreases the performance of object matching like

CBTF and MBTF.

We calculate one BTF curve for each color channel for camera pair C_i and C_j. Each BTF curve covers complete color range ([0..255]). The BTF mapping is actually many-to-one mapping similar to histogram equalization. There is a possibility that some color information might be lost. This can be seen in the figure 5.11.f. In figure 5.11, the same person is present in image (a) and (c) in two cameras C_i and C_j. The image (e) represents the same object after color calibration of image (c). It is evident from the images (b), (d) and (f) of figure 5.11, that MCBTF is able to transfer the color information of a one camera to another camera in an effective way.

In chapter 6, section 6.2, we will discus our experiments for object re-identification in a non-overlapping FOV multi-camera environment. The results which we will present in section 6.2 demonstrate that inter-camera object tracking with using camera color calibration has significantly better results. In camera calibration, our MCBTF technique works much better than MBTF and CBTF (section 6.2).

The idea of using the moving objects, when they pass through in the FOV of non-

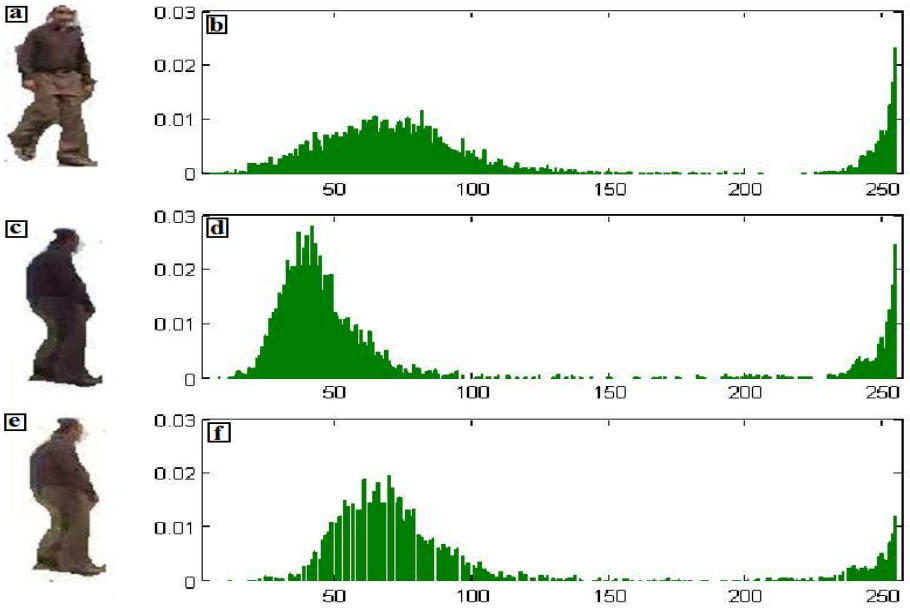

Figure 5.11: An object present in (a) camera C1 (JVC), (b) histogram of image (a), (c) same object in C2 (Fuji), (d) histogram of image (b), (e - f) illustrate image (c) after color correction and its histogram.

overlapping multi-camera is an attractive idea. It helps to find BTF between the cameras even they have no overlapping regions. It has also a limitation, when an object appearance within a same camera may be changed due to change of object view angle. This phenomenon is represented in figure 5.12. Same object in a same camera has different histogram due to change of its view angle. This might be a problem for calculating BTF.

Especially, in MBTF, where one BTF curve is computed for each object. It is better for computing MBTF to select same view angle of the object in camera pairs $C_1 - C_2$ and $C_1 - C_3$. This problem is not a significant issue in CBTF and MCBTF as they accumulate or find mean histogram respectively before calculating their BTF curve. MCBTF calculate one mean histogram per camera. If we have same numbers of objects in each camera then histogram order is not important.

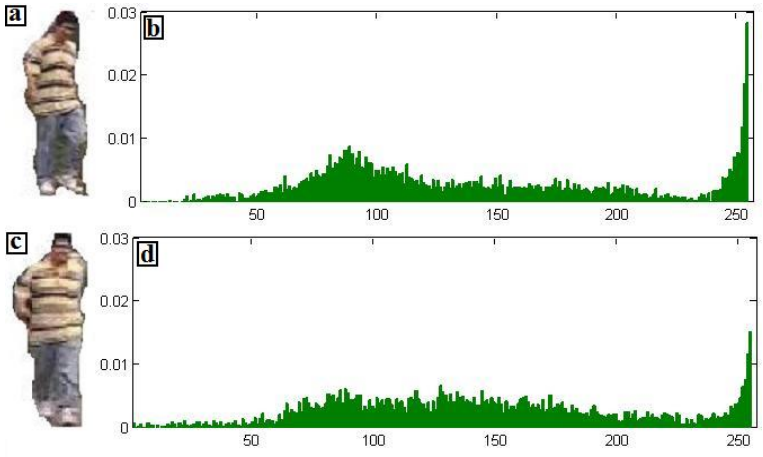

Figure 5.12: An object and its histograms with different camera view angle in same camera

5.5 Conclusion

In this chapter, we discussed the multi-camera color calibration for overlapping and non-overlapping FOV cameras environments. We proposed some modifications to the well known CBTF color calibration technique for non-overlapping camera environment. Our proposed MCBTF maps one camera color information to another camera better than CBTF and MBTF. The experimental results of our proposed MCBTF color calibration technique for object re-identification in non-overlapping FOV multi-camera will be examined in chapter 6. We show that our MCBTF method solves the inter-camera color calibration problems. In future works, we are planning to propose BTF updating strategies, in order to cancel out the effects of changes in illumination conditions after the training time of camera color calibration.

Similarly, new method to find BTF using illumination charts instead of objects, may improve CBTF performance. The possible advantage of using this method is that it has no problem of appearance view angle.

Chapter 6

Human Re-identification in a Multi-Camera Environment

In this chapter, we present a complete object tracking system in a multi-camera environment. We combine all the ideas, we have proposed in the previous chapters: background modeling, object tracking and re-identification and color calibration in multi-camera environments.

The task of observing an object in one camera's field of view (FOV) and recognizing that object again in the same or in another camera's FOV is often referred as the object re-identification. The problem has a significant importance for large area security scenarios like: airports, university campuses, shopping centers or train stations. In video surveillance systems, re-identifying a person in another camera is often done manually by an operator, which is a difficult task. Therefore, it is desirable to add computer-aided assistance to this task. Object re-identification in the presence of many objects is a still well focused topic.

Object re-identification helps to keep information about an object activity in all the cameras installed at different locations. Many approaches have been proposed for object re-identification in recent years. We discussed these approaches in section 2.3 of chapter 2 in the framework of a single camera or of multi-camera environments. The methods that have been proposed in the literature differ one another by the following parameters: the location of cameras (indoor, outdoor or mixed environment), the camera's type (gray scale, color or infra red) and the camera monitoring scene (overlapping FOV, non-overlapping FOV, fixed or moving cameras). The multiplication of the number of cameras in a video surveillance system increase the impact of following parameters: large scale video data handling, real time performance, object size variation, linking ob-

jects different view angles, object appearance dissimilarity in different cameras, etc.
In section 6.1, we explain our proposed object tracking and re-identification method in a non-overlapping multi-camera environment. The system presented in section 6.1 is optimized to get better object re-identification performance using cache and object database effectively. In section 6.2 we discuss the object re-identification performance of the solution which we propose in section 6.1. We perform object re-identification experiments using color calibration techniques and without color calibration. Section 6.3 concludes the chapter and discuss the future works.

6.1 Methodology

We have an environment containing several non-overlapping FOV cameras. We propose a system which can detect and track moving objects. The system should have the ability to re-identify an object when it passes in some other camera's FOV. All the object's Vertical Feature (VF), entering and exiting positions, timestamps, and camera ID are stored in a centralized database. Whenever the user needs some information about a particular event, then he can formulate a query to see the activities of all the objects at a particular moment or a particular object during a certain period of time.
The idea is to combine the proposed algorithms of foreground-background segmentation, object tracking and identification in a single camera environment and inter-camera color calibration is presented in [Ilyas et al., 2010c]. Figure 6.1 illustrates the proposed method for object re-identification and tracking in a multi camera environment. We use three non-overlapping FOV cameras, C_1, C_2 and C_3 in our experiments. This method can be extended to any number of cameras.

The first step of our algorithm is to calibrate the camera's colors during the training time. Camera color calibration becomes necessary, because object are re-identified using vertical feature. VF uses object color information for object recognition and re-identification. Significant object appearance changes produce false object recognition and re-identification. We use the camera C_1 as reference. The objective of camera color calibration is to maximize an object color appearance similarity in all the cameras. We calculate BTF for camera pairs $C_1 - C_2$ and $C_1 - C_3$ using our proposed algorithm MCBTF. The BTF curves of camera pair $C_1 - C_2$ using MCBT (see section 5.3.1), CBTF (see section 5.3.2) and MCBTF (see section 5.3.3) are presented in figure 5.10. The BTF curves of other camera pair $C_1 - C_3$ using MCBT, CBTF and MCBTF are shown in figure 6.2. In section 6.2 we discuss the object re-identification performance using MBTF, CBTF and MCBTF. Finally we select color calibration technique MCBTF for object re-

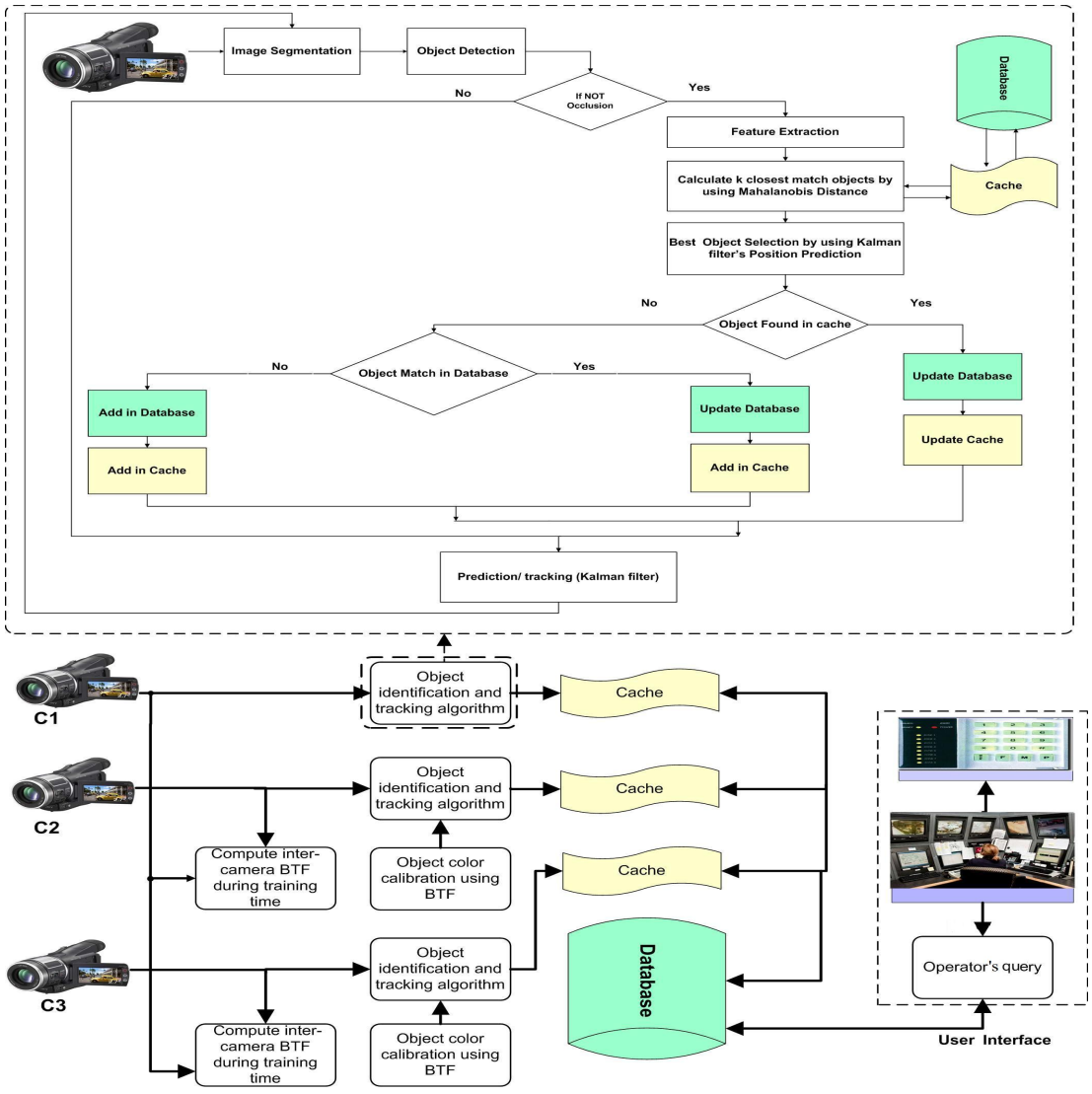

Figure 6.1: Object identification and tracking block diagram in a non-overlapping camera environment

identification and tracking due to its better performance.

During the color calibration training period, object re-identification and tracking is not activated. During the training time, the background modeling algorithm Modified CodeBook (MCB) model the background of each camera. Activating the background modeling on each camera during the color calibration do not affect the other object's tracking blocks performance, because background modeling is used to detect objects in camera's field of view. Please note that during the training time, we compute the BTF curves and after training time, we use them to correct only the object's colors present in the FOV of cameras C_2 and C_3.

After computing BTF for the cameras pairs $C_1 - C_2$ and $C_1 - C_3$, the system is ready to perform object tracking and re-identification. Images of cameras C_1, C_2 and C_3 are

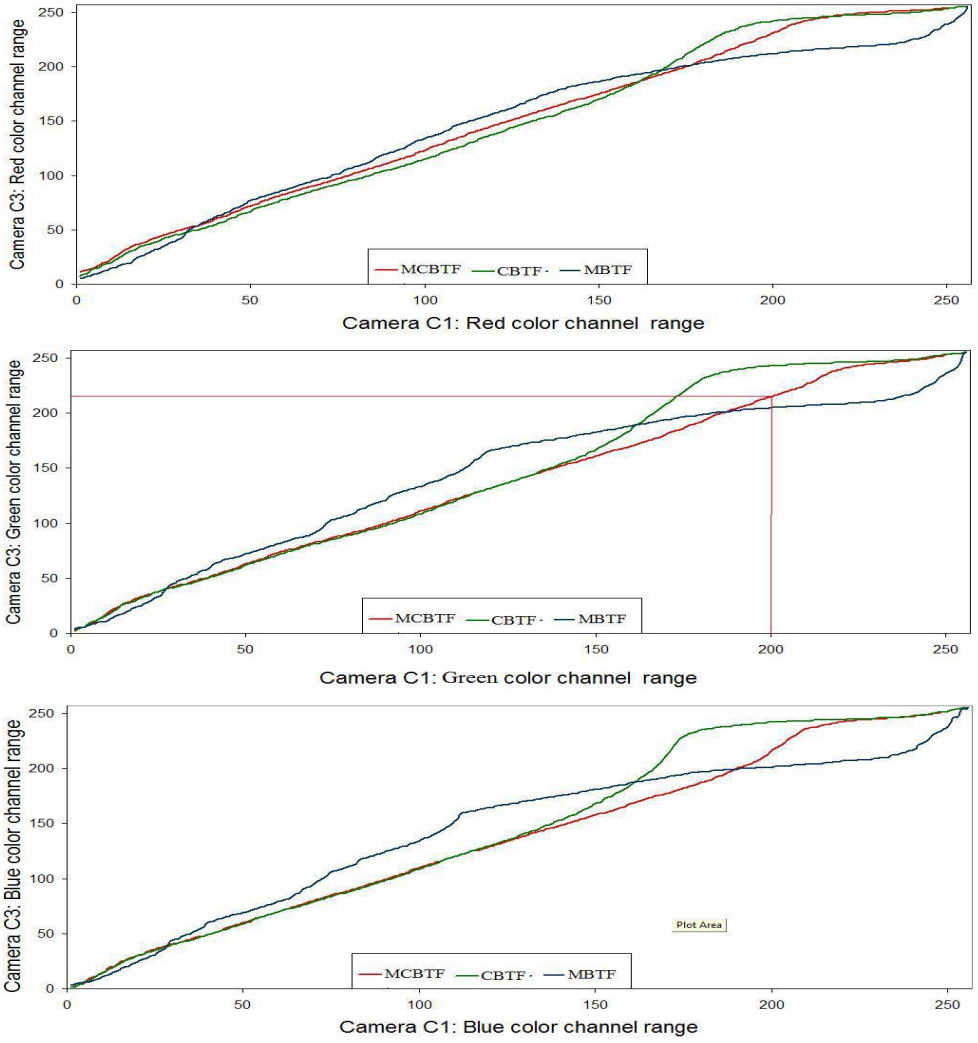

Figure 6.2: The BTF curve between cameras C_1 (JVC) and C_3 (Sony) using MBTF, CBTF and MCBTF are calculated during training time.

segmented into foreground and background using Modified CodeBook. Some detected objects are shown in figure 6.3. The objects having colors similar with background are also detected effectively.

After the objects detection using MCB, the objects of the cameras C_2 and C_3 are corrected using the BTF curves of MCBTF presented in figure 5.10 (BTF for camera pair $C_1 - C_2$) and 6.2 (BTF for camera pair pair $C_1 - C_3$), which we compute during the training time. Each pixel color (R, G, B) value of detected object in camera C_2 and C_3 are replaced with the corresponding value in the reference color model of camera C_1. For example, if an object's pixel color (green) has value=200 in camera C_3, is replaced with the value 220. The same procedure is adopted for all the pixels belonging to the objects of cameras C_2 and C_3.

Figure 6.3: Object detection using MCB method present in chapter 3

We extract object's VF, position, velocity, area, frame number and camera's ID. Object's predicted positions are also calculated using the Kalman filter. We improve the human re-identification performance in non-overlapping multi-camera environment using cache and database effectively. We introduce separate but identical caches for each camera. We store object information in a compact way in the cache: object's VF, position, velocity, frame number, area and object's predicted position. Please note that many human have similar clothes color, which increase the possibility of false recognition. In the database, when an object exits the camera's field of view, we store: the camera ID, the object's vertical feature, the entry and exits positions for the current scene and the corresponding frame numbers.

When an object exits from one camera and re-enters in same or another camera's FOV, we match this object in the database using VF only. If this object is re-identified then we label this object's ID as "under tracking". All the objects "under tracking" are added in cache. These objects are available in database for re-identification process when they exit camera's FOV again. When an object is not matched in the cache for some successive frames, then it is assumed that this object has exited from the camera's FOV. It is then deleted from the cache. Storing the camera's identity, frame number and position help to find the object activity pattern in different cameras. Storing frame numbers when object enters and exits from the scene helps to find at which time objects enter and exit from the camera's FOV.

Initially, the database and the caches are empty. The first detected object features (object's VF, position, velocity, area, frame number and camera's ID) are stored in cache without object matching and we label it "under tracking". Sample objects and their VF

Figure 6.4: Objects and their extracted representative VF using method present in chapter 4

feature are shown in the figure 6.4. Current object is matched with previous frame's objects stored in cache using its recognition features and using its motion model. If an object is matched in cache then its VF, position, area, frame number and next frame predicted position are updated. Else current object is matched with database objects. We only use VF to match current object with database objects. If current object is matched with database object, then we add object's VF, position, velocity, area and frame number in cache; re-initialize the Kalman filter using algorithm 4 and label the object "under tracking" in database. If the current object is not matched with any database object, then a new label is created and tracking process begins as described before. When an object is matched in database, we re-use the label it had before, and simply initialize its velocity to zero.

When the current object is matched with the previous frame's object, the matching time is decreased and the false object matching is minimized. Indeed, the cache consists of recent objects only, where as the database consists of all the objects which were presented any time in any camera. The caches of all the cameras are connected with the database. If an object is matched then its features are updated. The motion model is not useful for object recognition when it exits the FOV of one camera for some time, and re-enters in the FOV of any camera of the system. All the objects present in a cache or database are matched using the algorithm, presented in the section 4.5 of chapter 4. Object's position, next frame predicted position, area and frame number are updated by simply replacing their current values. VF is updated by using equation 4.1. In cache, we store the initial (when object enters camera's FOV) and final (when object exits from camera's FOV) frame numbers and the object's positions.

The algorithm of object tracking and re-identification in non-overlapping multi-camera environments is presented in algorithm 11. In the next section, we discuss the perfor-

mance of our proposed algorithm for object re-identification in non-overlapping camera environments.

Input: Image sequences from camera C_1, C_2 and C_2
Output: Object re-identification and tracking

```
if time ≤ training time then
    compute the BTF for camera pairs C1 − C2 and C1 − C3 using algorithm 10
else
    detect objects of cameras C1, C2 and C3 using algorithm 2 ;
    correct object's colors of cameras C2 and C3 using its corresponding BTF ;
    match this object in cache using algorithm 5 ;
    if object is matched in cache then
        update its VF, position, velocities, area and frame number ;
        get Kalman prediction for this object in next frame using algorithm 4 ;
    else
        object is matched in database using algorithm 5;
        if object is matched in database then
            add object's VF, position, velocities, area and frame number in cache ;
            re-initialize the Kalman filter for object using algorithm 4 ;
            label object "under tracking" in database ;
        else
            create a new object in cache ;
            add object's VF, position, velocities, area and frame number in cache ;
            initialize the Kalman filter for object using algorithm 4 ;
        end
    end
end
```

Algorithm 11: Object tracking and re-identification in non-overlapping camera environment

6.2 Results

We have applied our object tracking and re-identification algorithm for video sequences consisting of 4,000 frames. In this experiment, we installed three different cameras C_1, C_2 and C_3 (JVC, Fuji and Sony respectively). Figure 6.5 shows the camera's positions in our experiment. There are many possible paths for objects to enter in the camera's FOV. The gray color is representing walking tracks. Persons may enter in the camera's FOV without passing through the others camera's FOV.

In the experiment, we asked to 12 persons to enter and exit from the FOV of one camera to another camera more than 30 times, from different directions in arbitrary order. Figure 6.6 shows some images from cameras C_1, C_2 and C_3 respectively. In the figure, objects are passing through the camera's FOV in different time sequences. We distin-

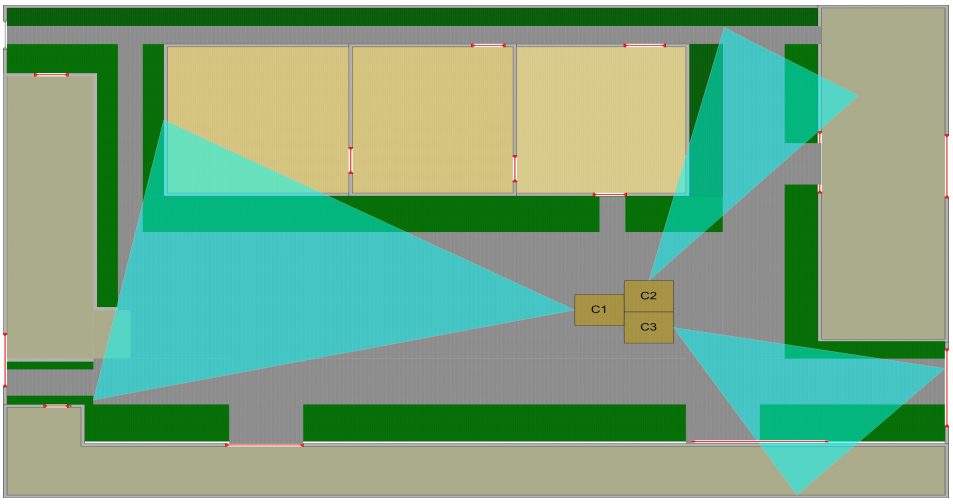

Figure 6.5: Camera topology for object identification in non-overlapping field of view cameras.

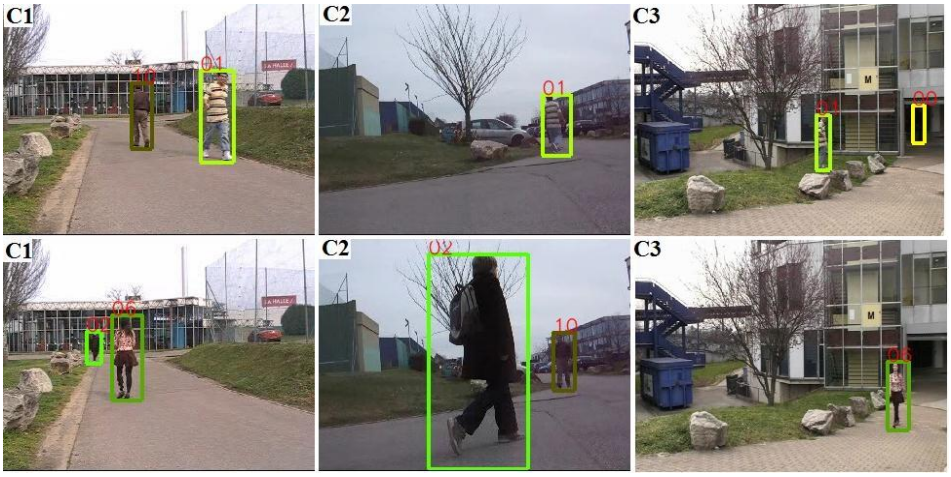

Figure 6.6: Object re-identification In non-overlapping camera environment.

guish them by drawing the bounding rectangle and object identification (ID) number on the top of rectangle around these objects. The consistent color of rectangles and ID in all the cameras for the objects confirms that objects are re-identified when they exit from one camera's FOV and enter in another camera's FOV. The figure, also shows that we are able to track and re-identify objects even when their size, view angle, illumination conditions and appearance are significantly different.

We also compared the object re-identification performance with and without camera color calibration. We use MBTF, CBTF and MCBTF color calibration functions to maximize the object color appearance in the cameras. We plot ROC curve between precision (PR) and Recall (RE) for a set of values of D_{ref}. We fix the value of $\zeta = 0.03$ and $D_{th} = 50$ pixels and we change the value of D_{ref} from 8 to 12. These parameters are discussed in

chapter 4. Higher value of ζ allows a quick update of VF and a very small value slow down the update of VF, which becomes the reason of false object recognition, because, object appearance changes with variation of illumination conditions.
PR and RE of object re-identification for each algorithm is calculated by using equations 6.1 and 6.2.

$$PR = \frac{TP}{TP + FP} \tag{6.1}$$

$$RE = \frac{TP}{TP + FN} \tag{6.2}$$

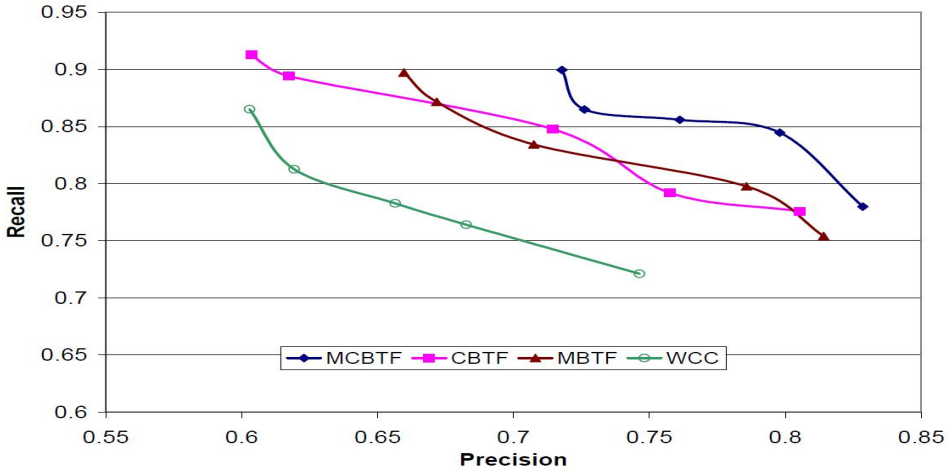

Figure 6.7: Object re-identification ROC curve between precision and recall with and without color calibration (WCC).

The three quantities TP, FP and FN being defined as follow:
True positive (TP): If an old label is correctly assigned to an object, or a new label is created for a new object, then it is considered as TP.
False positive (FP): If an old label is assigned to a wrong object, then it is considered as FP.
False negative (FN): If a new label is created for an already labeled object, then it is known as FN.
Using of ROC curves using PR and RE helps to understand the object re-identification performance. In our experiments of object re-identification, the precision indicates how many objects are re-identified without producing false positives. While recall tells the possibility of object re-identification. For example, the increase in false object re-identification decreases the object re-identification performance. Similarly, if many new labels are created for objects already stored in database, then it decreases the value of

RE. We plot the ROC curves for five different values of D_{ref}. In our experiments, we find that small values of the similarity threshold D_{ref} produce more precision but many new tokens are created for already labeled objects. Which becomes the reason of smaller recall. It is often considered that a situation having the same value for precision and recall is considered as appreciable, which suggests that a D_{ref} of value=8.5 has equal PR and RE (82%).

Figure 6.7 illustrates human re-identification performance using PR and RE curves. These graphs are plotted for the color calibration techniques, MBTF, CBTF, MCBTF and also Without Color Calibration (WCC). It is evident that our modification to CBTF increases significantly the object re-identification performance, both in terms of precision and of recall. MCBTF has higher value of precision and recall than all the three other techniques. The performances of CBTF and MBTF are comparable, without clear advantage of one approach in front of the other. These ROC curves helps to understand the contrasted results presented by [Prosser et al., 2008] and [Orazio et al., 2009]. The three color calibration techniques show improvements as compared to the basic approach that does not calibrate cameras colors. The test without color calibration produces more FP specially in camera C_2 which is installed in the region having the lowest illumination. Hence its re-identification results are worse than any of the techniques that use color calibration. We can then conclude from our object re-identification results that MCBTF gives re-identification performance significantly better than CBTF and MBTF.

6.3 Conclusion

In this chapter, we applied our previous proposed algorithms for foreground-background segmentation, object tracking, identification and camera color calibration in non-overlapping multi-camera environments. We also proposed to use a cache and the database communication effectively to increase object re-identification performance in non-overlapping cameras environments. The results show that the proposed object tracking and re-identification algorithm works satisfactory. The results also illustrate that using camera color calibration increases object tracking and re-identification performance significantly, especially using our new color calibration scheme MCBT. On the other hands, using the object correspondence probability function like Bayesian belief network (BBN) function may increase object re-identification performance in a multi-camera environment which should be further investigated in future works.

Contents

Chapter 7

Conclusion and Future Works

In this thesis, we have contributed to enhance the capabilities of conventional visual surveillance system to minimize the work load of operators. The system can be used to summarize the objects movements in a multi-camera environment. It can generate alerts allowing the operator to concentrate and look at specific regions. Similarly, the operator can see the summary of all the objects movements and activities if he needs.
We have discussed our contributions to improve the performance of visual systems in chapters 3, 4, 5 and 6. In this conclusion, we first summarize our contributions for object tracking system and discuss some of its limitations. In a second part, we will propose some possible perspectives for future works.

7.1 Conclusion

The performance of visual tracking systems depends on the individual performance of its different building blocks. The important steps of the system are: image noise filtering, moving object extraction from sequence of images, object fusions (object occlusion), object recognition, tracking and object interactions analysis, object activity monitoring, inter-camera object correspondence, object trajectories analysis, and inter-camera color calibration. It is not possible to review and improve all the above discussed parts during the limited duration of PhD studies. But we have proposed some algorithms for object detection, object tracking and inter-camera color calibration in a multi-camera environment. Object re-identification is improved by calibrating camera's colors in multi-camera environments. We discuss on the performance of different algorithms in the following paragraphs.
In chapter 3, we have proposed an improvement for a foreground-background segmentation algorithm. We have modified the use of the codeword matching frequency pa-

rameter for accessing, deleting, matching and adding a codeword in the codebook or to move cache codewords into codebook. The comparison of MCB with two other techniques CB and MOG shows that MCB is able to produce better results in almost all the situations. In short we can summarize the results as follow:

The authors claimed that CB works better than MOG and other techniques. In our experiment, this is the case only when few objects having sufficient contrast with background are present. CB adopts object pixels as a background pixel if several objects having similar colors are present in the scene. Our modified version use the parameter of frequency for improvement of the CB method. It does not introduce any additional parameters nor computational complexity so it is still computationally less expensive than MOG. But it is able to detect moving object more precisely than the original codebook and probability based mixture of Gaussians.

We evaluate our results using ROC analysis in terms of precision versus sensitivity or in the terms of TPR and FPR. Four methods for computing a unique quality factor are used. These methodologies indicate that our proposed MCB shows better result than MOG and CB. MCB performs better than CB in precision quality, specificity based error factor as well as weighted Euclidean distance. In comparison with MOG, MCB obtains better precision factor, less false alarms and is more robust with variation of light intensities. Moreover, it involves less floating point calculations.

We can conclude that MCB performance is better than CB in all the conditions, without introducing complex calculations in the algorithm. The choice between MCB and MOG depends on the application. If the precision is considered as the most important factor, then MCB is probably the best choice. If a compromise is possible on precision, shadow and false alarms but any small object should not be lost, then MOG can be a better choice. Nevertheless, like CB and MOG, MCB does not handle the case of still objects in a satisfying manner. If a foreground object stops for some time, then it will also be included in the background.

In chapter 4, we proposed a real time human tracking algorithm using probabilistic and deterministic models to increase object tracking accuracy. We also proposed a simple 1-D appearance model (VF) and have combined it with motion based features for object recognition in a single camera or in a non-overlapping multi-camera video surveillance system. We have compared our algorithm with two other algorithms. One algorithm uses dominant colors and the other one uses object motion features. Results in section 4.6 verified our claim that the proposed algorithm's results are better than motion based or color based models. Similarly, we propose a simple method for object-object occlusion detection. Our object tracking algorithm is simple to implement and able to track

several objects in real time. The algorithm can re-identify a human which exits from one camera's FOV and re-enters in the scene again. Object re-identification gives benefit to track the objects without assigning new label, when they re-enter in same or other camera's FOV. In general, the algorithm gives satisfactory results, but there are situations in which it fails: if object's height is smaller than 20 pixels, in the case of very low video quality and when some objects appearance do not have enough difference. Finally, it cannot detect occlusions if a compact group of peoples enter in one camera's FOV.
Our 1-D appearance model uses spatio-color information of objects. Object appearance may be very different in multi-camera environments. We maximize an object similarity by calibrating camera's colors. In chapter 5, we proposed a inter-camera color calibration algorithm for non-overlapping camera environments. Our algorithm maps one camera's color information to another camera better than well known CBTF and MBTF techniques discussed in chapter 5.
In chapter 6, we combine our proposed algorithms for background segmentation (see chapter 3), object tracking and identification (see chapter 4), and camera color calibration in non-overlapping multi-camera environment (see chapter 5). We discuss the object re-identification performance using ROC properties precision and recall. The results discussed in the chapter illustrate that our proposed object tracking and identification algorithm works in a satisfactory manner in multi camera environments. The results also confirm that our proposed MCBTF camera color calibration technique increases object re-identification performance.

7.2 Future Works

During the development and evaluation of object tracking systems, several possibilities for future research are uncovered. Some of them may lead to further improvement of the proposed techniques. Other directions are the application of the proposed system as a part of other application areas.
Our MCB background modeling technique uses several parameters. Find the optimal value of each parameter is a difficult task. Automatic selection of optimized values of these parameters, will certainly increase the object detection performance. Similarly, stationary objects are absorbed in the background. A more sophisticated algorithm based on object recognition might be useful to overcome this problem.
Adding the object correspondence probability function like Bayesian belief network (BBN) may increase object re-identification performance in multi-camera environments.

If we know the probabilities of an object exiting from camera C_i to enter in camera C_j after some time t, will certainly increase the performance of the re-identification task. We are interested in finding an inter-camera color calibration to improve the recognition performance in mixed indoor/outdoor environments, where the significant changes of brightness can be problematic. Future works will also concern the extension of our object tracking algorithm to overlapping multi-camera environments. Similarly, updating the camera's color calibration after the training period can help to minimize the luminosity variation in the scene. Existing techniques for camera color calibration during the training time allows objects to move in the camera's FOV to calculate the BTF. This technique has the problem for calculating the perfect BTF as object's view angles in camera pair $C_i - C_j$ may be different in multi-camera environments. The appearance of a given object may be very different in different views, which can become the reason of incorrect BTF. The possible solution is to use some standard or customize luminosity charts instead of objects during the training time.

One possible application of object recognition is the compression of video data. If we detect objects in videos, then objects and background can be stored using different compression rates. The objects can be stored with high accuracy (thus, a low compression) and background with high compressions rates (leading to some loss of unimportant details). This will help to preserve good video quality of objects without significantly increasing video data size. Similarly, motion estimation techniques are also used to compress the video information. The detection of objects and their motion estimation studied in this thesis, can also be useful for video encoding.

An other application, we would have liked to explore is content based media retrieval. In this application, some particular object is selected and information about this object is retrieved in videos. Object behavior and its activity analysis also needs object recognition for further analysis.

Appendix A

Résumé en Français

Titre : Suivi et ré-identification d'objets dans des environnements multi-caméras

A.1 Résumé de la thèse

Le domaine de la vidéosurveillance a connu une très forte expansion ces dernières années. Mais la multiplication des caméras installées dans des espaces publics ou privés, rend de plus en plus difficile l'exploitation par des opérateurs humains des masses de données produites par ces systèmes. De nombreuses techniques d'analyse automatique de la vidéo ont été étudiées du point de vue de la recherche, et commencent à être commercialisées dans des solutions industrielles, pour assister les opérateurs de télésurveillance. Mais la plupart de ces systèmes considèrent les caméras d'une manière indépendante les unes des autres. L'objectif de cette thèse est de permettre d'appréhender la surveillance de zones étendues, couvertes par des caméras multiples, à champs non-recouvrants. L'un des problèmes auxquels nous nous sommes intéressés est celui de la ré-identification d'objets : lorsqu'un objet apparaît dans le champ d'une caméra, il s'agit de déterminer si cet objet a déjà été observé et suivi par l'une des caméras du système. Nous souhaitons effectuer cette tâche sans aucune connaissance a priori du positionnement des caméras les unes par rapport aux autres.

Il existe dans la littérature beaucoup d'algorithmes permettant le suivi des objets en mouvement dans une vidéo. Ces algorithmes sont suffisants pour détecter des fragments de la trajectoire et vérifier que les objets ont un mouvement cohérent. Par contre, ces algorithmes ne sont pas suffisamment robustes aux occultations, aux intersections, aux fusions et aux séparations. Cette insuffisance des algorithmes actuels pose problème, dans la mesure où ils forment les briques de base d'un suivi multi-caméras. Une première partie du travail de thèse a été donc de perfectionner les algorithmes de segmen-

tation et de suivi de façon à les rendre plus robustes.

Dans un premier temps, nous avons donc proposé une amélioration aux algorithmes de segmentation premier plan/arrière plan basés sur les dictionnaires (codebooks). Nous avons proposé une méthodologie d'évaluation afin de comparer de la manière la plus objective possible, plusieurs techniques de segmentation basées sur l'analyse de la précision et du rappel des algorithmes. En nous basant sur un jeu d'essai issu de bases de données publiques, nous montrons le bon comportement de notre algorithme modifié.

Une deuxième contribution de la thèse concerne l'élaboration d'un descripteur robuste et compact pour le suivi des objets mobiles dans les vidéos. Nous proposons un modèle d'apparence simplifié, appelé caractéristique verticale (VF pour Vertical Feature), indépendant de l'angle de vue et de la taille apparente des objets. Ce descripteur offre un bon compromis entre les modèles colorimétriques très compacts, mais qui perdent toute l'organisation spatiale des couleurs des objets suivis, et les modèles d'apparence traditionnels, peu adaptés à la description d'objets déformables. Nous associons à ce descripteur un modèle de mouvement des objets suivis, et montrons la supériorité d'une approche combinant ces deux outils aux approches traditionnelles de suivi, basées sur le mean shift ou sur le filtre de Kalman.

Chaque objet suivi par une caméra peut ainsi être associé à un descripteur. Dans le cadre du suivi multi-caméras, nous sommes confrontés à une certaine variabilité de ces descripteurs, en raison des changements des conditions d'éclairage, mais également en raison des caractéristiques techniques des caméras, qui peuvent être différentes d'un modèle à l'autre. Nous nous sommes donc intéressés au problème de l'étalonnage des couleurs acquises par les caméras, qui visent à rendre identiques les descripteurs d'un même objet observé par les différentes caméras du système. Les approches existantes estiment les fonctions de transfert de luminosité (BTF pour Brightness Tranfert Function) en mesurant la réponse donnée par chaque caméra à des objets connus. Nous comparons les méthodes basées sur une moyenne (MBTF) ou sur un cumul (CBTF) des histogrammes de couleur, et montrons les faiblesses de ces approches lorsque certaines couleurs sont trop peu représentées dans les objets servant à l'étalonnage. Nous proposons une alternative (MCBTF) dont nous montrons la supériorité par rapport aux méthodes existantes.

Enfin, des expérimentations systématiques sont menées sur le problème de la ré-identification d'objets dans un environnement multi-caméras, qui permettent de valider l'ensemble de nos propositions.

A.2 Problématique

Dans les applications de vidéosurveillance l'aspect multi-caméras commence à jouer un rôle important. Non seulement les objets en mouvement doivent être segmentés et suivis, mais la machine doit être capable de reconnaitre un même objet qui sort puis refait son apparition dans le champ d'une caméra, ou qui passe du champ d'une caméra à celui d'une autre caméra.
Il y a deux situations possibles pour gérer ces aspects multi-caméra : soit les champs de deux ou plusieurs caméras se recouvrent, soit les caméras ont des champs de vue disjoints. Dans le premier cas, le même objet est filmé par deux ou plusieurs caméras et donc les caractéristiques prises en compte doivent être peu sensibles au changement du point de vue et à la distance de l'objet par rapport aux caméras. Il y a aussi le problème des occultations qui peuvent éventuellement être levées en utilisant plusieurs caméras bien positionnées les unes par rapport aux autres. Nous avons choisi de ne pas utiliser de connaissance a priori sur la position de la caméra. Dans le deuxième cas, il faut reconnaitre un même objet qui passe devant plusieurs caméras avec des champs non-recouvrants.
Un autre problème est lié au fait que la couleur d'un objet qui passe devant différentes caméras change à cause de :

1. Différences dans le calcul du gain, l'optique et l'électronique
2. Marques de caméra différentes
3. Cameras installées à l'intérieur et/ou à l'extérieur d'un bâtiment

Comme la couleur des objets est une caractéristique importante pour la reconnaissance d'objets, l'utilisation d'une méthode de normalisation des couleurs entre les différentes caméras s'impose.

A.3 Travail réalisé

La performance des algorithmes de suivi dépend des techniques de segmentation. En début de ce travail de thèse, nous avons proposé un algorithme de modélisation d'arrière-plan, basé sur la méthode des codebooks. Cet algorithme a été publié dans la conférence AVSS [Ilyas et al., 2009].
Dans un deuxième temps, pour créer une "signature" pour chaque objet en mouvement, nous avons aussi proposé une caractéristique verticale basée sur une projection des couleurs de l'objet sur son axe principal. Cette caractéristique est relativement invariante

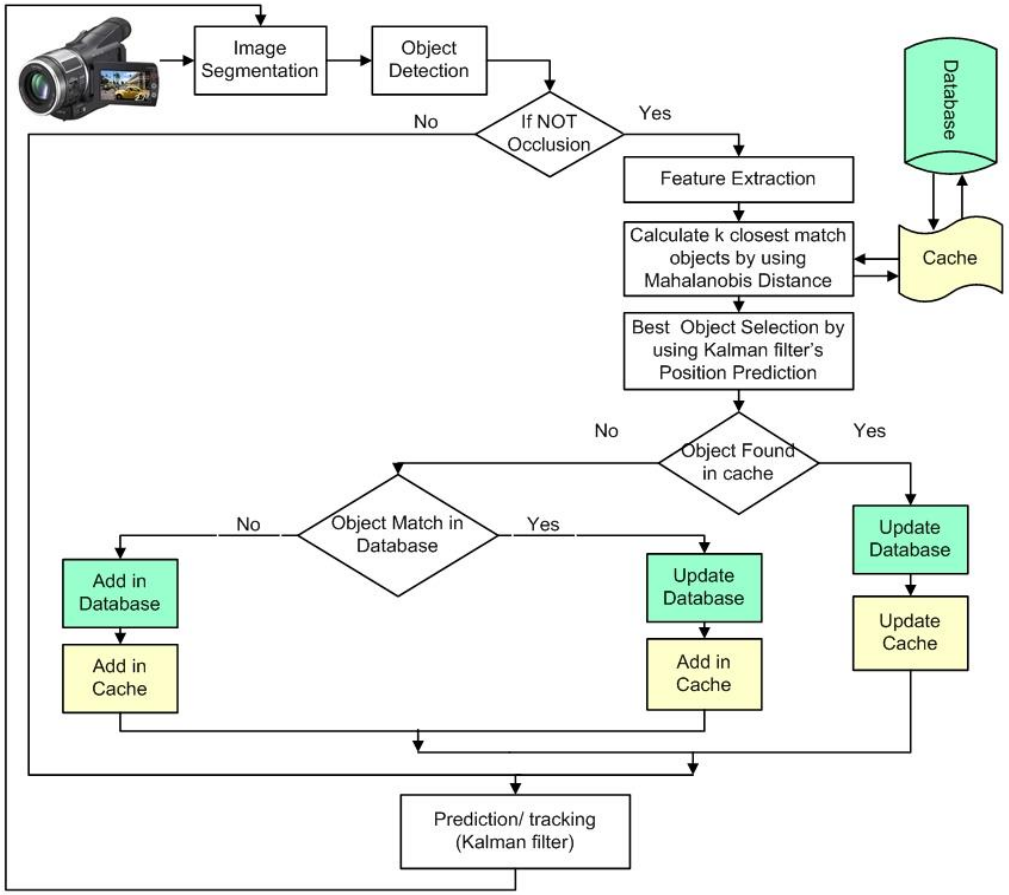

FIGURE A.1 – Méthode de suivi

au changement de point de vue de la caméra et à la distance de l'objet par rapport à la caméra, mais ne permet pas toujours un suivi robuste (par exemple dans le cas où dans l'image il y a beaucoup de personnes habillées dans des couleurs similaires). Pour le suivi, cette caractéristique est combinée avec la position de l'objet dans l'image d'une caméra et sa vitesse. Le schéma suivant (Figure A.1) présente la méthode de suivi des objets à l'intérieur d'une caméra. La caractéristique verticale de chaque objet est stockée dans une base de données pour pouvoir reconnaitre un objet qui sort du champ de la caméra et ensuite revient, ou un objet qui passe devant le champ de plusieurs caméras différentes. Pour gérer les cas d'occultation, nous utilisons un filtre de Kalman pour prédire la position de l'objet suivi, mais sans mettre à jour sa caractéristique verticale [Ilyas et al., 2010a].

Dans un troisième temps, nous avons travaillé sur la normalisation des couleurs dans un environnement multi-caméras. Nous avons fait la comparaison des techniques de calibration de couleur et nous avons proposé une amélioration de cette technique [Ilyas et al., 2010b].

A.3.1 Segmentation d'objets

La détection d'objets en mouvement et leur suivi est une étape essentielle de notre travail. Nous nous sommes concentrés dans un premier temps sur les méthodes de suivi d'objets mono caméra, dans le but de faire une comparaison et de sélectionner celle que nous pourrons étendre au cas multi-caméras. Après avoir passé en revue la littérature, nous avons finalement choisi deux méthodes de suivi d'objets en mouvement. [Kim et al., 2005] présentent une méthode pour la segmentation d'images en utilisant le "codebook" (CB). Cette technique montre des bons résultats. D'autres travaux de modélisation du fond, utilisant les mixtures de gaussiennes (MOG) sont présentées par [Stauffer et al., 2000].
Après avoir implémenté plusieurs algorithmes de modélisation de fond (en utilisant OpenCV, Microsoft Visual Studio V.8 sous Windows Vista), nous avons pu les comparer et identifier les points forts et les points faibles de chacune de ces méthodes. La conclusion a été que les deux méthodes les mieux adaptées à la modélisation du fond et donc au suivi d'objets en mouvement sont la mixture de gaussiennes [Stauffer et al., 2000] et le codebook [Kim et al., 2005]. En revanche, il existe certains cas dans lesquels les résultats de la mixture de gaussiennes et de la méthode codebook ne sont pas très bons : 1. lorsqu'une durée suffisante pour l'apprentissage n'est pas disponible. 2. quand de trop nombreux objets sont en mouvement dans la scène (cas des foules par exemple). 3. quand les objets ont une couleur trop proche de la couleur de fond.

A.3.1.1 Modification de la méthode "codebook" (MCB)

L'algorithme " codebook " fonctionne très bien lorsqu'aucun objet en mouvement n'est présent pendant la période d'apprentissage de l'arrière plan, ce qui ne peut pas toujours être assuré. Le deuxième problème est généré par le fait qu'une couleur appartenant aux objets en mouvement apparait également dans le fond de la scène. Le codebook adopte cette couleur comme l'un des codewords modélisant le fond alors que ce pixel appartient à l'objet. Pour surmonter cette difficulté, nous avons proposé une modification à l'algorithme de base, intégrant la "fréquence d'apparition" des objets, pour modifier la politique d'ajout ou de suppression d'un codeword dans le codebook.
Les résultats seront établis sur la base de nombreuses images dans différentes conditions d'éclairage, différentes situations environnementales et des nombres différents de personnes, avec des personnes en situation de déplacement mais aussi à l'arrêt. Quelques

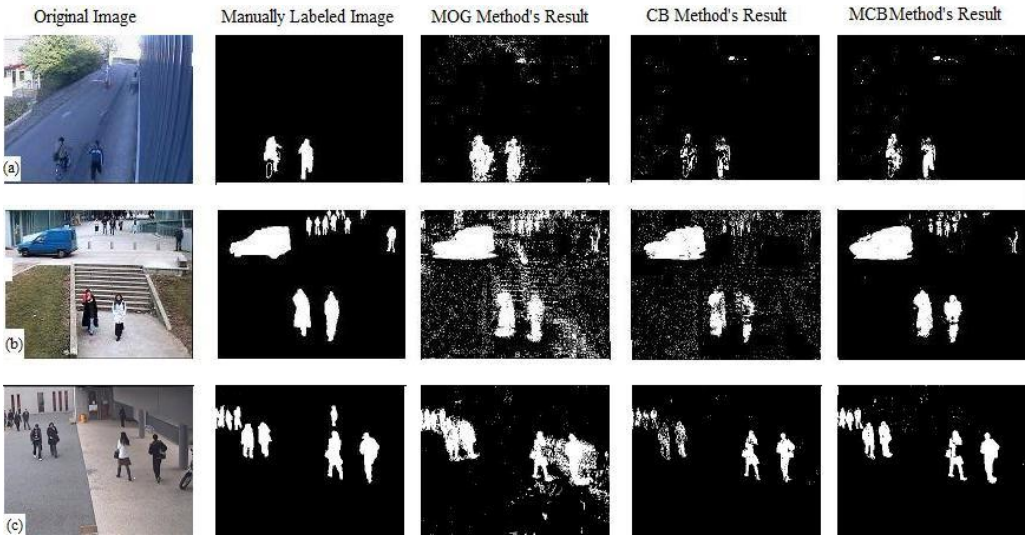

FIGURE A.2 – Les résultats de la segmentation

images sont présentées en Figure A.2.

A.3.1.2 Méthodologie de comparaison des différents algorithmes

Pour obtenir une analyse fiable il nous faut une vérité terrain et une comparaison des différentes méthodes en termes de rappel et précision. Nous avons développé un outil permettant de comparer de manière quantitative (donc plus objective que par une simple observation qualitative des images) les résultats des trois méthodes (mixture de gaussiennes, codebook et codebook modifié). Nous avons étiqueté manuellement plusieurs images test significatives, prises à intervalles de temps réguliers dans des séquences particulières, et avons comparé les résultats de notre étiquetage " manuel " aux résultats donnés par les trois algorithmes.

Ces résultats sont présentés sous la forme de courbes ROC. Nous basons nos calculs sur l'expression des vrais positifs (VP), des faux positifs (FP) et des faux négatifs (FN). Nous exprimons également la précision et le rappel pour chacune des images **??**. Ces expériences nous ont permis de construire une mesure quantitative de la performance des algorithmes comparés, et donc de valider nos résultats de manière objective. Les prin-

Method	*MOG*	*CB*	*MCB*
Facteur qualité utilisant précision et rappel (F)	28.28	24.15	**32.17**
Facteur de qualité utilisant le Coefficient de Jaccard	20.75	17.56	**25.22**

TABLE A.1 – Résultat des techniques de segmentation

cipaux résultats sont donnés dans le Tableau A.1, et démontrent ainsi la pertinence de nos propositions. Plus de précisions sur les mesures utilisées sont données dans l'article [Rosin and Ioannidis, 2003].

A.3.2 La reconnaissance d'objet

A.3.2.1 Présentation du problème

La reconnaissance d'objets est une autre étape nécessaire pour le suivi d'objets, principalement dans un contexte multi-caméras, mais également pour pouvoir ensuite effectuer des requêtes. Pour reconnaître les objets, il est nécessaire de les représenter par des de vecteurs de caractéristiques. Les techniques utilisées pour la reconnaissance d'objets dépendent de la situation. Nous nous intéressons au suivi de personnes en mouvement. La couleur est la caractéristique la plus importante, combinée avec une information spatiale. Les propriétés géométriques ne donnent pas des bons résultats pour des objets élastiques (non-rigides).
C'est la raison pour laquelle nous choisissons un modèle d'apparence 1-D pour la reconnaissance des objets (appelé caractéristique verticale). Dans des travaux similaires sur des caractéristiques d'objets, [Sato and Aggarwal, 2004] utilisent la taille, la texture verticale, la surface et l'accélération pour la reconnaissance d'objets en mouvement. L'algorithme de K. Sato et al. ne donne pas des bons résultats dans des situations réelles, parce que la taille/nombre de tranches des objet n'est pas fixé, donc il n'y a pas de normalisation des objets par rapport à leur taille apparente.
Les caractéristiques auxquelles nous nous sommes intéressés sont :

- La caractéristique verticale
- La position des objets dans la scène.
- La vitesse des objets dans la scène

La distance Mahalanobis est utilisée pour apparier l'objet courant avec un éventuel objet de la base de données. L'algorithme détaillé est présenté en Figure A.1.
Pour calculer la caractéristique verticale, nous considérons l'axe principal de l'objet. La hauteur de l'objet est fixée et l'objet est re-échantillonné si besoin. Les pixels de l'objet sont projetés sur l'axe principal. Pour chaque canal de couleur nous calculons la valeur moyenne. L'avantage d'utiliser la couleur moyenne, est d'obtenir un traitement plus robuste au changement d'angle de vue, et plus rapide en temps de calcul. La Figure A.3 représente les objets et représentation de leur caractéristique verticale.

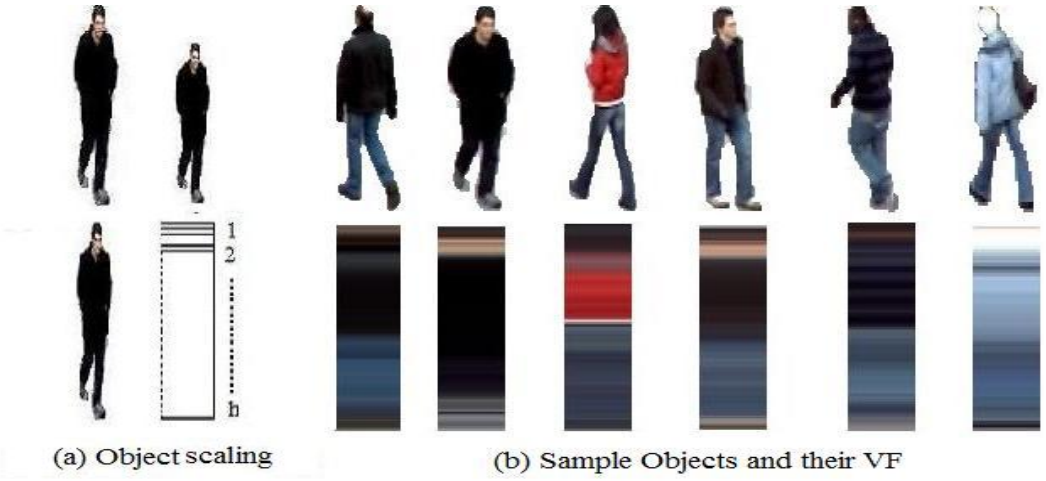

FIGURE A.3 – Des objets avec la représentation de leur caractéristique verticale

Dans un premier temps nous avons travaillé sur la représentation des objets en essayant de reconnaître les mêmes objets dans la même scène, mais filmés par deux caméras différentes. Ensuite nous avons travaillé sur un véritable environnement multi-caméras, qui filment des scènes différentes mais à travers lesquelles peuvent passer les mêmes objets. La Figure A.4 représente la reconnaissance des objets en séquences d'images filmées par une même caméra. La Figure A.5 montre le résultat dans un environnement multi-caméras. Comme on peut l'observer en Figure A.4 et en Figure A.5, notre technique de reconnaissance basée sur la caractéristique verticale, la position et la vitesse des objets donne de bons résultats, même en cas de couleurs similaires, de changement d'angle de vue et de changement de taille des objets.

A.3.2.2 Résultats de la reconnaissance d'objets

Pour évaluer notre technique de reconnaissance, nous avons étiqueté manuellement 7400 images de la base de données PETS, CAVIAR et VISOR. Nous avons fait la comparaison de notre technique avec d'autres méthodes qui utilisent seulement des caractéristiques de couleur ou de mouvement des objets. Parmi les caractéristiques de la base de données nous pouvons citer les éléments suivants :

- Il y a fusions et séparations des objets (occultations)
- Il y a des changements de l'angle de vue des objets
- Il y a des changements de l'intensité lumineuse globale
- Plusieurs personnes portent des vêtements de couleurs similaires.

FIGURE A.4 – Reconnaissance des objets qui sortent et reviennent dans la scène en mono caméra.

- Plusieurs personnes apparaissent à des endroits différents, dans la même vidéo
- Certains objets entrent et sortent puis reviennent dans la scène

Les résultats de plusieurs techniques de reconnaissance sont montrés dans le Tableau A.2. Notre technique donne des meilleurs résultats que les techniques qui utilisent seulement des caractéristiques de couleur ou de mouvement.
Ces premiers résultats illustrent qu'une utilisation combinée de la technique du " mean shift " et du " filtre de Kalman " donnent des résultats bien meilleurs que chacune des techniques prises séparément.
Enfin, le Tableau A.3 ci-dessous détaille le nombre de situations différentes que nous avons cherché à analyser, et met en évidence le petit nombre de cas dans lesquels notre

Base de Données	Kalman	Mean shift	Notre méthode
CAVIAR	93.27%	85.64%	96.72%
VISOR	81.27%	64.70%	89.02%
PETS	73.18%	67.04%	91.35%
Ensemble	81.57%	70.86%	91.97%
Suivi (fps)	85.63	7.18	39.32

TABLE A.2 – Résultat de Reconnaissance des objets et Suivi de 15 à 20 objets à chaque image de la séquence, et plus de 60 objets en base de données Configuration : Core Duo 1.86 GHz. Taille des images : 320 x 240

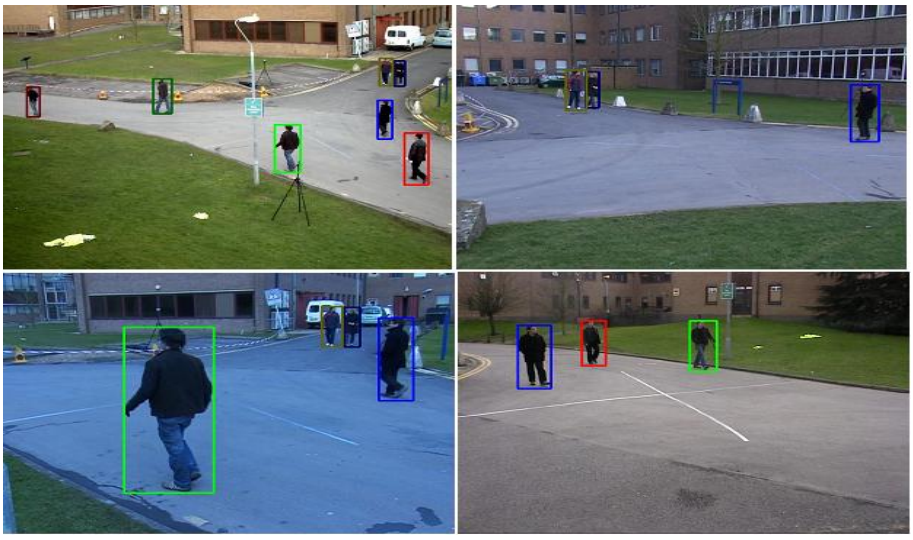

FIGURE A.5 – Reconnaissance des objets en multi-caméras.

Parameter of Recognition	*Result*
Nombre d'objets dans toutes les images	16011
Nombre d'objets reconnus	14725
Nombre d'objets non reconnus	1286
Pourcentage de réussite	91.96%
Nombre d'objets correctement reconnus après ré-apparition	56
Nombre d'objets incorrectement reconnus après ré-apparition	9
Reconnus après la fusion	246
Non reconnus après la fusion	16
Nombre d'objets correctement suivis en phase d'occultation	1700
Nombre d'objets incorrectement suivis en phase d'occultation	523

TABLE A.3 – Résultat de reconnaissance des objets

méthode échoue, même dans des situations complexes de ré-apparition des objets, et de séparation après une phase d'occultation.

A.3.3 Normalisation de couleurs pour plusieurs caméras

Dans notre cas, la couleur des objets est une caractéristique très importante pour la reconnaissance. La couleur d'un objet qui passe devant différentes caméras peut changer à cause des changements d'intensité de la lumière (soleil ou ombre), a cause du fait que les caméras sont installées à l'intérieur et/ou à l'extérieur d'un bâtiment.
La méthode choisie pour la normalisation des couleurs entre les différentes caméras est d'apprendre une fonction de transfert de luminosité : " brightness transfer function "

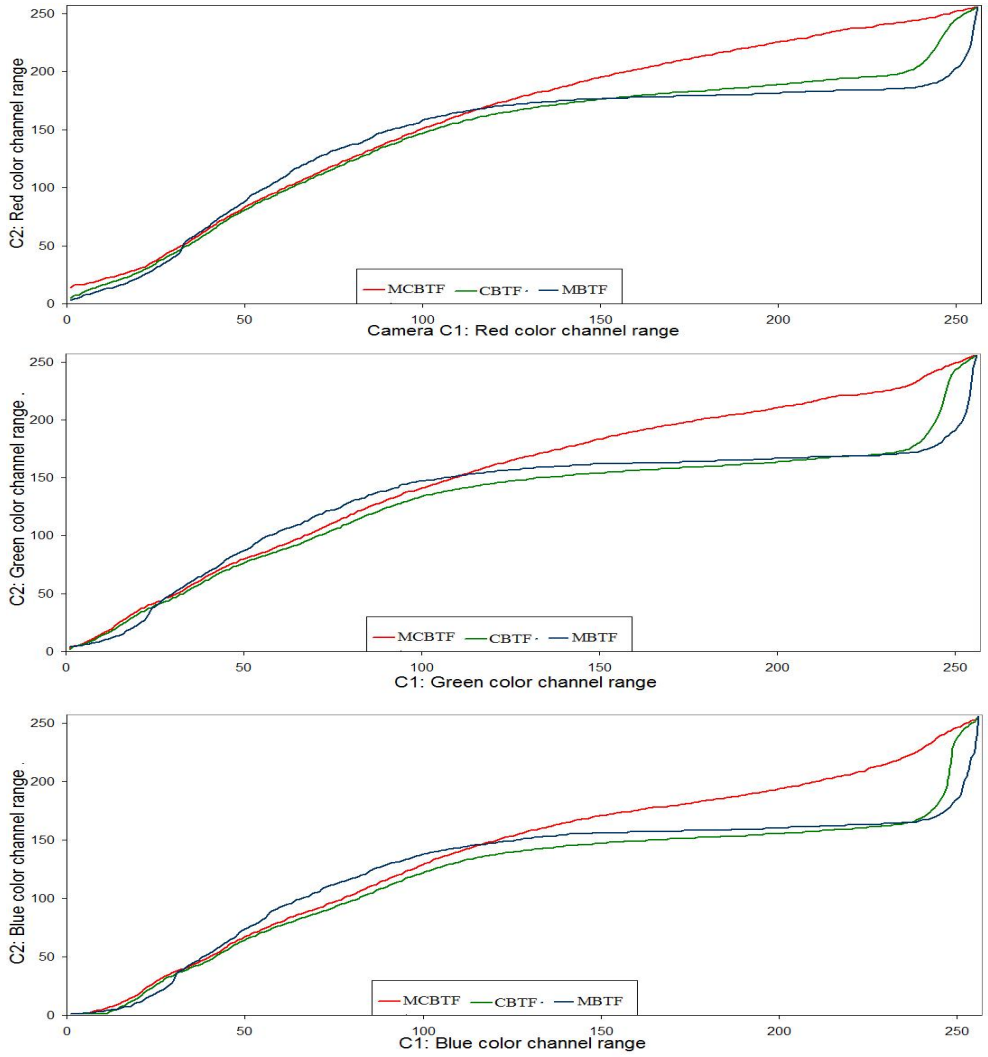

FIGURE A.6 – Courbe BTF pour la normalisation de couleur en multi caméra.

(BTF) entre deux ou plusieurs caméras avec les étapes suivantes :

1. Trouver l'histogramme de plusieurs objets identiques, passant devant les deux caméras.

2. Trouver l'histogramme moyen des histogrammes des objets. Pour ce calcul, nous ignorons les entrées de l'histogramme dont l'effectif est trop faible.

3. Trouver la distance minimale entre les histogrammes moyens d'un ensemble d'objets filmés par deux caméras.

La Figure A.6 donne les BTF entre deux caméras, pour chacun des trois canaux. Pour pouvoir établir de manière fiable une bonne BTF entre deux caméras, il faut utiliser pendant la période d'apprentissage, un grand nombre d'objets ou des objets présentant un nombre significatif de couleurs différentes. Notre technique est similaire à l'approche

présentée en [Prosser et al., 2008]. La différence consiste dans le fait que nous ignorons les entrées de l'histogramme n'étant pas représentées par un nombre suffisant de pixels. En Figure A.6 nous montrons les courbes de la méthode proposée [Ilyas et al., 2010b](en rouge) et de la méthode présentée en [Prosser et al., 2008] (en vert) et [Orazio et al., 2009] (en bleu).

La Figure A.7 montre que tous les objets filmés par une caméra Fuji sont plus foncés

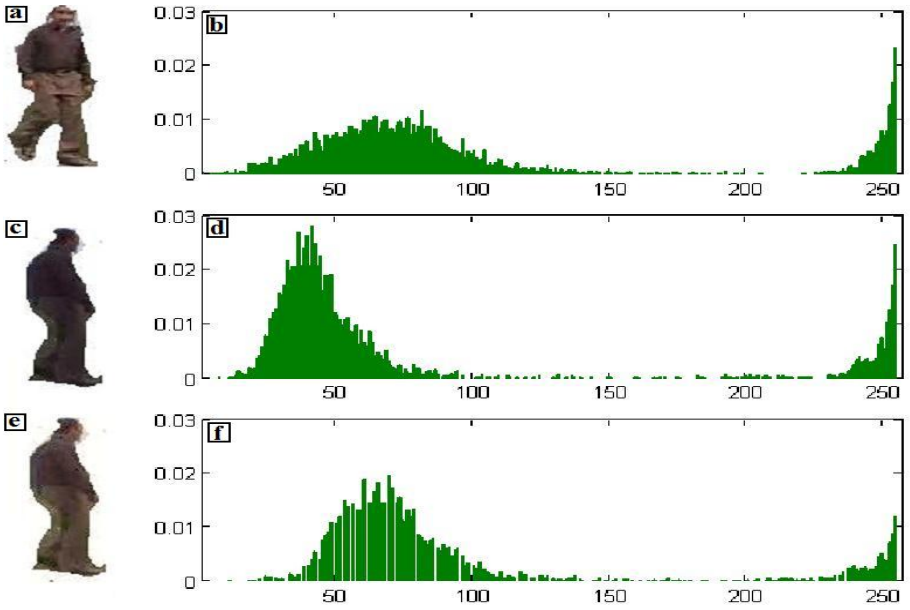

FIGURE A.7 – Calibration et correction de la couleur des objets en multi-caméras.

que ceux filmés par une caméra JVC, ce qui est mieux traduit par les courbes obtenues par notre méthode, en particulier pour les valeurs élevées des intensités.

A.3.4 Re-Identification humains dans un environnement multi-caméras champs non-recouvrants

[Ilyas et al., 2010c] combinons dans un système complet les algorithmes proposés pour la segmentation de fond (voir section A.3.1), le suivi d'objets et l'identification (voir section A.3.2), et l'étalonnage des couleurs caméra non-cumul environnement multi-caméras (voir section A.3.3). La Figure A.8 présente la méthodologie de re-identification et suivi d'objets dans un environnement multi-caméras à champs non-recouvrants. Nous calculons la BTF pour les paires de caméras C1-C2 et C1-C3 et pour les objets qui entrent dans le champ de chaque caméra pendant le temps de l'apprentissage. Egalement nous modélisons l'arrière de plan avec la méthode codebook modifiée pendant le temps d'apprentissage.

Après le calcul de MCBTF pour les paires de caméras C1-C2 et C1-C3, le système est

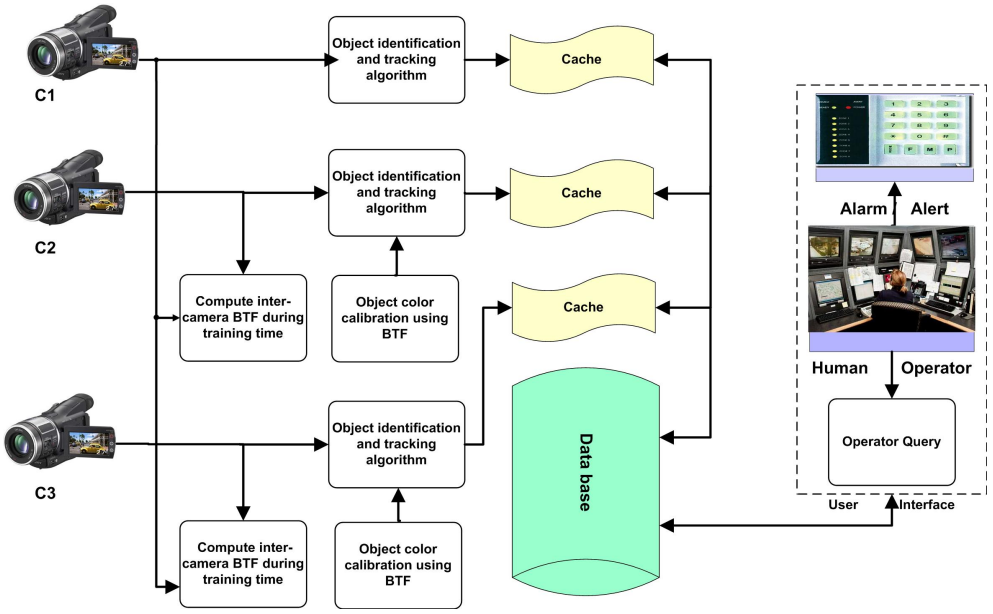

FIGURE A.8 – Méthode de l're-identification et suivi d'objets dans un environnement multi-caméras à champs non-recouvrants

prêt à effectuer le suivi d'objets et la ré-identification. Les images provenant des caméras C1, C2 et C3 sont segmentés en premier plan et arrière-plan à l'aide de la méthode codebook modifiée (MCB). Après la détection des objets en utilisant MCB, les couleurs des objets des caméras C2 et C3 sont corrigées avec les courbes de MCBTF. Nous utilisons l'algorithme de reconnaissance d'objets et de suivi expliqué dans la section A.3.2.

Pour les tests, nous utilisons 4000 images provenant des caméras C1, C2 et C3. Nous

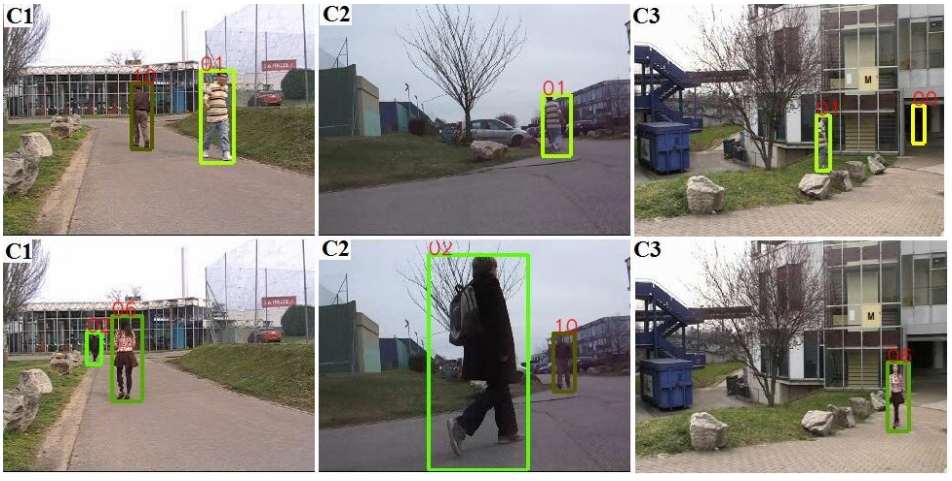

FIGURE A.9 – Ré-identification d'objets dans l'environnement multi-caméras à champs non-recouvrants

avons calculé les courbes d'étalonnage de couleurs entre les caméras C2 et C3 en utilisant plusieurs méthodes : MBTF, CBTF, MCBTF. Ensuite nous avons identifié automa-

tiquement tous les objets dans trois séquences vidéo (Figure A.9). Nous présentons les résultats de la ré-identification d'objets en utilisant la courbe basée sur la précision et le rappel. Les résultats présentés dans la Figure A.10 montrent que notre algorithme d'étalonnage des couleurs MCBTF augmente la performance de ré-identification d'objets par rapport à la non-utilisation d'étalonnage de couleurs (WCC) et aussi par rapport aux deux autres méthodes d'étalonnage de couleurs (CBTF et MBTF).

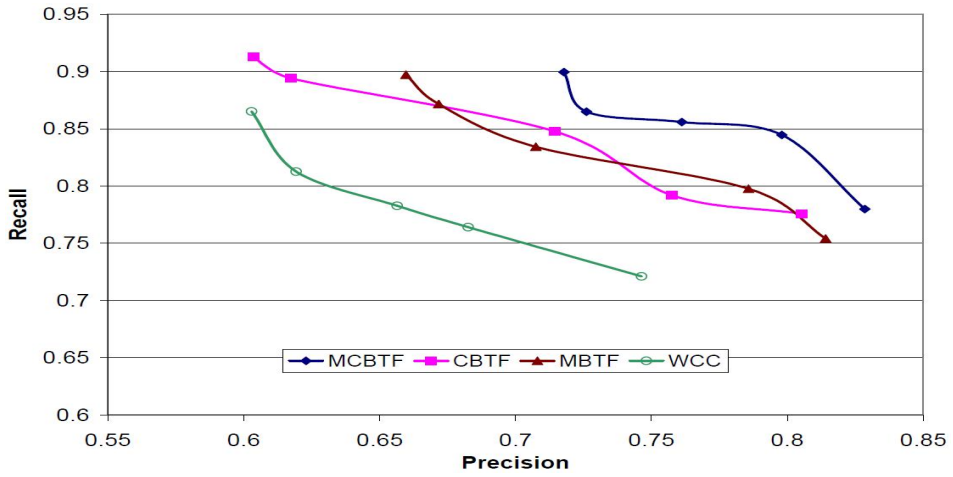

FIGURE A.10 – Courbe ROC pour la ré-identification d'objets avec et sans étalonnage des couleurs

A.4 Conclusion et perspectives

L'algorithme codebook modifié que nous avons proposé permet d'obtenir un meilleur modèle d'arrière-plan, donc une meilleure segmentation d'objets en mouvement. Un article a été publié sur ce sujet ([Ilyas et al., 2009]).

La caractéristique verticale proposée, utilisée avec la position et la vitesse des objets, permet d'obtenir une bonne reconnaissance des objets dans des vidéos filmées par plusieurs caméras. Notre algorithme de suivi d'objets est également capable de détecter les occultations [Ilyas et al., 2010a].

Nous avons enfin appliqué et amélioré une technique de calibration de couleurs pour plusieurs caméras, ce qui permet d'améliorer le taux de reconnaissance des objets [Ilyas et al., 2010b]. Un nouvel article sur la méthodologie de re-identification et suivi d'objets dans un environnement multi-caméras à champs non-recouvrants a été publié dans la conférence [Ilyas et al., 2010c] (à compléter).

Notre technique de modélisation du fond (MCB) utilise de nombreux paramètres, qui ont été déterminés de manière empirique à l'aide d'expérimentations nombreuses, mais

néanmoins non systématiques. Trouver la valeur optimale de chacun de ces paramètres est une tâche difficile. Des perspectives intéressantes à nos travaux pourraient être d'automatiser la recherche de valeurs optimisées de ces paramètres, grâce à des méthodes d'apprentissage. Des campagnes d'expérimentations systématiques permettront certainement d'accroître les performances de nos algorithmes de détection et de ré-identification d'objets.

Un autre axe de recherche que nous aimerions creuser consiste à effectuer des statistiques sur la correspondance entre les objets observés par différentes caméras. Des approches basées sur les réseaux bayésiens (BBN) peuvent sans aucun doute améliorer les performances de ré-identification des objets dans des environnements multi-caméra. Des travaux futurs porteront également sur la mise à jour de l'étalonnage des couleurs en fonction des changements de luminosité observés dans la scène.

Une application possible de la reconnaissance d'objets est la compression de données vidéo. Si nous détectons des objets dans les vidéos, les objets (mobiles) et le fond (statique) peuvent être stockés en utilisant des taux de compression différents.

Une autre application que nous aurions aimé explorer est la recherche par le contenu de données multimédia. Dans ce type d'applications, un objet particulier est sélectionné et des informations sur cet objet sont extraites dans les vidéos. L'ensemble des séquences faisant apparaître des objets à l'apparence ou au comportement similaire à l'objet requête peuvent alors être extraits de manière automatisée, et présentés à l'opérateur par ordre de ressemblance décroissante. Nous avons la conviction que la caractéristique verticale introduite dans cette thèse peut constituer un descripteur pertinent pour ce type de requêtes.

Bibliography

A. Alahi, P. Vandergheynst, M. Bierlaire, and M. Kunt. Cascade of descriptors to detect and track objects across any network of cameras. *CVIU*, 114(6):624–640, 2010. 30

C. Arth, C. Leistner, and H. Bishof. Object reacquisition and tracking in large-scale smart camera networks. In *IEEE International Conference on Smart Distributed Cameras*, 2007. 33

M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Statistics and Computing*, 50(2): 174–189, 2002. 25

G. P. Ashkar and J. W. Modestino. The contour extraction problem with biomedical applications. *Computer Graphics and Image Processing*, 7(3):331–355, 1978. 26

S. Bak, E. Corvee, F. Brémond, and M. Thonnat. Person re-identification using spatial covariance regions of human body parts. In *7th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, 2010. 34, 35

J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *Int. Journal of Computer Vision*, 12(1):43–77, 1994. 12

H. Bay, A. Ess, T. Tuytelaars, and LV Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008. 29

J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1000–1006, 1997. 34

S.T. Birchfield and S. Rangarajan. Spatiograms versus histograms for region-based tracking. In *CVPR05*, pages 1158–1163, 2005. 28, 60

M. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *Int. J. Comput. Vision*, 26(1):63–84, 1998. 29

I. Bouchrika, J. N. Carter, and M. S. Nixon. Recognizing people in non-intersecting camera views. In *3rd International Conference on Crime Detection and Prevention (ICDP 2009*, pages 1–6, 2009. 34

G. Bradski and J. Davis. Motion segmentation and pose recognition with motion history gradients. *Machine Vision and Applications*, 13(3):74–184, 2002. 11

A. Broggi, M. Bertozzi, A. Fascioli, and M. Sechi. Shape based pedestrian detection. In *IEEE Intelligent Vehicles Symposium*, pages 215–220, 2000. 13

R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signal and Applied Kalman Filtering*. John Wiley & Sons, Inc , USA, 2nd edition, 1992. 24

Q. Cai, A. Mitiche, and J.K. Aggarwal. Tracking human motion in an indoor environment. In *IEEE International Conference on Image Processing, ICIP03*, pages 215–218, 1995. 24

T. Camus. Real-time quantized optical flow. In *IEEE Conference on Computer Architectures for Machine Perception*, 1995. 12

J. N. Carter and M. S. Nixon. Measuring gait signatures which are invariant to their trajectory. *Measurement + Control*, 32(9):265–269, 1999. 34

R. L. T. Cederberg. Chain-link coding ans segmentation for raster scan devices. *Computer Graphics and Image Processing*, 10(3):224–234, 1979. 26

TH. Chalidabhongse, K. Kim, D. Harwood, and L. Davis. A perturbation method for evaluating background subtraction algorithms. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillances VSPETS*, 2003. 19, 20, 51

Y. T. Chen, C.S. Chen, C. R. Huang, and Y. P. Hung. Efficient hierarchical method for background subtraction. *Pattern Recognition*, 40(10):2706–2715, 2007. 20

D.E. Cheng and M. Piccardi. Matching of objects moving across disjoint cameras. In *ICIP*, pages 1769–1772, 2006. 36

G. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette for articulated objects and its use for human body kinematics estimation and motion capture. In *Computer Vision and Pattern Recognition, CVPR03*, 2003. 25

A.K.R. Chowdhury, A. Kale, , and A. Kale. Towards a view invariant gait recognition algorithm. In *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pages 143–150, 2003. 34

D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non rigid objects using mean shift. In *CVPR*, pages 142–149, 2000. 27, 71

D.N.T. Cong, L. Khoudour, C. Achard, and P. Phothisane. People re-identification by means of a camera network using a graph-based approach. *Machine Vision and Application*, 90:2362–2374, 2009. 17

T. Cootes, G. EDWARDS, and C. TAYLOR. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI*, 23(6):681Ű685, 2001. 27

R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts and shadows in video streams. *PAMI*, 25(10):1337–1342, 2003. 16

D. Cunado, M. S. Nixon, and J. N. Carter. Automatic extraction and description of human gait models for recognition purposes. *Computer Vision and Image Understanding*, 90(1):1–41, 2003. 34

J. Czyz, B. Ristic, and B. Macq. A particle filter for joint detection and tracking of color objects. *IVC*, 25(8):1271–1281, 2007. 25, 29, 30

S. D., Hordley, G. D. Finlayson, G. Schaefer, and G. Y. Tian. Illuminant and device invariant color using histogram equalization. *Pattern Recognition Letter*, 38(2):146–162, 2005. 35

A. Dailianas, R.B. Allen, and P. England. Comparison of automatic video segmentation algorithms. In *SPIE Conference on Integration Issues in Large Commercial Media Delivery Systems*, pages 2–16, 1995. 26

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, CVPR05*, pages 886–893, 2005. 22

J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *ICML*, pages 233–240, 2006. 19, 20, 51

P. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of SIGGRAPH*, pages 369–378, 1997. 36

J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Computer Vision and Pattern Recognition, CVPR00*, 2000. 25

Y. Dhome, N. Tronson, A. Vacavant, T. Chateau, C. Gabard, Y. Goyat, and D. Gruyer. A benchmark for background subtraction algorithms in monocular vision: a comparative study, 2010a. Benchmark data available at `http://gravir.univ-bpclermont.fr/visage/?q=node/44/` (access date: 12 September 2010),. xiii, 20

Y. Dhome, N. Tronson, A. Vacavant, T. Chateau, C. Gabard, Y. Goyat, and D. Gruyer. A benchmark for background subtraction algorithms in monocular vision: a comparative study. In *International Conference on Image Processing Theory, Tools and Applications, IPTA10*, 2010b. 20

P. Dickinson, A. Hunter, and K. Appiah. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *IEEE International Conference on Computer Vision*, pages 44–50, 2003. 14

P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *CVPR09*, pages 304–311, 2009. 27

G. Doretto, A. Chiuso, Y.N. Wu, , and S. Soatto. Dynamic textures. *IJCV*, 51(2):91–109, 2003. 16

A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000. 25

A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *6th European Conference on Computer Vision, ECCV00*, pages 751–767, 2000. xiii, 14, 15, 16, 43

A.E Elgammal and L.S. Davis. Probabilistic framework for segmenting people under occlusion. In *IEEE International Conference on Computer Vision, ICCV001*, pages 145–152, 2001. 24

A.M. Elgammal, R. Duraiswami, and L.S. Davis. Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. *PAMI*, 25(11):1499–1504, 2003. 27, 28

H. Elzein, S. Lakshmanan, and P. Watta. A motion and shape-based pedestrian detection algorithm. In *IEEE Intelligent Vehicles Symposium*, pages 500–504, 2003. 24

M. Enzweiler and D.M. Gavrila. Monocular pedestrian detection: Survey and experiments. *PAMI*, 31(12):2179–2195, 2009. 22, 60

P. Fieguth and D. Terzopoulos. Color-based tracking of heads and other mobile objects at video frame rates. In *CVPR97*, pages 21–27, 1997. 29

G. Foresti, C. Micheloni, L. Sindaro, P. Remagnino, and T. Ellis. Advanced image and video processing in active-based surveillance system. In *IEEE Signal Processing Magazine*, pages 25–37, 2005. 10

H. Freeman. On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, EC-10:260–268, 1961. 27

N. Funk and G. Bishop. *A Study of the Kalman Filter Applied to Visual Tracking*. University of Alberta, Chapel Hill, NC, USA, 2003. 25

T. Gandhi and M. Trivedi. Person tracking and reidentification: Introducing panoramic appearance map (pam) for feature representation. In *Machine Vision and Applications: Special Issue on Novel Concepts and Challenges for the Generation of Video Surveillance Systems*, 2007. 33

X. Gao, T.E. Boult, F. Coetzee, and V. Ramesh. Error analysis of background adaption. In *International Conference Computer Vision and Pattern Recognition, ICPRoo*, pages 503–510, 2000. 19

D. M. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *International Journal Computer vision*, 73(1):41–59, 2007. 22

D.M. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *ICCV99*, page 87Ű93, 1999. 29

D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE Transactions ON Pattern Analysis and Machine Intelligence, (PAMI)*, 32(2):1239–1258, 2010. 22, 27

N. Gheissari, T. Sebastian, and R. Hartley. Person reidentification using spatiotemporal appearance. In *International Conference on Computer Vision and Pattern Recognition,(CVPR)*, pages 1528–1535, 2006. 33

A. Gilbert, , and R. Bowden. Tracking objects across cameras by incrementally learning inter-camera calibration and patterns of activity. In *ECCV*, pages 125–136, 2006. 36

G. Gordon, T. Darrel, H. Harville, and J. Woodfill. Background estimation and removal based on range and color. In *Proceedings of the IEEE Computer Vision and Pattern Recognition*, 1999. 15

Y. Goyat, T. Chateau, L. Malaterre, and L. Trassoudaine. Vehicle trajectories evaluation by static video sensors. In *9th IEEE International Conference on Intelligent Transportation Systems*, pages 864–869, 2006. 15, 20

M. Grossberg and S. K. Nayar. What can be known about radiometric response function using images. In *Proceedings of of ECCV*, 2002. 36

O. Hamdoun, F. Moutarde, B. Stanciulescu, and B. Steux. Person re-identification in multi-camera system by signature based on interest point descriptors collected on short video sequences. In *ICDSCo8*, pages 1–6, 2008. 34, 36, 60

B. Han, D. Comaniciu, and L. Davis. Sequence kernel density approximation through mode propagation: Application to background modeling. *Proceedings of the IEEE Computer Vision and Pattern Recognition*, PP(99):1186–1197, 2008. 15

M. Heikkilä and M. Pietikäinen. A texture based method for modeling the background and detecting moving objects. *Transaction on Pattern Analysis and Machine Intelligence*, 28(4):657–662, 2006. 18

M. Holm. Towards automatic rectification of satellite images using feature based matching. In *International Geoscience and Remote Sensing Symposium*, pages 2439–2442, 1991. 26

B. K. P. Horn. *Robot Vision*. Cambridge, MA., MIT, USA, 1986. 12

B. K. P. Horn, , and B. G. Schunck. Determining optical flow. *Journal of Artificial Intelligence*, 17:185–203, 1981. 12

B.K.P. Horn and B.G. Rhunck. Determining optical flow: a retrospective. *Artificial Intelligence*, 59:81–87, 1993. 12

T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE International Conference on Computer Vision*, 1999. xiii, 16, 42

M. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8:179–187, 1962. 26

L. Huang and M. Wang. Image thresholding by minimizing the measures of fuzziness. *Pattern Recognition Letter*, 28:41–51, 1995. 13

P. S. Huang, C. J. Harris, and M. S. Nixon. Human gait recognition in canonical space using temporal templates. *IEEE Proceedings of Vision, Image and Signal Processing*, 146 (2):93–100, 1999. 34

A. Ilie and G.Welch. Ensuring color consistency across multiple cameras. In *Proceedings of the Tenth IEEE International Conference on Computer Vision, ICCV05*, pages 1268–1275, 2005. 36, 77

A. Ilyas, M. Scuturici, and S. Miguet. Real time foreground-background segmentation using a modified codebook model. In *AVSS09*, pages 454–459, 2009. 58, 113, 124

A. Ilyas, M. Scuturici, and S. Miguet. A combined motion and appearance model for human tracking in multiple cameras environment. In *6th IEEE International Conference on Emerging Technologies, ICET2010*, pages 198–203, 2010a. 58, 60, 114, 124

A. Ilyas, M. Scuturici, and S. Miguet. Object re-identification in multi camera environment. In *International Conference on Intelligence and Information Technology,ICIIT 2010*, 2010b. 78, 85, 114, 122, 124

A. Ilyas, M. Scuturici, and S. Miguet. Inter-camera color calibration for object re-identification and tracking. In *2nd International Conference of Soft Computing and Pattern Recognition (SoCPaR 2010)*, 2010c. 96, 122, 124

M. Isard and A. Blake. Condensation-conditional density propagation for visual tracking. *International Journal on Computer Vision*, 29(1):5–28, 1998. 25

M. Izadi and P. Saeedi. Robust region-based background subtraction and shadow removing using color and gradient information. In *19th International Conference on Pattern Recognition ICPR*, pages 1–5, 2008. xiii, 17, 18

R. Jain and H. Nagel. On the estimation of optical flow: Relations between different approaches and some new results. *IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI*, 1(2):299–324, 1979. 10

O. Javed, K. Shafique, and M. Shah. A hierarchical approach to robust background subtraction using color and gradient information. In *Motion02*, pages 22–27, 2002. 18

O. Javed, K. Shafique, , and M. Shah. Appearance modeling for tracking in multiple non-overlapping cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 26–33, 2005. 33

O. Javed, K. Shafique, Z. Rasheed, and M. Shah. Modeling inter-camera space-time and appearance relationships for tracking across non-overlapping views. *Comput. Vis. Image Underst.*, 109(2):146–162, 2008. xiii, 31, 33, 37, 78

G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14:201–211, 1973. 34

L. Juan and O. Gwun. A comparison of sift, pcasift and surf. *International Journal of Image Processing, IJIP*, 3:143–152, 2009. 29

A Kadyrov and M Petrou. Object descriptors invariant to affine distortions. In *British Machine Vision Conference, BMVC01*, pages 391–400, 2001. 26

P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for real time tracking with shadow detection. In *2nd European Workshop on Advanced Video Based Surveillance Systems*, 2001. 14, 20

R. E. Kalman. A new approach to linear filtering and prediction problems. *T-ASME*, pages 35–45, 1960. 24, 65, 71

J. Kapur, P. Sahoo, and A. Wong. A new method for gray level picture thresholding using the entropy of the histogram. *Computer Vision Graphics Image Process*, 29(3):273–285, 1985. 13

Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, pages 506–513, 2004. 34

R. Keeney and H. Raiffa. *Decisions with multiple objectives*. John Wiley & Sons, Inc , USA, 1st edition, 1976. 81

V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, pages 253–259, 1999. 33

S.M. Khan and M. Shah. Tracking multiple occluding people by localizing on multiple scene planes. *PAMI*, 31(3):505–519, 2009. 30, 64

K. Kim and L. Davis. Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In *Ninth European Conf. Computer Vision*, 2006. 29, 30

K. Kim, T. Thanarat, H. Chalidabbhognse, D. Harwood, and L. Davis. Real time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11 (3):172–185, 2005. xiii, 15, 16, 17, 42, 44, 46, 55, 115

T. Kohonen. Learning vector quantization. *Neural Networks*, 1:3–16, 1988. 44

J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *Proceedings of the Third IEEE International Workshop on Visual Surveillance (VS'2000)*, pages 3–10, 2000. 27

M. Lantagne, M. Parizeau, and R. Bergevin. Vip : Vision tool for comparing images of people. In *IEEE Conference on Vision Interface*, pages 35–42, 2003. 33

H. Lee and H. Ko. Detection of occluded multiple objects using occlusion activity detection and object association. *ISPACS*, pages 100–105, 2004. 25, 64

T. M. Lehmann, C. Gönner, and K. Spitzer. Survey: Interpolation methods in medical image processing. *IEEE Transaction Medical Imaging*, 18:1049–1075, 1999. 61

J.M. Lester, H.A. Williams, B.A. Weintraub, and J.F. Brenner. Two graph searching techniques for boundary finding in white blood cell images. *Computers in Biology and Medicine*, 8(4):293–308, 1978. 26

M. K. Leung and Y. H. Yang. Human body motion segmentation in a complex scene. *Pattern Recognition Letter*, 20:55–64, 1987. 10

L. Li, W. Huang, I.Y.H. Gu, , and Q. Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transaction on Image Processing*, 13(11):1459–1472, 2004. 18

H. Liu, T. Hong, M. Herman, and R. Chellappa. Motion-model-based boundary extraction. In *International Symposium on Computer Vision*, pages 587–592, 1995. 12

H. Liu, T.H. Hong, M. Herman, and R. Chellappa. Accuracy vs. efficiency trade-offs in optical flow algorithms. *Computer Vision and Image Understanding*, 7(3):271–286, 1998. xiii, 12

B.P.L. Lo and S.A. Velastin. Automatic congestion detection system for underground platforms. In *International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 158–161, 2001. 16

DG. Lowe. Object recognition from local scale-invariant features. In *ICCV99*, pages 1150–1157, 1999. 28, 60

DG. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 29

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981. 12

X. Luo and S. Bhandarkar. Multiple object tracking using elastic matchings. In *Advanced Video and Signal based Surveillance, AVSS05*, pages 123–128, 2005. 25

Y.F. Ma and H.J. Zhang. Detecting motion object by spatio-temporal entropy. In *IEEE Int. Conf. on Multimedia and Expo (ICME 2001)*, pages 379–382, 2001. 13

D. Makris, T. J. Ellis, and J. K. Black. Bridging the gaps between cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 32, 33

S. Mann and R. Picard. Extending dynamic range by combining different exposed pictures. In *Proceedings of Imaging Science and Technology*, pages 442–448, 1995. 36

A. Martelli. Edge detection using heuristic search methods. *Computer Graphics and Image Processing*, 1(2):169–182, 1972. 26

J. Martinez-Del-Rincon, C. Orrite-Urunuela, and JE. Herrero-Jaraba. An efficient particle filter for color-based tracking in complex scenes. In *Advanced Video and Signal Based Surveillance AVSS07*, pages 176–181, 2007. 25, 30

H. Medeiros, J. Park, and A. Kak. Distributed object tracking using a cluster-based kalman filter in wireless camera networks. *IEEE Journal of Selected Topics in Signal Processing*, 2(4):448–463, 2008. 25

I. Mikic, M. Trivedi, E. Hunter, and P. Cosman. Human body model acquisition and tracking using voxel data. *International Journal of Computer Vision*, 53(3):199–223, 2003. 25

A. Mittal and L.S. Davis. M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. *IJCV*, 51(3):189–203, 2003. 28, 30, 60, 64

T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, 2006. 22

H. Mori, N.M. Charkari, , and T. Matsushita. On-line vehicle and pedestrian detections based on sign pattern. *IEEE Transactions on Industrial Electronics*, 41:384–391, 1994. 13

B. Mughadam and A. Pentland. Probabilistic visual learning for object representation. *PAMI*, 9(7):696–710, 1997. 29

M. P. Murray. Gait as a total pattern of movement. *American Journal of Physical Medicine*, 46(1):290–333, 1967. 34

P. Nillius, J. Sullivan, and S. Carlsson. Multi-target tracking-linking identities using bayesian network inference. In *CVPR*, pages 2187–2194, 2006. 33

S.A. Niyogi and E.H. Adelson. Analyzing and recognizing walking figures in xyt. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR94*, pages 469–474, 1994. 13

N. Noceti, A. Destrero, A. Lovato, and F. Odone. Combined motion and appearance models for robust object tracking in real-time. *AVSS09*, pages 412–417, 2009. 30

K. Nummiaro, E. Koller-Meier, and L. V. Gool. An adaptive color-cased particle filter. *IVC*, 21(1):99–110, 2003. 29

T. D. Orazio, P. L. Mazzeo, and P. Spagnolo. Color brightness trasfer function evaluation for non overlapping multi camera tracking. In *International conference on distributed cameras (ICDS09)*, pages 1–6, 2009. 33, 38, 78, 104, 122

N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans Systems Man Cybernet*, 9:62–66, 1979. 10

U. Park, A. Jain, I. Kitahara, K. Kogure, and N. Hagita. Vise: Visual search engine using multiple networked cameras. In *18th International Conference on Pattern Recognition (ICPR'06)*, pages 1204–1207, 2006. 33

J. Parker. *Algorithms for Image Processing and Computer Vision*. John Wiley and Sons, Inc, NewYork, USA, 2nd edition, 1996. 13

N. Parragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, 46(3):223–247, 2002. 27

M. Petrou and P. Bosdogianni. *Image Processing: The Fundamental*. John Wiley and Sons, Inc, NewYork, USA, 2nd edition, 2010. 10

T. Pham, M. Worring, and A. Smeulders. A multi-camera visual surveillance system for tracking re-occurrences of people. In *International Conference on on Smart Distributed Cameras*, 2007. 33

M. Pic, L. Berthouze, and T. Kurita. Adaptive background estimation: Computing a pixel-wise learning rate from local confidence and global correlation. *IEICE Transaction Information and System*, E87-D(1):50–57, 2004. 15

M. Piccardi. Background subtraction techniques: a review. In *IEEE International Conference on Systems, Man and Cybernetics*, 2004. 16

F. Porikli. Inter-camera color calibration using cross-correlation model function. In *International Conference on Image Processing, 1CIP03*, pages 133–136, 2003. 80

F. Porikli and A. Divakaran. Multi-camera calibration, object tracking and query generation. In *ICME 03*, pages 653–656, 2003. xiv, 33, 36, 37, 62, 78, 81

B. Prosser, S.G. Gong, and T. Xiang. Multi-camera matching using bi-directional cumulative brightness transfer functions. In *BMVC08*, 2008. 33, 38, 78, 87, 104, 122

A. Rahimi and T. Darrell. Simultaneous calibration and tracking with a network of non-overlapping sensors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 32, 33

C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using kalman-filtering. In *Proceedings of International Conference on recent Advances in Mechatronics*, pages 193–199, 1995. 16, 42

T. Ridler and S. Calvard. Picture thresholding using an iterative selection method. *IEEE Transaction on Systems Man Cybernet*, 8:629–632, 1978. 10

G. X. Ritter and J. N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, USA, 2nd edition, 2000. 9

A. Rosenfeld and A. C. Kak. *Digital Picture Processing*. Academic Press, USA, 1976. 26

P. Rosin. Unimodal thresholding. *Pattern Recognition Letter*, 34(11):2083–2096, 2001. 10, 11

P. L. Rosin and E. Ioannidis. Evaluation of global image thresholding for change detection. *Pattern Recognition Letter*, 24:2345–2356, 2003. xiii, 10, 11, 19, 52, 117

P.M. Roth, H. Grabner, D. Skocaj, H. Bischof, and A. Leonardis. On-line conservative learning for person detection. In *IEEE Workshop on VS-PETS*, pages 223–230, 2005. 34

E. Salvador, P. Green, and T. Ebrahimi. Shadow identification and classification using invariant color model. In *Proceeding of IEEE ICASSP*, pages 1545–1548, 2001. 14

H. Sanchez-Cruz and RM. Rodriguez-Dagnino. Compressing bi-level images by means of a three-bit chain code. *Optical Engineering*, 44(9):97–104, 2005. 27

K. Sato and J.K. Aggarwal. Temporal spatio-velocity transform and its application to tracking and interaction. *CVIU*, 96(2):100–128, 2004. 28, 117

L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice Hall, 2002. 10

H.S. Sheshadri and A. Kandaswamy. Computer aided decision system for early detection of breast cancer. *Indian Journal of Medical Research*, 2006. 54

M. H. Sigari and M. Fathy. Real-time background modeling/subtraction using two-layer codebook model. In *International MultiConference of Engineers and Computer Scientists*, 2008. 20

P. H. A . Sneath and R. R. Sokal. *Numerical Taxonomy: the Principles and Practice of Numerical Classification*. W. H. Freeman, San Francisco:, 1973. 19

S. Soatto, G. Doretto, and Y.N. Wu. Dynamic textures. In *Proceedings of International Conference on Computer Vision, ICCV01*, page 439Ű446, 2001. 16

C. Stauffer, W. Eric, and L. Grimson. Learning patterns of activity using real time tracking. *Transaction on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000. xiii, 14, 15, 16, 20, 42, 43, 115

C. Su and A. Amer. A real-time adaptive thresholding for video change detection. In *International Conference on Image Processing, ICIP06*, pages 157–160, 2006. 10

N. Thome and S. Miguet. A robust appearance model for tracking human motion. In *Advanced Video and Signal Based Surveillance AVSS*, pages 528–533, 2005. 14, 28, 43

N. Thome, D. Merad, and S. Miguet. Human body part labeling and tracking using graph matching theory. In *AVSS06*, pages 38–43, 2006. 28

G.D. Tian and A.D. Men. An improved texture-based method for background subtraction using local binary patterns. In *CISP09*, pages 1–4, 2009. 18

Y. Tian, M. Lu, and A. Hampapur. Robust and efficient foreground analysis for real-time video surveillance. In *IEEE Conference on Computer Vision and Pattern Recognition CVPR05*, pages 1182–1187, 2005. 17

F. D. Torre, C. Vallespi, P.E. Rybski, M. Veloso, and T. Kanade. Learning to track multiple people in omni-directional video. In *International Conference on Robotics and Automation99*, page 4150Ű4155, 2005. 29

W. Tsai. Moment-preserving thresholding. *Computer Vision Graphics Image Processing*, 29: 377–393, 1985. 10

O. Tuzel, F. Porikli, and P. Meer. A bayesian approach to background modeling. In *In Conference on Computer Vision and Pattern Recognition, CVPR*, 2005. 20

L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 48–57, 2004. 33

C. Vazquez, M. Ghazal, and A. Amer. Occlusion and split detection for object tracking in surveillance applications. In *Proceeding of SPIE, EI*, pages 1–12, 2007. 30, 64

N. Verbeke and N. Vincent. A pca-based technique to detect moving objects. In *SCIA*, pages 641–650, 2007. 11

P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal Computer vision*, 33(2):153–161, 2005. 22

C. J. Walder and B. C. Lovell. Face and object recognition and detection using color vector quantization. In *Fourth Australasian Workshop on Signal Processing and Applications*, pages 27–30, 2002. 28

H. Wang and D. Suter. A consensus-based method for tracking: Modelling background scenario and foreground appearance. *Pattern Recognition*, 40(3):1091Ű1105, 2007. 8

Ji. Wang, Ze Wang, Geoffrey K. Aguirre, and John A. Detre. To smooth or not to smooth? roc analysis of perfusion fmri data. *Magnetic resonance imaging*, 23(1):75–81, 2005. 19, 54

Y. Wei, J. Sun, X. Tang, and H. Yeung. Interactive offline tracking for color objects. In *ICCV07*, pages 1–8, 2007. 28

G. Welch and G. Bishop. *An Introduction to the Kalman Filter*. University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995. 25, 66

C. Wöhler and J. Anlauf. An adaptable time-delay neural-network algorithm for image sequence analysis. *IEEE Trans. Neural Networks*, 10(6):1531–1536, 1999. 22

D. Xu and H.Li. Geometric moment invariants. *Pattern Recognition Journal*, 41(1):240–249, 2008. 26

L.Q. Xu and D.C. Hogg. Neural networks in human motion tracking - an experimental study. *Image and Vision Computing*, 15:607Ű615, 1997. 24

R. Yager. On the measure of fuzziness and negation. *International Journal on General Systems*, 5:221–229, 1979. 13

S. Yasutomi and H. Mori. A method for discriminating of pedestrian based on rhythm. In *International Conference on Intelligent Robots and Systems*, page 988Ű995, 1994. 24

A. Yilmaz, O. Javed, , and M. Shah. Object tracking: A survey. *ACM Computer Survey*, 38(4):1–45, 2006. ISSN 0360-0300. xiii, 22, 23

L. Zhao and C. Thorpe. Stereo and neural network-based pedestrian detection. *IEEE Transactions on Intelligent Transportation Systems*, 1(3):298–303, 2000. 29

J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *IEEE International Conference on Computer Vision (ICCV)*, pages 44–50, 2003. 16

L. J. Zhu, J. N. Hwang, and H. Y. Cheng. Tracking of multiple objects across multiple cameras with overlapping and non-overlapping vie. In *IEEE International Symposium on Circuits and Systems, (ISCAS)*, pages 1056–1060, 2009. 32, 33

S. Zhu and A. Yuille. Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI*, 18(9):884–900, 1996. 27

Author's Publications

International Conferences

1. A. Ilyas, M. Scuturici, and S. Miguet. Real time foreground-background segmentation using a modified codebook model. *6th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS 2009)*, pages 454-459, 2009.

2. A. Ilyas, M. Scuturici, and S. Miguet. A combined motion and appearance model for human tracking in multiple cameras environment. *In 6th IEEE International Conference on Emerging Technologies, ICET2010*, pages 198-203, 2010.

3. A. Ilyas, M. Scuturici, and S. Miguet. Object re-identification in multi camera environment. *In International Conference on Intelligence and Information Technology, ICIIT2010* 2010,

4. A. Ilyas, M. Scuturici, and S. Miguet. Inter-camera color calibration for object re-identification and tracking. *In 2nd International Conference of Soft Computing and Pattern Recognition (SoCPaR 2010)*, 2010.

Invited Talk

1. S. Miguet, A. Ilyas, I. Pop, R. L Robinault, M. Scuturici, M.N Thome. Analyse de l'activité humaine dans les séquences vidéo. *In Ecole de Préparation à la Recherche Appliquée : Vidéosurveillance Industrielle et Sécuritaire*, Ile de Kerkennah, Tunisie. 2010.

Submitted Article

1. A. Ilyas, M. Scuturici, S. Miguet. Human Tracking and Re-identification in a Distributed Camera System. *In Journal of Supercomputing, Springer* (2011)