

Caifeng Shan  
Fatih Porikli  
Tao Xiang  
Shaogang Gong (Eds.)

# Video Analytics for Business Intelligence

**Editor-in-Chief**

Prof. Janusz Kacprzyk  
Systems Research Institute  
Polish Academy of Sciences  
ul. Newelska 6  
01-447 Warsaw  
Poland  
E-mail: kacprzyk@ibspan.waw.pl

Caifeng Shan, Fatih Porikli, Tao Xiang,  
and Shaogang Gong (Eds.)

# Video Analytics for Business Intelligence

*Editors*

Dr. Caifeng Shan  
Philips Research  
Eindhoven  
The Netherlands

Dr. Fatih Porikli  
Mitsubishi Electric Research Laboratories  
Cambridge  
USA

Dr. Tao Xiang  
Dept. Computer Science  
Queen Mary University of London  
UK

Prof. Shaogang Gong  
Dept. Computer Science  
Queen Mary University of London  
UK

ISSN 1860-949X  
ISBN 978-3-642-28597-4  
DOI 10.1007/978-3-642-28598-1  
Springer Heidelberg New York Dordrecht London

e-ISSN 1860-9503  
e-ISBN 978-3-642-28598-1

Library of Congress Control Number: 2012933768

© Springer-Verlag Berlin Heidelberg 2012

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media ([www.springer.com](http://www.springer.com))

# Preface

Closed Circuit TeleVision (CCTV) cameras have been increasingly deployed pervasively in public spaces including retail centres and shopping malls. Intelligent video analytics aims to automatically analyze content of massive amount of public space video data and has been one of the most active areas of computer vision research in the last two decades. Current focus of video analytics research has been largely on detecting alarm events and abnormal behaviours for public safety and security applications. However, increasingly CCTV installations have also been exploited for gathering and analyzing business intelligence information, in order to enhance marketing and operational efficiency. For example, in retail environments, surveillance cameras can be utilised to collect statistical information about shopping behaviour and preference for marketing (e.g., how many people entered a shop; how many females/males or which age groups of people showed interests to a particular product; how long did they stay in the shop; and what are the frequent paths), and to measure operational efficiency for improving customer experience. In an airport, there is an urgent need for real-time measuring of congestion and queuing length in the departure security checking area and immigration control area, for improving efficiency and customer satisfaction. Despite the enormous potential for non-security oriented commercial applications, video analytics for business intelligence gathering and analysis has just started to receive attention and remains an under-explored area in the research community.

Compared to security oriented visual surveillance, business intelligence driven video analysis applications have different requirements and characteristics. For example, in security applications it is crucial to detect abnormal or suspicious behaviors, often in real-time. On the contrary, business intelligence applications focus on monitoring normal events (e.g., people entering a shop), and it is often sufficient to capture and store the observation for offline analysis. Furthermore, for security applications, emphasis is put on accurately evaluating each individual event. In contrast, business intelligence information is normally collected based on events measured statistically and holistically over a long period. Due to these differences, existing algorithms developed for security related applications may not be readily applicable to business intelligence applications. Innovative adaptation of existing techniques and/or development of novel approaches are required.

This book presents the latest developments on video analytics for business intelligence applications. It provides both academic and commercial practitioners an understanding of the state-of-the-art and a resource for potential applications and

practice. The book is intended to be accessible to a broader audience working on computer vision and video analysis applications.

Eleven peer-reviewed book chapters are included in the book, which covers different related topics. We have divided the chapters into four parts, each addressing a specific theme. The four themes presented include computational vision, demographics, behaviour analysis, and systems.

## Part I: Computational Vision

Video analytics in dynamic business environments attempts to detect, track, and recognize objects of interest from multiple videos, and more generally to interpret their behaviors and activities. In order to accomplish these, various computer vision and pattern recognition techniques are needed for building a computational framework for video analytics.

Detecting and tracking objects are key building blocks for a video analytics system. Under the business intelligence notion, an object can be a face, a head, a human, a queue of people, a crowd as well as a product on an assembly line. Chapter 1 presents a review of object detection and tracking. The main trends and taxonomy of popular methods for object detection, object modeling, and object tracking are introduced to give an insight into the underlying ideas of those methods as well as to show their limitations. This review supports a deeper appreciation of many applications presented in the rest of the book.

Camera calibration could help video analytics significantly in camera networks; however, a convenient calibration is still a challenge on its own. A calibration framework for large networks including non-overlapping cameras is introduced in Chapter 2, which relies on visual information coming from walking people. Since non-overlapping scenarios make point correspondences impossible, time constancy of a person's motion introduces the missing complementary information. The framework obtains calibrated cameras starting from single camera calibration thereby bringing the problem to a reduced form suitable for multi-view calibration. It extends the standard bundle adjustment by a smoothness constraint to avoid the ill-posed problem.

With video cameras installed almost ubiquitously in business environments, vast amounts of video data are generated. How to efficiently search and retrieve video data is one of challenges for video analytics applications. Recently Approximate Nearest Neighbor (ANN) methods have become popular for example-based video search, providing a trade-off between the accuracy of finding nearest-neighbors and the computational complexity. State-of-the-art Euclidean ANN methods do not perform well when applied to non-Euclidean datasets. Chapter 3 presents algorithms for performing ANN on manifolds by explicitly considering the Riemannian geometry of the non-Euclidean manifold and by taking advantage of the kernel trick in non-Euclidean spaces where performing Riemannian computations is expensive. The proposed methods are able to retrieve similar objects with very low complexity.

## Part II: Demographics

Collecting demographic information about customers in a business environment is an important part for business intelligence gathering and analysis, for instance, how many people visited a shopping mall; how many males and females visited; and how long they stayed. Instead of hiring humans to manually observe the customers, a computational system can be developed to automatically collect the demographic information by detecting the presence and analyzing the behaviour of people in the videos captured by cameras installed in a business environment.

Chapter 4 provides a detailed review on the computational approaches to human age and sex classification. Various methods for feature extraction and learning are discussed. The relation between age estimation and sex classification is also described, which is useful when designing a large-scale system. Major challenges and future research directions are discussed at the end, to inspire new research and investigation towards developing a working system.

Counting people from video is a challenging problem, and the difficulties lie at unconstrained imaging conditions such as illumination variation and low (spatial and/or temporal) resolution. Chapter 5 addresses the problem of counting mostly static people in indoor conditions with varying poses and illumination using very low frame rate video. Illumination issues are handled at the pixel level using photometry-based normalization, while the pose and low movement issues are addressed at feature level by exploiting the spatio-temporal coherence that is present among small body part movements. The motion of body parts is accumulated using a spatio-temporal autoregressive (AR) model to arrive at blob representations that are further grouped into people counts.

In public places, crowds indicate congestion, delay, instability, or abnormal events such as a fight, riot and emergency. Crowd information provides business intelligence, e.g., the distribution of people throughout spaces, throughput rates, and local densities. Chapter 6 introduces a scene-invariant crowd counting approach that uses local features to estimate the crowd size and its distribution across a scene. The use of local features allows the algorithm to calculate local occupancy statistics, scale to conditions which are unseen in the training data, and be trained on significantly less data. The algorithm uses camera calibration to scale features between multiple viewpoints, by taking into account the relative sizes of objects in these scenes.

In Chapter 7, soft biometrics (colour, height and luggage) are exploited to determine operational statistics relating to how people move through a space (e.g., dwell time). Soft biometrics, including traits such as height, weight, gender, hair, skin and clothing colour, can be used to describe, but not uniquely identify an individual. These traits can provide identification at long range, and aid in object tracking and detection in disjoint camera networks. An average soft biometric is proposed in this chapter to locate people who look distinct; these people are then detected at various locations within a disjoint camera network to gradually obtain operational statistics.

## Part III: Behaviour Analysis

One important problem for video analytics is to screen hundreds of hours of video for activity patterns that potentially impact the business. Chapter 8 presents algorithms that analyze surveillance video to automatically recognize various functional elements, such as walkways, roadways, parking-spots, and doorways, through their interactions with pedestrian and vehicle detections. The recognized functional element regions provide a means of capturing statistics related to particular businesses. For example, the owner may be interested in the number of people that enter or exit their business versus the number of people that walk past. Results shown on functional element recognition and business related activity profile extraction demonstrate the effectiveness of these algorithms.

Group monitoring is an important topic for video analytics. By considering relational connections among people, group modeling can provide more meaningful semantic description of the visual events. In Chapter 9, groups of people and group activities are analyzed from a social signaling perspective. Several aspects of groups are considered: 1) the life of a group, analyzing how the presence of a group can be detected in crowded situations (i.e., the birth and the death of a group), 2) how a moving group can be tracked (its evolution), and 3) which basic activities are carried out by their components in terms of interactions between the humans and the environment. In particular, the regions of the environment where the attention of humans is more focused can be detected.

## Part IV: Systems

Many issues need to be addressed when designing and validating a video analytics system for business intelligence applications, for example, video content browsing and retrieval, user interfaces, multi-modal sensor fusion, etc. In chapter 10, the ObjectVideo team presents an overview of the state of the art approaches that can be used for designing video analytics systems for business intelligence. Existing algorithms and techniques are described in detail for extraction and processing of target and scene information, multi-sensor cross camera analysis, inferring of simple, complex and abnormal video events, video data mining and retrieval, intuitive UIs for efficient customer experience, and text summarization of visual data. Concluded with the applications, this chapter provides a full picture of the design and implementation of video analytics for business intelligence.

Monitoring queue statistics like average wait time, average service time and queue length helps businesses enhance service efficiency, improve customer satisfaction and increase revenue. Chapter 11 presents the systematic design and validation of a general solution for automated queue statistics estimation. The design takes into account multiple variables such as queue geometry, service-counter type, illumination dynamics, camera viewpoints, people appearances, etc. These variables are addressed by decomposing the main task into two subtasks, queue size estimation and service time estimation, and combining their outputs. The validation process for a particular deployment of the proposed solution including the evaluation metrics and the obtained results is discussed.

## Acknowledgements

The preparation of this book has required the devotion of many people. We would like to thank all the authors for their tremendous effort and dedication in preparing the book chapters. We would also like to express our sincere thanks to all of the reviewers; their critical yet constructive feedback helped shape the final book. Finally, our special thanks go to Thomas Ditzinger and Holger Schäpe at Springer-Verlag Heidelberg for their constant support throughout the preparation of this book.

### Editors

Dr. Caifeng Shan

Philips Research, Eindhoven, The Netherlands

Email: caifeng.shan@philips.com

Dr. Fatih Porikli

Mitsubishi Electric Research Laboratories, Cambridge, USA

Email: fatih@merl.com

Dr. Tao Xiang

Queen Mary, University of London, UK

Email: txiang@eeecs.qmul.ac.uk

Prof. Shaogang Gong

Queen Mary, University of London, UK

Email: shaogang.gong@eeecs.qmul.ac.uk

# Contents

## Part I: Computational Vision

<b>Object Detection and Tracking .....</b>	3
<i>Fatih Porikli, Alper Yilmaz</i>	
<b>Auto-calibration of Non-overlapping Multi-camera CCTV Systems .....</b>	43
<i>Cristina Picus, Roman Pflugfelder, Branislav Micusik</i>	

<b>Fast Approximate Nearest Neighbor Methods for Example-Based Video Search .....</b>	69
<i>Rizwan Chaudhry, Yuri Ivanov</i>	

## Part II: Demographics

<b>Human Age Estimation and Sex Classification .....</b>	101
<i>Guodong Guo</i>	

<b>People Counter: Counting of Mostly Static People in Indoor Conditions .....</b>	133
<i>Amit Khemlani, Kester Duncan, Sudeep Sarkar</i>	

<b>Scene Invariant Crowd Counting and Crowd Occupancy Analysis .....</b>	161
<i>David Ryan, Simon Denman, Sridha Sridharan, Clinton Fookes</i>	

<b>Identifying Customer Behaviour and Dwell Time Using Soft Biometrics .....</b>	199
<i>Simon Denman, Alina Bialkowski, Clinton Fookes, Sridha Sridharan</i>	

## Part III: Behaviour Analysis

<b>Automatic Activity Profile Generation from Detected Functional Regions for Video Scene Analysis .....</b>	241
<i>Eran Swears, Matthew Turek, Roderic Collins, A.G. Amitha Perera, Anthony Hoogs</i>	

<b>Analyzing Groups: A Social Signaling Perspective . . . . .</b>	271
<i>Loris Bazzani, Marco Cristani, Giulia Paggetti, Diego Tosato, Gloria Menegaz, Vittorio Murino</i>	

## Part IV: Systems

<b>Video Analytics for Business Intelligence . . . . .</b>	309
<i>Asaad Hakeem, Himaanshu Gupta, Atul Kanaujia, Tae Eun Choe, Kiran Gunda, Andrew Scanlon, Li Yu, Zhong Zhang, Peter Venetianer, Zeeshan Rasheed, Niels Haering</i>	
<b>Design and Validation of a System for People Queue Statistics Estimation . . . . .</b>	355
<i>Vasu Parameswaran, Vinay Shet, Visvanathan Ramesh</i>	
<b>Author Index . . . . .</b>	375

**Part I**

**Computational Vision**

# Object Detection and Tracking

Fatih Porikli and Alper Yilmaz

**Abstract.** Detecting and tracking objects are among the most prevalent and challenging tasks that a surveillance system has to accomplish in order to determine meaningful events and suspicious activities, and automatically annotate and retrieve video content. Under the business intelligence notion, an object can be a face, a head, a human, a queue of people, a crowd as well as a product on an assembly line. In this chapter we introduce the reader to main trends and provide taxonomy of popular methods to give an insight to underlying ideas as well as to show their limitations in the hopes of facilitating integration of object detection and tracking for more effective business oriented video analytics.

**Keywords:** detection, tracking, representations, descriptors, features.

## 1 Introduction

Visual surveillance in dynamic business environments attempts to detect, track, and recognize objects of interest from multiple videos, and more generally to interpret object behaviors and actions. For instance, it aims to automatically compute the flux of people at public areas such as stores and travel sites, and then attain congestion and demographic analysis to assist in crowd traffic management and targeted advertisement. Such intelligent systems would replace the traditional surveillance setups where the number of cameras exceeds the capacity of costly human operators to monitor them.

---

Fatih Porikli  
Mitsubishi Electric Research Laboratories, USA  
e-mail: [fatih@merl.com](mailto:fatih@merl.com)

Alper Yilmaz  
The Ohio State University, USA  
e-mail: [yilmaz.15@osu.edu](mailto:yilmaz.15@osu.edu)

Proceeding with a low-level image features to high-level event understanding approach, there are three main steps of visual analytics: detection of objects and agents, tracking of such objects and indicators from frame to frame, and evaluating tracking results to describe and infer semantic events and latent phenomena. This analogy can be extended to other applications including motion-based recognition, access control, video indexing, human-computer interaction, and vehicle traffic monitoring and navigation. This chapter reviews fundamental aspects of the detection and tracking steps to support a deeper appreciation of many applications presented in the rest of the book.

Imagine waiting for your turn in a shopping line at a busy department store. Your visual system can easily sense humans and identify different layers of their interactions. As with other tasks that our brain does effortlessly, visual analytics has turned long out to be entangled for machines. Not surprisingly, this is also an open problem for visual perception.

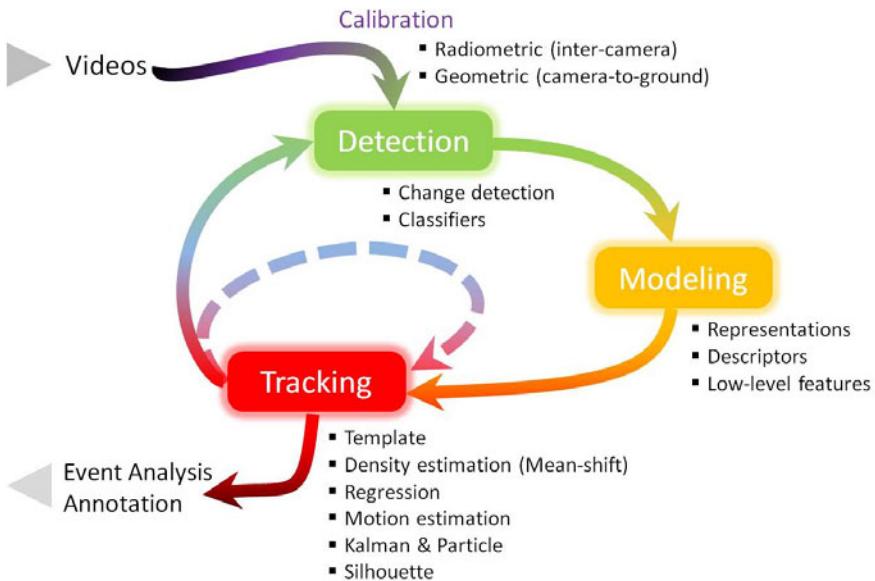
The main challenge is the problem of variability. A visual detection and tracking system needs to generalize across huge variations in object appearance such due for instance to viewpoint, pose, facial expressions, lighting conditions, imaging quality or occlusions while maintaining specificity to not claim everything it sees are objects of interest. In addition, these tasks should preferably be performed in real-time on conventional computing platforms.

One can ask the question which of the two tasks is easier and which comes first? Within detection, motion changes and appearance cues can be used to distinguish objects, which typically renders it relatively easily, and tracking techniques are often triggered by detection results. Combination of statistical analysis of visual features and temporal motion information usually lead to more robust approaches. For systems that face noisy conditions, however, tracking is suggested to be followed by detection to gather sufficient statistic as several track-before-detect algorithms propose. Besides, tracking steer to choose detection regions, source and sink areas. In any case, it has been common in the past few years, to assume that different strategies are required for these different tasks. Here we take the theoretical view that detection and tracking, rather than being two distinct tasks, represent two points in a spectrum of generalization levels.

Figure 1 illustrates a procedural flow of these reciprocal tasks intertwined with the object modeling. In the following sections, we attempt to give an overview of the popular choices for the object detection, modeling, and tracking stages for which a plethora of solutions have been evidently and inevitably produced over the past several decades.

## 2 Object Detection

Object detection is essential to initialize tracking process. It is continually applied in every frame. A common approach for moving object detection is to use temporal information extracted from a sequence of images, for instance by



**Fig. 1** Fundamental tasks of a video analytics framework based on object detection and tracking.

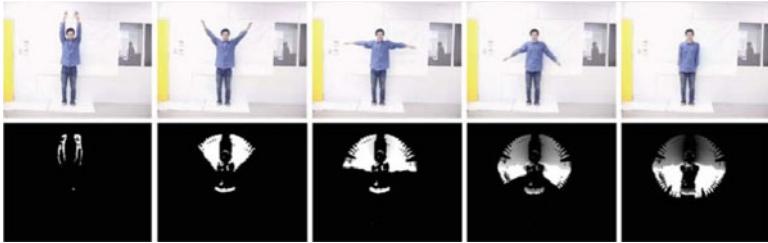
computing inter-frame difference, learning a static background scene model and comparing it with the current scene, or finding high motion areas. Another popular approach to object detection is to slide a window across the image (possibly at multiple scales), and to classify each such local window as containing the target or background. Alternatively, local interest points are extracted from the image, and then each of the regions around these points can be classified, rather than looking at all possible subwindows.

## 2.1 Change Detection

Change detection is the identification of changes in the state of a pixel through the examination of the appearance values between sets of video frames. Some of the most commonly used change detection techniques are frame differencing, background subtraction, motion segmentation, and matrix decomposition.

### 2.1.1 Frame Differencing and Motion History Image

Frame differencing is the intensity dissimilarity between two frames assuming that intensity change of a pixel apparently indicate something changing in the image, e.g. a moving object. It simply subtracts the current image from



**Fig. 2** Motion history images obtained by aggregating frame differences (© A. Ahad, 2010, Springer).

the previous or a reference image. It has been well studied since the late 70s. In general, the frame difference method is sensitive to illumination changes and it cannot detect an object once it becomes stationary.

Motion History Image [1] is obtained by successive layering of frame differences. For each new frame, existing frame differences are decreased in value subject to some threshold and the new silhouette (if any) is overlaid at maximal brightness. Motion history can also be reconstructed by foreground masks. It has the advantage that a range of times from frame-to-frame to several seconds may be encoded in a single image. A sample motion history image is shown in Figure [2].

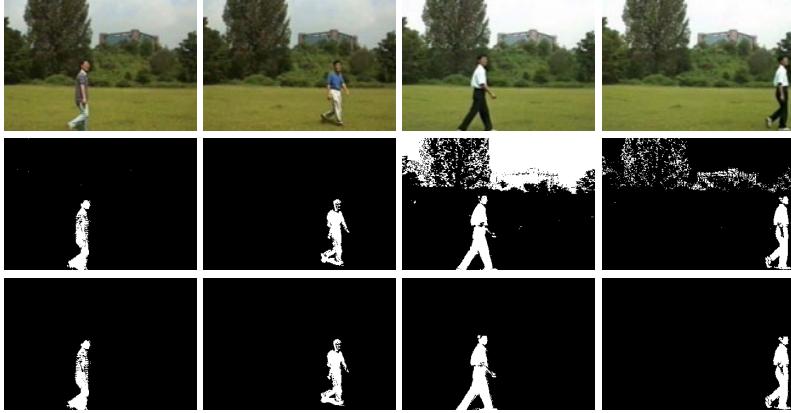
### 2.1.2 Background Subtraction

Detection can be achieved by building a representation of the scene, called the background model, and then observing deviations from this model for each incoming frame. Any gradual change from the background model is assumed to signify a moving object.

Earlier approaches use simple filters to make a prediction of background pixel intensities. In [2] Kalman filtering is used to model background dynamics. Similarly, in [3] Wiener filtering is used to make a linear predictions at the pixel level. Foreground regions consisting of homogeneous color are filled in at the region level, and in case most of the pixels in a frame exhibit sudden change, the background models are assumed no longer valid at the frame level. At this point either a previously stored pixel based background model is swapped in, or the model is reinitialized.

To learn changes in time, [4] proposed to model each pixel with a single Gaussian distribution. Once this background model is derived by updating in several consecutive frames, the likelihood of current pixel color coming from the corresponding model is computed, and the pixels that deviate significantly from their models are labeled as the foreground pixels.

However, a single Gaussian is not a good model for dynamic scenes [5] as multiple colors may be observed at a pixel due to repetitive object motion,



**Fig. 3** Sample foreground detection results for GMM (middle row) and Layered model with Bayesian Update (bottom row). When the background changes more than the preset variance of the GMM it produces false alarms.

shadows or reflectance. A substantial improvement is achieved by using multi-modal statistical models to describe background color. For instance, in [6] use a Gaussian Mixture Model (GMM) to represent a background pixel. GMM compares a pixel in the current frame with every model in the mixture until a matching Gaussian is found. If a match is found the mean and variance of the matched Gaussian is updated, otherwise a new Gaussian with the mean equal to the current pixel color and some initial variance is introduced into the mixture.

As an alternative to mixture models, a Bayesian approach that model each pixel as a combination of layered Gaussians is proposed in [7]. Recursive Bayesian estimation is performed to update the background parameters to better preserve the multi-modality of the background model than the conventional expectation maximization fitting, and to automatically determine the number of active layers for each pixel. Moving regions, which are detected using the GMM and layered approaches are shown in Figure 3.

Instead of relying only a pixel, GMM's can be trained to incorporate extended spatial information. In [8] a non-parametric kernel density estimation is used to refit a density function every time to multiple previous pixel values. During the subtraction process the current pixel is matched not only to the corresponding background model but also to the nearby pixel locations. Thus, some robustness against camera jitter or small movements in the background is obtained. Similar affects can be achieved by extending the support to larger blocks and using texture features that are less sensitive to inter-frame illumination variations. Although nonparametric models are robust against small changes they are computationally and memory-wise expensive. Besides, extending the support causes small foreground objects to disappear.

A shortcoming of above background methods is that they neglect the temporal correlation among the previous values of a pixel. This prevents them detecting a structured or near-periodic changes, for example the alternating signals in an intersection, the motion of plants driven by wind, the action of waves on a beach, and the appearance of rotating objects.

A frequency decomposition based background generation that explicitly harnesses the scene dynamics is proposed in [9]. To capture the cyclostationary behavior of each pixel, the frequency coefficients of the temporal variation of pixel intensity are computed in temporal windows and a background model that is composed of frequency coefficients is maintained and fused with distance maps to eliminate trail effects.

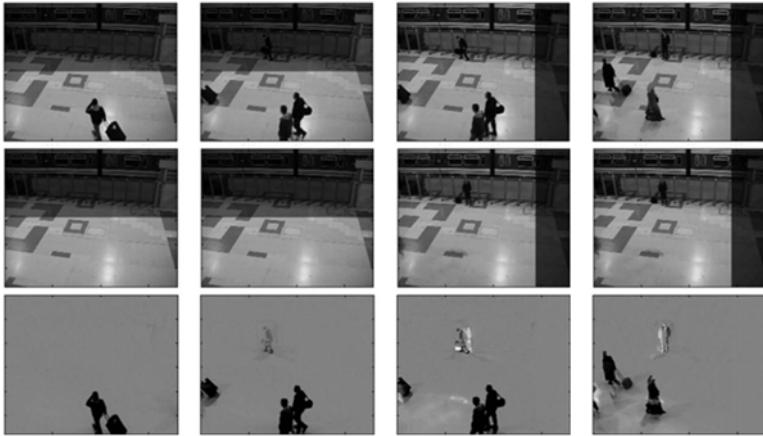
An alternate approach is to represent the intensity variations of a pixel in an image sequence as discrete states and using Hidden Markov Model (HMM), and switching among these states with the observations to classify pixels [10]. The advantage of using HMMs is that certain events, which may not be modeled correctly by unsupervised algorithms, can be learned using the provided training samples.

### 2.1.3 Motion Segmentation

Motion segmentation refers to the assignment of groups of pixels to various classes based on the speed and direction of their movements. Most approaches to motion segmentation first seek to compute the optical flow of the image sequence. Discontinuities in the optical flow can help in segmenting images into regions that correspond to different objects. Optical flow can arise from relative motion of objects and the camera. Using the rate of change of the image intensity and by assuming that the brightness function changes smoothly, the flow velocity is attained by minimizing a global error function. By assuming a locally constant flow, robustness against noise at the expense of the resolution of their optical field can be improved. From representing moving objects using sets of overlapping layers obtained by k-means clustering to variational methods that jointly solves the problems of motion estimation and segmentation for two consecutive frames in a sequence, many different flow based motion segmentation methods exist.

One way is to estimate the consistency of optical flow over a short duration of time [11] where the significant variation of accumulated local optical flows represent the dynamic features of nonstationary background objects. Towards the modeling of the dynamic characteristics, in [12] optical flow is computed and utilized as a feature in a higher dimensional space. In order to properly utilize the uncertainties in the features, a kernel based multivariate density estimation technique that adapts the bandwidth according the uncertainties in the test and sample measurements is incorporated.

Optical flow computation will be in error if the constant brightness and velocity smoothness assumptions are violated. In real imagery, their violation is quite common. Typically, the optical flow changes dramatically in highly



**Fig. 4** Background and foreground decompositions are obtained by matrix decomposition.

textured regions, around moving boundaries, at depth discontinuities, etc. Resulting errors propagate across the entire optical flow solution.

#### 2.1.4 Matrix Decomposition

Instead of modeling the variation of individual pixels, the whole image can be vectorized and used in background modeling. In [13] a holistic approach using eigenspace decomposition is proposed. For a certain number of input frames, a background matrix is formed by cascading the rows in each frame one after the other, and eigenvalue decomposition is applied to the covariance of the matrix. The background is then represented by the most descriptive eigenvectors, that encompass all possible illuminations to decrease sensitivity to illumination. The foreground objects are detected by projecting the current image to the eigenspace and finding the difference between the reconstructed and actual images.

Instead of the conventional background and foreground definition, an intrinsic image inspired method that decomposes a scene into time-varying background and foreground intrinsic images are proposed in [14]. The multiplication of these images reconstructs the scene. First, a set of previous images are formed into a temporal scale and their spatial gradients are computed. By taking advantage of the sparseness of the filter outputs, the background is estimated by median filtering of the gradients, the corresponding foreground is found using the background. The intrinsic background/foreground decomposition is robust even under sudden and severe illumination changes, yet computationally expensive.

A learning-based background subtraction approach based on the theory of sparse representation and dictionary learning is proposed in [15]. This

method makes the following two important assumptions: (1) the background of a scene has a sparse linear representation over a learned dictionary; (2) the foreground is sparse in the sense that majority pixels of the frame belong to the background. These two assumptions enable handling both sudden and gradual background changes. To build a correct background model when training samples are not foreground-free, a robust dictionary learning algorithm is used.

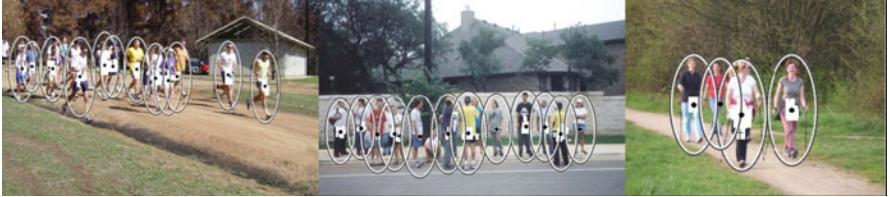
Learning based foreground and background decompositions for a sequence where the background significantly changes (upper and right half of the images are distorted) are shown in Figure 4.

## 2.2 Classifiers

Object (e.g. human) detection can be performed by learning a classification function that captures the variation in object appearances and views from a set of labeled training examples in a supervised framework. The input of the classification function is a test region and the output is the estimated label. i.e. object or not. Selection of features plays an important role in the performance of the classification, hence, it is important to use a set of features that discriminate one class from the other. Once the features are selected, different learning approaches can be applied including, but are not limited to, neural networks, boosting, decision trees, and support vector machines. These learning methods compute a hypersurface that separates one object class from the other in a high dimensional space.

Boosting is an iterative method of finding a very accurate classifier by combining many base classifiers, each of which may only be moderately accurate [16]. In the training phase of the AdaBoost algorithm, the first step is to construct an initial distribution of weights over the training set. The boosting mechanism then selects a base classifier that gives the least error, where the error is proportional to the weights of the misclassified data. Next, the weights associated with the data misclassified by the selected base classifier are increased. Thus the algorithm encourages the selection of another classifier that performs better on the misclassified data in the next iteration. In the context of object detection, weak classifiers can be simple operators such as a set of thresholds, applied to the object features extracted from the image windows.

Support Vector Machines (SVM) are used to cluster data into two classes by finding the maximum marginal hyperplane that separates one class from the other [17]. The margin of the hyperplane, which is maximized, is defined by the distance between the hyperplane and the closest data points. The data points that lie on the boundary of the margin of the hyperplane are called the support vectors. Despite being a linear classifier, SVM can also be used as a non-linear classifier by applying the kernel trick to the input feature vector extracted from the input. Application of the kernel trick to a set of data



**Fig. 5** Detection examples. The classifier is trained on the INRIA data set. White dots show all the detection results. Black dots are the modes generated by mean shift smoothing, and the ellipses are average detection window sizes. There are extremely few false positives and negatives.

that is not linearly separable, transforms the data to a higher dimensional space which is likely to be separable. The kernels used for kernel trick are polynomial kernels or radial basis functions, e.g. Gaussian kernel, and two layer perceptron, e.g. a sigmoid function. However, the selection of the right kernel for the problem at hand is not easy. Once a kernel is chosen one has to test the classification performance for a set of parameters which may not work as well when new observations are introduced to the sample set.

Leading approaches in classification based object detection can be separated into two groups based on the search technique. The first group of methods is based on sequentially applying a classifier at all the possible sub-windows in a given image. In [18], a polynomial support vector machine (SVM) was learned using Haar wavelets as object descriptors. Later, the work was extended to multiple classifiers trained to detect object parts, and the responses inside the detection window are combined to give the final decision [19]. In [20], a human detector was described by training an SVM classifier using densely sampled histogram of oriented gradients inside the detection window. In a similar approach [21], near real time detection performances were achieved by training a cascade model using histogram of oriented gradients (HOG) features by taking advantage of the integral histograms [22]. Similar to still images, in [23], a moving human detection algorithm was described using Haar wavelet descriptors but extracted from space-time differences in video. The operators in the space-time domain are in the form of frame differencing which encode some form of motion information, and frame differencing, when used as an operator in the temporal domain, is assumed to reduce the number of false detections by enforcing object detection in the regions where the motion occurs.

Instead of Haar wavelets or HOG features, region covariance matrices [24] are utilized as object descriptors in [25]. A region was represented by the covariance matrix of image features, such as spatial location, intensity, higher order derivatives, etc. Similarly, an object is modeled with several covariance matrices of overlapping regions. Since these descriptors do not lie on a vector space, conventional machine learning techniques are not adequate to learn

the classifiers. The space of nonsingular covariance matrices can be represented as a connected Riemannian manifold. For classification of points lying on a Riemannian manifold a manifold learning method is presented by incorporating the a priori information about the geometry of the space. Typical human detection results generated by this method is shown in Figure 5.

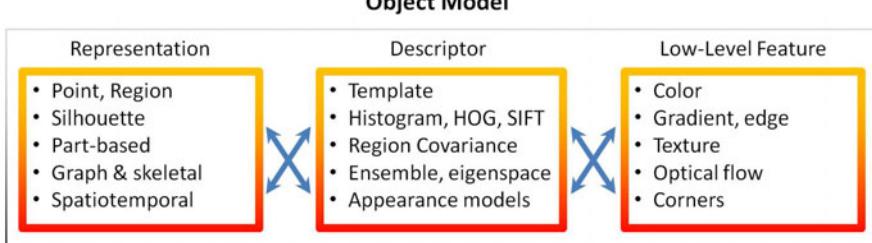
The second group of methods is based on detecting object parts [26] or common shapes [27] and assembling these local features according to geometric constraints to form the final model. In [28], parts were represented by co-occurrences of local orientation features and separate detectors were trained for each part using AdaBoost. Object location was determined by maximizing the joint likelihood of part occurrences combined according to the geometric relations. A human detection system for crowded scenes was described in [29]. The approach combined local appearance features and their geometric relations with global cues by top-down segmentation based on per pixel likelihoods. Other approaches include using silhouette information either in matching [30] or in classification framework [31].

Supervised learning methods usually require a large collection of samples from each object class. A possible approach to reduce the amount of manually labeled data is to accompany co-training to supervised learning [32]. The main idea behind co-training is to train two classifiers using a small set of labeled data, where the features used for each classifier are independent. After training is achieved, each classifier is used to assign unlabeled data to the training set of the other classifier. Co-training has been used to reduce the amount of manual interaction required for training in the context of AdaBoost [33] and SVM [34].

An alternative classifier based detection of the foreground regions is attempted using corner-based background models [35]. Instead of processing every pixel, this algorithm detects a certain number of feature points using a Harris corner detector and a scale-invariant feature point descriptor. It dynamically learns a single background model and classify each extracted feature as either a background or a foreground feature using a motion tracker to differentiate motion consistent foreground points from background points with random or repetitive motion. Since feature point extraction and descriptor generation are computationally expensive, this algorithm may not be suitable for real-time applications if number of feature points are excessive.

### 3 Object Modeling

To keep track of location and other properties of the detected objects, one must have an internal representation of an object suitable for matching its descriptor to image features. In this section, we will describe the various object model representations, descriptors, and features commonly employed for tracking (Figure 6). In general, there is a strong relationship between the object representations, descriptions, their low-level features and applications.



**Fig. 6** Different layers of object modeling can be categorized under the representations, descriptors, and low-level features.



**Fig. 7** From left to right: object region, elliptical region, silhouette (contour), part-based, skeletal representations.

### 3.1 Model Representations

Object representations are usually chosen according to the application domain. The model selected to represent object shape limits the type of motion or deformation it can undergo. For example if an object is represented as a point then only a translational model can be used. In case if a geometric shape representation, like ellipse, is used for the object then parametric motion models like affine or projective transformations are appropriate. These representations can approximate the motion of rigid objects in the scene. For a non-rigid object, silhouette or contour is the most descriptive representation and both parametric and non-parametric models can be used to specify their motion. We explain some of these representations (as illustrated in Figure 7) in more detail below.

#### 3.1.1 Point and Region

Many methods represent an object by a predefined shape around its centroid [36] or a set of points [37]. Object shape can be defined as a rectangle, ellipse, etc. on the imaging plane or on the physical ground plane. Often such shapes are normalized into a fixed size to simplify feature extraction. In general, motion for such representations is modeled by simple translation,

similarity, affine or homography transformations. Though primitive shapes are more suitable for representing simple rigid objects, they are also used for tracking non-rigid objects.

### 3.1.2 Silhouette

Silhouette is the region inside the contouring boundary of object. The most common silhouette representation is in the form of a binary indicator function, which marks the object region by ones and the non-object regions by zeros. For contour based methods, the silhouette is represented either explicitly or implicitly. Explicit representation defines the boundary of the silhouette by a set of control points. Implicitly representation defines the silhouette by means of a function defined on a grid. The most common implicit contour representation is the level sets representation.

A traditional explicit contour structure is composed of a set of points (tuples) on the contour along with a set of spline equations, which are real functions fit to the control points generated from the tuples. A natural choice is cubic spline defined using piecewise cubic polynomials, between the tuples. The piecewise analytical contour form provides the ability to estimate differential descriptors with ease. The analytical form, however, does not let changes in the contour topology (merging and splitting), which is necessary during contour evolution.

Alternatively, contour in an image can be implicitly defined using the level set [38]. In this formalism, the position of a contour is embedded as the zero level set in a two dimensional function. The value at a grid position (e.g. local object coordinates) is commonly set to its distance from the closest contour location and is computed by applying a distance transform. The level set representation has attracted much attention due to its ability to adapt to topology changes, direct computation of differential contour features (e.g. contour curvature), and extendibility to higher dimensions with no change in formulation.

### 3.1.3 Connected Parts

Articulated objects are composed of parts that are held together with joints. Depending on the granularity of choice, parts of a human body can be grouped into head, torso, arms and legs, which can be defined as geometric primitives, such as rectangles, cylinders and ellipses [39]. The relationship between the parts are governed by kinematic motion models, e.g. joint angle. An important issue that needs explicit handling is the occlusion problem that arises when one part is behind the others making it invisible. Occluded parts constitute missing observations and can be dealt with by applying heuristics or by enforcing learned part-arrangements. In addition, the degree of articulation increases the complexity of the models.

### 3.1.4 Graph and Skeletal

Another spatial representation is the skeletal models which are used to animate characters and humans in graphics. Skeleton is an articulated structure with a set of curve segments and joints connecting them [40]. Object skeleton can be extracted by applying medial axis transform, which takes the object silhouette and iteratively computes the set of points lying on its topological skeleton such that each point has more than one closest points to the bounding contour. Alternatively, it is termed as the loci of centers of bi-tangent circles that fit within the object silhouette.

Learned skeletal representations have many possible uses. For example, manually constructed skeletal models are often a key component in full-body tracking algorithms. The ability to learn skeletal structure could help to automate the process, potentially producing models more exible and accurate than those constructed manually.

The motion of an articulated object can be described as a collection of rigid motions, one per part, with the added constraint that the motions of connected parts must be spatially coherent. This constraint causes the motion subspaces of two connected objects to intersect, making them linearly dependent. In particular, for each pair of connected parts, the motion subspaces share one dimension (translation) if they are joined at a point and two dimensions (translation and one angle of rotation) if they are joined at an axis of rotation.

### 3.1.5 Spatiotemporal

While spatial representations lacks motion indicators, there are specific representations that are defined in the spatiotemporal space and inherently convey the motion information. Spatiotemporal representations can be extracted by local analysis or by looking at the space-time cube globally. Local representations are composed of a set of points that present characteristic motion and appearance content. The point set is commonly treated as a bag of features without temporal order. Space-time cube [41], which is generated by stacking video frames, can be considered a volumetric 3D image. An important observation about the space-time cube is that aside from providing temporal information, it also carries a unique view geometric information when we have two cameras observing a common scene. Alternative to using temporally sparse point sets, one can consider progression of the point set in time by extracting their trajectories [42]. Trajectory representation is constituted of a temporally ordered point series, which represent the position of a point starting from its initial observation at until it disappears from the scene.

### ***3.2 Model Descriptors***

Descriptors are the mathematical embodiments of object regions. The size of the region, dynamic range, imaging noise and artifacts play a significant role in achieving discriminative descriptors. Generally speaking, the larger the object region, the more discriminative the descriptor will be.

A major concern for most descriptors is the lack of a competent similarity criterion that captures both statistical and spatial properties, i.e., most approaches either depend only on the color distributions or structural models. Many different representations, from aggregated statistics to appearance models, have been used for describing objects. Color histograms are popular representations of nonparametric density, but they disregard the spatial arrangement of the feature values. Moreover, they do not scale to higher dimensions due to exponential size and sparsity. Appearance models map the image features onto a fixed size window. Since the dimensionality is a polynomial in the number of features and the window size, only a relatively small number of features can be used. Appearance models are highly sensitive to the pose, scale and shape variations.

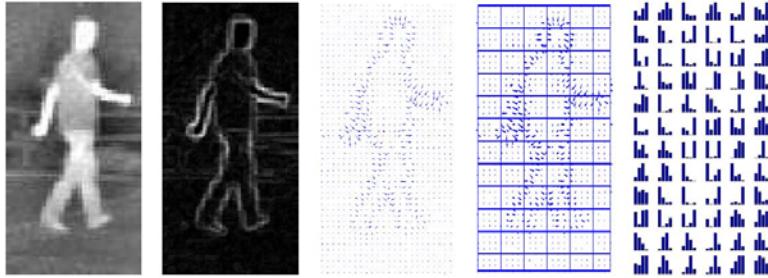
#### ***3.2.1 Template***

Templates are the most intuitive and commonly adopted descriptors and often formed from geometric shapes or silhouettes. A unique property of a template is that it is an ordered list of appearance observations inside its region. This property naturally provides the template descriptor with the capability to carry both spatial and appearance information. They can be 2D (spatial) or 3D (spatiotemporal) depending on their use, and commonly have a shape in the form of a geometric primitive, such as a rectangle, square, ellipse, circle or their 3D versions. They are often centered around a point and defined by weighted spatial kernels that may have varying scalars at different pixels within the kernel. The kernel function represents a convolution of the geometric primitive with the template. The motion of the kernel from one frame to the next follows a parametric model including translation, conformal and affine transformations.

Templates, however, only encode the object appearance generated from a single view. Thus, they are more suitable for problems where the viewing angle of the camera and the object pose remains constant or changes very slowly.

#### ***3.2.2 Histogram, HOG, SIFT***

Histogram is a distribution based descriptor that estimates the probability distribution from the observations within a spatial or spatiotemporal region defined by a template, silhouette or a volume. The observations considered



**Fig. 8** HOG concatenates the bins of the local gradient histograms into a vector form (Courtesy of E. Benhaim).

can be raw color values, derivative information or texture measures. Color distributions are generally estimated non-parametrically by a histogram, or parametrically by mixture models.

The histogram, which is a common choice, can be generated by first defining the number of bins (quantization levels) and counting the number of observations that fall into respective bins. While a histogram can be generated using raw color or intensity values, they may need to be processed, such as mean color adjustment, prior to estimating the histogram to remove the illumination and shadowing effects. This adjustment can be achieved by subtracting the mean color computed in the neighborhood of the region of interest.

Alternative to using color values, image gradient is adapted for generating a distribution based descriptor. Two closely related approaches are the Scale Invariant Feature Transform (SIFT) descriptors [43] and the Histogram of Oriented Gradients (HOG) [44, 20]. Both of these approaches compute the gradient of intensity and construct a histogram of gradient orientations weighted by the gradient magnitude (Figure 8). Shared steps between the two approaches can be listed as follows:

```

Input: Image and regions of interest
Output: Histograms
foreach region R do
  foreach  $(x, y) \in R$  do
    compute the gradient:  $\nabla I(x, y) = (I_x, I_y) =$ 
     $(I(x - 1, y) - I(x + 1, y), I(x, y - 1) - I(x, y + 1))$ ;
    compute gradient direction:  $\theta(\nabla I(x, y)) = \arctan(I_x, I_y)$ ;
    compute gradient magnitude:  $|\nabla I(x, y)| = (I_x * I_x + I_y * I_y)^{1/2}$ ;
    if  $|\nabla I(x, y)| \geq \tau$  then
      | Increment histogram bin for  $\theta(\nabla I(x, y))$ ;
    end
  end
  smooth histogram;
end

```

Aside from the common steps outlined above, SIFT approach computes the major orientation from the resulting histogram and subtracts it from computed orientations to achieve rotation invariance. HOG, on the other hand does not perform orientation normalization. SIFT generates more number of interest points compared to other interest point detectors. This is due to the fact that the interest points at different scales and different resolutions (pyramid) are accumulated. Empirically, it has been shown in [45] that SIFT is more resilient to image deformations. More recently the color based SIFT method is introduced and is widely adopted [46].

Another possibility is to generate the histogram defining the shape of the object. Shape histograms model the spatial relation between the pixels lying on the contour. The spatial relation between a reference point on the contour with respect to other points is modeled by a histogram, and a set of such histograms, which are generated by taking all or randomly chosen contour points individually as a reference, provides a distribution based descriptor. The spatial relation between two points is measured by computing the angle and magnitude of the vector joining them. Similarly, shape context uses a concentric circular template centered on a reference contour point, which provide the bins of the histogram in the polar coordinates [47].

In all cases, a commonly used approach to compare histograms is Bhattacharya distance.

### 3.2.3 Region Covariance

The region covariance matrix proposes a natural way of fusing multiple features. Its diagonal entries represent the variance of each feature and the nondiagonal entries represent the correlations. The noise corrupting individual samples are largely filtered out with an average filter during covariance computation.

For a given region  $R$ , let  $F(x, y, i)$  be the pixel feature values. Features  $i = 1, \dots, d$  can be any mapping including pixel's Cartesian coordinates, intensity, color, gradient, texture, filter responses, etc. For instance,  $F(\cdot, \cdot, 1)$  can be assigned to the  $x$ -gradient:  $F(x, y, 3) = I_x(x, y)$ . Region covariance descriptor  $C$  represents the region  $R$  with the  $d \times d$  covariance matrix of the pixel-wise features

$$C = \begin{bmatrix} c_{11} & \cdots & c_{1d} \\ \vdots & \ddots & \vdots \\ c_{d1} & \cdots & c_{dd} \end{bmatrix} \quad \text{and} \quad c_{ij} = \frac{1}{N_R - 1} \sum_{n=1}^{N_R} (F(x_n, y_n, i) - \mu_i)(F(x_n, y_n, j) - \mu_j) \quad (1)$$

where  $\mu$  is the mean of the corresponding feature for all  $N_R$  pixels in the region.

There are several advantages of using covariance matrices as region descriptors. It is a natural way of fusing multiple features. A single covariance matrix extracted from a region is usually enough to match the region in different views and poses. The covariance of a distribution is enough to discriminate it

from other distributions. The covariance matrices are low-dimensional compared to other region descriptors and due to symmetry it has only  $(d^2 + d)/2$  different values. This provides significant robustness to object variations while keeping the descriptor efficiently discriminative. Whereas if the same region is described with joint histograms a total of  $B^d$  values, where  $B$  is the number of histogram bins, are needed, which renders such joint histograms very fragile.

**Distance Calculation:** The covariance matrices do not lie on Euclidean space. For example, the space is not closed under multiplication with negative scalers. Most of the common machine learning methods work on Euclidean spaces and therefore they are not suitable for our features. The nearest neighbor algorithm which will be used in the following sections, only requires a way of computing distances between feature points. A distance measure is proposed in [48] to measure the dissimilarity of two covariance matrices

$$\rho^2(C_1, C_2) = \sum_{i=1}^d \ln^2 \lambda_i(C_1, C_2) \quad (2)$$

where  $\{\lambda_i(C_1, C_2)\}$  are the generalized eigenvalues of  $C_1$  and  $C_2$ . The distance measure follows from the Lie group structure of positive definite matrices and an equivalent form can be derived from the Lie algebra of positive definite matrices. We refer the readers to [48] for a detailed discussion on the distance metric.

### 3.2.4 Ensembles and Eigenspaces

Ensemble descriptors keep a combination of weak or partial descriptors. More specifically, the ensemble tracking [49] works by constantly updating a collection of weak classifiers to separate the object from its background. The weak classifiers can be added or removed at any time to reflect changes in the object appearance or incorporate new information about the background. Hence, object is not represented explicitly, instead an ensemble of classifiers is used to determine if a pixel belongs to the object or not. Each weak classifier is trained on positive and negative examples where, the examples coming from the object are positive examples and examples coming from the background are negative examples. The strong classifier is then used to classify the pixels in the next frame, producing a confidence map of the pixels, where the classification margin is used as the confidence measure. The peak of the map is assumed to be the location of the object in the current frame. Once the detection for the current frame is completed, a new weak classifier is trained on the new frame, added to the ensemble, and the process is repeated all over again.

Given a set of images, eigenspace approaches construct a small set of basis images that characterize the majority of the variation in the training set and

can be used to approximate any of the training images. For each image in a training set of  $p$  images a 1D column vector is constructed by scanning the image in the standard lexicographic order. Each of these 1D vectors becomes a column in a data matrix. The number of training images is assumed to be less than the number of pixels, and Singular Value Decomposition (SVD) is used to decompose the data matrix into 1) an orthogonal matrix of the same size as the data matrix representing the principal component directions in the training set, 2) a diagonal matrix with singular values sorted in decreasing order along the diagonal, and 3) an orthogonal matrix that encodes the coefficients to be used in expanding each column of the data matrix in terms of the principal component directions.

It is possible to approximate some new image vector in terms of the orthogonal matrix columns that have comparably higher singular values by taking the dot product of the image vector and the columns of the orthogonal matrix. This amounts to a projection of the input image onto the subspace defined by the largest basis vectors. The eigenspace can be thought of as a compact view-based object representation that is learned from a set of input images. Previously observed views of an object can be approximated by a linear combination of the basis vectors. This can be thought of as matching between the eigenspace and the image.

### 3.2.5 Appearance Models

Active appearance models are generated by simultaneously modeling the object shape and appearance [50]. In general the object shape is defined by a set of landmarks. Similar to the contour based representation, the landmarks can reside on the object boundary or alternatively, they can reside inside the object region. For each landmark, an appearance vector is stored which is in the form of color, texture or gradient magnitude. Active appearance models require a training phase where both the shape and its associated appearance is learned from a set of samples using, for instance, the principal component analysis.

## 3.3 Model Features

The use of a particular feature set for tracking can also greatly affect the performance. Generally, the features that best discriminate between multiple objects and, between the object and background are also best for tracking the object. Many tracking algorithms use a weighted combination of multiple features assuming that a combination of preselected features will be discriminative. A wide range of feature selection algorithms have been investigated in the machine learning and pattern recognition communities. However, these algorithms require off-line training information about the target and/or the background. Such information is not always available. Moreover, as the object

appearance or background varies, the discriminative features also vary. Thus, there is a need for online selection of discriminative features. The details of common visual features are as follows.

- **Color:** The apparent color of an object is influenced primarily by two physical factors: 1) the spectral power distribution of the illuminant, and 2) the surface reflectance properties of the object. In image acquisition, the RGB (red, green, blue) color space is usually used to represent color. However, the RGB space is not a perceptually uniform, that is, the difference between the colors in the RGB space does not correspond to the color differences perceived by the humans [51]. Instead, YUV and LAB are perceptually uniform, while HSV (Hue, Saturation, Value) is an approximately uniform color space. However, these color spaces are sensitive to noise. In summary, there is no last word on which color space is more efficient, therefore a variety of color spaces have been used in tracking.
- **Gradient:** Object boundaries usually generate strong changes in image intensities. Edge gradient identifies these changes. An important property of edges is that they are less sensitive to illumination changes as compared to color features. Algorithms that track the boundary of the objects usually use edges as the representative feature. Because of its simplicity and accuracy, the most popular edge detection approach is the Canny Edge detector [52].
- **Optical Flow:** Optical flow is a dense field of displacement vectors which defines the translation of each pixel in a region. It is computed using a brightness constraint, which assumes “brightness constancy” of corresponding pixels in consecutive frames and is usually computed using the image derivatives [53, 54]. Optical flow is commonly used as a feature in motion-based segmentation and tracking applications. A comparison of the popular optical flow techniques can be find in [55].
- **Texture:** Texture is a measure of the intensity variation of a surface which quantifies properties such as smoothness and regularity. Compared to color, texture requires a processing step to generate the descriptors. There are various texture descriptors including gray level co-occurrence matrices, Law’s texture measures (twenty five 2D filters generated from five 1D filters corresponding to level, edge, spot, wave and ripple), wavelets, Gabor filters, and steerable pyramids. A detailed analysis of texture features can be found in [56]. Similar to edge features, the texture features are less sensitive to illumination changes as compared to color.
- **Corner Points:** Corner points are one of the earliest and most commonly used features due to its low computational complexity and ease of implementation. Harris corner detector, like many others, defines texture-content by conjecturing that the change in the color content of pixels in the locality of a candidate interest point should be high:

$$E(x, y) = \sum_u \sum_v \left( I(x+u, y+v) - I(x, y) \right)^2. \quad (3)$$

The Taylor series approximation of this equation around  $(x, y)$  results in

$$E(u, v) = [u \ v] \underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} u \\ v \end{bmatrix}. \quad (4)$$

This equation contains the commonly termed structure tensor  $\mathbf{M}$ , which is a second moment computed from the template around the candidate. This matrix defines an ellipse with minor and major axes denoted by its eigenvectors and their extent by respective eigenvalues. The eigenvalues,  $\lambda_i$  of  $\mathbf{M}$  are computed from its characteristic equation:  $\lambda^2 + \det(\mathbf{M}) - \lambda \cdot \text{trace}(\mathbf{M}) = 0$ , which suggests that using determinant and trace of  $\mathbf{M}$  should suffice in marking interest points as stated in [57]. Therefore, the traditional texture content measure  $\min(\lambda_1, \lambda_2)$  can be approximated by  $\det(\mathbf{M}) - c \cdot \text{trace}(\mathbf{M})^2$  for constant  $c$ . The texture content measure is computed for all pixels and it is subjected to nonmaximal suppression which removes weak interest point candidates and eliminates multiple candidates in small neighborhoods.

Harris detector, when applied in scale space, such as by convolving the image with a set of different scaled Gaussian filters, provides feature points at multiple scales. The interest points coexisting at different scales can be combined to provide scale-invariant interest points. Considering that the shape tensor is invariant to rotations, Harris feature becomes invariant to similarity transform. The spatial point detection scheme outlined for Harris detector is later extended to spatiotemporal coordinates by introducing the time as an additional dimension to the formulation [58]. Limitations of the Harris feature include its inability to locate interest points at subpixel level, and difficulty setting the number of interest points it detects.

Mostly, features are chosen manually by the user depending on the application domain. However, the problem of automatic feature selection has received significant attention in the pattern recognition communities. Automatic feature selection methods can be divided into filter methods and wrapper methods [59]. The filter methods try to select the features based on a general criteria, e.g. the features should be uncorrelated. The wrapper methods select the features based on the usefulness of the features in a specific problem domain, e.g. the classification performance using a subset of features. Principal Component Analysis (PCA) is an example of the filter methods for the feature reduction. PCA involves transformation of number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called the principal components.

A common wrapper method of selecting the discriminatory features is boosting. More specifically, AdaBoost finds a strong classifier based on a

combination of moderately inaccurate “weak” classifiers. Given a large set of features, one classifier can be trained for each feature. Adaboosts discover a weighted combination of classifiers (representing features) that maximize the classification performance of the algorithm. These weights indicates the discriminative power of the features.

## 4 Object Tracking

Given the detected object, it is then the tracker’s task to perform find its correspondence in the following frames while constructing object’s trajectory. It is an essential component of several vision applications as well as the video analytics for business applications. Object tracker may also provide the complete region in the image that is occupied by the object.

The tasks of detecting the object and establishing correspondence between the object instances across frames can either be performed separately or jointly. In the first case, possible object regions in every frame are obtained by means of an object detection algorithm and then the tracker corresponds objects across frames. In the latter case, the object region and correspondence is jointly estimated by updating object location and region information obtained from previous frames.

Robust and accurate tracking of a deforming, non-rigid and fast moving object without getting restricted to particular model assumptions presents a major challenge. One can simplify tracking by imposing constraints on motion and appearance of objects. For example, almost all tracking algorithms assume the object motion to be smooth with no abrupt changes. One can further constrain the object motion to be of constant velocity, or constant acceleration based on a priori information. Prior knowledge about the number and the size of objects, or the object appearance and shape can also be used to simplify the problem.

Numerous approaches for object tracking have been proposed. These primarily differ from each other based on the way they approach the following questions: Which object representation is suitable for tracking? Which image features should be used? How should motion, appearance and shape of the object be modeled? The answers to these questions depend on the context/environment in which the tracking is being performed, and the end use for which the tracking information is being sought.

For business applications, they should perform mostly partitioned indoors spaces where sudden illumination changes, for instance due to on-and-off of a light switch, can occur. They are expected to handle significant object size changes due to oblique views and severe occlusions due to usually lower camera heights. They need to resolve multiple-object tracking even often the descriptors are insufficient as people tend to dress in for instance dark clothes in business environments.



**Fig. 9** Mean-shift tracking iterations: estimated object location at time  $t - 1$ , (b) Frame at time  $t$  with initial location estimate using the previous object position, (c), (d) location update using mean shift iterations, (e) final object position at time  $t$ .

We give a brief description of most common tracking techniques in the following.

#### 4.1 Template Matching

The most common approach in this category is template (or blob) matching. Template matching is a brute-force method of searching the image for a region similar to the object template defined in the previous frame. The position of the template in the current image is computed by a similarity measure, e.g. cross correlation. Usually image intensity or color features are used to form the templates. Since image intensity is very sensitive to illumination changes, image gradients can also be used as features. Note that instead of region templates, other object descriptors for instance, histograms or mixture models can also be used for matching, with the same spirit of cross correlation based exhaustive search.

A limitation of template matching is high computation cost due to the brute-force search. To reduce the computational cost, search window is usually limited to the vicinity of its previous position [60].

#### 4.2 Density Estimation: Mean-Shift

Mean-shift [61] is a nonparametric density gradient estimator to find the image window that is most similar to the object’s color histogram in the current frame. It iteratively carries out a kernel based search starting at the previous location of the object as shown in Figure 9.

The mean-shift tracker maximizes the appearance similarity iteratively by comparing the histogram of the object  $h_o$  and the histogram of the window around the candidate object location  $h_*$ . Histogram distance is defined in terms of the Bhattacharya distance. Minimizing the Bhattacharya distance or alternatively maximizing the Bhattacharya coefficient  $\rho(h_o, h_*) = \sum_{b=1}^B [h_o(b)h_*(b)]^{1/2}$  and expanding it to Taylor series suggests that the new location of the object can be iteratively computed [62] by estimating the likelihood ratio between the model  $h_o$  and candidate histograms  $h_*$ .

At each iteration the candidate window is shifted towards the direction that the Bhattacharya coefficient is maximum. The iterative process is repeated until the increase in the Bhattacharya coefficient becomes insignificant:

- Compute the candidate histogram  $h_*(\mathbf{m}_{t-1})$ ,
- Calculate  $\rho(h_o, h_*(\mathbf{m}_{t-1}))$  and the new weights  $w_n = \sum_b^B \delta[I(x_n, y_n)_B - b] \sqrt{h_o/h_*(\mathbf{m}_{t-1})}$ ,
- Find new location  $\mathbf{m}_t$  by,

$$\mathbf{m}_t = \frac{\sum_n^{N_R} w_n \dot{K}(\|\mathbf{x}_n - \mathbf{m}_{t-1}\|) \mathbf{x}_n}{\sum_n^{N_R} w_n \dot{K}(\|\mathbf{x}_n - \mathbf{m}_{t-1}\|)}, \quad (5)$$

- Stop if  $\rho(h_o, h_*(\mathbf{m}_t)) < \rho(h_o, h_*(\mathbf{m}_{t-1}))$  or  $\|\mathbf{m}_t - \mathbf{m}_{t-1}\| < \epsilon$ , else  $\mathbf{m}_t \leftarrow \mathbf{m}_{t-1}$  and iterate.

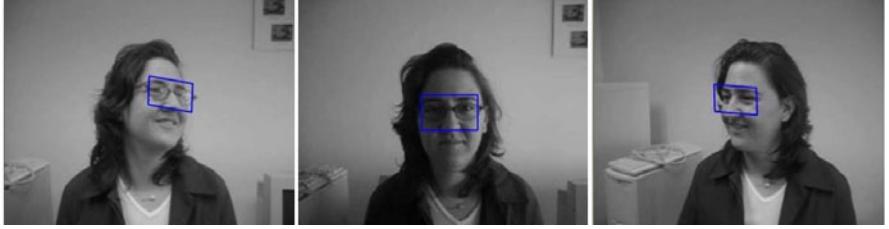
Above  $\mathbf{x}_n = [x_n, y_n]^T$ ,  $\mathbf{m}_t = [m_x, m_y]^T$  is the estimated location of the object,  $t$  is the iteration index,  $I(x_n, y_n)_B$  is the bin of the pixel value  $I(x, y)$ , and  $\dot{K}$  is the derivative of the kernel function  $K$ , which can be a 2D Gaussian. The kernel state is defined by the centroid of the object in spatial coordinates, such that the estimation process results in the new object position. Alternatively, the state can be defined by the scale, orientation and position of the object [63].

In other words, at each iteration, the mean-shift vector is computed such that the histogram similarity is increased. This process is repeated until convergence is achieved, which usually takes five to six iterations. For histogram generation, a weighting scheme is defined by a spatial kernel which gives higher weights to the pixels closer to the object center [36] extended the mean shift tracking approach used a joint spatial-color histogram instead of just color histogram.

An obvious advantage of the mean-shift tracker over the standard template matching is the elimination of a brute force search, and the computation of the translation of the object patch in a small number of iterations. However, the mean-shift tracker requires that a portion of the object is inside the previous object region upon initialization. Even though there are variants [64] to improve its localization by using additional modalities, the success of the mean-shift strongly depends on the discriminating power of the histograms. To overcome this shortcoming, a covariance matrix representation version is proposed in [65], and a multiple-kernel version in [66].

### 4.3 Regression

Regression refers to understand the relationship between multiple variables. Linear regression assumes the relationship depends linearly on a model in which the conditional mean of a scalar variable given the other variables is an affine function of those variables. Numerous procedures have been



**Fig. 10** Regression tracking on manifold for a given region. Note that the tracking is still valid even the region undergoes out-of-plane rotations.

developed for parameter estimation and inference in linear regression. Here a least squares estimator is described for object tracking, details can be found in [67].

Suppose  $(\alpha_i, X_i)$  are the pairs of observed data  $\alpha \in \mathbb{R}^d$  in vector space and the corresponding points on the manifold  $X \in \mathcal{M}$ . The regression function  $\varphi$  maps the vector space data onto the manifold  $\varphi : \mathbb{R}^d \mapsto \mathcal{M}$ . An objective function is defined as the sum of the squared geodesic distances between the estimations  $\varphi(\alpha_i)$  and the points  $X_i$

$$J = \sum_i \Delta^2 [\varphi(\alpha_i), X_i]. \quad (6)$$

Assuming a Lie algebra on the manifold can be defined, the objective function can be approximated as

$$J = \sum_i \|\log [\varphi^{-1}(\alpha_i) X_i]\|^2 \approx \sum_i \|\log [\varphi(\alpha_i)] - \log [X_i]\|^2 \quad (7)$$

up to the first order terms. The regression function  $\varphi$  can be written as

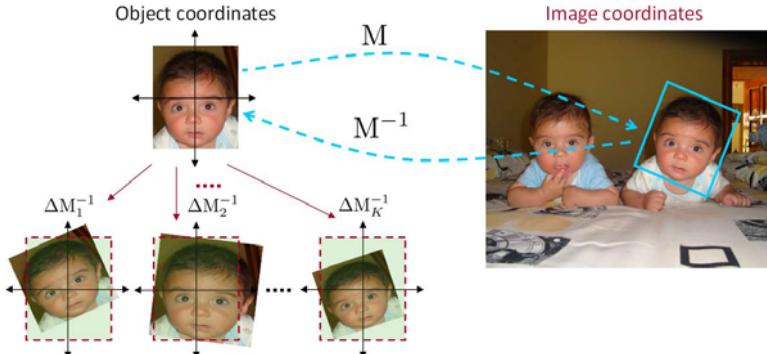
$$\varphi(\alpha_i) = \exp (\alpha_i^T \Omega) \quad (8)$$

to learn the function  $\Omega : \mathbb{R}^d \mapsto \mathbb{R}^r$  which estimates the tangent vectors  $\log (X_i)$  on the Lie algebra where  $\Omega$  is the  $d \times r$  matrix of regression coefficients. Thus, the objective function (7) becomes

$$J = \sum_i \|\alpha_i^T \Omega - \log [X_i]\|^2 \quad (9)$$

Let  $\mathbf{X}$  be the  $k \times d$  matrix of initial observations and  $\mathbf{Y}$  be the  $k \times r$  matrix of mappings to the Lie algebra

$$\mathbf{X} = \begin{bmatrix} [\alpha_1]^T \\ \vdots \\ [\alpha_k]^T \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} [\log(X_1)]^T \\ \vdots \\ [\log(X_k)]^T \end{bmatrix} \quad (10)$$



**Fig. 11** Random transformations are applied in object coordinates to generate the training features for regression function estimation.

Substituting (10) into (9), one can obtain

$$J = \text{tr}[(\mathbf{X}\Omega - \mathbf{Y})^T(\mathbf{X}\Omega - \mathbf{Y})] \quad (11)$$

where the trace replaces the summation in (7). Differentiating the objective function  $J$  with respect to  $\Omega$ , the minimum is achieved at  $\Omega = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$ . To avoid overfitting, additional constraints on the size of the regression coefficients can be introduced

$$J = \text{tr}[(\mathbf{X}\Omega - \mathbf{Y})^T(\mathbf{X}\Omega - \mathbf{Y})] + \beta \|\Omega\|^2 \quad (12)$$

which is called the ridge regression. The minimizer of the cost function  $J$  is given by  $\Omega = (\mathbf{X}^T \mathbf{X} + \beta \mathbf{I})^{-1} \mathbf{X}^T \mathbf{Y}$  where  $\mathbf{I}$  is an  $d \times d$  identity matrix. The regularization coefficient  $\beta$  determines the degree of shrinkage on the regression coefficients.

At the initialization of the object, the affine motion tracker estimates a regression function that maps the region feature vectors to the hypothesized affine motion vectors by first hypothesizing a set of random motion vectors within the given bounds, determining the transformed regions for these motions, and then computing the corresponding features within each warped region. In the tracking time, it extracts the feature vector only for the previous object region location and applies the learned regression function. Sample affine tracking results are shown in Figure ??

Let  $M$  transforms a unit square at the origin to the affine region enclosing the target object  $[x \ y \ 1]^T_I = M[x \ y \ 1]^T_O$  where the subscripts indicate the image and object coordinates respectively. The inverse  $M^{-1}$  is an affine motion matrix and transforms the image coordinates to the object coordinates. The aim of tracking is to estimate the transformation matrix  $M_t$ , given the previous images and the initial transformation  $M_0$ . The transformations are modeled incrementally

$$M_t = M_{t-1} \cdot \Delta M_t \quad (13)$$

and estimate the increments  $\Delta M_t$  at each time. The transformation  $\Delta M_t$  corresponds to motion of target from time  $t-1$  to  $t$  in the object coordinates.

Suppose the target region is represented with orientation histograms computed at a regular grid inside the unit square in object coordinates, i.e with  $\alpha(I(M_t^{-1})) \in \mathbb{R}^d$  where  $d$  is the dimension of the descriptor. Given the previous location of the object  $M_{t-1}$  and the current observation  $I_t$ , the new transformation  $\Delta M_t$  by the regression function is estimated as

$$\Delta M_t = \varphi(\alpha(M_{t-1}^{-1})). \quad (14)$$

The problem reduces to learning and updating the regression function  $\varphi$ . During the learning step, a training set of  $k$  random affine transformation matrices  $\{\Delta M_j\}_{j=1\dots k}$  are generated around the identity matrix to learn the regression function  $\Omega$  as illustrated in Figure ???. The approximation is good enough since the transformations are in a small neighborhood of the identity. The training set consists of samples  $\{\alpha_j, \Delta M_j\}_{j=1\dots k}$ . Since number of samples is smaller than the dimension of the feature space,  $k < d$ , the system is underdetermined. To relieve this, the ridge regression is applied to estimate the regression coefficients.

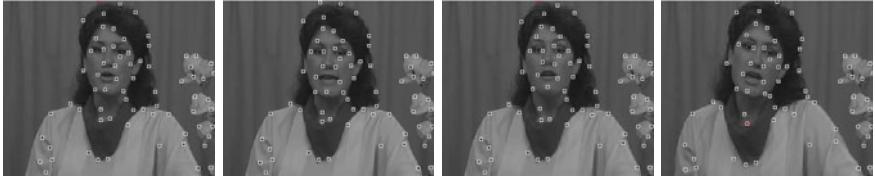
Since objects can undergo appearance changes in time, it is necessary to adapt to these variations. The model update achieves reestimating the regression function. During tracking, a set of random observations are generated at each frame with the same method described above. The observations stored for most recent frames constitute the update training set. More details and an importance sampling based adaptation can be found in [68].

#### 4.4 Motion Estimation

Optical flow methods are used for generating dense flow fields by computing the flow vector of each pixel under the brightness constancy constraint,  $I(x, y, t) - I(x + dx, y + dy, t + dt) = 0$ . This computation is always carried out in the neighborhood of the pixel either algebraically or geometrically. Extending optical flow methods to compute the translation of a rectangular region is trivial. In [69], the KLT tracker, which iteratively computes the translation  $(du, dv)$  of a region (e.g.  $25 \times 25$  patch) centered on an interest point, is proposed as:

$$\begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix} \begin{pmatrix} du \\ dv \end{pmatrix} = \begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}.$$

This equation is similar in construction to the optical flow [54]. Once the new location of the interest point is obtained, the KLT tracker evaluates the quality of the tracked patch by computing the affine transformation between the corresponding patches in consecutive frames. If the sum of square difference



**Fig. 12** Tracking features using the KLT tracker.

between the current patch and the projected patch is small, they continue tracking the feature, otherwise the feature is eliminated. The results obtained by the KLT tracker are shown in Figure 12.

In [70] an object tracker that tracks an object as a three component mixture, consisting of the stable appearance features, transient features and noise process is proposed. The stable component identifies the most reliable appearance for motion estimation, i.e. the regions of object whose appearance does not quickly change over time. The transient component identifies the quickly changing pixels. The noise component handles the outliers in the object appearance, which arise due to noise. An online version of the EM algorithm is used to learn the parameters of this three component mixture. The authors use the phase of the steerable filter responses as features for appearance representation. The object shape is represented by an ellipse. The motion of the object is computed in terms of warping the tracked region from one frame to the next one. The warping transformation consists of translation, rotation and scale parameters. A weighted combination of the stable and transient components is used to determine the warping parameters. The advantage of learning stable and transient features is that one can give more weight to stable features for tracking, for example, if the face of a person who is talking is being tracked, then the forehead or nose region can give a better match to the face in the next frame as opposed to the mouth of the person.

#### 4.5 Kalman Filtering

Let the location of a moving object is defined by a sequence of states  $X_t$ . The change in state over time is governed by the linear system,

$$X_t = A_t X_{t-1} + \eta, \quad (15)$$

where  $\eta$  is white noise with covariance  $\Sigma^\eta$ . The relationship between the measurement and the state is specified by the measurement equation  $Y_t = D_t X_t + \xi$ , where  $D_t$  is the measurement matrix and  $\xi$  is the white noise with covariance  $\Sigma^\xi$  independent of  $\eta$ . The objective of tracking is to estimate the

state  $X_t$  given all the measurements up to that moment, or equivalently to construct the probability density function  $p(X_t|Y_{1,\dots,t})$ .

Theoretically optimal solution is provided by a recursive Bayesian filter which solves the problem in two steps. The prediction step uses a dynamic equation and the already computed pdf of the state at time  $t - 1$  to derive the prior probability distribution of the current state. Then, the correction step employs the likelihood function of the current measurement to compute the posterior probability distribution.

Kalman filter is used to estimate the state of a linear system where the state is assumed to be distributed by a Gaussian. The prediction step of the Kalman filter uses the state model to predict the new state of the variables:

$$\begin{aligned} X_t^p &= AX_{t-1} \\ \Sigma_t^p &= A\Sigma_{t-1}A^T + \Sigma_t^\eta, \end{aligned}$$

where  $X_t^p$  and  $\Sigma_t^p$  are the state and the covariance predictions at time  $t$ .  $A_t$  is the state transition matrix which defines the relation between the state variables at time  $t$  and  $t - 1$ . Similarly, the correction step uses the current observations  $Y_t$  to update the object's state:

$$X_t = X_t^p + G_t[Y_t - D_t X_t^p], \quad (16)$$

$$G_t = \Sigma_t^p D_t^T [D_t \Sigma_t^p D_t^T + \Sigma_t^\xi]^{-1}, \quad (17)$$

$$\Sigma_t = \Sigma_t^p - G_t D_t \Sigma_t^p,$$

where  $G_t$  is the Kalman gain, which is used for propagation of the state models. Note that the updated state,  $X_t$ , is still distributed by a Gaussian. In case system is nonlinear, it can be linearized using the Taylor series expansion to obtain the Extended Kalman Filter. Similar to Kalman filter, Extended Kalman filter assumes that the state is distributed by a Gaussian.

Kalman filter has been extensively used in early vision community for tracking, e.g. to track points in noisy images [71] and to estimate 3D trajectory from 2D motion [72]. When tracking multiple objects using particle and Kalman filters, one needs to deterministically associate the most likely measurement for a particular object to that object's state, i.e. the correspondence problem needs to be solved before these filters can be applied. The simplest method to perform correspondence is to use the nearest neighbor approach. However, if the objects are close to each other, then there is always a chance that the correspondence is incorrect. An incorrectly associated measurement can cause the filter to fail to converge.

## 4.6 Particle Filtering

One limitation of the Kalman filter is the assumption that the state variables are Gaussian. Thus, Kalman filter will give a poor estimations of state variables that do not follow Gaussian distribution. This limitation can be overcome by using particle filtering [73] [74].

In particle filtering, the conditional state density  $p(X_t|Y_t)$  at time  $t$  is represented by a set of  $N_s$  samples  $\{s_{t,n}\}$  where  $n = 1, \dots, N_S$  with weights  $\pi_{t,n}$  corresponding to sampling probability. These samples are called as particles. The weights define the importance of a sample, i.e. its observation frequency [75]. To decrease computational complexity, for each tuple  $(s_n, \pi_n)$  a cumulative weight  $c_n$  is also stored, where  $\sum c_n = 1$ . The new samples at time  $t$  are drawn from  $S_{t-1} = \{(s_{t-1,n}, \pi_{t-1,n}, c_{t-1,n})\}$  at the previous time  $t-1$  step based on different sampling schemes. The most common sampling scheme is the “importance sampling” which can be stated as follows:

- Selection: Select  $N_S$  random samples  $\hat{s}_{t,n}$  from  $S_{t-1}$  by generating a random number  $r \in [0, 1]$ , finding the smallest  $j$  such that  $c_{t-1,j} > r$  and setting  $\hat{s}_{t,n} = s_{t-1,j}$ .
- Prediction: For each selected sample  $\hat{s}_{t,n}$ , generate a new sample by  $s_{t,n} = f(\hat{s}_{t,n}, W_{t,n})$ , where  $W_{t,n}$  is a zero mean Gaussian error and  $f$  is a non-negative function  $f(s) = s$ .
- Update: Weights  $\pi_{t,n}$  corresponding to the new samples  $s_{t,n}$  are computed using the measurements  $Y_t$  by  $\pi_{t,n} = p(Y_t|X_t = s_{t,n})$ , where the probability can be modeled as a Gaussian density.

Using the new samples  $S_t$  one can estimate the new object position by  $\sum_{n=1}^{N_S} \pi_{t,n} f(s_{t,n}, W)$ . Particle filter based trackers can be initialized by either using the first measurements,  $s_{0,n} \sim X_0$ , with weight  $\pi_{0,n} = \frac{1}{N_S}$  or by training the system using sample sequences. In addition to keeping track of the best particles, an additional resampling is usually employed to eliminate samples with very low weights. Note that the posterior density does not have to be a Gaussian.

Particle filtering suffers from sample degeneracy and impoverishment, especially for higher dimensional representations due to the importance sampling.

Note that the particle filter described above assume a single measurement at each time instant, i.e. the state of single object is estimated. Tracking multiple objects requires a joint solution of data association and state estimation problems. There exist several statistical data association techniques to tackle this problem. A detailed review of these techniques can be found in [76]. Joint Probability Data Association Filtering (JPDAF) and Multiple Hypothesis Tracking (MHT) are two widely used techniques for data association explained in the following.

#### 4.6.1 Joint Probability Data Association Filter

Suppose we have  $k$  tracks and at time  $t$  and  $\{Y_{1t}, \dots, Y_{mt}\}$  are the  $m$  measurements. We need to assign these measurements to the existing tracks. Let  $\Gamma$  be a set of assignments. It is assumed that the number of tracks will remain constant over time. Let  $\gamma_{ij}$  be the innovation associated with the track  $j$  due to the measurements. The JPDAF associates all measurements with each track [77]. The combined weighted innovation is given by

$$\gamma_j = \sum_{i=1}^m z_{ij} \gamma_{ij}, \quad (18)$$

where  $z_{ij}$  is the posterior probability that the measurement  $i$  originated from the object associated with track  $j$  and is given as:

$$z_{ij} = \sum_{\Gamma} p(\Gamma_j | Y_{1t}, \dots, Y_{mt}) L_{i,j}(\Gamma), \quad (19)$$

where  $L_{ij}$  is the indicator variable,  $i = 1, \dots, m$  and  $j = 1, \dots, k$ . It is equal to one if the measurement  $Y_{it}$  is associated with track  $j$ , otherwise it is zero. The weighted innovation given in (18) can be plugged in the Kalman filter update equations (16) for each track  $j$ .

#### 4.6.2 Multiple Hypothesis Tracking

The major limitation of the JPDAF algorithm is its inability to handle new objects entering the scene or already tracked objects exiting the scene. Since the JPDAF algorithm performs data association of a fixed number of objects being tracked over two frames, serious errors can arise if there is a change in the number of objects. The MHT algorithm does not have this shortcoming.

As motion correspondence is established using only two frames, there is always a finite chance of an incorrect correspondence. Better tracking results can be obtained if the correspondence decision is deferred until several frames have been examined. The MHT algorithm maintains several correspondence hypotheses for each object at each time frame. The final track of the object is the most likely set of correspondences over the time period of its observation. The MHT algorithm has the ability to handle occlusions, i.e. continuation of a track even if some of the measurements from an object are missing.

An MHT iteration begins with a set of current track hypotheses. Each hypothesis is a collection of disjoint tracks. For each hypothesis, a prediction of each object's position in the next frame is made. The predictions are then compared with actual measurements by evaluating a distance measure. A set of correspondences (associations) are established for each hypothesis based on the distance measure, which introduces new hypotheses for the next iteration. Each new hypothesis represents a new set of tracks based on the current measurements. Note that each measurement can belong to a new

object entering the scene, a previously tracked object, or a spurious measurement. Moreover, a measurement may not be assigned to an object because the object may have exited the scene, or a measurement corresponding to an object may not be obtained. The latter happens because either the object is occluded or it is not detected due to noise.

Since the MHT makes associations in a deterministic sense and exhaustively enumerates all possible associations it is computationally exponential both in memory and time. To reduce the computational load, a Probabilistic Multiple Hypotheses Tracker (PMHT) is proposed by [78] in which the associations are considered to be statistically independent random variables. Thus, there is no requirement for exhaustive enumeration of associations. Alternatively, [79] use Murty's algorithm [80] to determine k-best hypotheses in polynomial time for tracking points. Particle filters to handle multiple measurements to track multiple objects have also been proposed, e.g. [81] where data association is handled in a similar way as in PMHT, however the state estimation is achieved through particle filters.

## 4.7 Silhouette Tracking

The goal of a silhouette based object tracker is to find the object region by means of an object model generated using the previous frames. Silhouette trackers can be categorized into two categories: matching and contour evolution. Shape matching approaches search for the object silhouette in the current frame. Contour evolution approaches, on the other hand, track an initial contour to its new position in the current frame by either using the state space models or direct minimization of some energy functional.

Important factors to distinguish different silhouette trackers are: What features are used? How occlusion is handled? and If the training is required or not? Moreover some algorithms only use information about the silhouette boundary for tracking while other use the complete region inside the silhouette. Generally the region based approaches are more resilient to noise.

The most important advantage of tracking silhouettes is their flexibility to handle a large variety of object shapes. Occlusion handling is another issue of silhouette tracking methods. Usually methods do not address the occlusion problem explicitly. A common approach is to assume constant motion or constant acceleration, where during occlusion the object silhouette from the previous frame is translated to its hypothetical new position. Another important aspect related to silhouette trackers is their capability of dealing with object split and merge. For instance, while tracking a silhouette of person carrying an object, when the person leaves an object, a part of the person's contour will be placed on the left object (region split). These topology changes of region split or merge can be handled well by implicit contour representations.

#### 4.7.1 Shape Matching

Shape matching often assumes similarity transform from the current frame to the next, therefore nonrigid object motion is not explicitly handled. It is usually carried out by background subtraction. Once the object silhouettes are extracted, matching is performed by computing some distance between the object models associated with each silhouette. The object model, which is usually in the form of an edge map, is reinitialized to handle appearance changes in every frame after the object is located. This update is required to overcome tracking problems related to viewpoint and lighting condition changes as well as nonrigid object motion.

[82] performs shape matching using an edge based representation. Hausdorff distance is used to construct a correlation surface, from which the minimum is selected as the new object position. In the context of matching using an edge based model, Hausdorff distance measures the most mismatched edges. Due to this, this method emphasize parts of the edge map that are not drastically affected by object motion. For instance, in the case of a walking person, the head and the torso do not change their shape much, whereas the motion of the arms and legs will result in drastic shape changes, such that, removing the edges corresponding to arms and legs will improve the tracking performance.

In contrast to looking for possible matches in consecutive frames, shape matching can be performed by computing the flow vectors for each pixel inside the silhouette, such that the flow, which is dominant over the entire silhouette, is used to generate the silhouette trajectory.

#### 4.7.2 Contour Evolution

Contour evolution requires some part of the object in the current frame overlap with the object region in the previous frame. Tracking by evolving a contour can be performed using two different approaches. The first approach uses state space models for contour shape and motion. However, explicit representations do not allow topology changes such as split or merge. On the other hand, the second approach directly evolves the contour by minimizing the contour energy using direct minimization techniques, such as gradient descent.

The state spaces are updated at each frame such that the contour's a posteriori probability is maximized. The posterior probability depends on the prior state and the current likelihood, which is usually defined in terms of the distance of the contour from observed edges [83] defines the state space by the dynamics of the control points. The dynamics of the control points are modeled in terms of a spring model, which moves the control points based on the spring stiffness parameters. The new state (spring parameters) of the contour is predicted using Kalman filter. The correction step uses the image observations which are defined in terms of the image gradients.



**Fig. 13** Contour tracking results in presence of occlusion using the method proposed in [87] (©[2004] IEEE).

In [75], the state space is defined in terms of spline shape parameters and affine motion parameters. The measurements consist of image edges computed in the normal direction to the contour. The state is updated using a particle filter. [84], extends the particle filter to track multiple objects by including the “exclusion principle” for handling occlusion. [85] proposes the contour is parameterized as an ellipse. Each contour node has an associated HMM and the states of each HMM is defined by the points lying on the lines normal to the contour control point. The observation likelihood of the contour depends on the background and the foreground partitions defined by the edge along the normal line on contour control points. The state transition probabilities of the HMM are estimated using the JPDAF.

Contour evolution minimizes an energy functional either by greedy methods or by gradient descent. The contour energy is defined in terms of temporal gradient [86], or appearance statistics generated from, for instance a band around the object boundary and the background regions [87]. The width of the band serves as a means to combine region and boundary based contour methods contour tracking methods into a single framework. The object shape and its changes are modeled using level sets to resolve the object occlusions during the course of tracking. Sample results of the appearance statistics based approach are given in Figure 13.

## 5 Final Observations

Tracking approaches that employ a stable model can only accommodate small changes in the object appearance but do not explicitly handle severe occlusions or continuous appearance changes.

Occlusion, either partial or full, can be classified into self occlusion, inter-object occlusion and occlusion by the background scene structure. Self occlusion occurs when one part of the object occludes another, especially for articulated objects. Inter-object occlusion occurs when two objects being tracked occlude each other, which is the common case in surveillance video. Similarly, occlusion by the background occurs when a structure in the background e.g. a column, a divider, etc., occludes the tracked objects. Generally, for inter-object occlusion, the multi-object trackers can exploit the knowledge of position and appearance of the occluded and occluding objects to detect

and resolve occlusion. Partial occlusion of an object by a scene structure is hard to detect, since it is difficult to differentiate between the object changing its shape and the object getting occluded.

A common approach to handle full occlusions during tracking is to assume motion consistency, and in case an occlusion is detected, to keep on predicting the object location till the object reappears. Among such predictors, Kalman filter can be given as an example. Occlusion can also be implicitly resolved during generation of object tracks. The chance of occlusion can be reduced by an appropriate selection of camera positions. For instance, if the cameras are mounted on for birds eye view of the scene, most occlusions can be eliminated. Multiple cameras viewing the same scene can also be used to resolve object occlusions during tracking.

Multi-camera tracking methods have demonstrated superior tracking results as compared to single camera trackers in case of persistent occlusion between the objects. In many situations it is not possible to have overlapping camera views due to limited resources or large areas of interest. Non-overlapping multi-camera tracking has to deal with sparse object observations. Therefore additional assumptions have to be made about the object speed and the path in order to obtain the correspondences across cameras. Methods that establish object correspondence assume 1) the cameras are stationary and 2) the object tracks within each camera are available. The performance of these algorithms depends greatly on how much the objects follow the established paths and expected time intervals across cameras.

Object appearance changes can be included in the model update by introducing noise and transient models. Despite explicit modeling of noise and transient features, trackers often perform poorly, or even lose tracking, in cases when the performer suddenly turns around during an action and reveals a completely different appearance, which has not been learned before.

A potential approach to overcome this limitation is to learn different views of the object and later use them during tracking. In addition, a tracker that takes advantage of contextual information to incorporate general constraints on shape and motion of objects will usually perform better than the one that does not exploit this information. The capability to learn object models online may greatly increase the applicability of a tracker. Unsupervised learning of object models for multiple non-rigid moving objects from a single camera remains an unsolved problem. One interesting direction, that has largely been unexplored, is the use of semi-supervised learning techniques for modeling objects. These techniques (co-training, transductive SVMs, constrained graph cuts) do not require prohibitive amount of training data.

Overall, additional sources of information, in particular prior and contextual information, should be exploited whenever possible to attune the tracker to the particular scenario. A principled approach to integrate these disparate sources of information will result in a general tracker that can be employed with success in business intelligence applications.

## References

1. Davis, J., Bobick, A.: The representation and recognition of action using temporal templates. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Juan, Puerto Rico (1997)
2. Karman, K., von Brandt, A.: Moving object recognition using an adaptive background memory. In: Capellini (ed.) Time-varying Image Processing and Moving Object Recognition, vol. II, pp. 297–307. Elsevier, Amsterdam (1990)
3. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: principles and practice of background maintenance. In: Proc. 7th Intl. Conf. on Computer Vision, Kerkyra, Greece (1999)
4. Wren, C., Azarbayejani, A., Darell, T., Pentland, A.: Pfnder: real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997)
5. Gao, X., Boult, T., Coetzee, F., Ramesh, V.: Error analysis of background adaption. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Hilton Head, SC (2000)
6. Stauffer, C., Grimson, E.: Adaptive background mixture models for real-time tracking. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Fort Collins, CO (1999)
7. Tuzel, O., Porikli, F., Meer, P.: A bayesian approach to background modeling. In: IEEE Workshop on Machine Vision for Intelligent Vehicles (MVIV) in Conjunction with CVPR (2005)
8. Elgammal, A., Harwood, D., Davis, L.: Non-parametric Model for Background Subtraction. In: Vernon, D. (ed.) ECCV 2000, Part II. LNCS, vol. 1843, pp. 751–767. Springer, Heidelberg (2000)
9. Porikli, F., Wren, C.: Change detection by frequency decomposition: wave-back. In: Proc. of Workshop on Image Analysis for Multimedia Interactive Services, Montreux (2005)
10. Stenger, B., Ramesh, V., Paragios, N., Coetze, F., Buhmann, J.: Topology free Hidden Markov Models: application to background modeling. In: Proc. 8th Intl. Conf. on Computer Vision, Vancouver, Canada (2001)
11. Wixson, L.: Detecting salient motion by accumulating directionary consistent flow. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000)
12. Mittal, A., Paragios, N.: Motion-based background subtraction using adaptive kernel density estimation. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Washington, DC (2004)
13. Oliver, N., Rosario, B., Pentland, A.: A Bayesian computer vision system for modeling human interactions. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000)
14. Porikli, F.: Multiplicative background-foreground estimation under uncontrolled illumination using intrinsic images (2005)
15. Zhao, C., Wang, X., Cham, W.-K.: Background subtraction via robust dictionary learning. EURASIP Journal on Image and Video Processing (2011)
16. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Annual Conference on Computational Learning Theory (1995)
17. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Annual Conference on Computational Learning Theory (1995)

18. Papageorgiou, C., Poggio, T.: A trainable system for object detection. *International Journal of Computer Vision* 38 (2000)
19. Mohan, A., Papageorgiou, C., Poggio, T.: Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2001)
20. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, CA (2005)
21. Zhu, Q., Avidan, S., Ye, M., Cheng, K.-T.: Fast human detection using a cascade of histograms of oriented gradients. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY (2006)
22. Porikli, F.: Integral Histogram: a fast way to extract histograms in Cartesian spaces. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, CA (2005)
23. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: Proc. 9th Intl. Conf. on Computer Vision, Nice, France (2003)
24. Tuzel, O., Porikli, F., Meer, P.: Region Covariance: A Fast Descriptor for Detection and Classification. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) *ECCV 2006*, Part II. LNCS, vol. 3952, pp. 589–600. Springer, Heidelberg (2006)
25. Tuzel, O., Porikli, F., Meer, P.: Human detection via classification on riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2008)
26. Felzenszwalb, P., Huttenlocher, D.: Pictorial structures for object recognition. *International Journal of Computer Vision* 61 (2005)
27. Mikolajczyk, K., Leibe, B., Schiele, B.: Multiple object class detection with a generative model. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY (2006)
28. Mikolajczyk, K., Schmid, C., Zisserman, A.: Human Detection Based on a Probabilistic Assembly of Robust Part Detectors. In: Pajdla, T., Matas, J.(G.) (eds.) *ECCV 2004*, Part I. LNCS, vol. 3021, pp. 69–82. Springer, Heidelberg (2004)
29. Leibe, B., Seemann, E., Schiele, B.: Pedestrian detection in crowded scenes. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, CA (2005)
30. Gavrila, D., Philomin, V.: Real-time object detection for smart vehicles. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Fort Collins, CO (1999)
31. Opelt, A., Pinz, A., Zisserman, A.: Incremental learning of object detectors using a visual shape alphabet. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY (2006)
32. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Annual Conference on Computational Learning Theory (1998)
33. Levin, A., Viola, P., Freund, Y.: Unsupervised improvement of visual detectors using co-training. In: Proc. 9th Intl. Conf. on Computer Vision, Nice, France (2003)
34. Kockelkorn, M., Luneburg, A., Scheffer, T.: Using transduction and multi-view learning to answer emails. In: European Conf. on Principle and Practice of Knowledge Discovery in Databases (2003)

35. Zhu, Q., Avidan, S., Cheng, K.: Learning a sparse, corner-based representation for time-varying background modeling. In: Proc. 10th Intl. Conf. on Computer Vision, Beijing, China (2005)
36. Comaniciu, D., Meer, P.: Mean-Shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002)
37. Serby, D., Koller-Meier, S., Gool, L.V.: Probabilistic object tracking using multiple features. In: Proc. 17th Int'l Conf. on Pattern Recognition, Cambridge, UK (2004)
38. Sethian, J.: Level set methods: evolving interfaces in geometry, fluid mechanics computer vision and material sciences. Cambridge University Press (1999)
39. Andriluka, M., Roth, R., Schiele, B.: Pictorial structures revisited: people detection and articulated pose estimation. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Miami, FL (2009)
40. Ross, D., Tarlow, D., Zemel, R.: Learning articulated structure and motion. *International Journal of Computer Vision* (2010)
41. Zelnik-Manor, L., Irani, M.: Event-based video analysis. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Kauai, HI (2001)
42. Rao, R., Yilmaz, A., Shah, M.: View invariant representation and recognition of actions. *International Journal of Computer Vision* 50 (2002)
43. Lowe, D.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60 (2004)
44. Freeman, W., Roth, M.: Orientation histograms for hand gesture recognition. In: Intl. Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland (1995)
45. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Madison, WI (2003)
46. Sande, K., Gevers, T., Snoek, C.: Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (2010)
47. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002)
48. Förstner, W., Moonen, B.: A metric for covariance matrices. Dept. of Geodesy and Geoinformatics, Stuttgart University (1999)
49. Avidan, S.: Ensemble tracking. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, CA (2005)
50. Edwards, G., Taylor, C., Cootes, T.: Interpreting face images using active appearance models. In: International Conference on Face and Gesture Recognition (1998)
51. Paschos, G.: Perceptually uniform color spaces for color texture analysis: an empirical evaluation. *IEEE Trans. on Image Processing* 10 (2001)
52. Canny, J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (1986)
53. Horn, B., Schunk, B.: Determining optical flow. *Artificial Intelligence* 17 (1981)
54. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Intl Joint Conf. on Artificial Intelligence (1981)
55. Sun, D., Roth, S., Black, M.: Secrets of optical flow estimation and their principles. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Francisco, CA (2010)

56. Mirmehdi, M., Xie, X., Suri, J. (eds.): *Handbook of Texture Analysis*. Imperial College Press (2008)
57. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proc. of Alvey Vision Conf. (1988)
58. Laptev, I.: On space-time interest points. *International Journal of Computer Vision* 64 (2005)
59. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. *Artificial Intelligence* (1997)
60. Schweitzer, H., Bell, J.W., Wu, F.: Very Fast Template Matching. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) *ECCV 2002*, Part IV. LNCS, vol. 2353, pp. 358–372. Springer, Heidelberg (2002)
61. Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using Mean-Shift. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Hilton Head, SC (2000)
62. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (2003)
63. Yilmaz, A.: Kernel based object tracking using asymmetric kernels with adaptive scale and orientation selection. *Machine Vision and Applications* 22 (2011)
64. Porikli, F., Tuzel, O.: Multi-kernel object tracking. In: Proceedings of IEEE Int'l Conference on Multimedia and Expo., Amsterdam, Netherlands (2005)
65. Porikli, F., Tuzel, O., Meer, P.: Covariance tracking using model update based on Lie algebra. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY (2006)
66. Porikli, F., Tuzel, O.: Object tracking in low-frame-rate video. In: Proc. of PIE/EI - Image and Video Communication and Processing, San Jose, CA (2005)
67. Tuzel, O., Porikli, F., Meer, P.: Learning on Lie groups for invariant detection and tracking. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Anchorage, AK (2008)
68. Porikli, F., Pan, P.: Regressed importance sampling on manifolds for efficient object tracking. In: 6th IEEE Advanced Video and Signal based Surveillance Conference (2009)
69. Shi, J., Tomasi, C.: Good features to track. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Seattle, WA (1994)
70. Jepson, A., Fleet, D., ElMaraghi, T.: Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (2003)
71. Broda, T., Chellappa, R.: Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8 (1986)
72. Rosales, R., Sclaroff, S.: A framework for heading-guided recognition of human activity. *Computer Vision and Image Understanding* 91 (2003)
73. Tanizaki, H.: Non-Gaussian state-space modeling of nonstationary time series. *Journal of the American Statistical Association* (1987)
74. Bouaynaya, N., Qu, W., Schonfeld, D.: An online motion-based particle filter for head tracking applications. In: Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Philadelphia (2005)
75. Isard, M., Blake, A.: Condensation: conditional density propagation for visual tracking. *International Journal of Computer Vision* 29 (1998)

76. Bar-Shalom, Y., Foreman, T.: Tracking and Data Association. Academic Press Inc. (1988)
77. Rasmussen, C., Hager, G.: Probabilistic data association methods for tracking complex visual objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23 (2001)
78. Streit, R., Luginbuhl, T.: Maximum likelihood method for probabilistic multi-hypothesis tracking. In: *Proceedings of SPIE* (1994)
79. Cox, I., Hingorani, S.: An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18 (1996)
80. Murty, K.: An algorithm for ranking all the assignments in order of increasing cost. *Operations Research* 16 (1968)
81. Hue, C., Cadre, J., Perez, P.: Sequential Monte Carlo methods for multiple target tracking and data fusion. *IEEE Trans. on Signal Processing* 50 (2002)
82. Huttenlocher, D., Noh, J., Ruckridge, W.: Tracking non-rigid objects in complex scenes. In: *Proc. 4th Intl. Conf. on Computer Vision*, Berlin, Germany (1993)
83. Terzopoulos, D., Szeliski, R.: Tracking with kalman snakes. In: Blake, A., Yuille, A. (eds.) *Active Vision*. MIT Press (1992)
84. MacCormick, J., Blake, A.: Probabilistic exclusion and partitioned sampling for multiple object tracking. *International Journal of Computer Vision* 39 (2000)
85. Chen, Y., Rui, Y., Huang, T.: JPDAF based HMM for real-time contour tracking. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Kauai, HI (2001)
86. Mansouri, A.: Region tracking via level set PDEs without motion computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002)
87. Yilmaz, A., Li, X., Shah, M.: Contour based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26 (2004)

# Auto-calibration of Non-overlapping Multi-camera CCTV Systems

Cristina Picus, Roman Pflugfelder, and Branislav Micusik

**Abstract.** Deployment of existing vision approaches in camera networks for applications such as human tracking show a large gap between user expectation and current results. Calibrated cameras could push these approaches closer to applicability, as physical constraints greatly complement the ill-posed acquisition process. Calibrated cameras promise also new applications as spatial relationships among cameras and the environment capture additional information. However, a convenient calibration is still a challenge on its own. This paper presents a novel calibration framework for large networks including non-overlapping cameras. The framework purely relies on visual information coming from walking people. Since non-overlapping scenarios make point correspondences impossible, time constancy of a person's motion introduces the missing complementary information. The framework obtains calibrated cameras starting from single camera calibration thereby bringing the problem to a reduced form suitable for multi-view calibration. It extends the standard bundle adjustment by a smoothness constraint to avoid the ill-posed problem arising from missing point correspondences. The stratified optimization suppresses the danger to get stuck in local minima. Experiments with synthetic and real data validate the approach.

## 1 Introduction

CCTV systems for business intelligence in security critical infrastructures can easily comprise hundreds of cameras. The contextual information available within the camera network is currently less exploited in practice. One reason out of many is the inexistence of a feasible manual maintenance of a sufficiently precise geospatial map for a large camera network. Making integration of contextual information on a large scale just from visual information possible, would dramatically amplify the

---

Cristina Picus · Roman Pflugfelder · Branislav Micusik  
AIT Austrian Institute of Technology, Donau-City-Straße 1, 1220 Vienna  
e-mail: {cristina.picus, roman.pflugfelder,  
branislav.micusik}@ait.ac.at

prospects of camera networks, for example, image understanding would profoundly profit of it in terms of capability and robustness.

Automatic camera calibration would definitely be a way out of this maintenance dilemma. However, for reason of costs and efficiency, cameras are typically installed in sparse networks with non-overlapping fields of view, thus, preventing two or more cameras to observe the same part of the world. The absence of point correspondences among camera views is a clear problem for traditional calibration approaches.

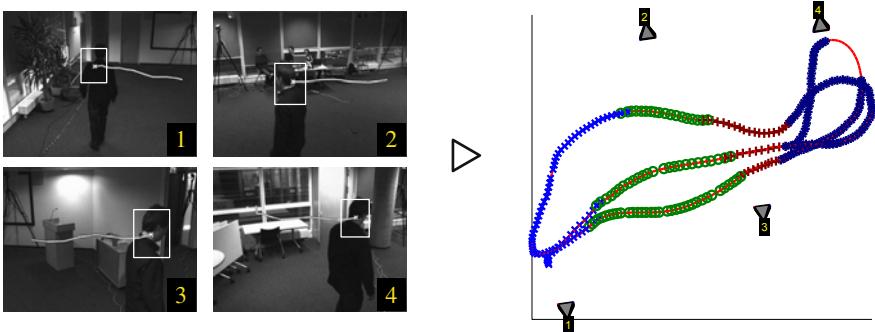
This paper presents a stratified calibration approach for solving the non-overlapping camera problem conveniently by using a human person as calibration target. At this point it is important to emphasize that the occurrence of people in the environment is a marginal assumption for business intelligence or surveillance applications as people are the most prominent target of interest in either cases. A calibration approach focusing on human targets therefore has significant practical relevance. As a precise localization of a person in the image is hard, the use of people as calibration targets is still a challenging research issue. This paper, however, tackles the underlying calibration approach excluding questions on person detection. The experiments show that available detectors allow reasonable results yet useful for particular business intelligence and surveillance tasks.

Neither internal nor external camera parameters are known in the beginning, only a synchronization in the acquisition of images and a commonly observed floor are assumed. The current approach works for normal lenses in the close- to mid-view range, that is, the distance human to camera is smaller than 15 m, because for larger distances the image resolution is insufficient to image the human shape with the necessary detail. A realistic, experimental indoor setup with four surveillance cameras shows localization errors in the camera centers below 1 m, which is within the limits the volume of a typical person occupies in space (Fig. 1). This result is expected to be sufficient for maintenance of geospatial maps in similar environments.

The proposed approach estimates the internal parameters of each camera independently, followed by a joint estimation of all external camera parameters. Each intrinsic calibration defines the local coordinate system in such a way that it advantageously constraints the problem of the extrinsic calibration where the problem of many local minima is greatly reduced.

The intrinsic calibration exploits frontally viewed human shapes whereas the single person has to stand at different locations on the floor but in the same pose in viewing direction camera. The problem is casted to a rectangular, quadratic eigenvalue problem which can be elegantly and to some extend robustly solved in quasi closed-form. This novel solution shows superior performance compared to state-of-the-art approaches in single view calibration [1] [2]. For example, HOG detection [3] and subsequent RANSAC verification [4] would basically make this step automatic and robust as long as the detector's outlier rate concerning frontal-poses of the person is below 50 %. However, as the paper's intention is to identify the approach's limits, we manually drew the bounding boxes for the considered human images.

The joint extrinsic calibration uses all visible parts of the human trajectory by using a human upper body part detector. Again, we identify the limits by using instead



**Fig. 1** The goal: Auto-calibration of non-overlapping cameras with a human as calibration target. Left: experimental setup of four cameras (1-4); this setup allows investigations of both the overlapping as well as the non-overlapping case. Right: cameras are localized in one coordinate frame; top view is shown. The trajectory (red) is reconstructed by the approach as byproduct; other colors depict available observations along the trajectory.

of the detector a projected bright light spots of a LED lamp which moves approximately planar and parallel to the floor through the environment. Such light spots are detectable with sub-pixel accuracy. Additionally the realistic case is tested which uses a human's trajectory from a state-of-the-art tracker. The standard approach by Ali Rahimi, Brian Dunagan and Trevor Darrell [5] for top view cameras is now applicable which extends its range of use to generally mounted cameras with varying focal lengths. Experiments show that both steps bring the problem very close to the optimal solution.

A final step performs a novel regularized bundle adjustment which accounts for the originally ill-posed problem when some points on the trajectory are not seen at all and when the most of the points are seen by at most one camera. It optimizes over all the camera parameters and enforces a smoothness constraint along the trajectory.

The remainder of the paper is organized as follows: After relating the proposed approach to the state-of-the-art (Section 2), Section 3 introduces the framework step by step from single-view calibration (Section 3.1), over trajectory alignment in the ground plane (Section 3.2), to the final bundle adjustment (Section 3.3). Section 4 gives results with a realistic experiment and Section 5 concludes the work.

## 2 Related Work

Photogrammetry and Computer Vision have dealt with camera calibration since their early beginning. Research questions in Photogrammetry were mainly questions about sufficient projection models among points in the world and in the images and questions about convenient optimizations methods to estimate the unknown model parameters, for example, the bundle adjustment [6]. Computer Vision research has additionally found answers to efficiently automate these optimization methods.

Today, Structure from Motion frameworks are available that allow camera calibration and reconstruction of whole cities with amazing accuracy and robustness [2]. These frameworks found even their way into commercial applications, for example, Microsoft's Photosynth™.

These algorithms deliver satisfying results as long as the user has full camera control and the scene is sufficiently structured and textured. In Video Surveillance, cameras are mounted in the scene with their fields of view often non-overlapping, hence, point correspondences are unavailable. Some environments are rich in structure, however, more often the scene seen by the cameras is without sufficient texture, for example, door views and corridor views.

Alternatively to texture, some approaches focus on additional calibration targets within the scene. Calibration targets such as checkerboards or laser pointers have been successfully used [8, 9]. Point light sources are detectable with high precision even with cameras having a small depth of field. Although, targeting mainly immersive multimedia applications, these approaches are only suitable in small environments, for example in indoor environments, where all camera views overlap. Reasonable checkerboards as well as laser pointers are invisible in the images of large environments, hence, algorithms rely on the surveilled targets such as vehicles or humans, however, with the challenge of low precision in their detection. As these targets usually move outdoor on the earth's surface or indoor on the floor, which is from a geometric point of view a critical situation in camera calibration, these algorithms necessarily assume a common ground plane in the environment.

Calibration frameworks tackling this situation in overlapping fields of view with known internal camera parameters can be traced back in literature to Lee et al. [10]. Planar trajectories in camera triplets are used to estimate the plane-induced inter-camera homographies, camera rotations and camera centers by non-linear trajectory alignment. The approach was further improved by successive dense alignment of the images [11] and additional vanishing points [12]. Estimating the inter-camera homographies from sole detections is also mentioned in literature [13, 14]. However, in many cases the detections are concentrated in a particular image region or they are collinear which is a critical situation for camera calibration. Flipping the viewpoint from cameras to the target allows an on-line, filtering approach with calibrated top view cameras. Kalman filtering with a linearized projection model has shown reasonable results with the additional benefit to provide confidence intervals for the model parameters [15]. To conclude, the requirement of having an overlap between fields of view is the main limitation of these approaches in practice.

Despite their importance, calibration frameworks for non-overlapping cameras are rarely treated in literature, only recently a work, which assumes calibrated, top view cameras, appeared [5]. Similar to bundle adjustment, an objective function based on known planar trajectories can be formulated. Although the problem is ill-posed, additional regularization by assuming smooth trajectories allows estimation of camera rotation, camera centers and trajectory's location simultaneously. The approach was also augmented for moving cameras and unknown trajectory association [16] and has also been improved in efficiency [17] by splitting the solution search space into parts corresponding to visible and not visible parts of the

trajectory. This paper basically extends this promising approach for uncalibrated cameras, general camera views and slightly non-planar trajectories to make it applicable in surveillance scenarios.

Estimating the topology of a camera network is related to camera calibration. Instead of calculating the geometric parameters of a quantitative model, topology is captured by a qualitative model such as a graph where vertices represent cameras and edges express spatial neighborhood of two cameras. Techniques [18] [19] are known that obtain topology of cameras with non-overlapping fields of view by exploiting time of appearance and disappearance of objects moving through the environment. Instead of accumulating positive evidence for an edge between two cameras, the dual approach of ruling out the inexistence of spatial neighborhood turned out to be more efficient, reliable and scalable to larger camera networks [20].

Topology can be used as a prior to simplify the correspondence problem during point matching [21] or object tracking. As this paper assumes identity maintenance of objects, topology is not considered in this work.

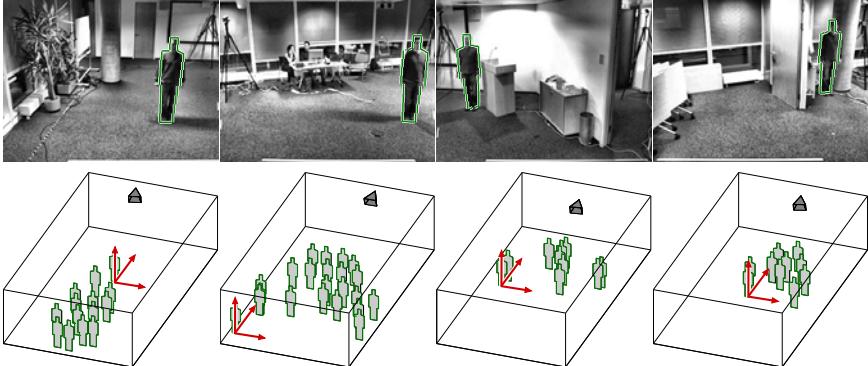
### 3 Network Calibration Framework

The calibration pipeline operates in three basic steps: first, a pre-calibration estimates the camera intrinsics and their calibration w.r.t. a local coordinate system [22]. Second, the local systems are aligned to a common reference frame using a Maximum A Posteriori (MAP) estimate of both camera parameters and trajectory after mapping on the ground-plane [5]. The combined solution of the pre-calibration and alignment step is used as initialization of a local optimization procedure which extends the standard BA [23] to non-overlapping cameras.

#### 3.1 Pre-calibration

The pre-calibration delivers for each  $i^{th}$  camera its focal length  $f^i$ , rotation  $\tilde{\mathbf{R}}^i$ , and translation  $\tilde{\mathbf{t}}^i$  w.r.t. a local coordinate system placed on the ground plane with one axis perpendicular to it, as shown in Fig. 2. The goal is to estimate the unknown camera parameters by observing a person standing at several roughly mutually parallel frontal positions, see Fig. 2. The local coordinate system is given by one of the person positions.

We present a pre-calibration method which is advantageous in two aspects: first, it does not require any special calibration target, just a person standing at different locations in fronto-parallel poses, which does not have to be necessarily parallel to the image plane. Second, coordinate systems of single cameras are related up to 1D rotation and 2D translation on the ground plane. That allows us to reduce the space of unknowns when trying to express all the cameras in one coordinate system shown in Fig. 5. For each camera we need then to search for 3 parameters (1 for rotation angle and 2 for translation) instead of 7 in the general case (1 for focal length, 3 for rotation angles and 3 for translation). Note, that the translations  $\tilde{\mathbf{t}}^i$  are already



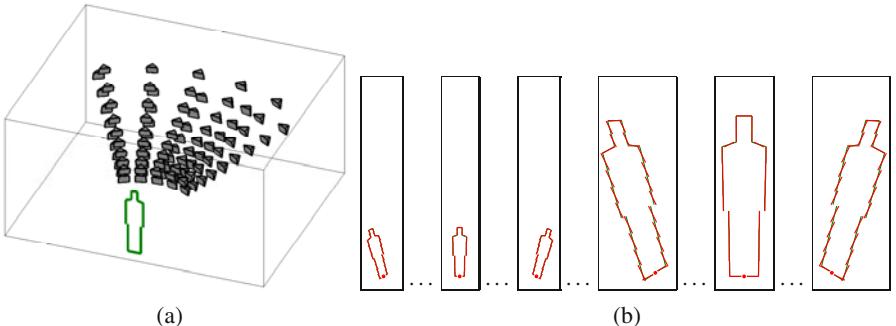
**Fig. 2** The single camera pre-calibration step. The cameras are independently pre-calibrated from an observed and automatically detected person (top row) standing at different locations. The calibrations are related to different local coordinate systems by  $[\tilde{\mathbf{R}}^i, \tilde{\mathbf{t}}^i]$  shown in the bottom views of the 3D plots.

estimated with the correct scale if the same person is used for calibration of all the cameras.

In the next we give more details on the pre-calibration method to get the parameters for each camera, i.e. the focal length  $f^i$ , rotation  $\tilde{\mathbf{R}}^i$ , and translation  $\tilde{\mathbf{t}}^i$ . We provide the basic concept, the full in-detail analysis can be found in [22]. The presented method has been shown to outperform techniques calibrating a single camera just from vanishing points [1] or similarly from human foot-head homology [2]. For simplicity in the next in this section we avoid the tildes above the rotation matrices and translation vectors which in fact indicate that the cameras are expressed in local coordinate systems and we drop the camera index number  $i$  as the pre-calibration is done independently for each camera.

### 3.1.1 Concept

The general concept of the method is the following. First, a person is detected in each frame by sliding a contour based human detector at all possible scales and rotations, some example detections are shown in Fig. 2. Second, the best  $N$  candidates are considered for each image and all the detections are used in a RANSAC-based estimation process to cope with outliers and to get the desired focal length, the camera extrinsics, and homology. The homology is a transformation which allows to compute the expected head location for each foot point and thus to select only feasible models to be evaluated at each point. It allows to re-run the contour detector again but with a finer resolution of the detector scale and rotation followed by the calibration method. This two-way iteration strategy allows to decrease computational burden which would be necessary for evaluating all the detailed models from the beginning.



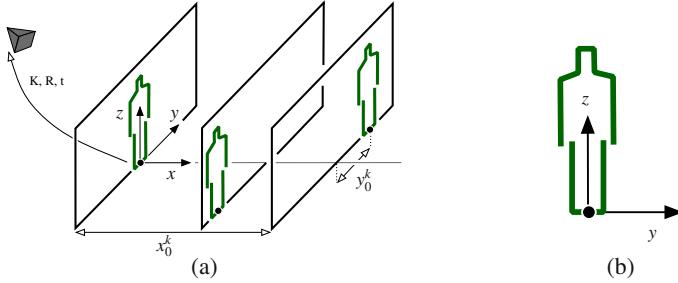
**Fig. 3** Generating the models for template contour matching. (a) A virtual camera is placed at different locations to capture many different human projections. Each camera is also rotated around its optical axis by a yaw angle which is not shown in the figure. (b) Some of the projections. The vertical dimension of the frame corresponds to the image vertical dimension.

### 3.1.2 Human Detection

At the beginning of the human detection step we have no prior about a foot-head relation. Therefore, when there is no information about the scene geometry, many hypothesized models capturing various person poses must be evaluated, see Fig. 3b. In order to avoid traversing the whole image when detecting a person, one can automatically pre-select frontal poses by a more general human detector, e.g. by the HOG detector [3]. However, one should notice that the detector's pose sensitiveness would require an extensive training data set of people in frontal pose. To efficiently validate all the poses we adopt the integral images based method [24] extended to generate synthetic projections of human contours from various camera viewpoints, giving us an ability to model much more human poses, roughly a thousand, than in [24], see Fig. 3.

The detection method we use falls into the group of generative approaches. Such approaches to recognizing objects have turned out to be a very promising direction [25] [26]. It has been shown [26] that the silhouettes alone without appearance features are very discriminative. The basic idea is to generate many silhouette images of a synthetic 3D model of objects of interest by changing a virtual camera viewpoint. Then, in the inference stage, a query image is traversed and each hypothesized image location is compared against all the stored object silhouette projections, or just the most representative ones.

The contour based detector provides richer information than e.g. the HOG detector [3]. The silhouette approximated by a piecewise linear contour allows us to detect interest points along the contour, e.g. when shoulders meet the neck. These serve as point correspondences, the key elements to define a mathematical framework for the following calibration method.



**Fig. 4** (a) A considered scene with a camera and a parallel standing person. (b) Coordinate system of the human edge model.

### 3.1.3 Shifted Homographies

Let us choose a world coordinate system at one human foot location and orientation of the coordinate axes as shown Fig. 4. The projection of a 3D point  $\mathbf{X}$  into a camera point  $\mathbf{u}$  reads as

$$\mathbf{u} = \alpha K [\mathbf{R} \ \mathbf{t}] \mathbf{X}, \quad (1)$$

where  $\alpha$  is a scale, and  $K, R, t$  stand for camera calibration matrix, rotation and translation w.r.t. the world coordinate system [27]. We assume that all human positions are parallel and  $x = 0$ , since fixing the coordinate system into the first detection. For each human pose thus holds

$$\begin{aligned} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} &= \alpha K [\mathbf{R} \ \mathbf{t}] \begin{pmatrix} x + x_0 \\ y + y_0 \\ z \\ 1 \end{pmatrix} = H \begin{pmatrix} y \\ z \\ 1 \end{pmatrix} = H\mathbf{x} \\ &= \alpha K [\mathbf{r}_2 \ \mathbf{r}_3 \ x_0\mathbf{r}_1 + y_0\mathbf{r}_2 + \mathbf{t}] \mathbf{x} \end{aligned} \quad (2)$$

where  $\mathbf{r}_i$  is the  $i$ -th column of the rotation matrix  $R$  and  $H$  is a  $3 \times 3$  homography matrix,  $u$  and  $v$  are detections in the image.  $\mathbf{x}$  are corresponding points on the world planes spanned by  $y$  and  $z$ . It is evident that if we compute the homography between the human edge model and all their projections, all the homographies will be the same up to the third column. This allows us to estimate the homography simultaneously from more, let's say  $K$ , detections and design the following linear system

$$M\mathbf{h} = M (\mathbf{h}_1^\top \ \mathbf{h}_2^\top \ \mathbf{h}_3^{1\top} \ \dots \ \mathbf{h}_3^{K\top})^\top = \mathbf{0}, \quad (3)$$

where  $\mathbf{h}_i$  is the  $i$ -th column of the homography, i.e.  $H^k = [\mathbf{h}_1 \ \mathbf{h}_2 \ \mathbf{h}_3^k]$  between the  $k$ th detection and the human edge model in Fig. 4b. The matrix  $M$  is composed of detected image model points and the corresponding human model points. These points are selected to be the intersections of lines in the piece-wise linear human model, shown in Fig. 4b. There are  $6 + 3K$  unknowns in Eq. (3). Each point correspondence adds 2 equations and for each detection, we must provide at least one point correspondence to give constraints on  $\mathbf{h}_3^k$  which is unique for each detection. Altogether

there must hold  $\text{rank}(\mathbf{M}) = (6 + 3K) - 1$ . That means, for one detection 4 correspondences are required, for two detections 6, for three 7 and so on. Including more detections and more correspondences leads to an overconstrained problem solvable in the least square sense by SVD. We use all the corner points of the model and two human detections, i.e. two images. Hartley's point normalization [27] is used before filling the matrix  $\mathbf{M}$  to improve the numerical stability.

### 3.1.4 Focal Length, Rotation, Translation

From Eq. (2) we know that the last column of the homography for each human pose factorizes to

$$\alpha \mathbf{K}(x_0^k \mathbf{r}_1 + y_0^k \mathbf{r}_2 + \mathbf{t}) = \mathbf{h}_3^k, \quad (4)$$

with the unknown calibration matrix  $\mathbf{K}$ ,  $\mathbf{t}$ ,  $x_0^k$ ,  $y_0^k$  and scale  $\alpha$ . Let assume square pixels and known principal point of the camera. Next, before solving Eq. (3), we express all image detections in the image coordinate system with its origin at the principal point. Thus, the calibration matrix is  $\mathbf{K} = \text{diag}(f, f, 1)$ , where  $f$  is the focal length. We can write

$$\alpha x_0^k \mathbf{K}(\mathbf{r}_2 \times \mathbf{r}_3) + \alpha y_0^k \mathbf{K}\mathbf{r}_2 + \alpha \mathbf{K}\mathbf{t} - \mathbf{h}_3^k = \mathbf{0}. \quad (5)$$

Recall that we have estimated  $\mathbf{h}_1$  and  $\mathbf{h}_2$  and we know from Eq. (2) and Eq. (3) that  $\mathbf{h}_1 = \alpha \mathbf{K}\mathbf{r}_2$  and  $\mathbf{h}_2 = \alpha \mathbf{K}\mathbf{r}_3$ . Then the first summand in Eq. (5) becomes

$$\frac{x_0^k}{\alpha} \det(\mathbf{K}^{-1}) \mathbf{K} \mathbf{K}^\top (\mathbf{h}_1 \times \mathbf{h}_2) = \frac{1}{f^2} \frac{x_0^k}{\alpha} \mathbf{K} \mathbf{K}^\top (\mathbf{h}_1 \times \mathbf{h}_2).$$

Substituting that back into Eq. (5) and multiplying by  $f^2$  we get

$$\frac{x_0^k}{\alpha} \mathbf{K} \mathbf{K}^\top (\mathbf{h}_1 \times \mathbf{h}_2) + y_0^k \mathbf{h}_1 f^2 + \alpha \mathbf{K} \mathbf{t} f^2 - \mathbf{h}_3^k f^2 = \mathbf{0} \quad (6)$$

Introducing  $\check{x}_0^k = \frac{x_0^k}{\alpha}$ ,  $\check{\mathbf{t}} = \alpha \mathbf{t} = (\check{t}_x \check{t}_y \check{t}_z)^\top$ , and  $\mathbf{e} = (\mathbf{h}_1 \times \mathbf{h}_2) = (e_x e_y e_z)^\top$  we get for each pose the following set of equations

$$\begin{aligned} \check{x}_0^k e_x + y_0^k h_{1x} + f \check{t}_x - h_{3x}^k &= 0 \\ \check{x}_0^k e_y + y_0^k h_{1y} + f \check{t}_y - h_{3y}^k &= 0 \\ \check{x}_0^k e_z + f^2 y_0^k h_{1z} + f^2 \check{t}_z - f^2 h_{3z}^k &= 0 \end{aligned} \quad (7)$$

with six unknowns  $f$ ,  $\check{\mathbf{t}}$ ,  $\check{x}_0^k$ ,  $y_0^k$ . Each additional human pose adds two new unknowns,  $\check{x}_0^k$  and  $y_0^k$ , but provides three more equations.

### Minimal Solution

To get a solution we need two detections, i.e. we get six equations with six unknowns. Recall that we set the world coordinate system to be in one of the detections, therefore

$\check{x}_0^1 = y_0^1 = 0$ , reducing the number of unknowns by two. To simplify the notation, let us denote symbols with index  $k = 1$ , i.e. denoting the first detection, as primed and symbols with  $k = 2$ , i.e. the second detection, as double primed.

From the first three equation we have then  $f\check{x} = h'_{3x}$  and  $f\check{y} = h'_{3y}$  and after elimination process of Eq. (7) we get finally

$$\begin{aligned}\check{x}_0'' &= \frac{h_{1y}h'_{3x} - h_{1x}h'_{3y} + h_{1x}h''_{3y} - h_{1y}h''_{3x}}{e_yh_{1x} - e_xh_{1y}}, \\ y_0'' &= \frac{e_yh'_{3x} - e_xh'_{3y} + e_xh''_{3y} - e_yh''_{3x}}{e_xh_{1y} - e_yh_{1x}}, \\ f &= \sqrt{\frac{-\check{x}_0''e_z}{y_0''h_{1z} + h'_{3z} - h''_{3z}}}, \\ \alpha &= \|K^{-1}\mathbf{h}_1\|_2, \\ \mathbf{t} &= \check{\mathbf{t}}/\alpha = (h'_{3x}/f \ h'_{3y}/f \ h'_{3z})^\top/\alpha.\end{aligned}\quad (8)$$

The estimated scale  $\alpha$  and the focal length  $f$  allow us to fully recover the rotation matrix  $R$ , see Eq. (2), thus  $\mathbf{r}_2 = (\alpha K)^{-1}\mathbf{h}_1$  and  $\mathbf{r}_3 = (\alpha K)^{-1}\mathbf{h}_2$ .

## Overconstrained Solution

We can rewrite Eq. (6), resp. Eq. (7), in the form of the Quadratic Eigenvalue Problem (QEP),

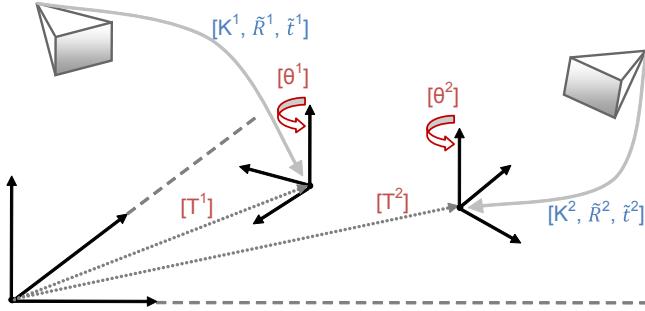
$$(D_1 + D_2f + D_3f^2)\mathbf{v} = \mathbf{0}, \quad (9)$$

with known design matrices  $D_i$ , the unknown focal length  $f$  and the vector  $\mathbf{v}$  with unknown parameters

$$\mathbf{v} = (\check{\mathbf{t}}^\top \ 1 \ \check{x}_0^2 \ y_0^2 \ \cdots \ \check{x}_0^K \ y_0^K)^\top. \quad (10)$$

Eq. (9) is linear in elements of  $\mathbf{v}$  and quadratic in the focal length. The design matrices  $D_1, D_2, D_3$  are sparse of size  $3K \times (2K + 2)$  such that each detection adds 3 rows. If  $K = 2$ , they are square and Eq. (9) is equivalent to the aforementioned Minimal Solution. For  $K > 2$ , the matrices become rectangular as each new pose adds three equations but only two unknowns.

When the design matrices  $D_{\{1,2,3\}}$  in Eq. (9) are square, they yield a square QEP and there are many solvers available, e.g. `polyeig` in MATLAB. Alternatively the QEP can be easily converted to a generalized eigenvalue problem (GEP) and for which even more solvers exist. The QEP as an interesting solvable non-linear algebraic problem [28] which has appeared in many scientific disciplines and many of such problems are summarized in [29]. Many computer vision has been already formulated as QEP, e.g. estimating epipolar geometry with unknown radial distortion [30] or unknown omnidirectional camera model [31], 5pt and 6pt relative pose problem [32], essential matrix estimation with a single unknown focal length [33], relative pose problem for non-overlapping cameras [34].



**Fig. 5** The estimated camera parameters after the pre-calibration  $[K^i, \tilde{R}^i, \tilde{t}^i]$  refer to different local ground-plane reference systems. The task of the ground-plane alignment is to find for each camera  $i$  the 2D translation and the rotation about the vertical axes  $\mu^i = [\mathbf{T}^i, \theta^i]$  used to define the camera parameters in a global reference system.

If the matrices are rectangular, as in our case, one gets a *rectangular* QEP. The simplest step to make the rectangular QEP square is to left multiply Eq. (9) by  $D_1^\top$ , as suggested by [30]. This trick has virtue of preserving the true solution in the noiseless case. However, such a solution suffers from a significant bias and variance in the presence of noise. We follow the work of [35] based on a perturbation approach. To find a global optimum for  $\mathbf{v}$  and  $f$  we proposed a solution which is more robust to noise. More details can be found in [22].

### 3.1.5 Foot-Head Homology

From the estimated calibration matrix  $K$ , rotation  $R = [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{r}_3]$  and translation  $t$ , the foot-head homology can be constructed as

$$\mathbf{H}_{FH} = K[\mathbf{r}_1 \ \mathbf{r}_2 \ l\mathbf{r}_3 + \mathbf{t}][\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}]^{-1}K^{-1}, \quad (11)$$

such that  $\mathbf{u}_H \simeq \mathbf{H}_{FH}\mathbf{u}_F$ , where  $\mathbf{u}_F$  are homogeneous image points at foot locations and  $\mathbf{u}_H$  their corresponding head positions. The constant  $l$  is the height of a person in pixels, set to the height of the human model used for estimating the homographies in Eq. (2).

## 3.2 Ground-Plane Alignment

The ground-plane alignment is the second step in the stratified optimization approach. It builds up on the output delivered by the pre-calibration, i.e. the camera parameters  $[K^i, \tilde{R}^i, \tilde{t}^i]$  for each camera  $i$  and delivers relative camera rotations and translation in a common reference system,  $\mu^i = [\mathbf{T}^i, \theta^i]$ . As the pre-calibration is done for each camera independently, an obvious consequence is that any information about the relative positioning of the cameras is missing. Recovering such information is the goal of the ground-plane alignment step.

The single camera parameters are defined in different local coordinate systems. The origin of the local systems is located on the ground-plane in a position arbitrarily defined by one of the detected person positions used for the pre-calibration. The task is to find for each camera a 2D translation of the origin,  $\mathbf{T}^i$ , and a rotation about the axis perpendicular to the ground-plane,  $\theta^i$ , in order to align the local systems to a common reference system. The task and the parameters involved are sketched in Fig. 5.

The cue used to recover the global information about the camera network is the trajectory of a moving target, e.g. a person. By the assumption that the cameras are synchronized, the problem can be fully solved given that there is an overlap between the field of views of the cameras. This is however a very restrictive assumption that usually does not hold. Therefore in the following we focus on non-overlapping scenarios, in which the motion of the target in the area of visual gap between the cameras is not observed. In order to compensate for the lack of information, the target's motion is assumed to be smooth and is described by a stochastic dynamics. Such assumption is generally true for small gaps.

The optimization problem is formulated following [5] as a MAP estimate of camera parameters and trajectory. Although such approach is thought for general positioning of the cameras given the local homography, the authors constraints experiments to the top camera view scenario with all cameras at the same height. This is to avoid pre-calibration and the determination of the focal length, which however could potentially break the assumption of planarity of the 2D trajectory on which the method is based. In Sec. 4 we show that the method works even if the planarity condition is only approximately fulfilled.

From a practical point of view, the approach has the clear advantage of exploiting information easily available in the environment, such as that provided by a moving person. Moreover, as already pointed out by the authors in [5], constraining the optimization to a reduced parameter space enables in many cases convergence of an otherwise very hard optimization problem. This is not the case in the general optimization by BA. To the best of our knowledge, currently there are no alternative closed-form techniques for non-overlapping cameras to deal with the non-linearity coming from the rotation parameters [36].

In the next sections we provide the general background of the method. For implementation details we refer to the original work in [5].

### 3.2.1 MAP Formulation

Camera parameters  $\mu$  and target's state  $\mathbf{x}$  are jointly estimated as the solution of a MAP optimization problem.

The state space vector  $\mu$  is defined as the collection of camera parameters  $\mu^i = [\mathbf{T}^i, \theta^i]$  for all  $N$  cameras  $\mu = [\mu^1, \dots, \mu^N]$ . Similarly, the target's state space vector is the cumulation of target states for each time stamp  $\mathbf{x} = [\mathbf{x}_1, \dots, \mathbf{x}_T]$ . The state at time  $t$  is defined by position and velocity of the target in the global coordinate

system,  $\mathbf{x}_t = [x_{u,t}, \dot{x}_{u,t} \ x_{v,t} \ \dot{x}_{v,t}]^T$ ,  $u$  and  $v$  representing the horizontal and vertical directions.

Considering the measurements  $\mathbf{y}$  as the full set of target coordinates in the camera views, the optimal solution is given by the maximum of the a-posteriori probability

$$[\mathbf{x}^*, \mu^*] = \arg \max_{\mathbf{x}, \mu} p(\mathbf{y}|\mathbf{x}, \mu) p(\mathbf{x}) p(\mu). \quad (12)$$

The first term in Eq. (12) is the likelihood of the target's measurements. As explained in Sec. 3.2.2, the likelihood criterion is the reprojection error, i.e. the error made by back-projecting the estimated target state in the image views using the estimated camera parameters.

$p(\mathbf{x})$  and  $p(\mu)$  are the a priori distribution of trajectory and camera parameters respectively.  $p(\mathbf{x})$  is defined using a linear Gaussian Markov dynamics to describe the state transition of the target  $\mathbf{x}_t \rightarrow \mathbf{x}_{t+1}$ . Contrary to standard BA in SfM, which always considers overlapping views, in the non-overlapping case part of the trajectory is not observed. In this case the contribution of  $p(\mathbf{x})$  becomes essential to gain enough constraints for solving the optimization and is therefore handled explicitly in Sec. 3.2.3.

Finally, the last term in Eq. (12) is the prior for the camera parameters. This is modeled using a multi-variate Gaussian distribution with zero mean and variance  $\Sigma_\mu$ . If no a-priori information about the cameras is available, the standard deviation is set to high values and the prior is non informative. To fix the gauge freedom, the parameters of one of the cameras are set at the origin of the global reference system.

### 3.2.2 Data Term

The likelihood of the full set of measurements is computed using a naive Bayes model, which considers all measurements as independently and identically distributed:

$$p(\mathbf{y}|\mathbf{x}, \mu) = \prod_{i,t} \delta(i, t) p(\mathbf{y}_t^i | \mathbf{x}_t, \mu^i) \quad (13)$$

$i$  is the camera index and  $t \in T$  the time stamps for which measurements are available. The indicator function  $\delta(i, t)$  takes the value 1 at times  $t$  in which the target is detected in camera  $i$ , otherwise it is 0. Therefore only time stamps for which measurement data are available contribute to the likelihood. This is also the reason for the name *data term*.

Given the camera parameters,  $\mu^i = [\mathbf{T}^i, \theta^i]$ , the  $i$ -th measure and its corresponding global state in the reconstructed trajectory are related by a homography given by composing a 1D rotation of angle  $-\theta^i$ , defined by the rotation matrix  $\mathbf{R}_{1D}^i$ , and a 2D translation  $\mathbf{T}^i$ :

$$\mathbf{y}_t^i = \pi^i(\mathbf{x}_t) + w_t = \mathbf{R}_{1D}^i(\mathbf{C}\mathbf{x}_t - \mathbf{T}^i) + w_t \quad (14)$$

$C$  is an operator that extracts the coordinate components out of the state  $\mathbf{x}_t$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (15)$$

$w_t$  is a Gaussian noise with covariance  $\Sigma_y = \sigma_y^2 I$ . This implies the following Gaussian likelihood probability for single detections:

$$p(\mathbf{y}_t^i | \mathbf{x}_t, \mu^i) = \mathcal{N}(y_t^i | \pi^i(\mathbf{x}_t), \sigma_y^2 I) \quad (16)$$

### 3.2.3 Smoothness Term

The a-priori distribution of the target's state is generated assuming an evolution model following a Markov dynamics. Therefore it is a concatenation of transition densities between consecutive time stamps:

$$p(\mathbf{x}) = p(\mathbf{x}_0) \prod_{t=1}^T p(\mathbf{x}_t | \mathbf{x}_{t-1}) \quad (17)$$

We assume a state space model that evolves according to a linear Gaussian Markov dynamics:

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + v_t \quad (18)$$

defined by the state space matrix  $A$

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (19)$$

$v_t$  is a Gaussian random noise of covariance  $\Sigma_v$ , which is used to model small fluctuations in the target motion. The state evolution model enforces a smoothness constraint on the trajectory, as the predicted position of the target is given by the previous position plus the velocity.

From Eq. (18) the transition density is given:

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; A\mathbf{x}_{t-1}, \Sigma_v) \quad (20)$$

which leads to an a-priori distribution in the form of a multi-variate Gaussian  $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \Lambda_x)$  with covariance matrix  $\Lambda_x$  expressed in terms of single term covariances  $\Sigma_x$ . The  $t$ -th row of  $G = \sqrt{\Lambda_x^{-1}}$  is given by:

$$G_t = \sqrt{\Sigma_x^{-1}} [0_{4 \times 4}, \dots, \underbrace{-A}_{t-1}, \underbrace{I_{4 \times 4}}_t, \dots, 0_{4 \times 4}] \quad (21)$$

### 3.2.4 MAP Cost Function

The optimization problem in Eq. (12) is equivalent to finding the maximum of the logarithm of the posterior probability. As all the terms in the posterior are Gaussian probabilities, the optimization is equivalent to minimizing the cost function given by the sum of the arguments of the Gaussian distributions, in particular the data and the smoothness terms

$$[\mathbf{x}^*, \boldsymbol{\mu}^*] = \arg \min_{\mathbf{x}, \boldsymbol{\mu}} \sum_{t \in T} \sum_i \delta(i, t) \|\mathbf{y}_t^i - \pi^i(\mathbf{x}_t)\|_{\Sigma_y}^2 + \mathbf{x}^T \boldsymbol{\Lambda}_x \mathbf{x} + \|\boldsymbol{\mu} - \boldsymbol{\mu}_0\|_{\Sigma_\mu}^2 \quad (22)$$

The optimization is a non-linear least squares problem which can be solved using Newton-Raphson.

## 3.3 Bundle-Adjustment for Non-overlapping Cameras Scenario

The last step of the calibration framework refines the solution of the first two steps by solving the full 3D calibration and reconstruction problem.

The output of the first steps is already a full reconstruction of target's trajectory  $\mathbf{x}$  and camera parameters,  $[\mathbf{K}^i, \mathbf{R}^i, \mathbf{t}^i]$ , for each camera  $i$ . However the optimization is performed separately in disjoint portions of the original parameter space. The joint optimal solution is provided by the bundle adjustment optimization.

Bundle adjustment [23] is a standard optimization method which is commonly used as the last step of 3D reconstruction problems to find a local solution which is optimal under Gaussian measurements noise. The general reconstruction problem is to find, starting from an approximate estimate of the solution and given the temporal sequence of 3D points  $\mathbf{X}_1 \dots \mathbf{X}_T$  and the set of projections  $\mathbf{u}_t^i$  of the point visible at time instance  $t$  in the camera  $i$ , the camera parameters and the 3D points that minimize the reprojection error between predicted and observed image points. This requires the solution of a non linear least square problem over a large parameter space, a hard optimization problem that can be solved only locally by proper initialization. Pre-calibration and ground-plane alignment provide the approximate solution needed to initialize bundle adjustment.

In the usual reconstruction at least some of the 3D points are observed from multiple views, which provide the necessary constraints to infer the reciprocal location of the cameras. In the non-overlapping camera scenario such constraints are again missing. Analogue to the approach of Rahimi et al. for planar trajectories we generalize BA to the non-overlapping case by introducing in the corresponding cost function a smoothness constraint.

The advantage of the bundle adjustment is that it provides a solution which is the true maximum likelihood estimate, at least when the error is modeled as zero-mean Gaussian distributed. Moreover, assumptions that were made in the first two steps in order to define the two sub-problems, may be released in the last optimization step. This leads to a more general problem definition. For instance, in the ground-plane alignment the trajectory is enforced to be planar. However it is a matter of fact

that by tracking humans, either the body center or the head top are tracked, whose motion is only approximately planar. The bundle-adjustment is able to correct for such deviations.

### 3.3.1 Concept

BA is formulated in a Bayesian framework which leads to a MAP formulation equivalent to Eq. (12).

The state variables are again  $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_T]$ , the collection of target's states over time, and  $\mu = [\mu_1, \dots, \mu_N]$ , the set of parameters for all cameras. However they are defined differently in this case.  $\mathbf{X}_t = [X_u, X_v, X_z]$  are the coordinates of the target trajectory in 3D at time  $t$ . The camera parameters,  $\mu^i = [K^i, R^i, t^i]$ , include focal length in the  $3 \times 3$  intrinsic calibration matrix  $K^i$  and 3 rotation and 3 translation parameters that define the extrinsic orientation.

The likelihood criterion for target's measurements is also equivalent and corresponds to minimizing the re-projection error. Given the higher dimensionality of the parameter space, special attention is paid to the parametrization [37]. In order to avoid singularities, which are inherent in the representation of rotations in terms of Euler angle, rotations are parameterized using quaternions under the constraint of unitary norm,  $\|\mathbf{q}\|^2 = 1$ .

The prior on target's state is the smoothness constrained extended to 3D.

The usually non-informative prior about camera parameters is skipped at this stage of the optimization.

The MAP formulation leads to an equivalent criterion as in Sec. 3.2 namely a cost function which is the sum of negative log-likelihoods coming from the different error estimates. In particular, we use again Gaussian error distributions, which leads to cost terms in form of sum of squares of the predicted errors, weighted by the respective covariance matrices. In the following we directly provide the data and smoothness terms adding up in the cost function.

### 3.3.2 Data Term

The data term is represented by the squared reprojection error computed using the projective camera model in Eq. (1). The error is weighted by the measurement covariance  $\Sigma_U$  using the Mahalanobis distance  $\|\cdot\|_{\Sigma_U}$

$$d_U^2 = \sum_t \sum_i \delta(i, t) \left\| \mathbf{u}_i^i - \alpha K^i [R^i \ t^i] \mathbf{X}_t \right\|_{\Sigma_U}^2. \quad (23)$$

$\alpha$  stands for the scale, which is arbitrarily fixed.  $K$ ,  $R$  and  $t$  are the camera intrinsic calibration matrix, the rotation and translation w.r.t. the world coordinate system [27], the latter derived by composing the local  $\tilde{R}^i$  and  $\tilde{t}^i$  with the parameters from the plane alignment  $[T^i, \theta^i]$ . The indicator function  $\delta(i, t)$  takes the value 1 at times  $t$  in which the target is detected in camera  $i$ , otherwise 0.

### 3.3.3 Smoothness Term

Smoothness is enforced on the trajectory by minimizing the second derivative

$$\|\mathbf{X}_{t+1} - 2\mathbf{X}_t + \mathbf{X}_{t-1}\|_2 \rightarrow 0. \quad (24)$$

Given the temporal sequence of 3D points  $\mathbf{X} = [\mathbf{X}_1 \dots \mathbf{X}_T]^\top$ , the smoothness term corresponding to the temporal constraint is defined as

$$d_X^2 = \mathbf{X}^T \mathbf{G}^T \mathbf{G} \mathbf{X}, \quad (25)$$

where  $\mathbf{G}$  is defined as the stack over all time stamps of the matrices  $\mathbf{G}_t$

$$\mathbf{G}_t = \sqrt{\Sigma_X^{-1}} [0_{3 \times 3}, \dots, \underbrace{\mathbb{I}_{3 \times 3}}_{t-1}, \underbrace{-2\mathbb{I}_{3 \times 3}}_t, \underbrace{\mathbb{I}_{3 \times 3}}_{t+1}, \dots, 0_{3 \times 3}]. \quad (26)$$

with smoothness covariance  $\Sigma_X$ .

### 3.3.4 BA Cost Function

Finally, the minimization problem w.r.t.  $\mathbf{X}$ , the camera rotations  $\mathbf{R} = \{\mathbf{R}^i\}$  and translations  $\mathbf{t} = \{\mathbf{t}^i\}$  is defined by the sum of the data and smoothness terms

$$[\mathbf{X}^*, \mathbf{R}^*, \mathbf{t}^*] = \arg \min_{\mathbf{X}, \mathbf{R}, \mathbf{t}} d_U^2(\mathbf{X}, \mathbf{R}, \mathbf{t}) + d_X^2(\mathbf{X}). \quad (27)$$

The covariance matrices of the data and smoothness terms are the model parameters. We use diagonal matrices and set the variance associated to the data term a few orders of magnitude smaller than that of the smoothness term. Therefore the minimization of the reprojection error is the leading criterion of the reconstruction if points are observed in at least one image view and the smoothness constraint is the only criterion when observations are missing.

The optimization is solved using the Levenberg-Marquardt (LM) algorithm which has proven to be efficient for non-linear least-squares problems.

### 3.3.5 Algorithm Outline

The combined pre-calibration, ground-plane alignment and bundle adjustment consists finally of the following steps

1. Detection of human bodies via the contour based method in Sec. 3.1.2 returning  $N$  best detections per image.
2. RANSAC: Random sampling of image pairs and in each image one of the best  $N$  detections. For each pair of the detections, the homography is estimated, Eq. (8), then the Minimal Solution solved, Eq. (8), and finally the foot-head homology estimated, Eq. (11). The Sampson distance [27] is used to indicate inliers and so

to measure the quality of the homology estimate. The procedure is iterated until no further inliers can be found.

3. The inliers are used to solve the overconstrained system in Eq. (9) via the rectangular QEP to improve the estimate of the unknown variables.
4. Steps 1-3 are repeated, utilizing the current homology estimate  $H_{FH}$ , with more densely sampled contour template models. At each image location only models with head points in some pre-defined radius from the expected ones given by the homology are evaluated.
5. As an intermediate output of pre-calibration, the camera calibration matrix  $K$ , local rotation  $R$ , translation  $t$ , and homology  $H_{FH}$  are provided.
6. Transform the detected person positions into the local camera reference systems on the ground plane using the output of pre-calibration (see Fig. 5).
7. Estimate jointly the relative camera rotations, translations and target state in a common reference system on the ground plane, Eq. (22).
8. Estimate jointly the relative camera rotations, translations and target state in full 3D, Eq. (27).
9. As an output of final calibration, camera global rotation  $R^*$ , translation  $t^*$ , and as a side effect the reconstructed 3D person positions  $X^*$  are provided.

## 4 Experimental Results

To establish the limits of the approach for camera calibration a set of experiments on both synthetic and real data is carried out.

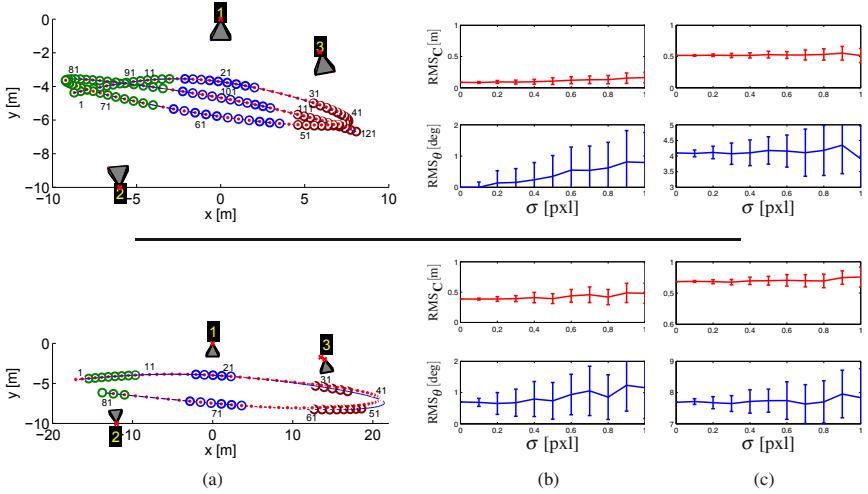
The synthetic experiments focus on evaluation of the ground-plane alignment approach for side views. We use two different sources of noise to simulate the effect in real data of noisy detections and of the imperfect estimate of the pre-calibration parameters. Moreover, we evaluate how the gap between camera views affects the reconstruction. Two different scenarios corresponding to almost overlapping and fully non-overlapping views are therefore considered.

In the real experiments we consider the full pipeline in a setup equivalent to the almost overlapping scenario of the synthetic experiment. For the pre-calibration detections of almost parallel human poses are used. The full reconstruction is done using first, highly accurate detections from a moving led-light source, second, tracking results using the TLD method in [38] manually initialized to track the head-shoulder upper body part.

### 4.1 Synthetic Data

We emulate a three camera scenario, each camera observing a part of a synthetic spline curve located in the  $z = 0$  plane. The quality of the reconstruction is strongly influenced by deviations of the trajectory from the Markov dynamics used to model it.

We evaluate the quality of calibration by ground-plane alignment in two realizations of the synthetic scenario, represented in the first column (a) of Fig. 6. In the

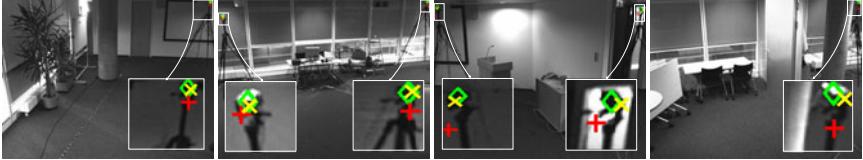


**Fig. 6** Estimating camera parameters  $[\mathbf{T}^i, \theta^i]$  and trajectory reconstruction for slightly (top) and largely (bottom) non-overlapping synthetic camera setup. (a) Top-view of the reconstructions. The true and estimated camera centers are shown by red crosses connected by red lines. The true and reconstructed trajectories are plotted by a blue line and connected red dots respectively. Circles mark the points observed by the cameras. (b) The RMS error on camera center localization and angle estimation vs. image noise level. (c) The same but with 3D noise added to the trajectory.

first case, called slightly non-overlapping, the gap between the field of view of the cameras is of one up to few meters, while in the second case distances are about 5 to 10 meters. Camera centers and rotations are known up to translation and rotation on the ground-plane  $[\mathbf{T}^i, \theta^i]$ . Fig. 6 shows qualitatively a good agreement between ground-truth and reconstruction, even though in the second example the sharp turn is reproduced with less accuracy than other parts of the trajectory because it falls outside the field of view.

We evaluate the camera localization, which is the cue we are mostly interested in surveillance applications, by computing average and standard deviation of the RMS error for different realization of the same trajectory, obtained by adding a Gaussian noise to the trajectory points observed in the camera views. This noise emulates the noisy detections in the image data. Each trajectory is sampled 50 times for each noise level with standard deviation between 0 and 1 pixel. Moreover, we compare the two cases with and without a uniform noise added to the 3D trajectory. A 3D noise of standard deviation  $\sigma = 20$  cm and 40 cm is added to the two scenarios in Fig. 6. This emulates for a real scenario the effect of inaccurate local pre-calibration and trajectory non-planarity on the ground-plane trajectory.

The mean value and standard deviation of the reconstruction of camera centers and of rotations is shown in columns (b) and (c) of Fig. 6. For almost overlapping cameras, the reconstruction of the camera centers is, even with added Gaussian noise, in the range of 10 cm. The error increases in the full non-overlapping case to about



**Fig. 7** LED-light experiment in a slightly overlapping case. ‘◊’ ground-truth epipoles, ‘+’ projected camera centers using the estimated parameters after ground-plane alignment, and ‘×’ after BA. The corresponding camera calibration and trajectory estimation is shown in Fig. II

50 cm. The estimate of the error on the angle is comparable in the two scenarios, in the range up to 2 degrees without 3D noise and up to 9 degrees with added 3D noise.

## 4.2 Real Data

The experimental setup is made up of four cameras in an indoor environment. The field of view of the cameras is slightly overlapping, so that each camera is visible at least once at the edge of the field of view of another camera. The overlap of fields of view is used in order to mark manually the epipoles which are used as ground-truth. An epipole  $e$  is related to the estimated parameters by  $e = -KRC$ , therefore it provides implicitly information about rotations and camera centers.

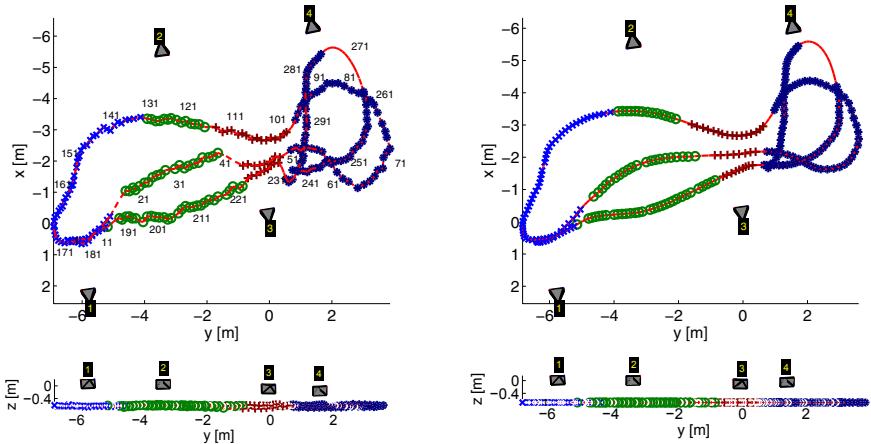
We measure with a laser based device the distance between camera centers **C** with a precision of  $\pm 10$  cm. The cameras have a resolution of  $640 \times 480$  pxl and are synchronized by an external trigger. The lens distortion is negligible as we used  $1/2''$  normal lens with  $1/3''$  cameras.

For pre-calibration, a video sequence is acquired in which a person is visible in almost parallel positions in front of each camera. The result is a local reconstruction as shown in Fig. 2. For an in-depth analysis we refer to the original paper [22].

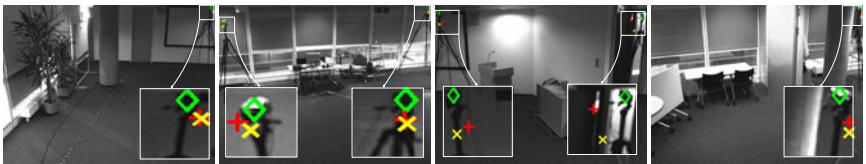
Two video sequences are acquired for global reconstruction, in which 1) a LED-light and 2) a person walking are visible, moving back and forth in the area covered by the cameras, Fig. II.

First, very accurate detections from a LED-light source are used in order test the limits of accuracy of the method, independently of the precision of the detector. However in generating the trajectory no special attention was paid in having it perfectly planar. Therefore, although the detection of the LED-light source is sub-pixel accurate, the up and down motion of the human hand holding the light partially reproduces the non planar motion of a person’s head due to gait transitions. In the LED-light experiment, few detections fall in the overlap area between the cameras.

In the tracking experiment, the trajectory is automatically generated by tracking the upper body part of a person using the TLD tracker [38]. The tracker fails in the overlapping region therefore no correspondences between detections in different cameras are available. The experiment is equivalent to the synthetic experiment in the slightly non-overlapping camera setup.



**Fig. 8** LED-light experiment in a fully non-overlapping case. Top- and side-views of camera calibrations and trajectory reconstruction before (left) and after (right) bundle adjustment.

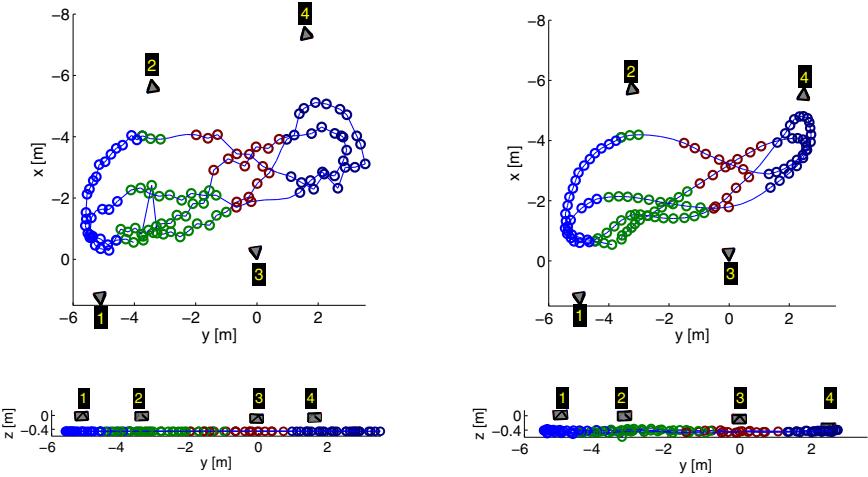


**Fig. 9** Epipole analysis for non-overlapping case from Fig. 8. See caption of Fig. 7

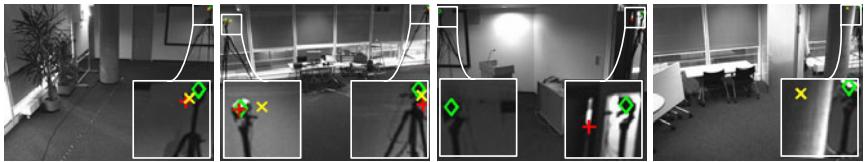
### LED-Light Data

Using the led-light data, we perform two set of experiments: in the first one we include detections in the overlap area while in the second one we eliminate them manually, in order to fall into the slightly non-overlapping scenario equivalent to the first case in Fig. 6. For the overlapping case, the full reconstruction after BA is shown in Fig. 11. The target detections are mapped by the local homographies to a noisy trajectory on the ground plane. As the ground-plane alignment enforces planarity of the reconstructed 3D trajectory, it is not able to match continuously the trajectory from one camera to the other and moreover the reconstructed trajectory is noisy. The result of the last optimization step in Fig. 11 is instead a smooth trajectory which is as expected slightly non planar. Moreover, the corrections of the BA in this scenario improves noticeably the estimate of the epipoles, as shown in Fig. 7. The average error on the estimated distance between the cameras is about 50 cm.

Reconstruction in the non-overlapping scenario is shown in Fig. 8. There is a good qualitative agreement of both experiments in terms of the reconstructed trajectory. The error on the distance between camera centers is also approximately the same. However, the epipoles are estimated with less accuracy and the BA is not able to improve the result of the ground-plane alignment, Fig. 9. The reason for



**Fig. 10** Real experiment overlapping case. Top- and side-views of camera calibrations and trajectory reconstruction before (left) and after (right) bundle adjustment.



**Fig. 11** Epipole analysis for real-data experiment case from Fig. 10. See caption of Fig. 7

that is evident: even with few points in the overlapping area, triangulation provides stronger support to the correct solution than the smoothness constraint for the non-overlapping case.

## Tracking Data

Similar results are achieved using the trajectory generated by tracking data, Fig. 10. Although there are less detections and the accuracy is lower, the reconstruction is still qualitatively good, with an average error in the estimated distance between pair of cameras of about 1 m. The estimate of the epipoles in Fig. 11, in some cases the epipoles falls outside of the field of view.

## 5 Conclusion

Calibrated camera networks are an important pre-requisite for the commercial exploitation of visual surveillance and business intelligence applications. The presented stratified auto-calibration framework is suitable for realistic scenarios with

non-overlapping cameras and data originating from observations of non-specific calibration targets such e.g. humans. We use in the pipeline a recently introduced single camera calibration in conjunction with an augmented non-linear optimization. For the method to support visual tasks, we expected an error in the calibration result not larger than the typical volume a person occupies in space which is approximately  $1\text{ m} \times 1\text{ m} \times 2\text{ m}$ . The experiments show satisfying precision within this limit of applicability. In the proposed approach the 3D reconstruction of the target's trajectory is presented almost as a side-effect. However, we would like to highlight here the relevance of this additional information for applications in event recognition and image understanding. Furthermore a topology graph as well as a spatial coverage map are easily extractable from the camera matrices to provide infrastructure providers and researchers valuable information for decisions about camera placement. Finally, the quality of the reconstruction improves with increasing constraints coming from the observed trajectories. Therefore the method is suited to be extended for online learning, as the precision of the estimate is increasing over time. Still a strong limitation is that discontinuities occurring in the non visible-gap negatively influence the reconstruction, the effect being stronger as larger the gap, however, and this is also a strength of this approach, the reconstruction degrades gracefully.

**Acknowledgement.** This research work was funded by the Wiener Wissenschafts-, Forschungs-, und Technologiefonds - WWTF, project No ICT08-030. We thank Prof. Arnold Neumaier for fruitful discussions.

## References

1. Lv, F., Zhao, T., Nevatia, R.: Self-calibration of a camera from video of a walking human. In: ICPR (2002)
2. Krahnstoever, N., Mendonca, P.R.S.: Bayesian autocalibration for surveillance. In: ICCV, pp. II:1858–II:1865 (2005)
3. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)
4. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM 24, 381–395 (1981)
5. Rahimi, A., Dunagan, B., Darrell, T.: Simultaneous calibration and tracking with a network of non-overlapping sensors. In: CVPR, pp. (I):187–(I):194 (2004)
6. McGlone (ed.): Manual of Photogrammetry. ASPRS (2004)
7. Szeliski, R.: Computer Vision: Algorithms and Applications. Springer (2009)
8. Baker, P., Aloimonos, Y.: Complete calibration of a multi-camera network. In: Proceedings of IEEE Workshop on Omnidirectional Vision, pp. 134–141 (2000)
9. Svoboda, T., Martinec, D., Pajdla, T.: A convenient multi-camera self-calibration for virtual environments. PRESENCE: Teleoperators and Virtual Environments 14(4), 407–422 (2005)
10. Lee, L., Romano, R., Stein, G.: Monitoring activities from multiple video streams: Establishing a common coordinate frame. IEEE Transaction on Pattern Analysis and Machine Intelligence (PAMI) 22(8), 758–767 (2000)

11. Stein, G.: Tracking from multiple view points: Self-calibration of space and time. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 1521–1527 (1999)
12. Jaynes, C.: Multi-view calibration from planar motion trajectories. *Image and Vision Computing* 22(7), 535–550 (2004)
13. Thaler, M., Mandrzinger, R.: Automatic inter-image homography estimation from person detections. In: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 456–461 (September 2010)
14. Kahn, S., Shah, M.: Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25(10), 1355–1360 (2003)
15. Funiak, S., Guestrin, C., Paskin, M., Sukthankar, R.: Distributed localization of networked cameras. In: Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN), Nashville, pp. 34–42 (April 2006)
16. Sheikh, Y., Li, X., Shah, M.: Trajectory association across non-overlapping moving cameras in planar scenes. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–7 (June 2007)
17. Rudoy, M., Rohrs, C.E.: Simultaneous sensor calibration and path estimation. In: Proc. IEEE Asilomar Conference on Signals, Systems, and Computers (2006)
18. Ellis, T.J., Makris, D., Black, J.K.: Learning a multi-camera topology. In: Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, pp. 165–171 (2003)
19. Tieu, K., Dalley, G., Grimson, W.E.L.: Inference of non-overlapping camera network topology by measuring statistical dependence. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1842–1849 (2005)
20. van den Hengel, A., Dick, A., Hill, R.: Activity topology estimation for large networks of cameras. In: IEEE International Conference on Video and Signal Based Surveillance, AVSS 2006, p. 44 (November 2006)
21. Devarajan, D., Cheng, Z., Radke, R.: Calibrating distributed camera networks. Proceedings of the IEEE 96(10), 1625–1639 (2008)
22. Micusik, B., Pajdla, T.: Simultaneous surveillance camera calibration and foot-head homology estimation from human detections. In: CVPR (2010)
23. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle Adjustment – A Modern Synthesis (chapter Bundle adjustment - a modern synthesis). In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, pp. 298–375. Springer, Heidelberg (2000)
24. Beleznai, C., Bischof, H.: Fast human detection in crowded scenes by contour integration and local shape estimation. In: CVPR (2009)
25. Liebelt, J., Schmid, C., Schertler, K.: Viewpoint-independent object class detection using 3D feature maps. In: CVPR (2008)
26. Toshev, A., Makadia, A., Daniilidis, K.: Shape-based object recognition in videos using 3D synthetic object models. In: CVPR (2009)
27. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
28. Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., van der Vorst, H. (eds.): Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide. SIAM (2000)
29. Grammont, L., Higham, N.J., Tisseur, F.: A framework for analyzing nonlinear eigenproblems and parametrized linear systems. *Linear Algebra and its Applications* (2010)
30. Fitzgibbon, A.W.: Simultaneous linear estimation of multiple view geometry and lens distortion. In: CVPR, pp. (I):125–(I):132 (2001)

31. Micusik, B., Pajdla, T.: Structure from motion with wide circular field of view cameras. *PAMI* 28(7) (2006)
32. Kukelova, Z., Bujnak, M., Pajdla, T.: Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems. In: *BMVC* (2008)
33. Bujnak, M., Kukelova, Z., Pajdla, T.: 3D reconstruction from image collections with a single known focal length. In: *ICCV* (2009)
34. Micusik, B.: Relative pose problem for non-overlapping surveillance cameras with known gravity vector. In: *CVPR* (2011)
35. Boutry, G., Elad, M., Golub, G.H., Milanfar, P., Milanfar, G.H.G.P.: The generalized eigenvalue problem for non-square pencils using a minimal perturbation approach. *SIAM J. Matrix Anal. Appl.* 27, 582–601 (2005)
36. Pflugfelder, R., Bischof, H.: Localization and trajectory reconstruction in surveillance cameras with non-overlapping views. *PAMI* 32(4), 709–721 (2009)
37. Lourakis, M.I.A., Argyros, A.A.: SBA: A Software Package for Generic Sparse Bundle Adjustment. *ACM Trans. Math. Software* 36(1), 1–30 (2009)
38. Kalal, Z., Matas, J., Mikolajczyk, K.: P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. In: *CVPR* (2010)

# Fast Approximate Nearest Neighbor Methods for Example-Based Video Search

Rizwan Chaudhry and Yuri Ivanov

**Abstract.** The cost of computer storage is steadily decreasing. Many terabytes of video data can be easily collected using video cameras in public places for modern surveillance applications, or stored on video sharing websites. However, the growth in CPU speeds has recently slowed to a crawl. This situation implies that while the data is being collected, it cannot be cheaply processed in time. Searching such vast collections of video data for useful information requires radically different approaches, calling for algorithms with sub-linear time complexity.

One application of a search in a large data set is query-by-example. A video clip is used as a query for an algorithm to find a set of similar clips from the collection. A naive solution to such problem would specify some sort of a similarity metric and exhaustively compute this similarity between the query and all other video clips in the collection. Then the clips with the highest similarity values can be returned as the answer-set. However, as the number of the videos in the collection grows, such computation becomes prohibitively expensive. In order to show sub-linear growth any large-scale algorithm needs to exploit some properties of the data that does away with the need to compute explicit distances between a query point and any other point in the set. To this end, Approximate Nearest Neighbor methods have recently become popular. These algorithms provide a trade-off between the accuracy of finding nearest-neighbors and the corresponding computational complexity. As a result, searches in very large datasets can be performed very quickly albeit at the cost of very few incorrect matches.

Most of the recent work in developing ANN methods has been done for data points that lie in a Euclidean space. However, several applications in computer

---

Rizwan Chaudhry  
Center for Imaging Science,  
Johns Hopkins University, Baltimore, MD  
e-mail: [rizwanch@cis.jhu.edu](mailto:rizwanch@cis.jhu.edu)

Yuri Ivanov  
Mitsubishi Electric Research Laboratories, Cambridge, MA  
e-mail: [yivanov@merl.com](mailto:yivanov@merl.com)

vision such as object and human activity recognition use non-Euclidean data. State-of-the-art Euclidean ANN methods do not perform well when applied to these datasets. In this chapter, we present algorithms for performing ANN on manifolds 1) by explicitly considering the Riemannian geometry of the non-Euclidean manifold and 2) by taking advantage of the kernel trick in non-Euclidean spaces where performing Riemannian computations is expensive. For a data set with  $N$  samples, the proposed methods are able to retrieve similar objects in as low as  $O(K)$  time complexity, where  $K \ll N$ . We test and evaluate our methods on both synthetic and real datasets and get better than state-of-the-art results.

## 1 Introduction

Video cameras have become almost ubiquitous in public places as well as in retail and business environments. Even though the primary application of these cameras still remains in security and surveillance, researchers and businesses alike are starting to look at methods for automatically mining video data to collect business intelligence. Analytics such as automatic human behavior analysis, event classification, and abnormal event detection from long periods - e.g. months or years - of recorded video would be very important for businesses and organizations to gain more knowledge about customer behaviour and how they should improve their business for maximum profit or maximum efficiency. For example, a department store would be interested in finding all the occasions in recorded video from the last several years, when customers exhibited certain behaviors such as running, or stealing merchandise etc. Airport officials would be interested in finding and analyzing all the occasions when a passenger gave a bag to another, or find all instances of abnormal behavior such as the breaking out of fights. Traffic officials would be interested in finding all times of the day from the last few years of data, when a certain highway had heavy traffic. Enabling applications based on video search would allow policy makers and business planners to make better decisions based on statistical analysis of events and behaviors across large time frames.

Presently, in most cases, video data is recorded continuously and stored for later access. Humans have to look at multiple screens, displaying videos from several cameras, either to monitor in real time, or to sift through archives of videos to find the exact place and time when an event of interest occurred. This is a very time-consuming process and causes delays in responding to sensitive scenarios such as robberies or accidents in a timely manner. These delays could potentially cause businesses huge losses. Automating processes that rely on looking at videos for long hours to find significant events or human actions is of utmost importance for businesses to cut costs and become more efficient.

In most applications for collecting business intelligence, the goal is to search for events or activities that are similar to a particular event or activity of interest from large collections of previously recorded video. Therefore it is extremely important to develop very efficient methods for automatic semantic retrieval of videos from large databases or archives. As the amount of video data increases, a simple linear search

through long hours of video, even if done automatically, will become prohibitively expensive. Hence there is a need to develop sub-linear methods for video retrieval. This chapter aims to address the problem of fast example-based video search which would not only benefit the above video-based business intelligence applications but in general any semantic retrieval application from large video databases.

## 1.1 Technical Approach

Real-life data is usually observed from a continuum. This implies that observations that are *similar* to each other can be mathematically represented by points that lie close to each other in some, possibly non-linear, coordinate system. One of the simplest methods, therefore, for finding previously observed data that is similar to a given test observation is by using the notion of a *distance* defined on that coordinate system and searching for points that are close to the test point under this distance. This is referred to as Nearest Neighbor matching. Nearest Neighbor matching is especially useful in classification tasks, where we want to determine the class label of a test data point by using the majority label of its  $k$  nearest neighbors. Because of its simplicity,  $k$ -Nearest-Neighbor ( $k$ -NN) is used in almost all classification tasks as a baseline algorithm. Moreover, there is often a trade-off between the complexity of data representation and the complexity of the classifier. For example, we could use a simple Euclidean data representation but choose a non-linear classifier such as kernel SVM. On the other hand, we could choose a complex data representation involving a non-Euclidean manifold, define a non-linear metric between data points, and choose  $k$ -NN as a simple classifier.

Despite its simplicity,  $k$ -NN has been successfully used across many scientific disciplines. It gives extremely good results when the number of training data points is large and densely sampled, and especially when the decision boundary between classes is very irregular. Moreover, it has also been shown in [20] that asymptotically, the error of a 1-NN classifier is always smaller than twice the Bayes error which guarantees the optimality and consistency of the  $k$ -NN classifier under certain assumptions. Against all these advantages, the biggest drawback when using a  $k$ -NN classifier is its computational complexity -  $O(N)$ , where  $N$  is the size of the training set. This linear relationship implies that as the size of the training set gets larger such as in the millions, the cost of computing the distance of a test point with all training data points becomes prohibitively large.

To address the computational complexity of the  $k$ -NN algorithm, several efficient strategies have been proposed in literature. These strategies pre-process the training data and extract salient properties that can be used to get sub-linear algorithms for nearest neighbor matching. Generally, these algorithms fall into two categories: 1) exact methods and 2) approximate methods. Exact methods generally use tree-based algorithms and clustering approaches to effectively reduce the complexity to  $O(\log N)$ , and are guaranteed to provide the correct nearest neighbors in all cases. On the other hand, approximate methods provide even greater efficiency, such as

$O(\log N)$  with a smaller constant factor or even  $O(1)$  as in Hashing based methods, but at the cost of slightly decreased accuracy.

Among tree based methods for nearest-neighbor matching, one of the most popular algorithms is the kd-tree algorithm by Friedman *et al.* [17]. This algorithm divides the training data into equal parts along all dimensions to build a tree structure. However as the dimensionality of the data increases, the number of sub-trees to be traversed for exact matching increases and the method becomes computationally infeasible compared to linear search. Several approximate versions [2], [4], [35] attempt to increase the efficiency of this algorithm at the cost of decreased accuracy. Another class of algorithms, [18], [28], constructs a tree by recursively clustering the data using the k-means algorithm to get exact or approximate sub-linear algorithms for nearest neighbor matching. The study in [27] compares these and several other tree-based algorithms and develops a scheme to automatically select the best algorithm given a dataset and design requirements such as trade-offs between the speed and accuracy of the method and memory requirements.

Traditional hashing methods are very successful at returning exact matches to a query if exactly the same point as the query exists in the database. A hash function is used to generate a binary key corresponding to each data-point and the key is used to address the location in memory to store the data-point. Hence, to check whether a query data point exists in the database, we only need to compute its key and directly refer to the corresponding memory location. Traditional hash functions, however do not generate keys that preserve the metric structure of the space of data points. Essentially, nearest neighbors in the space of data-points, do not get mapped as nearest neighbors in the corresponding space of binary hash keys. This is undesirable as in many machine learning applications, we are not interested in finding the exact match to a query. Instead, we are interested in finding objects in the database that are *similar* to the query object.

Recently, new hashing algorithms have been developed that preserve neighborhood relationships between derived codes. Approximate Nearest-Neighbor methods such as the variants of Locality Sensitive Hashing (LSH), [1], Semantic Hashing, [30], and Spectral Hashing, [39], provide efficient algorithms for constructing binary codes for points in a high dimensional space. These methods have the property that codes for points that are nearby in the high-dimensional space are also close to each other in the binary code space under the Hamming distance. The Hamming distance between two bit-strings is defined as the total number of mismatched bits. For example, the Hamming distance between 01100 and 01010 is 2. Hashing-based ANN methods provide an excellent method for creating hash tables because even if the key for a query object is not in the table, the keys for similar objects can be found by simply flipping a few bits of the binary code. As these *neighboring* keys will correspond to objects similar to the query object, they can be easily retrieved from the database by looking them up in the hash table.

## 1.2 Nearest-Neighbor Methods in Video Retrieval

Nearest Neighbor methods have received wide application in content based multimedia retrieval such as finding similar images or videos from huge corpus of the same. There has been great interest in fast content-based image retrieval for the past many years and research has increasingly been shifting toward web-scale databases of tens of millions of images. However there has been little work in the same direction for videos. A summary of notable work dominant in the field follows.

Karpenko *et al.* in [21] introduced a method where all the videos in a dataset were compressed to very small frame sizes and only a few key-frames. Using intensity statistics in the frames, the comparison of the new query is performed with the entire dataset. This technique leads to faster video comparison, but does not use semantically meaningful features and cannot be performed faster than  $O(N)$ . An important application of semantic video retrieval is searching for specific human activities in a large collection of videos. Human action analysis is considered one of the most important problems in computer vision. It enables such applications as automatic surveillance, behavior analysis, elderly care, etc. There has been a tremendous amount of work towards automatic analysis of human motion in videos. However, due to extensive amount of computation required for video analysis, this work by necessity is often restricted to smaller models and datasets. As mentioned in the introduction, in a real-life surveillance scenario video data is continuously recorded for a long period of time and saved for later analysis. Search in such extensive volumes of data remains a difficult task. Biswas *et al.* [9] provided a method that used two-level hash tables based on the invariant geometric properties of object shapes for efficient search and retrieval. Turaga *et al.* [38] proposed a dynamical-systems based model for human activities that can be used for clustering different types of activities in a continuous video. Sidenbladh *et al.* [34] used an approximate probabilistic tree search to find the closest match in the database for a query motion. Ben-Arie *et al.* in [5] used a sparsely sampled sequence of body poses and velocity vectors of body parts as humans move in a scene to construct multi-dimensional hash tables. For a test video, these features were extracted and the key was used to find the match in the hash-tables. Several other methods proposed in [13, 14, 22] cluster features derived from motion and appearance information for semantic retrieval. All of these methods either use exact-match hashing, which generally has difficulties in performing a neighborhood search, or tree-based approaches, which often help increase performance, but are not as fast as hashing techniques.

An important aspect to consider about algorithms for finding nearest neighbors is the metric structure of the space of data-points. Most state-of-the-art algorithms assume that the L-2 Euclidean distance,  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$ , is used to compute distances between two data points. However, in many practical applications, the data lies on a non-Euclidean manifold and Euclidean distance might not provide the best (dis)similarity measure, leading to sub-par algorithms. For instance, histograms created as part of a bags of video words classification procedure on local features proposed by Laptev, [25] and Dollar, [16], or dynamical systems proposed

in [7, 8, 3, 12], naturally lie on a non-trivial manifold that has strong non-Euclidean properties. Hence the above methods are not directly applicable.

In this chapter, we will address the above open issues by proposing algorithms for performing Approximate Nearest Neighbors matching on Non-Euclidean manifolds. In Section 2, we will briefly outline the Spectral Hashing algorithm and address its limitations when data lies on non-Euclidean manifolds. We will also look at some recent attempts at addressing the problem of performing ANN matching on non-Euclidean manifolds. In Section 3, we will first introduce some key concepts from Riemannian geometry before explaining our proposed methods. In Section 4, we will extensively evaluate our algorithm on both synthetic datasets as well as on two real video databases.

## 2 Background

In this section, we will describe the Spectral Hashing algorithm by [39] for Euclidean data and its variants for non-Euclidean data. We will also describe the strengths and weaknesses of both to motivate our algorithms for ANN on non-Euclidean manifolds.

### 2.1 Spectral Hashing

As presented by Weiss *et al.* in [39], given data points,  $\{\mathbf{x}_i\}_{i=1}^N \in \mathbb{R}^d$ , the goal is to find  $k$ -bit binary vectors,  $\{y_i\}_{i=1}^N \in \{-1, 1\}^k$  such that similar points in  $\mathbb{R}^N$ , under the similarity measure,  $W_{ij} = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\varepsilon^2})$  map to binary vectors that are close to each other under the Hamming distance weighted by  $W$ . Thus spectral hashing aims to minimize the objective function:

$$\begin{aligned} \text{minimize} \quad & \sum_{ij} W_{ij} \|y_i - y_j\|^2 \\ \text{s.t.} \quad & y_i \in \{-1, 1\}^k \\ & \sum_i y_i = 0 \\ & \frac{1}{N} \sum_i y_i y_i^\top = I. \end{aligned} \tag{1}$$

The intuition behind the above constraints is to get codes that are unbiased, *i.e.* -1 and 1 appearing an equal number of times on average, and independent w.r.t. their location in the bit-string. Since this is an NP-hard problem [39], a spectral relaxation is used by constructing matrices  $Y = [y_1^\top, y_2^\top, \dots, y_N^\top]^\top \in \{-1, 1\}^{N \times k}$  and  $D$  such that  $D_{ii} = \sum_j W_{ij}$ . The resulting problem is thus

$$\begin{aligned}
& \text{minimize} && \text{trace}(Y^\top (D - W) Y) \\
& \text{s.t.} && Y_{ij} \in \{-1, 1\} \\
& && Y^\top \mathbf{1} = 0 \\
& && YY^\top = I.
\end{aligned} \tag{2}$$

Relaxing the first constraint and allowing  $Y_{ij} \in \mathbb{R}$  results in the solution as the  $k$  eigenvectors of the Laplacian,  $D - W$ , corresponding to the  $k$  smallest eigenvalues excluding 0. This relaxation however, is only applicable to data in the training set. To compute codes for new data points, using Nyström's method [6], requires time linear in the size of the training set. If we assume that the data,  $\mathbf{x}_i \in \mathbb{R}^d$ , is sampled from a probability distribution  $p(\mathbf{x})$ , Spectral Hashing (SH) solves the following optimization problem:

$$\begin{aligned}
& \text{minimize} && \int \|y(\mathbf{x}_1) - y(\mathbf{x}_2)\|^2 W(\mathbf{x}_1, \mathbf{x}_2) p(\mathbf{x}_1) p(\mathbf{x}_2) d\mathbf{x}_1 d\mathbf{x}_2 \\
& \text{s.t.} && y(\mathbf{x}) \in \{-1, 1\}^k, \quad \int y(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = 0, \quad \text{and} \quad \int y(\mathbf{x}) y(\mathbf{x})^\top p(\mathbf{x}) d\mathbf{x} = I
\end{aligned} \tag{3}$$

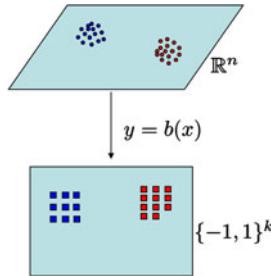
Relaxing the first constraint gives the solution of the problem,  $y$  as the first  $k$  non-zero *eigenfunctions* of the weighted Laplace-Beltrami operator on the manifold. If  $p$  is the separable multi-dimensional uniform distribution on a subset of  $\mathbb{R}^d$  and the weighting function,  $W$ , is defined as above, there exists a one-shot closed form solution for these eigenfunctions as an outer product of all the 1-dimensional eigenfunctions. Specifically, for the uniform distribution over the interval  $[a, b]$ , the eigenfunctions  $\Phi_k(x)$  and eigenvalues  $\lambda_k$  are [39]:

$$\begin{aligned}
\Phi_k(x) &= \sin\left(\frac{\pi}{2} + \frac{k\pi}{b-a}x\right) \\
\lambda_k &= 1 - \exp\left(-\frac{\varepsilon^2}{2} \left|\frac{k\pi}{b-a}\right|\right)
\end{aligned}$$

Therefore, the  $k$ -bit code for the  $d$ -dimensional vector,  $[x_1, x_2, \dots, x_d]^\top$  is found by thresholding to zero, the outer-product vector,

$$\begin{bmatrix} \Phi_1^1(x_1) \Phi_1^2(x_2) \dots \Phi_1^d(x_d) \\ \Phi_2^1(x_1) \Phi_2^2(x_2) \dots \Phi_2^d(x_d) \\ \vdots \\ \Phi_k^1(x_1) \Phi_k^2(x_2) \dots \Phi_k^d(x_d) \end{bmatrix}, \tag{4}$$

where  $\Phi_i^j$  denotes the  $i$ -th eigenfunction computed by assuming a 1-D uniform distribution on  $[a_j, b_j]$  corresponding to the  $j$ -th dimension. The spectral hashing algorithm, as in [39], is summarized in Algorithm 1.



**Fig. 1** Spectral Hashing (SH)[39]

**Alg. 1: Spectral Hashing [39]**

1. Compute principal components of data using PCA.
2. Compute the  $k$  smallest single-dimension analytical eigenfunctions of the Laplace-Beltrami operator under the specified weighting function and probability distribution by using a rectangular approximation along every PCA direction.
3. Threshold the analytical eigenfunctions computed for each data-point at zero, to obtain binary codes.

## 2.2 Addressing Non-euclidean Manifolds

Spectral hashing has a very appealing mathematical formulation. Ideally, one could take any probability distribution on a general manifold and a weighting function and analytically compute the eigenfunctions of the corresponding Laplace-Beltrami operator. However, even in the simple case of Euclidean data that follows a Gaussian distribution, there does not exist a closed form solution and we have to resort to iterative numerical techniques. Furthermore, the weighting function,  $W$ , is computed from geodesic distances and thus, is no longer a simple exponential similarity. This makes the exact computation of the solution of the minimization problem in Eq. (3) computationally intractable.

As described in Alg. 1, the method in [39] employs PCA to compute a basis for the dataset, and then computes closed-form one-dimensional eigenfunctions of the weighted Laplace-Beltrami operator in each principal component direction according to the spread of the data in that direction. This will only work if the spread of the data is approximated to be uniform when computing the closed form eigenfunctions. However for general manifolds, the spread of the data in a direction need not follow any such assumption and therefore the original approach in [39] would in general perform poorly. The authors in [39] mentioned this limitation of Spectral Hashing and assumed that a suitable Euclidean embedding can be used. However, finding such an embedding is not always possible.

A workaround originally proposed by Kulis and Grauman [24] for LSH uses the *kernel trick* to implicitly embed the data in a high-dimensional Euclidean space. Kulis and Darrell in [23] further used a similar kernel trick for Spectral Hashing by using Kernel PCA instead of PCA in step 1 of Alg. 1. We will refer to this method as *Kernel Spectral Hashing* (KSH). Even though their method is theoretically correct, as the kernels would embed the points in a high-dimensional Euclidean space, finding the value of the eigenfunction at each new test data-point would involve computing the kernel of the test point with all the points in the training set used to compute the kernel PCA components. Because of this, even though a well-chosen kernel might give good retrieval accuracy, the computational complexity of this method is at least  $O(N)$  and no better than performing exact nearest neighbor search.

Recently Turaga *et al.* in [37] proposed a method similar to our original approach in [11] that explicitly considers the Riemannian geometry of the manifold to construct hash codes by performing LSH in the tangent space of the manifold. As shown in [39], to perform accurately, LSH usually requires very large codewords, whereas Semantic Hashing and Spectral Hashing give compact binary codewords and therefore are more useful for mapping objects directly to memory addresses in a computer. That is why we specifically consider Spectral Hashing and propose two new fast approximate methods for performing Spectral Hashing on non-Euclidean data.

### 3 Non-euclidean Spectral Hashing

Noting the computational difficulty with directly applying Spectral Hashing techniques to non-Euclidean manifolds, we propose two new methods for finding compact binary codes for data lying on manifolds with which this difficulty can be circumvented. Before presenting our methods, we will briefly introduce some concepts from Riemannian geometry that will be utilized in our formulation.

#### 3.1 Basic Concepts from Riemannian Geometry

A manifold  $\mathcal{M}$ , is a topological space with the characteristic that it is locally Euclidean, *i.e.* for each point  $p \in \mathcal{M}$ , on the manifold, a diffeomorphic mapping  $\phi : U \rightarrow \mathbb{R}^n$  maps each point in an open neighborhood  $U$  around  $p$  to the Euclidean space,  $\mathbb{R}^n$ . The pair  $(U, \phi)$  is also called the (local) chart of  $U$  and  $n$  is the dimension of the manifold. Given a curve,  $\gamma(t)$  lying completely on the manifold and passing through the point  $p$  such that  $\gamma(0) = p$ , the derivative,  $\dot{\gamma}(0)$  is tangent to the manifold at the point  $p$ . The set of all tangents to all differentiable curves along the manifold through  $p$  denotes the tangent space to  $\mathcal{M}$  at  $p$ , also represented as  $T_p\mathcal{M}$ . The tangent space at a point on the manifold is a Euclidean space and has the same dimension as the manifold.

A manifold is called Riemannian if, besides being real and differentiable, it also has a notion of inner product,  $g(.,.)$ , associated to it. Given two points  $p$  and  $q$  on the manifold, the *geodesic distance* between the points is defined as the infimum of

the length of all possible differential curves along the manifold that pass through both points. More formally,

$$d(p, q) = \inf_{\{\gamma: [0, 1] \rightarrow \mathcal{M} | \gamma(0) = p, \gamma(1) = q\}} L(\gamma), \text{ where} \quad (5)$$

$$L(\gamma) = \int_0^1 (g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t)))^{\frac{1}{2}} dt. \quad (6)$$

Computing geodesics therefore in general requires variational calculus and finding analytic solutions might not be possible for arbitrary manifolds.

The *exponential map*,  $\exp_p: T_p \mathcal{M} \rightarrow \mathcal{M}$  is a mapping from the tangent space at the point  $p$  on the manifold to the manifold. It is defined as,  $\exp_p(v) = \gamma_v(1)$  where  $\gamma_v$  is a geodesic in the direction of the tangent  $v$  at point  $p$ . The *logarithm map*,  $\log_p: \mathcal{M} \rightarrow T_p \mathcal{M}$  is the inverse of the exponential map and maps points from the manifold to the tangent space at the point  $p$ , also called the *pole*. As in the case of geodesics, computing exponential and logarithm maps requires variational calculus and closed form analytic solutions exist for only a few special manifolds.

### 3.2 Riemannian Spectral Hashing

Since it is hard to compute closed form eigenfunctions in the SH algorithm for non-Euclidean data, we can embed the data in a Euclidean space. Then, under the assumption that it is drawn from a uniform distribution in that space, spectral hashing can be applied in this embedding space. Our first method, *Riemannian Spectral Hashing (RSH)*, follows this strategy.

The tangent space,  $T_y \mathcal{M}$ , to a manifold,  $\mathcal{M}$  at a point  $y$  is a Euclidean space. Therefore, assuming that the manifold is geodesically complete<sup>1</sup>, the data,  $\{\mathbf{x}_i\}_{i=1}^N$ , can be projected onto the tangent space at  $y$  by using the *logarithm map*,  $\Delta_i = \overrightarrow{y\mathbf{x}_i} = \log_y(\mathbf{x}_i)$ . This makes it possible to perform Spectral Hashing on the tangent space projections,  $\{\Delta_i\}_{i=1}^N$  locally, around  $y$  acting as the pole. However, the projection error tends to accumulate as points move further away from the pole. In order to minimize projection errors the RSH algorithm approximates a manifold with a *set of tangent hyperplanes*, positioned on a set of representative points (poles) which follow the distribution of the data on the manifold. The poles are found by clustering, for which we can use any extrinsic manifold clustering algorithm such as [36], [19] to cluster the data on the manifold into  $K$  clusters. We use the Riemannian  $k$ -means procedure:

1. Initialize cluster centers,  $\{\mathbf{c}_j\}_{j=1}^K$  by randomly choosing  $K$  points from the training data.
2. For each point  $\mathbf{x}_i$  in the data set, compute the geodesic distance to each cluster center,  $d(\mathbf{c}_j, \mathbf{x}_i) = \|\log_{\mathbf{c}_j}(\mathbf{x}_i)\|$ . Assign the cluster center that is the closest to the data point as the cluster membership,  $w_i = \operatorname{argmin}_j \|\log_{\mathbf{c}_j}(\mathbf{x}_i)\|$ .

---

<sup>1</sup> A geodesically complete manifold is one for which every maximal geodesic is defined on  $\mathbb{R}$ . Informally, there are no holes in the manifold.

3. Recompute each cluster center as the Karcher mean of the points in each cluster,  $\mathbf{c}_j = \text{mean}\{\mathbf{x}_l | w_l = j\}$ . The Karcher mean of a set of points in a manifold is found by projecting all the points on the tangent space at a randomly chosen point, finding the mean and projecting it back to the manifold and repeating this step around the mean found in the previous iteration until the final mean converges. This requires repeated uses of the *exponential map* and the *logarithm map* on the manifold until convergence to a mean.
4. Repeat until convergence.

This clustering algorithm is a simple extension of the k-means algorithm to a single geodesically complete manifold under the assumption that no two points in the same cluster are antipodes, essentially meaning that all the points can be covered by the same coordinate chart. This method inherits the convergence properties of regular Euclidean k-means. Once the clusters,  $\{\mathbf{c}_j\}_{j=1}^K$ , and memberships,  $\{w_i\}_{i=1}^N$ , have been assigned, all the points in the same cluster are projected to the tangent space around the cluster center using the corresponding logarithm maps. A separate spectral hashing algorithm is then trained on each tangent space.

For computing the binary code of a new test point,  $\mathbf{z}$ , we first compute the geodesic distance of  $\mathbf{z}$  with all the cluster centers and project it to the tangent space of the closest cluster center,  $\mathbf{c}_k$ , where  $k = \arg \min_j \|\mathbf{c}_j \vec{\mathbf{z}}\|$  to get  $\Delta_{\mathbf{z}} = \log_{\mathbf{c}_k}(\mathbf{z})$ . We then use spectral hashing to find the binary code of  $\Delta_{\mathbf{z}}$ . Since finding the right cluster center, only requires  $K$  geodesic distance evaluations, this results in a computational cost of  $O(K)$ . Even though this is greater than  $O(1)$  as in Spectral Hashing, it is much less than  $O(N)$  as in Kernel Spectral Hashing, where  $K \ll N$ . Moreover, by clustering all the data, we better approximate the uniform distribution assumption in each cluster. We summarize RSH in Algorithm 2. Figure 2 provides an illustration of the method.

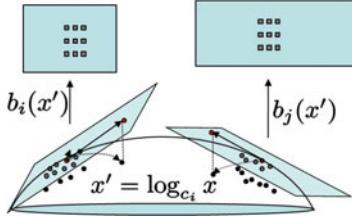
### Alg. 2: Riemannian Spectral Hashing

#### Training

1. Cluster training data using a manifold clustering algorithm such as Riemannian  $k$ -means or the methods in [36], [19].
2. Compute log-maps and project each cluster to the tangent space around center.
3. Train spectral hashing on points in each tangent space separately.

#### Testing

1. Find closest cluster center using geodesic distances on the manifold.
2. Project onto tangent space around closest cluster center.
3. Compute binary code of the projected point using Spectral Hashing.
4. Retrieve the nearest neighbor.



**Fig. 2** Riemannian Spectral Hashing (RSRH)

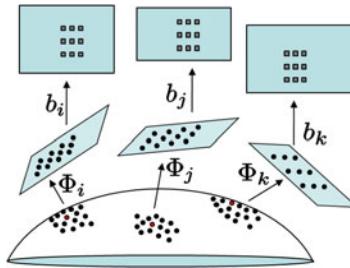
### 3.3 Distributed Kernel Spectral Hashing (DKSH)

In certain cases, closed form expressions for the logarithm and exponential maps for manifolds are not available. As iterative procedures will need to be used at each step when computing these maps, this limits the applicability of extrinsic manifold clustering algorithms as required in Alg. 2. If however, a kernel or other affinity measure,  $W(.,.)$  is defined on the manifold, a non-linear dimensionality reduction method such as Multidimensional Scaling (MDS) [20] can be employed to project the data into a low-dimensional Euclidean space before performing  $k$ -means on this low-dimensional space. Alternatively, a non-linear clustering algorithm such as kernel  $k$ -means [31] or Spectral Clustering [33] can be used to compute cluster associations of the data. As a result, we would not have cluster centers but only cluster associations for the training data. After the clustering stage, one representative point is chosen in each cluster to represent all data within it. One method to choose this point is as follows [29]:

1. Compute the  $N \times N$  affinity matrix,  $W$ , of the training data based on a kernel or affinity defined on the manifold.
2. Perform MDS using  $W$  to get a low-dimensional Euclidean representation  $\{\mathbf{u}_i\}_{i=1}^N$  and perform  $k$ -means on these points to get  $K$  cluster centers  $\{\mathbf{v}_j\}_{j=1}^K$  in the low-dimensional space.
3. Within each cluster center, choose the point  $\mathbf{u}$  in the projected data that is closest to each cluster center  $\mathbf{v}_j$ .
4. Find the original points  $\{\mathbf{x}_{p;j}\}_{j=1}^K$  on the manifold that mapped to the points  $\{\mathbf{v}_j\}_{j=1}^K$  after MDS and use these points as cluster representatives (pivots).

Once a representative, or pivot, for each cluster has been computed, we train Kernel Spectral Hashing (KSH) separately for each cluster.

As in RSH, to find the binary code for a test point,  $\mathbf{z}$ , we first compute its affinity,  $W(\mathbf{x}_{p;j}, \mathbf{z})$  with each pivot point and assign  $\mathbf{z}$  to the  $j$ -th cluster if  $\mathbf{x}_{p;j}$  has the highest affinity with  $\mathbf{z}$ . We then use Kernel Spectral Hashing trained for that specific cluster and compute the binary code for  $\mathbf{z}$  to retrieve the nearest neighbors. Assuming that in the best case, all the points are equally divided between  $K$  clusters, the query time complexity of this method is  $O(K + N/K)$  on average, which is more computationally expensive than RSH. However, it is still significantly better than the complexity of KSH. In the worst case when only one cluster is chosen,



**Fig. 3** Distributed Kernel Spectral Hashing (DKSH)

the complexity is  $O(N)$ , the same as KSH. We summarize DKSH in Algorithm 3. Figure 3 provides an illustration of the method.

### Alg. 3: Distributed Kernel Spectral Hashing

#### Training

1. Cluster training data using Non-Linear clustering (MDS, Spectral clustering etc.) using kernel similarity
2. Pick a pivot point, representing each cluster.
3. Train Kernel Spectral Hashing on points in each cluster separately.

#### Testing

1. Use kernel similarity to compute pivot with the highest affinity to test point.
2. Compute binary code with kernel spectral hashing for that pivot.
3. Retrieve the nearest neighbor.

### 3.4 Parameter Selection

By optimizing Eq. (3), Euclidean Spectral Hashing provides the optimal unbiased and uncorrelated  $k$ -bit codes for a given dataset assuming an L-2 exponential similarity and a separable  $d$ -dimensional uniform distribution. As the value of the objective function in Eq. (3) increases by increasing the number of bits, the Hamming distances between the codes for points in the dataset tends to increase. This can be seen from the fact that a 20% bit mismatch between two codes translates to a  $0.2k$  Hamming distance. Hence, when retrieving neighbors by finding points within a Hamming distance of 2, we can find many neighbors when using 10 bit code-words and very few when using 100 bits. This also relates the precision of the method with the recall. When using a fixed Hamming distance for retrieval, a small number of bits gives high recall but low precision and a large number of bits gives low recall and high precision. Therefore, when retrieving nearest neighbors irrespective of a cap on the Hamming distance, the accuracy of the method goes up as the number of bits increases.

The above discussion translates directly to our proposed methods. The codes computed for each cluster in RSH and DKSH are optimal in the same sense as for SH and the relationship between accuracy and number of bits follows as well. However, the number of clusters is an additional parameter in our framework. A large number of clusters will decrease projection errors in the manifold-to-tangent space mappings and lead to more accurate codes. However having more than  $\log N$  clusters increases the complexity of the method. Moreover, a large number of clusters will segment the data into too many disjoint regions. This has the undesirable affect of removing part of the true nearest neighbors of boundary points that fall in a different cluster. With the current approach, the only way to recover these points is to run our algorithms for all cluster centers/pivots. A soft-weighting scheme could be adopted to address some of these problems. In this chapter we will use our original clustering approaches that divide the data into a small number of clusters as this already gives better than state-of-the-art results. We will also see later that using a small number of clusters alongside a small number of bits requires much less memory resources as compared to using a large number of bits with just one cluster. In summary, there exists the traditional trade-off between precision and recall based on the number of bits and the additional trade-off between memory usage and accuracy based on the number of clusters.

## 4 Experiments

In this section we compare the proposed methods, Riemannian Spectral Hashing (RSH) and Distributed Kernel Spectral Hashing (DKSH), against exact Nearest Neighbors (NN), and state-of-the-art Hashing methods: Kernel Locality Sensitive Hashing (KLSH) [24], Euclidean Spectral Hashing [39] (SH), and Kernel Spectral Hashing [23] (KSH).

### 4.1 Experiments on Synthetic Data

To compare the performance of our Approximate Nearest-Neighbor algorithms, we will perform two different types of tests. First, we will use our algorithms in a classification framework where the approximate nearest neighbors are used to find the class label of test data. Second, we will compare the ratio of true nearest neighbors to all the ANNs provided by each algorithm to compare their precision using the Hamming distance and  $k$ -NN as the criteria.

We will test the proposed methods on synthetic datasets of points lying on two non-Euclidean manifolds: the 100 dimensional unit hypersphere,  $S^{99}$ , and the manifold of all 3-dimensional subspaces of  $\mathbb{R}^{10}$ , i.e., the Grassmann manifold,  $G_{10,3}$  or  $G_{3,10-3}$ . The evaluation is performed on an 8-core Intel Xeon 3.4 GHz machine with 32 GB of RAM. In each experiment, we restrict the number of processing cores to exactly one so that the run-times of various algorithms are comparable. When comparing our methods with Spectral Hashing, we treat the points on both the above mentioned manifolds as points in  $\mathbb{R}^{100}$  and  $\mathbb{R}^{30}$  respectively.

As a technical detail, it is noted that when the data size grows larger than  $10^4$  samples, the memory requirements of the PCA computation in SH and the kernel PCA in KSH become extremely large and can not be handled by our computational resources. As an example, consider computing the all pair kernel matrix for  $10^5$  points. Storing the result as a double precision matrix in memory requires a minimum of  $(10^5)^2 \times 8 = 72$  GB of memory, which is not available in our system. Therefore for datasets larger than  $10^4$  points, we randomly sample 1000 points, equally sampling from each class, and pre-train all the hashing algorithms on this smaller set. We then compute the binary hash codes for all the training points and store them for comparison against the test sets. Since the number of exponential and logarithm map evaluations as well as kernel evaluations will decrease, we will distinguish the training and testing times for the hashing methods where all the data was used for training ( $100 - 10^4$  samples), and for methods where a pre-training approach was used, ( $10^5, 10^6$  samples).

**Unit Hypersphere -  $S^{99}$ .** The unit hypersphere,  $S^{99}$ , is the set of all points,  $\mathbf{x} \in \mathbb{R}^{100}$  that satisfy the constraint,  $\sum_{i=1}^{100} x_i^2 = 1$ . The geodesic distance between two points,  $\mathbf{x}$  and  $\mathbf{y}$ , on a hypersphere is defined as  $d_G(\mathbf{x}, \mathbf{y}) = \cos^{-1}(\mathbf{x}^\top \mathbf{y})$ . Moreover, the logarithm and exponential maps on the sphere are defined as,

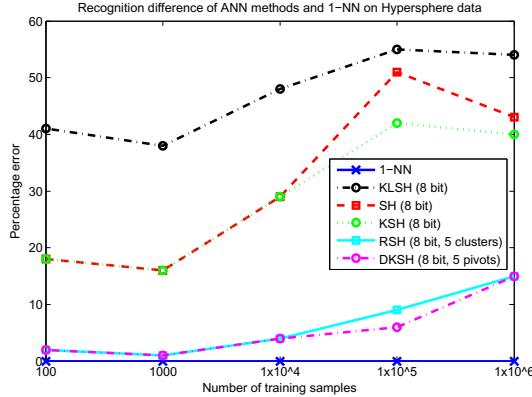
$$\log_{\mathbf{x}}(\mathbf{y}) = \frac{\mathbf{y} - (\mathbf{x}^\top \mathbf{y})\mathbf{x}}{\|\mathbf{y} - (\mathbf{x}^\top \mathbf{y})\mathbf{x}\|} \cos^{-1}(\mathbf{x}^\top \mathbf{y}),$$

$$\exp_{\mathbf{x}}(\Delta) = \cos(\|\Delta\|)\mathbf{x} + \sin(\|\Delta\|) \frac{\Delta}{\|\Delta\|},$$

where  $\Delta$  is a tangent vector at the pole  $\mathbf{x}$ . Finally, the standard inner product also defines a kernel on the sphere, i.e.  $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$ .

We generate 5 sets of 5-class each training datasets containing 100, 1000,  $10^4$ ,  $10^5$  and  $10^6$  points on  $S^{99}$ . For testing, we generate 100 more points in each case. Figure 4 displays the difference between the recognition percentages of exact 1-NN and the state-of-the-art methods (Kernel LSH (KLSH), SH and KSH) and the proposed hashing algorithms (RSRH and DKSH). We use 8 bits for all hashing algorithms and 5 clusters for the proposed methods. Both RSH and DKSH have the lowest percentage difference compared to the state-of-the-art methods for all training sizes. Moreover, the error percentages remain within 10-15% of the exact 1-NN method. This can clearly be attributed to the fact that the proposed methods specifically take into account the manifold structure of the space and thus result in better recognition performance.

Table 1 shows the training times required for each algorithm against the number of training samples. 1-NN does not require any training, whereas SH and RSH are the fastest to train. The training times for KLSH, KSH and DKSH increase greatly with the number of training samples. This is attributed to the increasing number of kernel computations. Table 2 provides the total test time for 100 samples. Coupled with higher accuracy, this is where we observe the real advantage of the proposed methods. As the size of the training data increases, not surprisingly, the time taken for 1-NN also increases. All test times for SH and KSH remain low but are still



**Fig. 4**  $S^{99}$  - Comparison of NN and ANN methods

**Table 1**  $S^{99}$  - Training times

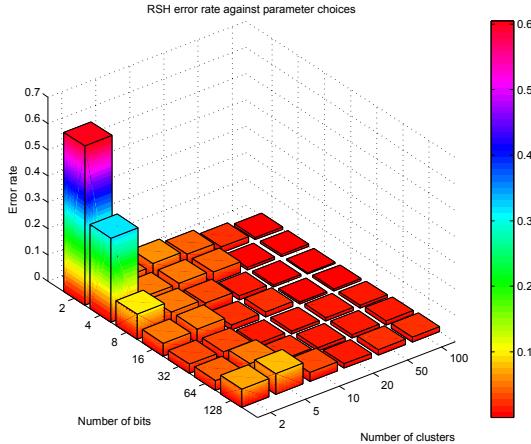
Method	Training time (sec)				
	100	1000	$10^4$	$10^5$	$10^6$
NN	0	0	0	0	0
KLSH	0.01	3.44	1.5h	85.0	11.7m
SH	0.01	0.40	56.02	46.11	6.4m
KSH	0.02	7.30	2.0h	50.7m	7.5h
RSH	0.35	1.90	33.8	70.17	8.8m
DKSH	0.08	7.62	2.0h	72.02	10.1m

**Table 2**  $S^{99}$  - Testing times

Method	Testing time (sec)				
	100	1000	$10^4$	$10^5$	$10^6$
NN	0.01	0.02	0.41	4.81	1.3m
KLSH	0.03	0.02	0.37	1.96	17.0
SH	0.04	0.04	0.04	0.09	1.06
KSH	0.06	3.13	4.1m	2.32	4.02
RSH	0.06	0.06	0.05	0.07	0.28
DKSH	0.06	0.07	10.09	0.07	0.25

higher than the test times for RSH and DKSH. This again illustrates the superiority of the proposed methods.

Figure 5 displays the dependence of the *error rate* of RSH on the algorithm parameters, *i.e.*, the number of bits and the number of clusters. We can see that if the number of bits is kept constant, increasing the number of cluster centers decreases the testing error rate. Similarly, keeping the number of clusters constant, and increasing the number of bits also decreases the testing error rate. The first quality is highly



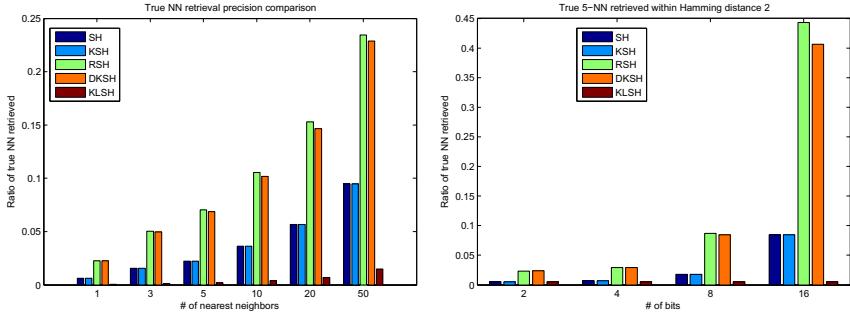
**Fig. 5**  $S^{99}$  - RSH error rates against algorithm parameters

desirable, since in a real scenario, the binary code will represent the memory location for a pointer to the data. Thus having more than 64 bits is not practical. In fact this shows that we can use relatively fewer number of bits and pack the data points in memory by using more clusters. Since the clusters can be located arbitrarily in memory, this reduces the need for large chunks of contiguous memory.

To compare the performance of NN retrieval, we generated 10,000 random points on the 99-dimensional hypersphere,  $S^{99}$ , and trained all the previously mentioned algorithms. For each point in the set, we computed the true nearest neighbors using the correct geodesic distance. We then computed the approximate nearest neighbors and compared the number of ANN that corresponded with true NN for each algorithm. The results were averaged over all 10,000 points for 20 replications of the same experiment. Figure 6(a) shows the mean ratio of true NN found in the closest 1, 3, 5, 10, 20, and 50 ANN found using each algorithm. As in the previous experiments, we used 8 bits for all the algorithms and 5 clusters for our proposed approaches.

To compare the precision of the approaches, we randomly generated 1000 points on the manifold and retrieved all the ANNs that were within a Hamming distance of 2 from each point. We then computed the precision of the approach as the ratio of the number of points retrieved that correspond to the true 5-NN for each point and averaged the results across all the points in a leave-one-out fashion. Figure 6(b) shows the precision of all the approaches against the number of bits. As we can see, both RSH and DKSH perform much better than both SH and KSH whereas KLSH seems to perform the worst.

**Grassmann Manifold -  $G_{10,3}$ .** In an analogous fashion to the previous section, we generate several training samples of different sizes on the Grassmann manifold,  $G_{10,3}$ , which is the manifold of all the 3-dimensional subspaces of  $\mathbb{R}^{10}$ . The data lies in 5 classes and is generated using the method in [10]. For non-linear clustering



(a) Ratio of true NN retrieved using various ANN algorithms for 10000 points on  $S^{99}$  (b) Precision as measured by ratio of correct 5-NN retrieved against number of bits -  $S^{99}$

and tangent space to manifold projections and vice-versa, we use the expressions for the exponential and logarithm maps on the Grassmann manifold in [36]:

$$\log_x(\mathbf{y}) = \mathbf{u} \sin^{-1}(\mathbf{s}) \mathbf{v}^\top, \quad (7)$$

where  $\mathbf{u}\mathbf{s}\mathbf{d}^\top = \mathbf{y} - \mathbf{x}\mathbf{x}^\top\mathbf{y}$  and  $\mathbf{v}\mathbf{c}\mathbf{d}^\top = \mathbf{x}^\top\mathbf{y}$  is the generalized SVD with  $\mathbf{c}^\top\mathbf{c} + \mathbf{s}^\top\mathbf{s} = \mathbf{e}$  and  $\sin^{-1}$  acts element-wise on the diagonal matrix  $S$ . Here  $\mathbf{e}$  is the identity matrix. Similarly,

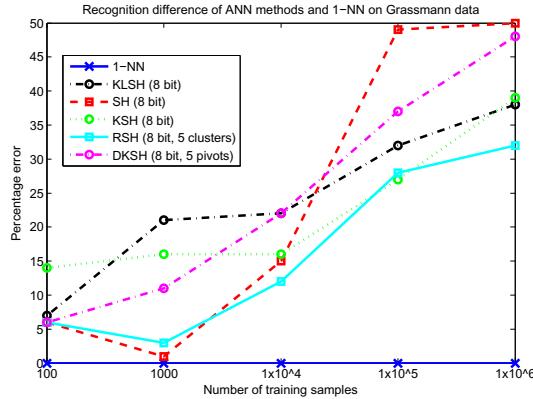
$$\exp_x(\Delta) = \mathbf{x}\mathbf{v} \cos(\mathbf{s}) \mathbf{v}^\top + \mathbf{u} \sin(\mathbf{s}) \mathbf{v}^\top, \quad (8)$$

where  $\Delta$  is a tangent vector at the pole  $\mathbf{x}$  and  $\mathbf{u}\mathbf{s}\mathbf{v}^\top$  is the SVD of  $\Delta$ .

For computing the kernel on the manifold we use the product of the cosines of the subspace angles between the subspaces [15]. Briefly, the subspace angles  $\{\theta_i\}_{i=1}^n$  between two  $n$ -dimensional subspaces,  $\text{range}(A)$  and  $\text{range}(B)$ , in  $\mathbb{R}^d$ , where  $A, B \in \mathbb{R}^{d \times n}$ , are defined by  $\theta_i = \cos^{-1} \lambda_i$ , where  $\lambda_i^2$  is the  $i$ -th eigenvalue of  $(A^\top A)^{-1} (A^\top B) (B^\top B)^{-1} (B^\top A)$ .

Again, we use 8 bits for the binary codes for all hashing algorithms and 5 clusters for our proposed methods. Figure 6 displays the difference between the recognition percentage of 1-NN and all other methods. At first it might seem that SH performs better than the other methods for small data sizes, the trend is offset drastically with large training sizes where it performs the worst. Overall, RSH performs better than all state-of-the-art methods and the error stays within at most 30% of that of 1-NN.

Table 3 provides the training times for each of the training datasets. We notice that KLSH, KSH and DKSH require the largest training times, whereas RSH and SH require the least. For all the methods, the training time increases with the number of data points but due to the large number of kernel computations during the training stage, the increase in time is greatest for KLSH, KSH and DKSH. Table 4 gives the total test time for 100 test samples for each of the training sizes. We again see the computational advantage of the proposed methods against the exact method as well as the state of the art KLSH and KSH methods. The test time increases steeply



**Fig. 6**  $G_{10,3}$  - Comparison of NN and ANN methods

**Table 3**  $G_{10,3}$  - Training times

Method	Training time (sec)				
	100	1000	$10^4$	$10^5$	$10^6$
NN	0	0	0	0	0
KLSH	0.01	5.07	1.2h	34.0m	5.5h
SH	0.01	0.08	9.60	16.63	3.9m
KSH	0.12	17.2	2.5h	1.5h	13.2h
RSH	0.56	10.52	6.1m	10.2m	16.6m
DKSH	0.17	21.27	3.2h	36m	6.3h

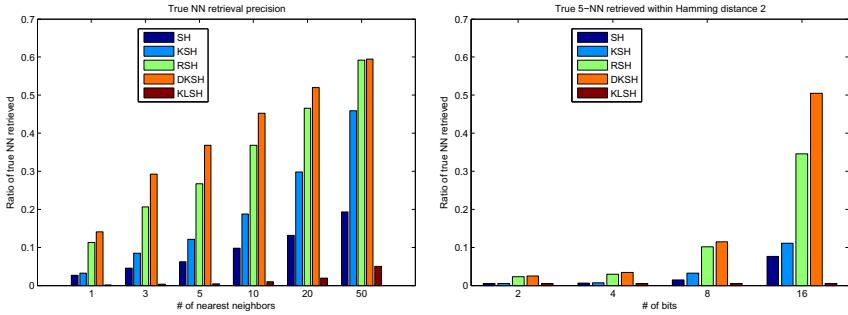
**Table 4**  $G_{10,3}$  - Testing times

Method	Testing time (sec)				
	100	1000	$10^4$	$10^5$	$10^6$
NN	2.06	21.2	3.3m	41.1m	5.7h
KLSH	0.20	2.29	22.5	4.15	23.1
SH	0.04	0.03	0.04	0.10	1.27
KSH	0.28	3.60	3.5m	4.83	5.47
RSH	0.12	0.11	0.10	0.13	1.21
DKSH	0.17	1.95	1.5m	1.16	1.88

with the size of the data for the kernel-based methods, whereas the corresponding increase in test time stays low for RSH.

As in the case of the hypersphere, we also compared the accuracy of ANN retrieval of the proposed approaches against the state-of-the-art. We generated 1000 random points on the Grassmann manifold,  $G_{10,3}$ , and trained all the algorithms with 8 bit codewords and 5 clusters. Fig. 7(a) shows the ratio of true  $k$ -NN retrieved out of the first  $k$  ANN, for different values of  $k$ . As we can see, Euclidean SH performs the worst and our methods, RSH and DKSH, perform far better for all values of  $k$ .

KLSH performs even worse than Euclidean SH, partly because of the low number of bits used and LSH-based algorithms require a large number of bits for good performance. Fig. 7(a) shows the ratio of true 5-NN retrieved within a Hamming distance of 2 for all the algorithms against the number of bits. We again see that our methods, RSH and DKSH, perform the best across all the other methods.

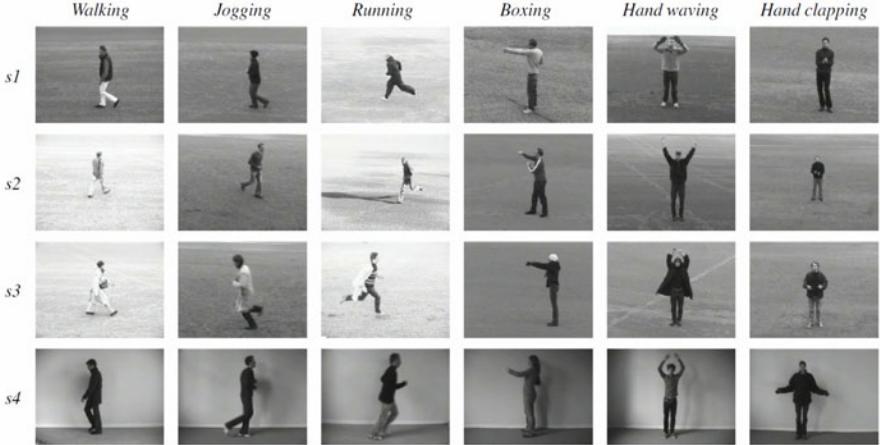


(a) Ratio of true NN retrieved using various ANN algorithms for 1000 points on  $G_{10,3}$  (b) Precision as measured by ratio of correct 5-NN retrieved against number of bits -  $G_{10,3}$

From the above set of experiments, we have shown that the proposed approximate nearest-neighbor methods, RSH and DKSH, by explicitly considering the manifold structure of the space of data, provide great computational advantage against exact Nearest Neighbors while having low to modest decrease in accuracy. Moreover, our methods always perform better than KLSH and KSH, the state-of-the-art non-Euclidean Hashing methods.

## 4.2 Human Action Dataset

Recent approaches in human activity recognition use features such as (1) distributions over a bag of spatial-temporal keypoints to represent the activity in a scene, or (2) dynamical systems learnt from a time-series of features extracted from the frames of the video. Both these features lie in non-Euclidean spaces and therefore the proposed approach is directly applicable for the purpose of retrieving activities from a large dataset of activity videos. Even though, human activity analysis has been a vibrant field in computer vision, to the best of our knowledge, no datasets are publicly available that contain more than a few thousand instances of human actions. Videos of unstructured scenes with multiple activities and events are available, however, the ground-truth activity segmentation and tracking is not provided and automatic extraction of these remains an open problem in computer vision. One of the most popular and largest datasets available is the KTH human action dataset [32]. This dataset contains six actions: walking, running, jogging, boxing, handwaving and handclapping. There are 25 persons performing these actions under four different scenarios: outdoors, outdoors across different scales, outdoors with bulky clothes



**Fig. 7** Sample actions/scenarios from the KTH database (Image from [32])

on and indoors. There are a total of 2391 sequences in the dataset. Figure 7 shows a few sample frames from the KTH database.

For our first experiment, we use the approach of [16] and divide the data as follows: All the videos of the first 16 subjects are used for training whereas the videos of the remaining 9 subjects are used for testing. We extract several spatio-temporal keypoints and their corresponding descriptors in all the videos. Briefly, these keypoints are extracted as the local maxima found after applying spatial-temporal Gaussian and Gabor filters to the video. A feature descriptor is then computed from the intensity data in a small volume around each detected keypoint. A  $k$ -means procedure is then used to cluster the descriptors extracted from the training data to form a dictionary of 100 elements, or words. For each video, we compute the frequency distribution of all the keypoints extracted according to closest distance from these words. This provides a 100 dimensional histogram per video that represents the action in that video.

The space of normalized histograms,  $\mathcal{H} = \{\mathbf{h} | \sum_{i=1}^n h_i = 1, h_i \geq 0\}$  is characterized by the  $n$ -simplex. However it is often more convenient to represent it using the square-root representation,  $\mathcal{H}' = \{\mathbf{h}' | \sum_{i=1}^n (\mathbf{h}'_i)^2 = 1\}$  which is equivalent to the positive orthant of the  $(n-1)$ -dimensional hypersphere,  $S^{n-1}$ . Using the previously developed tools for hyperspheres, we use the 100-dimensional feature histograms embedded in  $S^{99}$  for training and testing the proposed ANN methods. Note that for a fair comparison to nearest-neighbor algorithms, we will test our method against the simple nearest-neighbor algorithm and not against the state-of-the-art methods for human activity recognition that use sophisticated classification algorithms to achieve superior performance. The error rates reported below are not state-of-the-art on the KTH human action database; instead, they are the error rates achieved when using exact NN and state-of-the-art ANN methods and our proposed methods on the dataset. We emphasize that our goal here is not to find the best classification

algorithm on the KTH database, but to compare the performance of the proposed ANN methods against state-of-the-art ANN and exact NN methods.

Table 5 compares the performance of the proposed methods with Nearest Neighbors and state-of-the-art hashing methods. All the hashing methods use 8-bits for the binary codes. The proposed methods, RSH and DKSH divide the training data into 3 clusters. The results show that RSH has the highest recognition percentage other than exact NN, whereas the state-of-the-art KLSH has the worst recognition percentage. Moreover, RSH is also the most efficient method in terms of retrieval time, even though it requires the largest training time. Furthermore, the best recognition rate achievable using RSH was 69% with 64 bit code-words and 2 clusters, which is only 7% below the error rate achieved by exact 1-NN.

For our second experiment, we use the approach in [12] and compute the Histogram of Oriented Optical Flow (HOOF) features at each frame to get a normalized histogram time-series corresponding to the evolution of the optical flow in each video. HOOF features are computed as follows: 1) quantize the space of optical flow vectors into  $B$  bins according to the angle that the vectors make with the horizontal. 2) For each optical flow vector, add its magnitude to the corresponding bin. 3) Normalize the histogram to sum to 1. This gives a normalized histogram for each frame of the video. The dynamics of optical flow in the scene encapsulate the type of action that the person is performing. We then learn a linear-state non-linear dynamical system (NLDS) using the approach in [12] with the Bhattacharrya kernel on histograms. Each activity video is now represented as a NLDS of the form,

$$\begin{aligned}\mathbf{x}_{t+1} &= A\mathbf{x}_t + B\mathbf{v}_t \\ \Phi(\mathbf{y}_t) &= C\mathbf{x}_t + \mathbf{w}_t,\end{aligned}$$

where  $A \in \mathbb{R}^{n \times n}$  represents the temporal dynamics of the transition of the (hidden) state  $\mathbf{x}_t \in \mathbb{R}^n$  at time  $t$  to  $t+1$ .  $\Phi$  is a Euclidean embedding of the non-linear output, in our case a histogram  $\mathbf{y}_t \in \mathbb{R}^p$ , implicitly applied by using the Bhattacharrya kernel on the space of histograms.  $C$  represents the implicit linear mapping from the hidden state to the output whereas,  $B\mathbf{v}_t \sim \mathcal{N}(0, BB^\top)$  and  $\mathbf{w}_t \sim \mathcal{N}(0, \sigma^2 I)$  are the state- and output- noise at time  $t$ . The dynamical system parameters,  $(A, C, B, R)$ , therefore represent the dynamics of human activity in the scene. There are several methods for comparing dynamical systems, e.g. those proposed in [12] and the references therein.

We represent the dynamics and output transformation functions using the *observability matrix* for each dynamical system. The range-space of the observability matrix is composed of all possible outputs of the dynamical system given an initial state and is another common representation for a dynamical system. The observability matrix can be computed as  $\mathcal{O} = [C^\top, (AC)^\top, (A^2C)^\top, \dots, (A^{n-1}C)^\top]^\top \in \mathbb{R}^{pn \times n}$ . Since we are using the Bhattacharrya kernel, which is essentially an inner-product on the sphere, we can simply use PCA to learn the approximate dynamical system parameters and thus get the parameter matrices,  $A \in \mathbb{R}^{n \times n}$  and  $C \in \mathbb{R}^{p \times n}$ . Here  $n$  is the system order, and  $p$  is the size of the output. See [12] and the references therein for more details. Notice that the columns of  $\mathcal{O}$  span an  $n$ -dimensional subspace of

$\mathbb{R}^{np}$  and thus  $\mathcal{O}$  lies on the Grassmann manifold,  $G_{np,n}$ . We can therefore follow the experiments in section 4.1 using these observability matrices as the data points.

Since the approach in [12] is directly applicable only for stationary cameras, we choose sequences from the first scenario, i.e. outdoors with stationary camera (around 600 sequences) to test our algorithms. For the system parameters, we learn a system of order  $n = 10$  use 64 bins for the optical flow histograms, i.e.  $p = 64$ . We use 64% of the data for training and the remaining 36% for testing. Moreover, we use 64 bits for the binary codes and 15 clusters/pivots for the proposed methods.

Table 6 shows the recognition percentages and training and testing times for exact KNN using the Martin distance for dynamical systems, and the proposed and state-of-the-art hashing methods. We can see that our method, RSH, has the best recognition rate, slightly above KLSH. Notice that even though exact NN does not require any training, which could be as high as 321.7 seconds for RSH, the speed up in terms of test times is significant. Exact KNN requires 149.3 seconds for testing whereas RSH requires only 10.3 seconds and DKSH requires 15.8 seconds. Even though KLSH performs well in this scenario, due to the many kernel computations required, its testing time is at least 3 times greater than RSH, limiting the former's advantage.

**Table 5** BOW histograms

Method	Correct %	Train $t$	Test $t$
NN	76	0	11.5
KLSH	24	38.4	1.04
SH	51	3.4	0.45
KSH	51	41.1	81.1
RSH	62	58.6	0.39
DKSH	51	31.1	3.34

**Table 6** Observability matrices

Method	Correct %	Train $t$	Test $t$
NN	72	0	149.3
KLSH	64	0.547	35.0
SH	22	5.262	1.67
KSH	17	31.24	38.7
RSH	65	321.7	10.3
DKSH	58	266.5	15.8

### 4.3 Web Videos Database

Finally, we evaluate our method on the challenging Web Videos Database collected by Zanetti *et al.* in [40]. The Web Videos Database is composed of 11 categories of videos with an average of about 90 videos per category. The categories are Autos & Vehicles, Comedy, Entertainment, Film & Animation, Gadgets & Games, Howto & DIY, Music, News & Politics, Pets & Animals, Sports and Travel & Places. These videos were directly downloaded from Youtube, and therefore contain a number of unstructured artifacts, such as on-screen text, fade-ins and outs, camera shakes, zoom-in and out, animations, and encoding-decoding artifacts. Moreover, they do not contain artificially set up scenarios such as the ones in the KTH database in the previous sub-section. The average classification rate reported on a more extensive, but publicly unavailable, version of the database in [40] is around 34% when using exact k-NN. This low number highlights the difficulty of the semantic video retrieval

problem in general and the need to develop efficient and accurate methods for this important problem.

We divided the database in a training set containing 75% of the sequences selected equally from all the 11 categories, and a test set containing the remaining 25% of the sequences. Similar to the approach in the previous sub-section, we extracted spatio-temporal interest points using the method proposed by Laptev *et al.* in [26] from all the training videos. Histograms of gradients (HOG), and histograms of optical flow (HOF) were extracted from spatio-temporal windows around each interest point. The HOG-HOF features from the interest points collected from all the training videos were then quantized into 1000 codewords. For each training video, a normalized histogram of occurrences of these codewords was computed as a representation of the training video. Therefore, each video sequence was represented as a 1000-dimensional histogram. For the test videos, we similarly extracted the spatio-temporal features and computed a normalized histogram of occurrences of the codewords learnt during the training phase. Therefore, the problem of finding the nearest neighbors of a test video sequence boils down to the problem of computing nearest neighbors of the test histogram in the training set. Following the approach in the previous sub-sections, we used the square-root representation for histograms and the corresponding log- and exp-maps as well as kernels were used to retrieve the nearest neighbors.

Table 7 shows the performance of exact NN and several ANN methods including our proposed RSH and DKSH algorithms. As mentioned earlier, this is a very challenging database and even the exact NN recognition rate on this database is very low. Our proposed algorithms, RSH, and DKSH still outperform state-of-the-art ANN algorithms in correctly classifying the category of test sequences. In fact, RSH performs at just under a percent below the classification accuracy of exact NN. Moreover, as expected, our proposed methods perform much faster than exact NN, even though they require more time for training.

Figure 8 and 9 show the first four nearest neighbors for sample queries from the Autos & Vehicles and Howto & DIY categories respectively. A sample frame from the test sequence is shown in the first row. The subsequent rows show a sample frame from each of the first four nearest neighbor videos with their category label in the caption. The second row, (b-e) show exact NN matches. The third row, (f-i) shows results from KLSH. The fourth row, (j-m) shows the nearest neighbor results found using SH. The fifth row, (n-q) shows the nearest neighbor results found using

**Table 7** Comparison of exact NN and ANN methods on the Web Videos Database.

Method	Correct %	Train $t$	Test $t$
NN	33	0	5.58
KLSH	6.7	0.91	0.53
SH	23	3.85	0.29
KSH	25	1.01	1.10
RSH	32	33.2	0.31
DKSH	20	35.0	0.44



(a) Autos &amp; Vehicles



(b) Pets &amp; Animals



(c) Autos &amp; Vehicles



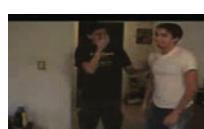
(d) Autos &amp; Vehicles



(e) Autos &amp; Vehicles



(f) Autos &amp; Vehicles



(g) Comedy



(h) Film &amp; Animation



(i) Film &amp; Animation



(j) Autos &amp; Vehicles



(k) Autos &amp; Vehicles



(l) Pets &amp; Animals



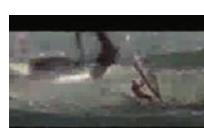
(m) Sports



(n) Autos &amp; Vehicles



(o) Pets &amp; Animals



(p) Sports



AlleyKat - Turbo s2k  
Custom turbo kit  
w/gt35r,Walbro fp,precision  
intercooler & injectors,STOCK  
internals,clutch w/ tranny,  
limited slip  
@ 14psi....

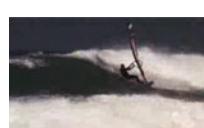
(q) Autos &amp; Vehicles



(r) Autos &amp; Vehicles



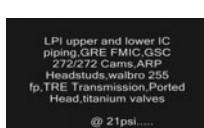
(s) Autos &amp; Vehicles



(t) Sports



(u) Autos &amp; Vehicles



(v) Autos &amp; Vehicles



(w) Travel &amp; Places

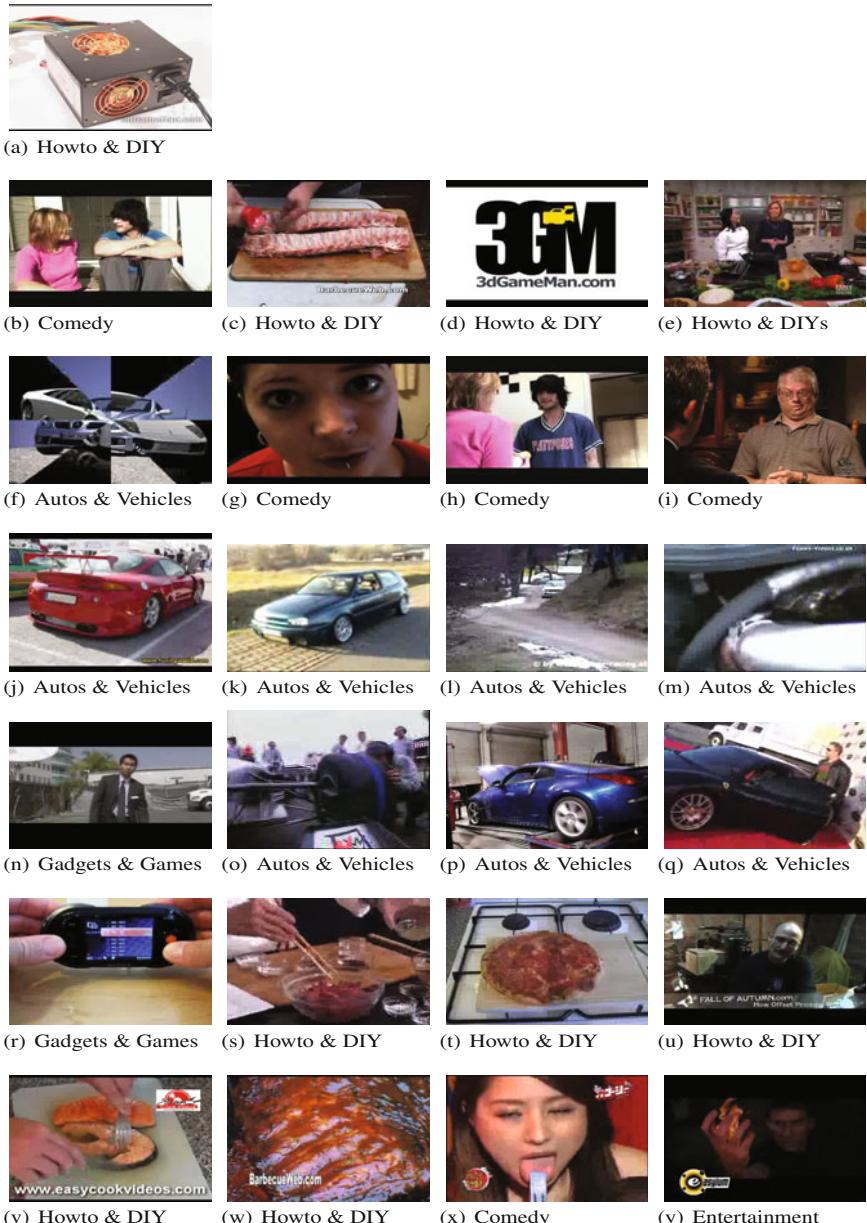


(x) News &amp; Politics



(y) Autos &amp; Vehicles

**Fig. 8** The result of applying exact NN and several ANN algorithms on the test video in (a) from the category Autos & Vehicles. Each row shows random frames from the first four nearest neighbors to the test video from left to right and their category name in the caption. Row 2 (b-e) Exact 1-NN, Row 3 (f-i) KLSH, Row 4 (j-m) SH, Row 5 (n-q) KSH, Row 6 (r-u) RSH and Row 7 (v-y) DKSH.



**Fig. 9** The result of applying exact NN and several ANN algorithms on the test video in (a) from the category Howto & DIY. Each row shows random frames from the first four nearest neighbors to the test video from left to right and their category name in the caption. Row 2 (b-e) Exact 1-NN, Row 3 (f-i) KLSH, Row 4 (j-m) SH, Row 5 (n-q) KSH, Row 6 (r-u) RSH and Row 7 (v-y) DKSH.

KSH. The sixth (r-u), and seventh (v-y) rows provide the nearest neighbor results found using our proposed algorithms RSH and DKSH respectively. As we can see, even exact NN does not return always return all the sequences from the correct category. KLSH seems to perform the worst, as also highlighted in Table 7. SH and KSH perform better than KLSH, but worse than our proposed methods, and seem to provide only one (SH) or two (KSH) correct matches, as opposed to three (RSH) and two (DKSH), as shown in Figure 8. In Figure 9, SH and KSH perform even worse and fail to find a single proper match, whereas our algorithms get two or more correct category matches. We can again see, that our methods perform close to exact NN and much better than state-of-the-art ANN methods for non-Euclidean manifolds. Moreover, our methods are very efficient and give orders of magnitude better search times as compared to exact NN.

## 5 Conclusion

In this chapter, we have addressed the problem of sub-linear search through large data sets of videos for business analytics such as human behaviours, events of interest or abnormal activities. We have introduced the approximate nearest neighbor search problem as a very efficient method for finding similar items to a query item in such large data sets. We have briefly described a number of applications where ANN methods have been successfully used in video retrieval applications. We observed that in many applications, video content is represented using nonlinear models and most state-of-the-art ANN algorithms have been designed to deal with data that usually lies in a Euclidean space. To address data that lies on non-Euclidean manifolds, we have proposed two new methods, Riemannian Spectral Hashing (RSH), and Distributed Kernel Spectral Hashing (DKSH), for performing fast approximate nearest-neighbor matching on non-Euclidean data. We have shown that state-of-the-art methods either do not take into account the manifold structure of the data, or are computationally inefficient and can in fact be slower in performance than exact nearest neighbors. Our experiments on synthetic and real data have shown that our methods are applicable to points that lie on simple manifolds such as the unit hypersphere as well as to points that lie on highly complicated manifolds such as the space of dynamical systems. The proposed methods provide immense computational savings at the cost of a small decrease in accuracy and hence are ideal for approximate nearest neighbor matching in large datasets.

**Acknowledgements.** The technical contribution in this work was originally done as part of a summer research internship at the Mitsubishi Electric Research Laboratories, Cambridge, MA, USA and published in [11]. The authors would like to thank Kinh Tieu, Ashok Veeraraghavan and Oncel Tuzel for their comments and discussions that helped improve the presentation of this work. Rizwan Chaudhry was also partially supported by the grant NSF 0941463 during the preparation of this chapter.

## References

- [1] Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In: Symposium on Foundations of Computer Science (2006)
- [2] Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM* 45, 891–923 (1998)
- [3] Basharat, A., Shah, M.: Time series prediction by chaotic modeling of nonlinear dynamical systems. In: International Conference on Computer Vision (2009)
- [4] Beis, J.S., Lowe, D.G.: Shape indexing using approximate nearest-neighbor search in high dimensional spaces. In: IEEE Conference on Computer Vision and Pattern Recognition (1997)
- [5] Ben-Arie, J., Wang, Z., Pandit, P., Rajaram, S.: Human activity recognition using multidimensional indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 1091–1104 (2002)
- [6] Bengio, Y., Delalleau, O., Roux, N.L., Paiement, J.F., Vincent, P., Ouimet, M.: Learning eigenfunctions links spectral embedding and kernel pca. *Neural Computation* 16 (2004)
- [7] Bissacco, A., Chiuso, A., Ma, Y., Soatto, S.: Recognition of human gaits. In: IEEE Conference on Computer Vision and Pattern Recognition (2001)
- [8] Bissacco, A., Chiuso, A., Soatto, S.: Classification and recognition of dynamical models: The role of phase, independent components, kernels and optimal transport. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(11), 1958–1972 (2007)
- [9] Biswas, S., Aggarwal, G., Chellappa, R.: Efficient indexing for articulation invariant shape matching and retrieval. In: IEEE Conference on Computer Vision and Pattern Recognition (2007)
- [10] Çetinçül, H.E., Vidal, R.: Intrinsic mean shift for clustering on stiefel and grassmann manifolds. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
- [11] Chaudhry, R., Ivanov, Y.: Fast Approximate Nearest Neighbor Methods for Non-Euclidean Manifolds with Applications to Human Activity Analysis in Videos. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 735–748. Springer, Heidelberg (2010)
- [12] Chaudhry, R., Ravichandran, A., Hager, G., Vidal, R.: Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
- [13] Chen, D.Y., Lee, S.Y., Chen, H.T.: Motion activity based semantic video similarity retrieval. In: Advances in Multimedia Information Processing (2002)
- [14] Chen, X., Zhang, C.: Semantic event retrieval from surveillance video databases. In: IEEE International Symposium on Multimedia (2008)
- [15] Cock, K.D., Moor, B.D.: Subspace angles and distances between ARMA models. *System and Control Letters* 46(4), 265–270 (2002)
- [16] Dollar, P., Reabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: VS-PETS (2005)
- [17] Freidman, J.H., Bentley, J.L., Finkel, R.A.: An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3, 209–226 (1977)
- [18] Fukunaga, K., Narendra, P.M.: A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers* 24, 750–753 (1975)
- [19] Goh, A., Vidal, R.: Clustering and dimensionality reduction on Riemannian manifolds. In: IEEE Conference on Computer Vision and Pattern Recognition (2008)

- [20] Hastie, T., Tibshirani, R., Friedman, J.H.: *The Elements of Statistical Learning*. Springer (2003)
- [21] Karpenko, A., Aarabi, P.: Tiny videos: Non-parametric content-based video retrieval and recognition. In: IEEE International Symposium on Multimedia (2008)
- [22] Kashino, K., Kimura, A., Kurozumi, T.: A quick video search method based on local and global feature clustering. In: International Conference on Pattern Recognition (2004)
- [23] Kulis, B., Darrel, T.: Learning to hash with binary reconstructive embeddings. Tech. Rep. UCB/EECS-2009-101. Electrical Engineering and Computer Sciences. University of California at Berkeley (2009)
- [24] Kulis, B., Grauman, K.: Kernelized locality-sensitive hashing for scalable image search. In: International Conference on Computer Vision (2009)
- [25] Laptev, I.: On space-time interest points. International Journal of Computer Vision 64, 107–123 (2005)
- [26] Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: IEEE Conference on Computer Vision and Pattern Recognition (2008)
- [27] Muja, M., Lowe, D.G.: Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Applications (2009)
- [28] Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: IEEE Conference on Computer Vision and Pattern Recognition (2006)
- [29] Ravichandran, A., Chaudhry, R., Vidal, R.: View-invariant dynamic texture recognition using a bag of dynamical systems. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
- [30] Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning 50, 969–978 (2009)
- [31] Schölkopf, B., Smola, A.: *Learning with Kernels*. MIT Press, Cambridge (2002)
- [32] Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: A local svm approach. In: International Conference on Pattern Recognition (2004)
- [33] Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
- [34] Sidenbladh, H., Black, M.J., Sigal, L.: Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 784–800. Springer, Heidelberg (2002)
- [35] Silpa-Anan, C., Hartley, R.: Optimised kd-trees for fast image descriptor matching. In: IEEE Conference on Computer Vision and Pattern Recognition (2008)
- [36] Subbarao, R., Meer, P.: Nonlinear mean shift over riemannian manifolds. International Journal of Computer Vision 84, 1–20 (2009)
- [37] Turaga, P., Chellappa, R.: Nearest-neighbor search algorithms on non-euclidean manifolds for computer vision applications. In: Indian Conference on Computer Vision, Graphics and Image Processing (2010)
- [38] Turaga, P., Veeraraghavan, A., Chellappa, R.: From videos to verbs: Mining videos for events using a cascade of dynamical systems. In: IEEE Conference on Computer Vision and Pattern Recognition (2007)
- [39] Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: Neural Information Processing Systems Conference (2008)
- [40] Zanetti, S., Zelnik-Manor, L., Perona, P.: A walk through the web's video clips. In: IEEE Workshop on Internet Vision, in Computer Vision and Pattern Recognition, CVPR (2008)

## **Part II**

# **Demographics**

# Human Age Estimation and Sex Classification

Guodong Guo

**Abstract.** Collecting demographic information from the customers, such as age and sex, is very important for marketing and customer group analysis. For instance, the marketing study has an interest to know how many people visited a shopping mall, and what is the distribution of the customers, such as how many males and females; how many young, adult, and senior people. Instead of hiring human workers to observe the customers, a computational system might be developed to analyze people who appeared in images and videos captured by cameras installed in a shopping mall, and then gather the demographic information. To develop a real system for age estimation and sex classification, many essential issues have to be addressed. In this chapter, a detailed introduction of the computational approaches to human age estimation and sex classification will be given. Various methods for feature extraction and learning will be described. Major challenges and future research directions will also be discussed. The goal is to inspire new research and encourage deeper investigation towards developing a working system for business intelligence.

## 1 Introduction

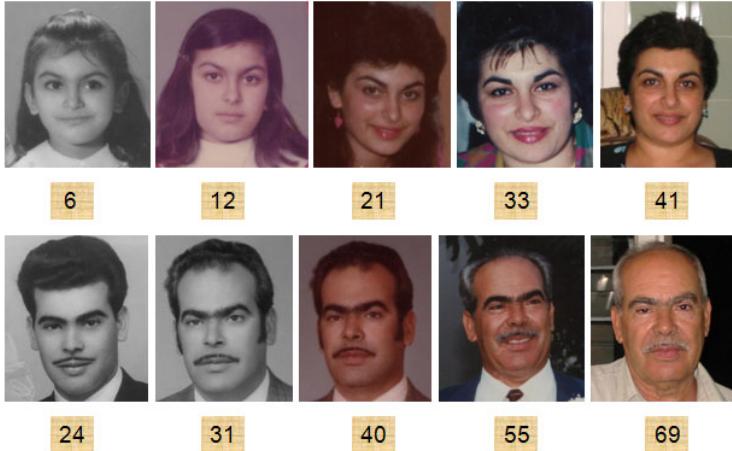
Automatic demographic information extraction by a machine is very useful for many real applications. For example, in business intelligence, the extracted age and sex information of the customers could provide important information for marketing study, such as *how many people visited a store or a shopping mall; how many males and females; and how many young, adult, and senior people of the customers?* Human observers could be hired to record related demographic information for a short time period, but it is not practical for a long term. Developing a computational system to perform this task may save the labor and operate at any time as long as you want.

---

Guodong Guo

Department of Computer Science and Electrical Engineering, West Virginia University  
e-mail: [guodong.guo@mail.wvu.edu](mailto:guodong.guo@mail.wvu.edu)

Human age estimation and sex classification should be an important component for business intelligence. Inspired by the great importance and practical value, researchers have been working on these topics for more than a decade. Significant progresses have been made in both age estimation and sex classification.



**Fig. 1** Face images of a female and a male at different ages (from FG-NET).

Usually, face images are used for age estimation and sex classification. Some example face images with ground truth ages are shown in Fig. 1. Other modalities, such as human body, can be used for sex recognition, too. In this chapter, the main focus is face-based age estimation and sex classification, while the body-based sex recognition is also introduced.

*Age estimation* is defined by using a computer or machine to estimate the specific age of a person, e.g., 10 years or 81 years, given his or her face image. The age interval is usually one year. In other words, the age estimation result will be integer numbers, e.g., 3 years, but not like 3 years and 2 months. A person's age can also be “estimated” by a human observer. In that case, we call it “age perception” rather than age estimation. Sometimes, the problem is reduced to just knowing the age group, such as young, adult, and senior, rather than the specific age of a person. In that situation, we call it “age group classification” to differentiate from age estimation, since different methods could be used given the problem difference. *Sex classification* is defined by using a computer or machine to determine the sex (male or female) of a person, given the face image (or other modalities).

Age perception and sex recognition are also active research topics in other disciplines than computational visual processing, such as psychology [74], gerontology [55, 18], public health [17, 87], and social science [16]. Rhodes [74] presented a review of human perception of ages based on psychology studies. In sex recognition, the main focus of psychology studies [7, 100, 94] is about how humans discriminate

between males and females and what kind of features are more useful and discriminative. Wild et al. [94] even showed that sex recognition is more difficult for subjects when children's faces are presented compared to adults' faces. Different from the studies in those disciplines, here I will focus on the computational aspect of human age estimation and sex classification.

In this chapter, I will provide some overview of typical approaches to age estimation and sex classification. Various methods are introduced for solving each problem. I will also discuss the relation between age estimation and sex classification, which is useful in providing a practical guide to design and develop a large-scale working system. I will also present major challenges in age estimation and sex classification, and hope to inspire new investigation and advance the research on these topics.

This chapter is hopefully written for a broad range of readers with different background. For readers who have worked on these topics, you will get a view of the current state-of-the-art approaches, understand the great challenges, and possibly get an inspiration of new thoughts. For readers who do not have any background in these areas or are not interested in technical details, at least you can get the knowledge about typical approaches to age estimation and sex classification, and what kinds of results or recognition accuracies you can expect.

## ***1.1 Applications***

Before presenting the algorithms and methods to process age and sex information, I would like to describe some potential applications of age estimation and sex classification first. To some extent, this can answer the question of why we are interested in age and sex processing, especially for people who are not familiar with these topics, or people have no technical background. Hopefully the readers can understand why age estimation and sex classification are useful and why researchers are working hard to develop methods to solve these problems.

**Customer Statistics for Business Intelligence:** Age and sex are useful information about the customers. Business managers or store owners may want to know how many people visited a shopping mall or a specific store, how many males and females among the customers, and how many of them belong to different age groups, e.g., young, adult, and senior people. Different age and sex groups of customers may have different shopping habit and preferences. Customers of different groups may have different responses and expectations to a new product. Companies or stores may target specific age and sex groups to advertise their products or attract people in some special groups. Knowing the customer responses and expectations corresponding to different groups may help the companies to get better feedback from the market and then improve their products and make more money. For example, restaurant owners want to know what percentage of people in each age and sex group purchased what kinds of sandwiches; store owners want to target some groups of people (potential customers) to advertise their products by showing business suits as a man walks by or jeans as an teenager is coming [28].

**Access Control Constrained by Age and Sex:** Suppose an age estimation system is installed in a door access, it can give a warning when an underage teenager enters bars or wine shops. It can also prevent minors from purchasing tobacco products from a vending machine. In another scenario, the age estimation system can send warnings when people who are too young or too old want to ride a roller coaster in a theme park [28]. For a computer and Internet access, an age estimation system can prevent young children (e.g., under 18 years old) to access Web sites with adult images, videos, or other adult materials. A “smart building” might use sex information for surveillance and control of access to certain areas [41]. For example, some areas may only be accessed by a man or woman. A “smart environment” may use an age estimation system to find elderly people who may need special care or help. In Japan, the money transfer frauds on ATMs are more and more severe. It was reported that the victims were swindled 2.75 billion Japanese Yen of 20,481 cases in 2008 [63]. Most victims are women over age 60. If age and sex processing systems can be installed in the ATMs, so that a warning signal could be sent to the bank when a woman over age 60 is transferring money, it will help to avoid possible money transfer frauds.

**Photo Organization Based on Age and Sex:** Digital cameras and camcorders are cheaper and cheaper, portable devices such as smart phones and tablet PCs are equipped with cameras. As a result, people are capturing photos and videos very frequently. One can imagine how many photos and videos can be captured by a person or family over the life period. Given the huge amount of data, it is necessary to study how to organize the photos or videos to make it easy to access and find the related photos when you want to see. Using age and sex information to label and annotate the visual data will provide some help for family photo organization and retrieval. For example, you may give a query, “find my daughter’s photos when she was five years old,” and get the relevant photos quickly rather than browse all photos one by one.

**Image Filtering for Search Engines:** Images and videos are being uploaded into the hosting sites, such as Flickr and YouTube. Social network web sites, such as Facebook and Myspace, have more and more images and videos in addition to text and other materials. By performing age estimation and sex classification on those images and videos with human faces, it can help the search engines to filter the images into different categories, such as images of young girls or old men. These can be a useful component to help the search engine to refine the search results or speed up the search process.

**Cross-Age Human Identification:** Human identification is important for homeland security, border control, and surveillance. Face is a popular biometric trait among many other traits, such as iris, fingerprint, voice, palmprint, hand geometry, etc. However, the human face undergoes significant changes when he or she grows from baby to adult and to elderly. As a result, the face templates have to be updated many times after the first time enrollment [71, 69]. If an age estimation system can be built, it can help the face recognition system to determine when to update and if it is necessary to update in some fixed time period. The amount of facial changes because of aging can be very different for people at different ages.

For example, a young child may undergo significant facial shape changes after several months or one year, while an adult at 30s may not change too much even after several years. On the other hand, age and sex are important traits of soft biometrics for video surveillance [27]. Soft biometrics can be used to improve the recognition accuracies of the primary biometric traits [54]. Guo et al. [50] showed that the soft biometrics, e.g., sex, ethnicity, weight, and height, can be used to improve the accuracies of cross-age face recognition significantly on a very large database with more than 10,000 individuals.

## 1.2 *Organization*

In the following of this chapter, I will present age estimation in Section 2 and sex classification in Section 3. Then I discuss the relation between age estimation and sex classification in Section 4. Major challenges of age and sex processing in real world applications will be presented in Section 5. Finally, some conclusions will be drawn.

# 2 Age Estimation

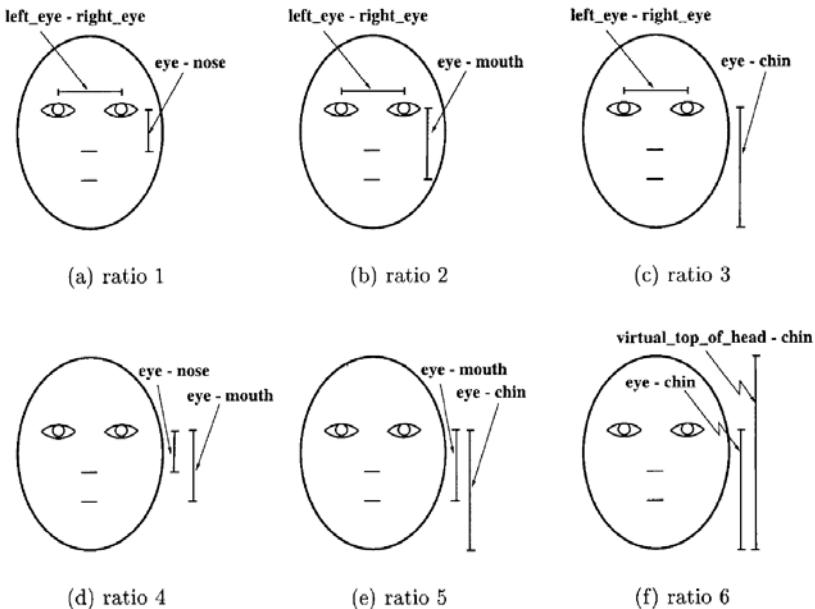
Human age estimation has become an active research in computer vision and pattern recognition [28, 73]. Usually age estimation has two major parts: Face representation and aging function learning. Each part will be introduced in detail in this section. After that, I will also discuss related issues, such as age group classification, evaluation of age estimation methods, and aging databases. Finally, other problems related to but different from age estimation will be mentioned.

## 2.1 *Face Representation*

For age estimation, various methods can be used to represent face images in order to extract useful features. In the following, various methods for facial aging representation will be described.

**Wrinkle Models:** When talking about human aging, one may immediately think about facial wrinkles. From adulthood to old age, skin aging becomes more and more significant because of the underneath muscle changes. Facial muscles lose elasticity and become loose gradually, which causes the appearance of wrinkles, creases, and sagging in faces. Based on the observation of facial wrinkles, one can develop a computational wrinkle model to recognize old people. Kwon and Lobo [57] presented the earliest work of detecting wrinkles from face images to separate young adults from senior adults. The wrinkles can be computed in several regions such as on the forehead, next to the eyes, and near the cheek bones. The detection of wrinkles in a facial region is based on the survival test of some randomly dropped snakelets in the specified regions.

The wrinkle model can be used to separate old people from young adults and babies, but it cannot predict the real age of a person, such as 56 or 57 years old. In practice, it might be combined with other models for age group classification. For example, the wrinkle model was used with anthropometry measures to classify people into three age groups: babies, young adults, and senior [57].



**Fig. 2** The six ratios used for facial anthropometric measure, originally shown in [57].

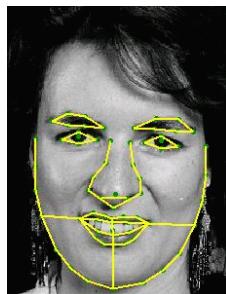
**Anthropometric Models:** Anthropometry is a branch of anthropology concerned with comparative measurements of the human body and its parts. Face anthropometry deals with measuring sizes and proportions on human faces. Farkas [24] provided a comprehensive measurement of facial distances and angles. Specifically, Farkas [24] reported the measurements over 57 facial landmarks or fiducial points at different ages, ranging from infancy to adulthood. Based on these anthropometric measures, one can develop computational methods to characterize the faces at different ages. A typical use of the facial anthropometry is to compute the distance ratios measured from facial landmarks. For example, Kwon and Lobo [57] computed six ratios of distances on frontal face images, as shown in Fig. 2, and used them to separate babies from adults. The experiment was performed on a small database of 47 faces. The authors did not report the overall performance on this small database, but showed the results of using each ratio.

Another approach proposed by Ramanathan and Chellappa [72] is to use eight ratios of distance measures to model age progression in young faces, e.g., 0 to 18 years, rather than just separating young kids from adults. The purpose is to predict

an individual's appearance across age and to perform face recognition across age progression. Their experiment was performed on a database of 233 images of 109 individuals, collected partially from the FG-NET aging database [25] and their own data. Gunay and Nabiyev [39] proposed a variant of Ramanathan and Chellappa's approach to detect facial anthropometric features and then used for age estimation.

In practice, there are some issues in using the anthropometric models: (1) it is not trivial to localize facial fiducial points accurately, although some progress has been made in recent years, and (2) dealing with facial shape variations among different people at the same age. The anthropometric models could be used to separate young people from adults, although they may not be accurate enough to estimate the exact age. So far, the anthropometric models have not been evaluated on a large database yet.

**Active Appearance Models:** The active appearance model (AAM) was proposed by Cootes et al. [20] as a statistical model to represent face images. The AAM method begins with a set of manually labeled landmarks to encode the facial structure, such as eyes, nose, mouth, facial contour, and so on. The principal component analysis (PCA) technique is then applied to these landmarks to learn a shape representation. Similarly the PCA method can also be applied to the image intensities given the structure or shape constraint. The learned AAM models can be applied to new face images to obtain the corresponding representation for each new face, as illustrated in Fig. 3. The AAM is a general method for face encoding and can be applied to different tasks of face processing. Lanitis et al. [59, 58] adapted the AAMs to represent face images for facial aging study. Later on, the AAM models were also used in other approaches, e.g., [44, 42, 43, 61], for age estimation.



**Fig. 3** Illustration of the AAM model on a face image, originally shown in [20].

One should notice that it is not trivial to localize the facial fiducial points accurately for the AAM models when faces undergo some variations, e.g., expression change and head rotations. Inaccurate point locations may have a significant influence on using the AAM models for age estimation.

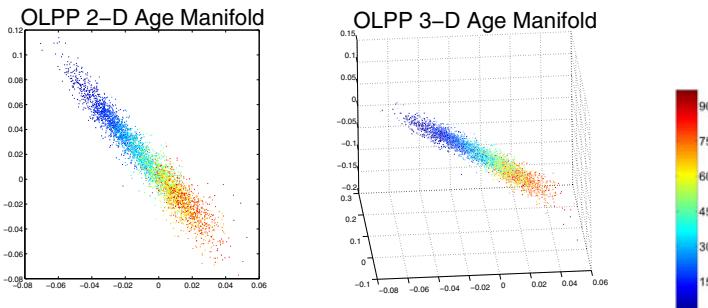
**Aging Pattern Subspace:** Since human ages can be a continuous sequence of numbers, such as 1, 2, ⋯. How about constructing a model to represent all ages together for each individual? Geng et al. [36, 35] explored this idea, and proposed

a method called AGing pattErn Subspace (AGES). An aging pattern is defined as a sequence of face images of the same individual arranged in the temporal order [36].

In the AGES approach, each face image is represented by the AAMs [20] as Lanitis et al [58]. But the AGES method uses all face images of the same individual at different ages.

The AGES approach requires to have all face images at different ages available. If some face images are not available, the AGES method can synthesize the missing faces by using an EM-like iterative learning algorithm. In practice, the synthesis results might not be good if there are too many missing faces.

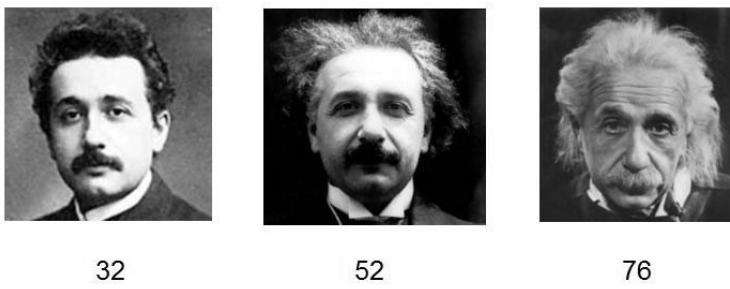
**Age Manifold:** Manifold learning [88, 77] is an important research topic in exploiting a low dimensional embedding of the original high dimensional input, such as face images. It can be used for non-linear dimensionality reduction or visualization of the proximity of the data. Usually the linear methods, such as the PCA or ICA (independent component analysis) [19], are not good enough for visualizing either the proximity of original data or the variation “trend.” There are many existing manifold learning methods that can be used for aging pattern representation, such as the orthogonal locality preserving projections (OLPP) [9]. The OLPP basically produces orthogonal basis functions to preserve essential manifold structures by measuring the local neighborhood distances. It pursues the projections that make the similar data points close while the dissimilar ones far away. Fu et al. [30, 29] applied the OLPP method to learn the age manifold. Guo et al. [44, 42] showed that the age manifold can represent the facial aging as well as obtain low age estimation errors when a proper age estimation method is used. See Fig. 4 for a visualization of the age manifold.



**Fig. 4** Visualization of the age manifold learned by the OLPP method, showing in 2D (left) and 3D (middle), respectively, from [42].

Usually the age manifold needs to be learned from a large training database. When a small database, e.g., the FG-NET [25], is used, it might not be a good idea to learn the age manifold. Another issue is that when raw face images are used for manifold learning, the image alignment problem may have an impact on it [48], since the manifold learning methods themselves cannot deal with the face mis-alignment problem.

**Biologically-Inspired Models:** It is interesting to understand the human vision system and adapt visual cortex models for computational image processing and recognition. A recent theory of the feedforward path of object recognition in the cortex accounts for the first 100-200 milliseconds of processing in the ventral stream of the primate visual cortex [76]. This biologically-inspired model consists of simple (S) and complex (C) layers alternately, and has shown good performance in object category recognition [78, 66]. Guo et al. [49] adapted the bio-inspired model to human age estimation in face images, and call it biologically-inspired features (BIFs). In previous approaches to object category recognition [79, 66, 78], a number of prototypes (approximately 1,000) are randomly selected from the learning images. These prototypes are stored for template matching in  $S_2$  units. Guo et al. [49] found that the  $S_2$  (and then  $C_2$ ) features using pre-learned prototypes do not work well for age estimation. They proposed to use a model that has  $S_1$  and  $C_1$  units and a “STD” operation in  $C_1$  feature extraction. Guo et al. [49] applied their method to some of Einstein’s faces collected from the Internet, and the estimated ages are pretty reasonable as shown in Fig. 5.



**Fig. 5** Age estimation on Einstein’s faces using the biologically-inspired method presented in [49].

Guo et al. [48] combined the biologically inspired model with various manifold learning techniques for facial aging representation. Several manifold learning methods are evaluated based on their combination with the BIF, including the orthogonal locality preserving projections (OLPP) [9], marginal fisher analysis (MFA) [104] and locality sensitive discriminant analysis (LSDA) [10].

It is interesting to combine the BIFs and manifold learning techniques [48]. The BIF has the capability to deal with small translation, rotation, and scale changes in addition to its discriminative power in image representation, while manifold learning techniques are usually sensitive to image mis-alignment caused by many factors, such as shape variations in facial growth and aging, head rotation, and even facial expression changes. It will make the manifold learning methods not sensitive to the image alignment problem when they work on the BIF rather than on raw images directly. On the other hand, the BIF usually has a high dimensionality. Using the manifold learning methods can efficiently reduce the dimensionality and thus make

the BIF more efficient. Further, Guo et al. [48] showed that the manifold learning methods can increase the age estimation accuracies over the pure BIF. As a result, the combination of BIF and manifold learning can take advantages from each other and becomes a useful representation for facial aging representation.

**A Compositional and Dynamic Model:** Face images can be represented at different level-of-details in a hierarchical fashion. For example, the first layer is the whole face, the second layer could be facial parts, such as eyes, nose, and mouth, and the third layer could be the skin details and wrinkles. Xu et al. [99] investigated this idea in detail and derived a general method for face modeling. In order to combine the different levels and consider different individuals together, they used a so-called AND-OR graph which is a top-down tree structure. The root node of the tree is the whole face, while the second and third layers consist of the local parts and facial details. The AND operation will composite all these nodes to form the whole face. The appearances of different individuals form alternatives for each facial parts or details, and combined with an “OR” operation. Suo et al. [86] adapted this hierarchical compositional face model to model face aging, and call it a compositional and dynamic model, especially focusing on the dynamic wrinkle variations with aging. This model can synthesize new face images at different ages as well as age estimation for a given face image. Please notice that the compositional and dynamic model is complex, and it is not easy to implement the whole model without sufficient knowledge in each step. Another issues is the need to localize each facial part precisely, which is not trivial in practice.

**Local Spatially Flexible Patches:** Rather than using a compositional model as in [86] to organize facial parts and details, local facial patches can be used directly for facial age estimation. Yan et al. [101, 105] used facial patches to represent face images for age estimation. They call it local spatially flexible patches, which encodes both the facial patch appearance and its location information,  $(u, v)$ . When all facial patches are extracted with overlap, the cost for face representation might be high in comparison with the compositional model [86].

**Other Methods:** In addition to the typical approaches presented above, there are also some other face representations for age estimation. Hayashi et al. [51] computed wrinkles and facial geometry to represent aging faces. Fukai et al. [31] took the fast Fourier transform (FFT) and genetic algorithm (GA) for feature extraction and selection for aging face representation. Gunay and Nabihev [40] and Yang and Ai [106] adopted the popular local binary pattern (LBP) method which has become a standard face encoding method for face recognition [4]. Gao and Ai [34] used the Gabor filters for feature extraction and showed its superb performance over the LBP features in age classification.

**A Summary on Facial Aging Representation:** A number of facial representations have been developed for age estimation. Some are adapted from the methods that were originally proposed for object recognition or face recognition, while some others are specifically related to aging, e.g., wrinkles and facial anthropometry. Depending on the application scenarios, each representation may have its advantages as well as disadvantages, as we discussed above. For practical use in business intelligence, it is demanding to perform a comprehensive evaluation of the face

aging representations using a *common testing strategy* and *standard evaluation protocol*.

Given the facial aging representation, the next step is to learn the aging function for age estimation.

## 2.2 Aging Function Learning

Given a face image  $I$  with the corresponding age  $y$ , we know how to represent the face using various methods and models presented in Section 2.1. Denote the derived face representation by  $X$ , now the problem is how to obtain an aging function that builds the relationship between  $X$  and  $y$ . Let us use  $y = f(X)$  to indicate the relation, and  $f$  is the so-called aging function.

It is not trivial to write the aging function explicitly, and even not clear what form the aging function could be. Lanitis et al. [58] proposed to use a quadratic function for aging. But there is no evidence to show why the aging function is quadratic. We believe that the aging function is not a simple polynomial or exponential function, considering a large span of ages, e.g., from 0 to 90 years. To alleviate the difficulty in deriving and proving a specific function which is proper for aging, a practical approach is to learn the aging function from data. The learned aging function is just a mapping from  $X$  to  $y$ , and we may not know the explicit form of the mapping. For example, when the support vector machine (SVM) is used with a radial basis function (RBF) as the kernel, we cannot know the specific function form or the order of aging function. To learn the aging function, many different methods can be used, which can be categorized into three broad categories: classification, regression, and a hybrid approach.

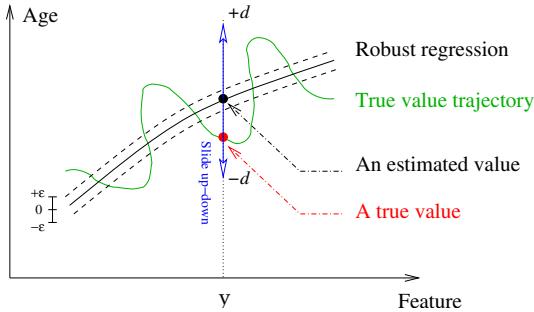
**Classification:** When we consider each age label as one class, the aging function can be learned by classifiers. Then age estimation can be viewed as a classification problem. Many classifiers developed in the machine learning society can be adopted, such as the artificial neural networks (ANN) [6] and SVMs [91]. Lanitis et al. [58] used the nearest neighbor classifier and ANN for age estimation, based on the AAM face representation. They also used a quadratic function for aging mapping. In their experiments performed on a relatively small database of 400 images at ages ranging from 0 to 35 years, they reported average error of 5.04 years using the quadratic aging function, which is slightly lower than the nearest neighbor classifier, but higher than the ANN. Guo et al. [44, 42] applied the SVM for age estimation, based on the age manifold for face image representation. They performed age estimation on a large database of 8,000 images and reported a the mean absolute errors (MAEs) of 5.55 and 5.52 years for females and males, respectively. They also reported an MAE of 7.16 years on the FG-NET [25] aging database using the AAM face representation.

**Regression:** Since the age labels are continuous numbers, e.g., 0, 1, 2, ..., age estimation can also be viewed as a regression problem. Various regressors might be used to learn the aging function  $f$ . Lanitis et al. [58] used a quadratic function for regression. The face representation  $X$  is derived from the AAM models. Fu et al.

[30] applied a multiple linear regressor to learn the aging function where  $X$  is the age manifold obtained by using the manifold learning method OLPP [9]. Guo et al. [42, 44] applied the support vector regression (SVR) [91] method for aging function learning. The input  $X$  is also the age manifold learned by the OLPP method. They performed age estimation on the public FG-NET database [25] and the large YGA database. A nice property of the SVR method is that it can deal with outliers in age labeling. The SVR optimization is based on the non-linear programming technique that has the capability to deal with outliers using slack variables. Later one, Luu et al. [61] also used the SVR method combined with the AAM models for age estimation. Yan et al. [103] employed a semi-definite programming (SDP) technique to learn the regression function. The SDP formulation is nice but the optimization usually has a high computational cost. Yan et al. [102] extended their work by using an Expectation-Maximization (EM) [23] optimization method to speed up the learning process and reduce the age estimation errors. Ni et al. [67] proposed a regression method called robust multi-instance regression (RMIR) to deal with age label outliers in regression. Their database was collected from the Internet and manually labeled later, the labeled ages may not be accurate compared with the true ages, which can be considered as label outliers. Xiao et al. [97] used metric learning to find latent semantic similarity between ages for SDP regression. Chen et al. [15] compared different methods among Least Angle Regression (LAR), Principle Component Analysis (PCA), and Locality Preserving Projections (LPP), and select the best one for age estimation.

**Hybrid:** We have seen that the aging function can be learned by either classifiers or regressors. A natural question you may ask is – Which is better for age estimation: classification or regression? To get an answer, one may compare the two different approaches empirically on multiple databases. Guo et al. [44, 42] studied this problem empirically on two databases: the FG-NET and YGA databases. They chose the SVMs as the classifier and the SVR as the regressor, using the same input  $X$  that was derived by manifold learning [9] or AAM [20]. From their experiments, the SVMs perform much better than the SVR on the YGA database (5.55 versus 7.00, and 5.52 versus 7.47 years, for females and males, respectively), while the SVMs perform much worse than the SVR on the FG-NET database (7.16 versus 5.16 years). From this experimental comparison on different databases, one can understand that there is no unique aging function for age estimation. Depending on databases, the aging function can be learned using either classifiers or regressors.

Since you have already observed that the SVMs performed much better than the SVR on the YGA database while worse on FG-NET, you may have a thought: How about combining the two approaches to take advantage of each other? Guo et al. [44, 42] explored this idea and proposed a method, called locally adjusted robust regressor (LARR). They showed that a consistently better performance can be obtained by combining classifiers with a regressor. Using the LARR scheme, the MAEs can be 5.25 and 5.30 years for the female and male on YGA, and 5.07 years on FG-NET, respectively. Specifically, the LARR method first performs a regression using all available data and then uses the regression results to guide the classifiers using a small local search range, as shown in Fig. 6. Since the global regression



**Fig. 6** Illustration of the idea of locally adjusted robust regression, originally shown in [42].

provides a constraint for local search, the LARR method improves the age estimation performance significantly. Theoretically the LARR method provides a nice way to combine classifiers with a regression, but practically it cannot determine how big the local search range could be. To determine the local search range automatically in a data-driven manner, Guo et al. [43] proposed a probabilistic approach to combine regression and classification results. The idea is to transform or interpret the regression results into probabilities using a uniform distribution, and then the probabilities from the classification result are cut off by the uniform distribution.

In sum, age estimation can be approached by either classifiers or regressors. One can say that age estimation is not purely a classification or regression problem. A promising approach is to combine regression and classification methods as demonstrated in [44, 42, 43]. It is worth investigating more advanced schemes in combining classifiers and regressors for aging function learning, and better performance might be expected in age estimation.

### 2.3 Age Group Classification

When talking about age estimation, usually we consider continuous integer values for ages, e.g., 0, 1, 2, …, and the task of age estimation is to determine the specific age of a person, i.e., 5 or 60 years old. In some applications, however, one may just want to know the age groups, such as babies, adults, or senior people. We call this “age group classification.”

Age group classification can be approached in different ways: (1) a pure group classification problem with a limited number of groups, e.g., young, adult, and senior, and (2) performing the regular age estimation and then reporting the age group information based on an estimated age. For pure group classification, the task is simpler since there is only a small number of groups, e.g., three or more, to consider. The problem can be solved using standard classifiers. Further, the task of age labeling is simpler since there is no need to know the specific age of a face. In practice, if one wants to collect a learning database from the Internet, he or she can collect the face images first, and then label the age for each face by just providing the age group information. On the other hand, if the “regular” age estimation

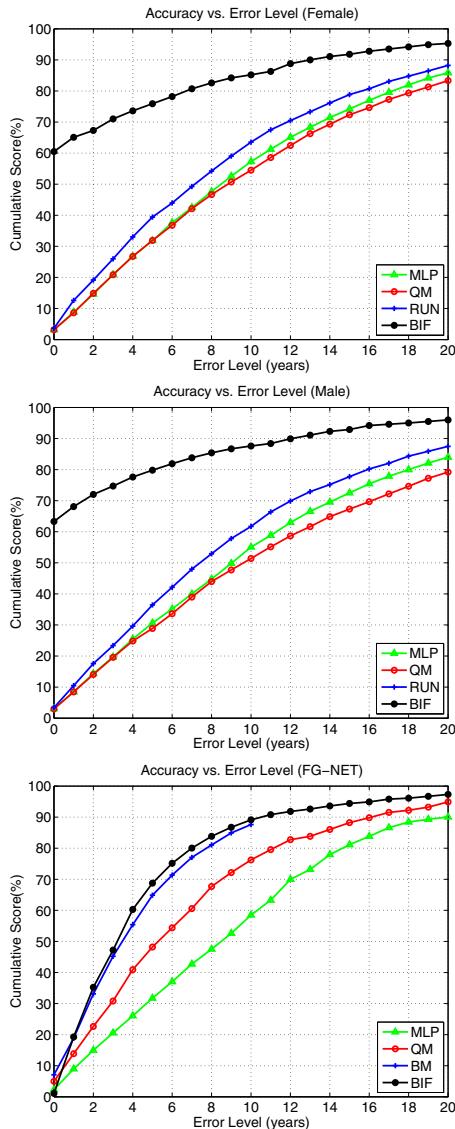
problem can be solved, the problem of age group classification will be trivial. Since the age groups can be determined easily given the estimated ages.

Generally speaking, age estimation is harder and more complex than age group classification. In age group classification, only classifiers can be used, while in age estimation, various methods can be adopted, including classifiers, regressors, or even a combination of classifiers with a regressor [44, 42, 43].

On the other hand, Guo et al. [48] showed that the age estimation accuracy can be improved if the age estimation follows an age and gender group classification module. When gender and age are considered together in group classification, we may have multiple groups, such as “young male,” “young female,” “male adult,” “female adult,” “senior male,” and “senior female.” It was showed in [48] that the features used for age and gender group classification can be different from the features used for age estimation.

In the literature, there are published works that only deal with age group classification. For example, An earliest work by Kwon and Lobo [57] presented a three-age-group classification system. They used an anthropometric model, i.e. ratios of distances between facial landmarks to separate babies from adults, and a wrinkle model to recognize senior people. Ueki et al. [90] built 11 Gaussian models in a low-dimensional feature space. A test face image is classified into one of the age groups based on the likelihood to each of the learned Gaussian models. They computed the classification accuracies in terms of different age ranges. Kanno et al. [56] dealt with a 4-class age group classification problem using the artificial neural networks (ANN).

Recently, Gallagher and Chen [32] analyzed images of groups of people using social context latent in a picture. They investigated both image content and social context for age and gender classification, and collected a database of people’s photos for evaluation. The authors manually labeled age group information (seven age groups: 0-2, 3-7, 8-12, 13-19, 20-36, 37-65, 66 or above), since no ground truth of age is available in data collection. It was reported that a gender classification accuracy of 66.9% was reached using the social context, 69.6% using image content, and 74.1% based on the combination of both sources of information on a large database that they collected. Their age group classification accuracies are 32.9%, 38.3%, and 42.9%, based on social context, image content, and a combination of both, respectively. Shan [82] used Local Binary Patterns (LBP) and Gabor features as face representation, AdaBoost for local feature learning, and SVM for age group classification. The database in [32] was used to evaluate the classification performance, and a result was obtained better than the original approach presented in [32]. Gao and Ai [33] used Gabor features with a fuzzy version of the linear discrimination analysis (LDA) for facial age classification on consumer images. Four age groups were considered: baby, child, adult, and elder people.

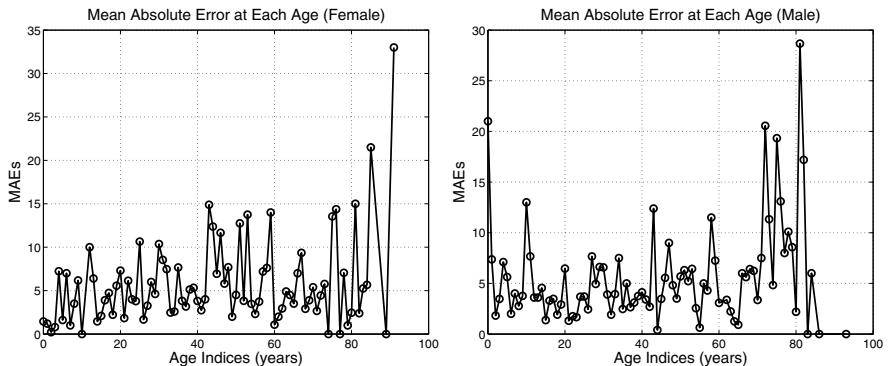


**Fig. 7** Cumulative scores of some typical approaches on the YGA and FG-NET databases (from [49]). Four methods are compared: MLP is the multi-layer perceptron (ANN) [58], QM is the quadratic modeling function [58], RUN is the regression with uncertainty [102], and BIF is the biologically-inspired feature approach [49].

## 2.4 Performance Evaluation for Age Estimation

Two quantitative measures are usually used to evaluate the age estimation performance. One is called the mean absolute error or MAE and another is called cumulative score or CS [58, 59, 36]. The MAE is defined as the average of the absolute errors between the estimated ages and the ground truth, i.e.,  $\text{MAE} = \sum_{k=1}^N |\hat{l}_k - l_k|/N$ , where  $l_k$  is the ground truth age for the test image  $k$ ,  $\hat{l}_k$  is the estimated age, and  $N$  is the total number of test images. The CS is defined as  $\text{CS}(j) = N_{e \leq j}/N \times 100\%$ , where  $N_{e \leq j}$  is the number of test images on which the age estimation makes an absolute error no higher than  $j$  years.

These two measures are usually used together to measure the age estimation accuracy. The CS measure draws curves to visualize the age estimation accuracies at different acceptable error levels, while the MAE is a specific number of years of estimation error. Some examples of CS curves from [49] are shown in Fig. 7.



**Fig. 8** MAEs per age for females and males, originally shown in [42].

Besides the often used MAE and CS curves, there are some other measures that might be used to get more details about the age estimation performance. One is the so-called “MAE per age,” which measures the mean absolute error at each specific age, e.g., 5 or 70 years. Usually it is better to draw a figure to show the MAE per age results, rather than using a table presentation, since the table could be very large containing many items (one average error per age, given a large span of ages). For example, Guo et al. [42] used the MAE per age measure to present more details in age estimation, as shown in Fig. 8. A nice property of this measure is that it allows a better understanding of an algorithm’s performance at each age, not just the overall performance, or it can show a detailed comparison between various approaches by displaying which method performs better at which age.

Another detailed measure is called “MAE per decade,” which is something between the overall MAE and “MAE per age.” It computes the mean absolute error for every ten years instead of per year. Usually the measure of MAE per decade is

presented by using a table, e.g., in [105, 49], although a bar graph display might be used as well.

To obtain performance measure, one needs to divide the database into training and test sets. The training set may also contain a small part for tuning parameters. Usually, the leave-one-person-out (LOPO) test scheme is used on FG-NET [36, 102, 42], and a 4-fold cross-validation scheme on YGA [102, 42]. On MORPH, a random selected set of about 10,000 images are used for training, and the remaining about 40,000 images for testing [45, 46], considering the memory issue.

In age group classification, usually the classification accuracies are measured and reported, since it can be viewed as a standard pattern classification problem.

## 2.5 Aging Databases

It is quite difficult to develop a facial aging database in comparison with many other face databases. The difficulty comes in two aspects: (1) face image capture with a large span of age variations, and (2) age labeling for each face. In face image acquisition, one has to recruit many people with a large range of ages, e.g., from babies to adults and to very old people. In age labeling, it is better to let the person whose faces are captured tell his or her true ages at the capture time. Since it is hard for other people to label the accurate age for each face afterwards. Further, if the captured person tells a wrong age, it is not trivial to find and correct the mistake after he or she left. Finally, even if a face database could be collected from the Internet containing a large span of ages, it is still very hard to label the groundtruth ages.

When considering an aging database, the first impression one may have is that each individual's face images have to be captured at different ages. The first public available aging database, FG-NET [25] was developed based on this consideration. One can imagine how difficult it is. In order to capture an individual's faces at different ages, there are three cases to consider: (1) if the person is young currently, we have to take many years to capture his or her faces at older ages; (2) if the person is very old now, we may request his or her photos that were captured before; and (3) if the person is a middle-aged, we have to request the photos years ago and wait for years to get his or her "future" face images. The FG-NET aging database contains many images scanned from people's photos years ago resulting in low qualities and uncontrolled variations.

Realizing the difficulties to cover a large span of years for each individual, recent aging databases capture just one face for each individual at the current age or a small range of age span for each individual. The private YGA database [103] captures each individual at a single age, the MORPH database [75] captures many individuals (more than 10,000 in Album II), but each individual has only a small number of face images at the same age or different ages. The LHI face database [86] has only one face image per individual at a single age. It makes the data collection easier by requiring only a small number of ages or a single age for each individual. However, the database captured in this way might not be good to study the age progression for each individual, since different people age differently, and there is no ground truth to verify whether an age progression model is correct or not.

There are other aging databases available, please see [28] for a list of them. For age estimation, a good database should have a balanced number of faces at each age or at least each age group within 5 to 10 years. Further, the aging database should have a large span of ages. Currently, the FG-NET and MORPH are the most popular databases that are public available for age estimation study, although both are not perfect in data capture and some other databases are available. The FG-NET contains faces mostly at young ages although the age range is from 0 to 69 years, while the MORPH database has no face below 16 years or above 68 years. It is still demanding to have a large database with a balanced distribution of individuals, age range, gender, and even ethnicity. Industrial or business people could contribute for this.

## 2.6 Other Traits

Age estimation is usually based on face regions, while other traits or cues might be used to determine a person's age. For instance, the white or gray hair may indicate a senior person. Human height may be another cue for age, e.g., young people are shorter than adults. Recent studies show that the height can be estimated from still images and videos, see Chellappa and Turaga [14] and the references therein. We expect new traits to be explored for age determination in the near future.

After completing age estimation, the next topic is about sex classification.

## 3 Sex Classification

Sex classification is a two-class classification problem: male and female. In this sense, the problem of sex classification is simpler than age estimation. The simplicity can be further understood in another two aspects: (1) data collection – it is seldom to design and develop a large face database that is specifically used for sex classification. Usually, the face recognition databases are adopted, e.g., the FERET database [70], for sex classification; and (2) sex labeling – the male and female labels can be annotated relatively accurately and quickly by a third person without difficulty after the image acquisition.

Historically, many researchers used and are still using “gender recognition” or “gender classification.” We use “sex classification” to focus on the biological aspect of sex, not a social role of gender. In describing previous approaches, however, sometimes we mention gender based on the original work.

In this section, I introduce face-based sex classification approaches, which include frontal face based and arbitrary view based. I will also introduce body-based sex classification. Finally I discuss the performance measure for sex classification.

### 3.1 Frontal Face Based

Most existing sex classification approaches use frontal view face images. Earlier works applied the artificial neural networks for sex classification. For example,

Golomb et al. [37] built the SEXNET for sex classification and reported an accuracy of 91.9% obtained from the experiments on 90 photos of 45 males and 45 females. This is probably the first computational approach to sex classification. Brunelli and Poggio [8] used the HyperBF networks for gender classification with an accuracy of 79% experimented on 168 images of 21 males and 21 females, Gutta et al. used a hybrid classifier combining the artificial neural networks and decision trees and reported an accuracy of 96% on the FERET database [70], a popular face recognition database.

Later on, the SVM classifier [91] became a dominating method in gender recognition. For example, Moghaddam and Yang [64] used the non-linear SVM for gender recognition on the FERET database with an accuracy of 96.6% on 1,755 images of 1,044 males and 711 females. The feature is just the raw face images with reduced sizes. Graf and Wichmann [38] used dimensionality reduction methods (e.g., PCA and LLE) with the SVM classifier. Jain and Huang [53] used independent component analysis (ICA) and linear discriminant analysis (LDA) with the SVM for gender classification. Costen et al. [21] presented an SVM-like sparse regularization function for gender classification.

AdaBoost [26] is another effective classifier for gender recognition. For example, Shakhnarovich et al. [80] reported a gender recognition system using web-collected face images. Their sex classification framework is about the same as the face detector [92] with rectangle filters on integral images for feature extraction and the AdaBoost for feature selection and classification. Baluja and Rowley [5] used simple features (intensity comparisons on neighborhood pixels) with the AdaBoost method and reported an accuracy of 94.4% on 2,409 face images of 1,495 males and 914 females from FERET. Wu et al. [95] proposed to use a Look Up Table (LUT) based AdaBoost for gender classification. Xu and Huang [98] described a SODA-Boosting method (a variant of the AdaBoost) for gender classification. They reported an accuracy of 92.82% on 4,109 face images (703 males and 498 females) from FERET database. Yang and Ai [106] used the local binary patterns (LBP) feature with the AdaBoost classifier for age, gender, and ethnicity classification. An accuracy of 93.3% was reported on 3,540 face images from FERET. Sun et al. [85] used the genetic algorithm (GA) for optimal feature selection in gender classification.

Makinen and Raisamo [62] performed an evaluation of various gender classification methods using two databases, the IMM [84] and the FERET [70] face databases. Based on extensive experiments, they found that the SVM with  $36 \times 36$  face image size as input achieved the best gender classification rate. Further, facial alignment is important for sex classification. A good alignment of facial features could improve the sex classification accuracy.

### 3.2 Arbitrary View Face

Although most sex classification approaches focus on frontal view faces, there are some works using multiple or arbitrary view face images. For instance, Toews and Arbel [89] attempted to recognize sex in face images from arbitrary views. The

main idea in [89] is to use scale-invariant feature transform (SIFT) [60] feature that is also affine invariant locally. They showed a probabilistic modeling in using the SIFT feature so that face detection, localization, and sex classification can be integrated together. Experimentally the sex classification accuracy from arbitrary views is lower than the frontal views. Wu and Lu [96] built a hierarchical classifier to deal with faces at different views for gender classification. The face images are first classified into different views based on an angle classifier, and then the sex classification is performed in that corresponding angle.

Shan [81] presented a sex classification study on real-world face images, using the LFW face database [52] which is a recent face database collected from the Internet and usually used for studying face recognition or verification in real world face images. There are pose variations together with other variations, such as illumination, facial expression, indoor/outdoor, eyeglass or without, etc. Shan [81] used the LBP features [4] and AdaBoost [26] for feature selection, and then the SVM [91] for sex classification. A high accuracy of 94.4% was reported on the LFW database.

### 3.3 Other Modalities

Face images are the major modality for sex classification. Besides faces, there are other modalities that can also be used for sex recognition. In recent studies, researchers used the human body for sex classification. Cao et al. [11] presented an early work for sex recognition from the body rather than face. They used the histogram of oriented gradient (HOG) [22] feature for body image representation, which is a popular feature for pedestrian detection, and the AdaBoost method [26] for classification. They reported a classification accuracy of about 75% on the MIT pedestrian database [68], which is a popular database for pedestrian detection, containing mixed body poses of frontal and back views.

Guo et al. [47] adapted the biologically-inspired features [76, 78] for body image representation and used various manifold learning methods for dimensionality reduction, and then the SVM [91] for body-based sex classification. They reported a classification accuracy of about 80% on the MIT pedestrian database, which is higher than using the HOG feature. More interestingly, they found that the back view of the body delivers higher accuracy than the frontal view. They also proposed a view classification method preceding the sex classification module to develop an automated sex classification system.

In reality, one may find other modalities for sex classification besides the face, such as the body [11, 47] and even the gait [83]. However, the face-based sex classification usually gives the highest classification accuracy based on current experimental results. The use of body has lower accuracy than the face, e.g., 80% vs. 95%. In real world applications, using body images for sex classification has some advantages over faces, however. For example, the body images can be used in back views while faces cannot, the body images can be viewed at a distance where the faces might have too low resolutions to recognize. A complete sex recognition system may combine different modalities to deal with various situations.

### 3.4 Performance Measure for Sex Classification

Sex classification performance is usually measured by the classification rate [62]. Sometimes, people reported the classification error rate which is essentially equivalent to 100% minus the recognition accuracy. In our opinion, other measures could also be used. For instance, the receiver operating characteristic (ROC) can be used, especially when the number of male and female faces is very unbalanced, either in training or testing. One can also use confusion matrix to measure the sex classification performance, which contains true positive (TP) and true negative (TN) rates.

## 4 The Mutual Influence of Age Estimation and Sex Classification

Traditionally, age estimation and sex classification are two separate and independent problems. One reason is because of the issue in data collection. It is seldom to develop a database that is used uniquely for sex classification. Usually the face recognition databases are adopted where the captured individuals are mainly adults. Few faces come from young children or senior people in traditional face recognition databases. As a result, the traditional sex classification approaches have no consideration of aging variation. On the other hand, it is difficult to collect a large aging database in early aging studies. For example, the popular FG-NET aging database [25] is relatively small, which limits the study of influencing factors to aging, e.g., sex difference or other factors.

Guo et al. [48, 45] studied the influence of sex difference on age estimation quantitatively on a large database with a balanced number of males and females. Guo et al. [41] also studied the sex classification rate with respect to the age group differences.

To study the relation between age estimation and sex classification, Guo et al. [48, 41] used the Yamaha Gender and Age (YGA) database to study the aging effect on sex classification. The YGA database is usually used to evaluate the performance of age estimation. It contains 8,000 face images taken from 1,600 subjects (800 females and 800 males) captured outdoors with half males and half females, in the age range from 0 to 93 years.

### 4.1 Effect of Sex Difference on Age Estimation

To study the performance of age estimation with respect to sex classification, Guo et al. [48] used the YGA database which has a balanced number of male and female faces in each age group.

They studied the problem systematically: *How much can age estimation performance be affected by sex difference?* In order to get a quantitative result, they compared age estimation errors under two situations: case I, age estimation is performed on all faces without discriminating between males and females; and case II, age estimation is performed for males and females separately, with the assumption that

the sex is known for each face. They compared age estimation errors in cases I and II using a variety of facial representations. The basic representation is to learn the age manifold from raw face images using the OLPP method, which has shown good performance for age estimation [30] [44] [42] [29]. They reported that the MAE is 7.04 years when the OLPP was applied to all training face images, while the MAEs are 5.24 and 5.69 years when it was applied to females and males separately. The average of the MAEs over females and males is 5.47 years, which is reduced from 7.04 years by 22.3%. This large reduction of error shows that age estimation is affected by the sex difference significantly.

They also proposed a framework to deal with sex difference in age estimation. For example, one of their methods is to perform age and sex group classification using a proper face representation, and then perform age estimation within the classified age and sex group. In this way, they reported a small MAE on age estimation [48], much better than previous approaches to age estimation. More recently, Guo and Mu [45] studied quantitatively the influence of sex and ethnicity on age estimation using the MORPH database [75]. Significant influence was found as that using the YGA database [48].

## 4.2 Sex Classification with Respect to Aging

Guo et al. [41] divided the 8,000 faces in the YGA database into three age groups: (1) “young” for ages from 0 to 19 years, containing 1,000 males and 1,000 females; (2) “adult” for ages 20 to 60, with 2,050 males and 2,050 females; and (3) “senior” for ages 61 to 93, with 950 males and 950 females. In each age group, there are half males and half females. The number of males and females in each age group is shown in Table I.

**Table 1** Numbers of male and female faces in the three age groups of the YGA face database: young, adult, and senior. It is from [41].

	Young (0-19)	Adult (20-60)	Senior (61-93)	All Ages (0-93)
Male	1,000	2,050	950	4,000
Female	1,000	2,050	950	4,000
Both	2,000	4,100	1,900	8,000

To study the aging effect on sex classification, Guo et al. [41] performed sex classification in each age group separately using the same “Raw+SVM” approach with two-fold cross validation. They found that the sex classification rate is 94.56% for “adults,” but 84.38% for “young” people, and 85.32% for “seniors.” Clearly, the recognition accuracies are very different for different age groups. It is about 10% difference between the “adult” and “young” or between the “adult” and “senior.” They also performed sex classification on the whole database without age group

discrimination. The accuracy is 89.28%, which is lower than that in the adult group, and is also lower than many previously reported results in sex classification (see Section 3.1). This experimental evaluation demonstrates sex classification rates can be very different in different age groups. They also verified other face representations, including LBP, HOG, and BIF, for sex classification in each age group, as shown in Table 2, and found similar results. For classifiers, the linear and non-linear SVMs are used to show the difference in sex classification. Based on the extensive experiments, Guo et al. [41] showed that sex classification is really influenced by people's ages.

**Table 2** Sex classification using different representations (raw pixels, LBP, HOG, and BIF) and different classifiers (the linear SVM (L-SVM) and non-linear SVM (N-SVM)). It is from [41].

Methods	Young (0-19)	Adult (20-60)	Senior (61-93)
Raw + L-SVM	78.59%	89.91%	81.17%
Raw + N-SVM	84.38%	94.56%	85.32%
LBP + L-SVM	79.76%	92.65%	87.55%
LBP + N-SVM	81.93%	94.96%	90.64%
HOG + L-SVM	75.83%	88.00%	77.13%
HOG + N-SVM	86.44%	94.03%	89.04%
BIF + L-SVM	83.01%	94.22%	91.81%
BIF + N-SVM	87.13%	96.03%	92.34%
Average(L-SVM)	80.52%	91.20%	84.42%
Average(N-SVM)	84.97%	94.90%	89.34%

Based on the aging effect on sex classification, Wang et al. [93] recently developed a method to deal with the problem of gender classification on infants and senior people. They used a Locality Preserving Projection (LPP) method combined with a Random Forest (RF) learning to perform gender classification on FG-NET [25], and compared with human performance. Chang et al. [12] used a similar approach but with the incorporation of a Genetic Algorithm (GA) [65] for feature selection to process gender classification containing different age groups.

### 4.3 Practical Value

Guo et al. [48, 41, 45] showed that age estimation is affected by sex difference, and sex classification rates are related to age groups. As a result, there is a mutual influence between age estimation and sex classification. So these two problems are actually related and interwound.

The studies of the mutual influences [48, 41, 45] can provide a practical guide in a real system design for business intelligence. A working system should take into

account the mutual influence in order to be robust and have high accuracies for both age estimation and sex classification. Recently, Wang et al. [93] and Chang et al. [12] have worked on sex classification with aging variations. On the other hand, the studies by Guo et al. [41, 45] indicate that the sex classification rates can be very high for adults, e.g., about 95% or even higher. So, the aging effect could be ignored when only working on face images of adults.

## 5 Major Challenges and Future Research Directions

Human aging is a complex process. Facial aging is determined by not only the gene but also many external factors, such as health, living style, living location, and weather conditions. Males and females may age differently [48]. People belonging to different ethnic groups may also have different aging process [45]. In sex classification, it is still hard to recognize sex from faces of young and elderly people [41], although significant progresses have been made for sex classification in adult faces. Given the current understanding of age estimation and sex classification and the state-of-the-art results, some challenges can be identified and presented here:

**Age and Sex Databases.** The progress of computational visual processing depends heavily on the data. The research results rely on what kinds of databases one can access. Currently, the FG-NET aging database [25] is relatively small, the MORPH database [75] (Album II) is large with many individuals, but the ages are just from 16 to 68 years without enough young and elderly people. Further, the numbers of males and females are not balanced. A good database for age estimation should consider the sex balance [48], and can provide a benchmark for both age estimation and sex classification, since these two problems are very related. Another human attribute is ethnicity which needs to be taken into account in age estimation [45]. The private YGA database [103] considers both age and sex, but there is only one ethnic group.

Considering the external factors like health, living style and location, and weather conditions, one needs to collect an age and sex database covering different geolocations and people with different health conditions to avoid the bias in data collection.

**Age Labeling.** Considering the difficulty to recruit many individuals to collect a large age and sex database, another way is to collect the data from the Internet. The nice property is that one can collect a very large database of faces under many different image capture situations (e.g., indoor/outdoor, face pose change, facial expression change, illumination variation, ...). However, a new problem is raised: how to label the ground truth of ages? Unlike facial identity or facial expressions, it is really hard to label the true age of a face by a third person. Ni et al. [67] attempted to collect a face database from the Web, and proposed a method to allow outliers in labeling, but the ground truth of ages is still needed in order to evaluate the age estimation results. Researchers may explore new approaches to label the facial ages in the wild.

**Age Estimation in Moving Faces.** So far, most age estimation approaches deal with still face images. How to extend the current works to moving faces? One can use a video camcorder to capture moving faces. In video capture, the motion blur may be significant depending on how fast a person moves. In addition to body movement, the person may look at various directions and may talk on a cell phone (e.g., in a shopping mall). All these factors will influence the age estimation performance. So the challenges here include: How to deal with the motion blur? How to deal with the pose variations or non-frontal views of faces? How to deal with facial occlusions (occluded by the hand or a cell phone)? Is it possible to use a talking face with expression changes for age estimation?

On the other hand, one may think about: Can we use the motion information to help age estimation? Or can we extract other useful information from moving faces for age estimation?

Further, it is also a great challenge for sex classification in moving faces. All the challenges raised for age estimation are applied to sex classification as well.

**Influence Factors.** It is well-known that face recognition is influenced by head pose change, illumination variation, and facial expression. The same influence applies to face-based age estimation and sex classification. It is still an unsolved problem for face recognition in dealing with these variations, although significant progress has been made in recent years [13]. Age and sex recognition also needs to consider these influencing factors. Please notice that, in face recognition, there exists a large number of face databases that can be found from the Face Recognition Homepage [1], and thus can be used to study effective algorithms to deal with the influences. However, there are few databases that are publicly available for studying the influences on age estimation.

In addition to pose, illumination and expression, age estimation is also influenced by other factors, such as sex and ethnicity [48, 45]. On the other hand, sex discrimination is influenced by age groups [41]. Recently, Guo and Mu [46] proposed to recognize age, sex, and ethnicity jointly based on a partial least squares (PLS) formulation, in order to deal with the mutual influence in age estimation. In sum, age estimation and sex classification are influenced by many factors. New research efforts are needed to perform extensive exploration and find good solutions to deal with the influence factors.

**Developing a Robust Age Estimation System in Practice.** Currently the age estimation and sex classification studies are mainly in academia research. We are eager to see real applications in industry and business. Recently, some commercial products are being developed for age estimation and sex classification. For instance, NEC develops an ultra-compact sensor that estimates age and gender [3], Intellio builds a system, called VisiScanner [2], for custom analysis including age and sex. In real applications, new problems may be found and then new research can be inspired. Hopefully industrial and business applications can provide new databases to challenge the academia researchers. The eventual goal is to develop a robust age estimation and sex classification system that is comparable to or surpass human capabilities.

## 6 Concluding Remarks

In this chapter, the state-of-the-art approaches to human age estimation and sex classification are described and discussed. Age estimation and sex classification are related problems and can have mutual influence especially on a large database. The main trait for age estimation and sex classification is the face, but other modalities may also be used. The body-based sex classification has been discussed, which may provide reasonable recognition accuracies when face images are unavailable or not useable. There are still many challenges towards the goal of developing a robust system to do age estimation and sex classification for business intelligence. Main challenges have been identified which may inspire new investigations in the near future.

## References

1. Face Recognition Homepage, <http://face-rec.org/>
2. Intellio visiscanner retail customer analytics solution, <http://www.intellio.eu/visiscanner.php>
3. NEC develops an ultra-compact sensor that estimates age and gender, <http://tweetbuzz.jp/entry/56752845/www.nec.co.jp/press/en/1105/3101.html>
4. Ahonen, T., Hadid, A., Pietikäinen, M.: Face Recognition with Local Binary Patterns. In: Pajdla, T., Matas, J.(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 469–481. Springer, Heidelberg (2004)
5. Baluja, S., Rowley, H.A.: Boosting sex identification performance. *Intl. J. of Comput. Vision* 71(1), 111–119 (2007)
6. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
7. Bruce, V., Burton, A., Hanna, E., Healey, P., Mason, O.: Sex discrimination: How do we tell the difference between male and female faces? *Perception* 22, 131–152 (1993)
8. Brunelli, R., Poggio, T.: Hyperbf networks for gender classification. In: Proc. DARPA Image Understanding Workshop, pp. 311–314 (1992)
9. Cai, D., He, X., Han, J., Zhang, H.: Orthogonal laplacianfaces for face recognition. *IEEE Trans. on Image Processing* 15, 3608–3614 (2006)
10. Cai, D., He, X., Zhou, K., Han, J., Bao, H.: Locality sensitive discriminant analysis. In: Proc. Int. Joint Conf. on Artificial Intell. (2007)
11. Cao, L., Dikmen, M., Fu, Y., Huang, T.: Gender recognition from body. In: ACM Multimedia (2008)
12. Chang, Y., Wang, Y., Ricanek, K., Chen, C.: Feature selection for improved automatic gender classification. In: IEEE Workshop on Computational Intelligence in Biometrics and Identity Management (CIBIM), pp. 29–35 (2011)
13. Chellappa, R., Sinha, P., Phillips, P.: Face recognition by computers and humans. *IEEE Computer* 43(2), 46–55 (2010)
14. Chellappa, R., Turaga, P.: Recent advances in age and height estimation from still images and video. In: IEEE Conf. on AFGR (2011)
15. Chen, C., Chang, Y., Ricanek, K., Wang, Y.: Face age estimation using model selection. In: IEEE CVPR Workshop, pp. 93–99 (2010)

16. Christensen, K., Doblhammer, G., Rau, R., Vaupel, J.: Ageing populations: the challenges ahead. *Lancet* 374, 1196–1208 (2009)
17. Christensen, K., Johnson, T., Vaupel, J.: The quest for genetic determinants of human longevity: challenges and insights. *Nature Reviews Genetics* 7, 436–448 (2006)
18. Christensen, K., Thinggaard, M., McGue, M., Rexbye, H., Hjelmborg, J., Aviv, A., Gunn, D., Ouderaa, F., Vaupel, J.: Perceived age as clinically useful biomarker of ageing: Cohort study. *British Medical Journal* 339, b5262 (2009)
19. Comon, P.: Independent component analysis: A new concept? *Signal Processing* 36(3), 287–314 (1994)
20. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active Appearance Models. In: Burkhardt, H., Neumann, B. (eds.) *ECCV 1998. LNCS*, vol. 1407, pp. 484–498. Springer, Heidelberg (1998)
21. Costen, N., Brown, M., Akamastu, S.: Sparse models for gender classification. In: *IEEE Int'l. Conf. on Automatic Face and Gesture Recognition* (2004)
22. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: *IEEE Conf. on Comput. Vision and Pattern Recognit.*, pp. 886–893 (2005)
23. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society, Series B* 39(1), 1–38 (1977)
24. Farkas, L.: *Anthropometry of the Head and Face*. Raven Press, New York (1994)
25. FGNET: The fg-net aging database (2002), <http://www.fgnet.rsunit.com/>
26. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: *Proc. the Thirteen International Conference on Machine Learning*, pp. 148–156 (1996)
27. Fu, Y., Guo, G.D., Huang, T.S.: Soft biometrics for video surveillance. In: Ma, Y., Qian, G. (eds.) *Intelligent Video Surveillance: Systems and Technology*. Taylor and Francis Group, LLC (2009)
28. Fu, Y., Guo, G.D., Huang, T.S.: Age synthesis and estimation via faces: A survey. *IEEE Trans. Pattern Analysis and Machine Intelligence* 32(11), 1955–1976 (2010)
29. Fu, Y., Huang, T.S.: Human age estimation with regression on discriminative aging manifold. *IEEE Trans. on Multimedia* 10(4), 578–584 (2008)
30. Fu, Y., Xu, Y., Huang, T.S.: Estimating human ages by manifold analysis of face pictures and regression on aging features. In: *IEEE Conf. on Multimedia and Expo.*, pp. 1383–1386 (2007)
31. Fukai, H., Takimoto, H., Mitsukura, Y., Fukumi, M.: Apparent age estimation system based on age perception. In: *SICE Annual Conference*, pp. 2808–2812 (2007)
32. Gallagher, A., Chen, T.: Understanding images of groups of people. In: *CVPR*, pp. 256–263 (2009)
33. Gao, F., Ai, H.: Face age classification on consumer images with gabor feature and fuzzy lda method. In: *The 3rd IAPR Intl. Conf. on Biometrics* (2009)
34. Gao, W., Ai, H.: Face gender classification on consumer images in a multiethnic environment. In: *Intl. Conf. on Biometrics* (2009)
35. Geng, X., Zhou, Z.H., Smith-Miles, K.: Automatic age estimation based on facial aging patterns. *IEEE Trans. on PAMI* 29(12), 2234–2240 (2007)
36. Geng, X., Zhou, Z.H., Zhang, Y., Li, G., Dai, H.: Learning from facial aging patterns for automatic age estimation. In: *ACM Conf. on Multimedia*, pp. 307–316 (2006)
37. Golomb, B., Lawrence, D., Sejnowski, T.: Sexnet: A neural network identifies sex from human faces. In: *Advances in Neural Information Processing Systems*, vol. 3, pp. 572–577 (1991)
38. Graf, A., Wichmann, F.: Gender classification of human faces. In: *Int'l Workshop on Biologically Motivated Computer Vision*, pp. 491–500 (2002)

39. Gunay, A., Nabiyev, V.V.: Automatic detection of anthropometric features from facial images. In: IEEE Conf. on Signal Processing and Communications Applications (2007)
40. Gunay, A., Nabiyev, V.V.: Automatic age classification with LBP. In: Proc. Int'l Symp. Computer and Information Science (2008)
41. Guo, G.D., Dyer, C., Fu, Y., Huang, T.S.: Is gender recognition affected by age? In: IEEE International Workshop on Human-Computer Interaction, pp. 2032–2039 (2009)
42. Guo, G.D., Fu, Y., Dyer, C., Huang, T.S.: Image-based human age estimation by manifold learning and locally adjusted robust regression. *IEEE Trans. Image Processing* 17(7), 1178–1188 (2008)
43. Guo, G.D., Fu, Y., Dyer, C., Huang, T.S.: A probabilistic fusion approach to human age prediction. In: International Workshop on Semantic Learning Applications in Multimedia (2008)
44. Guo, G.D., Fu, Y., Huang, T., Dyer, C.: Locally adjusted robust regression for human age estimation. In: IEEE Workshop on Application of Computer Vision (2008)
45. Guo, G.D., Mu, G.: Human age estimation: what is the influence across race and gender? In: IEEE International Workshop on Analysis and Modeling of Faces and Gestures (2010)
46. Guo, G.D., Mu, G.: Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 657–664 (2011)
47. Guo, G., Mu, G., Fu, Y.: Gender from Body: A Biologically-Inspired Approach with Manifold Learning. In: Zha, H., Taniguchi, R.-I., Maybank, S. (eds.) ACCV 2009. LNCS, vol. 5996, pp. 236–245. Springer, Heidelberg (2010)
48. Guo, G.D., Mu, G., Fu, Y., Dyer, C., Huang, T.S.: A study on automatic age estimation on a large database. In: IEEE International Conference on Computer Vision, pp. 1986–1991 (2009)
49. Guo, G.D., Mu, G., Fu, Y., Huang, T.S.: Human age estimation using bio-inspired features. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 112–119 (2009)
50. Guo, G.D., Mu, G., Ricanek, K.: Cross-age face recognition on a very large database: the performance versus age intervals and improvement using soft biometric traits. In: International Conference on Pattern Recognition (2010)
51. Hayashi, J., Yasumoto, M., Ito, H., Koshimizu, H.: A method for estimating and modeling age and gender using facial image processing. In: Seventh Int. Conf. on Virtual Systems and Multimedia, pp. 439–448 (2001)
52. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep., 7–49. University of Massachusetts, Amherst (2007)
53. Jain, A., Huang, J.: Integrating independent component analysis and linear discriminant analysis for gender classification. In: IEEE Int'l Conf. on Automatic Face and Gesture Recognition (2004)
54. Jain, A.K., Dass, S.C., Nandakumar, K.: Soft Biometric Traits for Personal Recognition Systems. In: Zhang, D., Jain, A.K. (eds.) ICBA 2004. LNCS, vol. 3072, pp. 731–738. Springer, Heidelberg (2004)
55. Johnson, T.: Recent results: Biomarkers of aging. *Experimental Gerontology* 41, 1243–1246 (2006)
56. Kanno, T., Akiba, M., Teramachi, Y., Nagahashi, H., Agui, T.: Classification of age group based on facial images of young males by using neural networks. *IEICE Trans. on Information and Systems* E84-D(8), 1094–1101 (2001)

57. Kwon, Y., Lobo, N.: Age classification from facial images. *Computer Vision and Image Understanding* 74(1), 1–21 (1999)
58. Lanitis, A., Draganova, C., Christodoulou, C.: Comparing different classifiers for automatic age estimation. *IEEE Trans. on SMC-B* 24(4), 621–628 (2002)
59. Lanitis, A., Taylor, C.J., Cootes, T.F.: Toward automatic simulation of aging effects on face images. *IEEE Trans. on Pattern Anal. Mach. Intell.* 34(1), 442–455 (2002)
60. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *IJCV* 60(2), 91–110 (2004)
61. Luu, K., Ricanek, K., Bui, T., Suen, C.: Age estimation using active appearance models and support vector machine regression. In: *IEEE Conf. on BTAS*, pp. 1–5 (2009)
62. Makinen, E., Raisamo, R.: Evaluation of gender classification methods with automatically detected and aligned faces. *IEEE Trans. Pattern Anal. Mach. Intell.* 30(3), 541–547 (2008)
63. Martin, A.: Bank transfer fraudsters have that gift of gab. *The Japanese Times* (2009), <http://search.japantimes.co.jp/cgi-bin/nn20090203i1.html>
64. Moghaddam, B., Yang, M.H.: Learning gender with support faces. *IEEE Trans. Pattern Anal. Mach. Intell.* 24(5), 707–711 (2002)
65. Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms (1989)
66. Mutch, J., Lowe, D.: Object class recognition and localization using sparse features with limited receptive fields. In: *IEEE Conf. on Comput. Vision and Pattern Recognit.*, pp. 11–18 (2006)
67. Ni, B., Song, Z., Yan, S.: Web image mining towards universal age estimator. In: *ACM Multimedia* (2009)
68. Oren, M., Papageorgiou, C., Sinha, P., Osuna, E., Poggio, T.: Pedestrian detection using wavelet templates. In: *IEEE Conf. on Comput. Vision and Pattern Recognit.*, pp. 193–199 (1997)
69. Park, U., Tong, Y., Jain, A.K.: Face recognition with temporal invariance: A 3d aging model. In: *Intl. Conf. on Automatic Face and Gesture Recognition* (2008)
70. Phillips, P., Moon, H., Rizvi, S., Rauss, P.: The feret evaluation methodology for face recognition algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(10), 1090–1104 (2000)
71. Ramanathan, N., Chellappa, R.: Face verification across age progression. *IEEE Trans. on Image Processing* 15(11), 3349–3361 (2006)
72. Ramanathan, N., Chellappa, R.: Modeling age progression in young faces. In: *IEEE CVPR*, pp. 387–394 (2006)
73. Ramanathan, N., Chellappa, R., Biswas, S.: Age progression in human faces: A survey. *Visual Languages and Computing* (2009)
74. Rhodes, M.G.: Age estimation of faces: A review. *Applied Cognitive Psychology* 23, 1–12 (2009)
75. Ricanek, K., Tesafaye, T.: Morph: A longitudinal image database of normal adult age-progression. In: *IEEE Conf. on AFGR*, pp. 341–345 (2006)
76. Riesenhuber, M., Poggio, T.: Hierarchical models of object recognition in cortex. *Nature Neuroscience* 2(11), 1019–1025 (1999)
77. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326 (2000)
78. Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., Poggio, T.: Robust object recognition with cortex-like mechanisms. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(3), 411–426 (2007)

79. Serre, T., Wolf, L., Poggio, T.: Object recognition with features inspired by visual cortex. In: IEEE Conf. on Comput. Vision and Pattern Recognit. (2005)
80. Shakhnarovich, G., Viola, P., Moghaddam, B.: A unified learning framework for real time face detection and classification. In: Intl. Conf. on Automatic Face and Gesture Recognition (2002)
81. Shan, C.: Gender Classification on Real-Life Faces. In: Blanc-Talon, J., Bone, D., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2010, Part II. LNCS, vol. 6475, pp. 323–331. Springer, Heidelberg (2010)
82. Shan, C.: Learning local features for age estimation on real-life faces. In: ACM Intl. Workshop on Multimodal Pervasive Video Analysis (2010)
83. Shan, C., Gong, S., McOwan, P.W.: Fusing gait and face cues for human gender recognition. Neurocomputing 71(10-12), 1931–1938 (2008)
84. Stegmann, M., Ersbøll, B., Larsen, R.: FAME - A flexible appearance modelling environment. IEEE Trans. Medical Imaging 22(10), 1319–1331 (2003)
85. Sun, Z., Bebis, G., Yuan, X., Louis, S.: Genetic feature subset selection for gender classification: A comparison study. In: IEEE Workshop on Application of Computer Vision (2002)
86. Suo, J., Zhu, S., Shan, S., Chen, X.: A compositional and dynamic model for face aging. IEEE Trans. Pattern Anal. Mach. Intell. 32(3), 385–401 (2010)
87. Tan, Q., Kruse, T., Christensen, K.: Design and analysis in genetic studies of human ageing and longevity. Ageing Research Reviews 5, 371–387 (2006)
88. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for non-linear dimensionality reduction. Science 290, 2319–2323 (2000)
89. Toews, M., Arbel, T.: Detection, localization, and sex classification of faces from arbitrary viewpoints and under occlusion. IEEE Trans. Pattern Anal. Mach. Intell. 31(9), 1567–1581 (2009)
90. Ueki, K., Hayashida, T., Kobayashi, T.: Subspace-based age-group classification using facial images under various lighting conditions. In: IEEE Conf. on AFGR (2006)
91. Vapnik, V.N.: Statistical Learning Theory. John Wiley, New York (1998)
92. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple. In: Proc. IEEE CVPR (2001)
93. Wang, Y., Ricanek, K., Chen, C., Chang, Y.: Gender classification from infants to seniors. In: IEEE Conf. on BTAS, pp. 1–6 (2010)
94. Wild, H.A., Barrett, S.E., Spence, M.J., O'Toole, A.J., Cheng, Y.D., Brooke, J.: Recognition and sex categorization of adults' and children's faces: examining performance in the absence of sex-stereotyped cues. J. of Exp. Child Psychology 77, 269–291 (2000)
95. Wu, B., Ai, H., Huang, C., Lao, S.: Lut-based adaboost for gender classification. In: Intl. Conf. on Audio and Video-Based Person Authentication (2003)
96. Wu, T.X., Lu, B.L.: Multi-View Gender Classification Using Hierarchical Classifiers Structure. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) ICONIP 2010. LNCS, vol. 6444, pp. 625–632. Springer, Heidelberg (2010)
97. Xiao, B., Yang, X., Xu, Y.: Learning distance metric for regression by semidefinite programming with application to human age estimation. In: ACM Multimedia (2009)
98. Xu, X., Huang, T.S.: SODA-Boosting and Its Application to Gender Recognition. In: Zhou, S.K., Zhao, W., Tang, X., Gong, S. (eds.) AMFG 2007. LNCS, vol. 4778, pp. 193–204. Springer, Heidelberg (2007)
99. Xu, Z., Chen, H., Zhu, S., Luo, J.: A hierarchical compositional model for face representation and sketching. IEEE Trans. Pattern Anal. Mach. Intell. 30(6), 955–969 (2008)
100. Yamaguchi, M.K., Hirukawa, T., Kanazawa, S.: Judgment of sex through facial parts. Perception 24, 563–575 (1995)

101. Yan, S., Liu, M., Huang, T.: Extracting age information from local spatially flexible patches. In: IEEE Conf. on ICASSP, pp. 737–740 (2008)
102. Yan, S., Wang, H., Huang, T.S., Tang, X.: Ranking with uncertain labels. In: IEEE Conf. on Multimedia and Expo., pp. 96–99 (2007)
103. Yan, S., Wang, H., Tang, X., Huang, T.: Learning auto-structured regressor from uncertain nonnegative labels. In: IEEE Conf. on ICCV (2007)
104. Yan, S., Xu, D., Zhang, B., Zhang, H., Yang, Q., Lin, S.: Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 40–51 (2007)
105. Yan, S., Zhou, X., Liu, M., Hasegawa-Johnson, M., Huang, T.: Regression from patch-kernel. In: IEEE Conf. on CVPR (2008)
106. Yang, Z., Ai, H.: Demographic classification with local binary patterns. In: Intl. Conf. on Biometrics, pp. 464–473 (2007)

# People Counter: Counting of Mostly Static People in Indoor Conditions

Amit Khemlani, Kester Duncan, and Sudeep Sarkar

**Abstract.** The ability to count people from video is a challenging problem. The scientific challenge arises from the fact that although the task is relatively well-defined, the imaging scenario is not well constrained. The background scene can be uncontrolled along with the illumination being complex and varying. Additionally, the spatial and temporal image resolution is usually poor. The context of most works in people counting is in counting pedestrians from single frames in outdoor settings or moving subjects in indoor settings from standard frame rate video. There is little work done on counting of persons in varying poses, who are mostly static (sitting, lying down), in very low frame rate video (4 frames per minute), and under harsh illumination variations. In this chapter, we explore a design that handles illumination issues at the pixel level using photometry-based normalization, and pose and low-movement issues at feature level by exploiting the spatio-temporal coherence that is present among small body part movements. The motion of each body part, such as the hands or the head, will be present even in mostly static poses. These short duration motions will occur spatially close together over the image location occupied by the subject. We accumulate these using a spatio-temporal autoregressive (AR) model to arrive at blob representations that are further grouped into people counts. We show quantitative performance on real datasets.

---

Amit Khemlani

Computer Science and Engineering, University of South Florida, 4202 E. Fowler Avenue, Tampa FL 33620, now at Healthsystems, Tampa, Florida

Kester Duncan

Computer Science and Engineering, University of South Florida, 4202 E. Fowler Avenue, Tampa FL 33620

e-mail: [kkduncan@cse.usf.edu](mailto:kkduncan@cse.usf.edu)

Sudeep Sarkar

Computer Science and Engineering, University of South Florida, 4202 E. Fowler Avenue, Tampa FL 33620

e-mail: [sarkar@cse.usf.edu](mailto:sarkar@cse.usf.edu)

## 1 Introduction

### 1.1 Motivation

Counting people is a challenging scientific problem that has many practical applications. These include monitoring the number of people sitting in front of a television set, counting the number of people in an elevator, counting people passing through security doors in malls, and counting the number of people working in a laboratory. In general, people counting is important in surveillance-based applications, market research, and people management. Manually counting people given a sequence of images is a very tedious process and automation helps. There are many work on people detection that work when the pose of the person can be constrained, e.g. in the case of pedestrian detection. Another class of approach to people counting is through detection of motion in a video sequence. These approaches assume people would be in motion for a substantial amount of time and use normal frame rate video. These assumptions are not suited for static scenes such as people working on a computer or watching television, where we have the issues of uncontrolled background scene, poor or complex illumination, poor image resolution (e.g. 320x240), and low frame rate (e.g. 4 frames per minute). In this chapter, we demonstrate methods to deal with the task of counting people under these challenging conditions.

### 1.2 State of the Art

Tables 1 and 2 categorize the work that has been done in people counting in terms of the application context, the degree of illumination it can handle, the computational methods employed to solve the problem, and the performance of the work. It also contains work done in people tracking that can be applied to count people. Most approaches use some form of background subtraction or frame to frame differencing which makes them susceptible to illumination changes. They exploit only the spatial information to count people thereby disregarding the importance of temporal information which could prove to be a very useful cue to deal with illumination changes. Many approaches use a setup in which an overhead camera is installed [18, 19, 20, 26, 1, 12]. Some use skin color to detect people [22]. Some works use static images to count people and some use video. The frame rates used ranges from 1 frame per second to 30 frames per second (fps). The lowest frame rate used was 1 fps [19]. In some instances, multiple sensors have been used to facilitate people counting. Infra-red cameras have also been used to detect irides, resulting in people counts [25, 7].

For low-frame rates, counting can be done by detecting people in each frame as in [15], where limbs and faces are separately detected using Quadratic Support Vector Machines (SVM) classifiers and then fused. There are also many pedestrian detection works in the literature that are also based on machine learning based approaches. These approaches works well for upright people. We present an approach that does not make assumptions about the posture of the person. The number of

possible configurations is large. In some cases there are subjects sitting with their backs to the camera. In this case, face and leg detection are not possible.

There is little work done in people counting when people are not moving around. Additionally, there is little work done that can handle harsh variations in illumination conditions. We address counting relatively static people using a static camera and we also deal with varying illumination conditions. We address these issues through pixel normalization and the use of spatiotemporal coherence information obtained from the video sequence.

**Table 1** Prior work done in people counting, categories into the application context, nature of illumination variations they can handle, algorithm employed, and the nature of performance achieved.

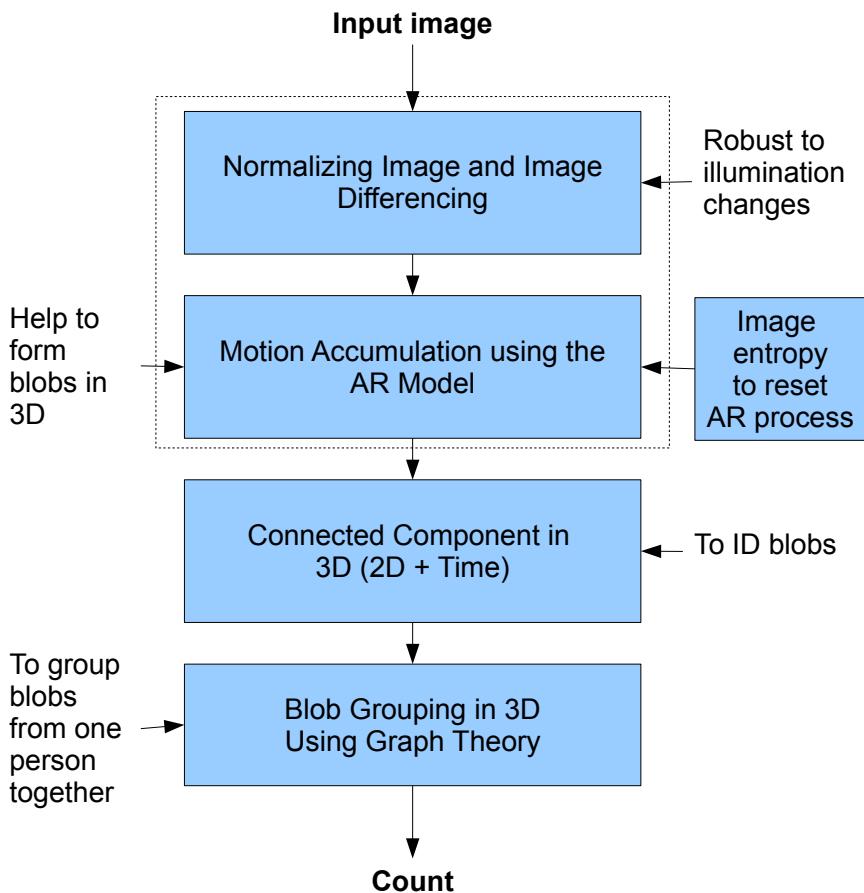
Work	Context	Illumination	Method	Performance
Conrad and Johnsonbaugh, 1994 [4]	Overhead view, moving people	Varying	Frame differencing	95.6% accuracy for a sampling of 7491 people
Rossi and Bozzoli, 1994 [18]	Top-down view at a security gate, moving people	Fixed	Frame differencing, vertical projection, texture analysis within blob	20 to 40 persons, 90% correct
Schofield et al., 1996 [19]	Top-down view of elevator	Fixed	Threshold based on complex background model learnt using neural networks	1 fps up to 7 people
Segen and Pingali, 1996 [20]	Elevated views at a mall, moving persons	Fixed	Track blobs, group short traces into larger tracks	14 fps, couple of people
Kuno, Watanabe, Shimosakoda, and Nakagawa, 1996 [13]	Horizontal view of people walking and non-humans	Fixed	Background subtraction, threshold using extraction function, get 3 projection histograms and calculate 3 features for counting people and handling occlusion	2500 images, 98% humans detected correctly.
Stillman et al., 1998 [22]	Fixed and Pan-Tilt-Zoom cameras	Fixed	Skin-tone detection, face blob detection based on vertical and horizontal projection	2 persons
Kettmaker and Zabih, 1999 [11]	Multiple cameras along corridor, moving person	Fixed	Moving blobs found using grouping of optic flow, matching across cameras using bipartite matching	4 cameras, 13 people
Terrada, Yoshida, Oe, and Yamaguchi, 1999 [26]	Stereo camera hung from ceiling	Fixed	Template matching between space-time images from two cameras to generate 3D data, two pairs of space-time images are generated to detect direction	22 people in and 21 people out
Beymer, 2000 [1]	Stereo camera on a door and pointing down	Fixed	Calibrate and specify VOI at setup, reconstruct the scene in 3D and select disparity pixels (occupancy map) in a VOI and Gaussian mixtures to model multiple people	20-25 fps, 905 enter/exit events, 4 hrs of data, error rate=1/4%
Lee, 2000 [14]	Horizontal view of office scenes	Fixed	Frame differencing, vertical projection, upper silhouette enhancement by voting, shape analysis to detect heads	3 people
Jabri et al., 2000 [10]	Horizontal views of walking people	Slow illumination change	Background (adaptive) subtraction, edge subtraction	1 person in 14 sequences under two different illumination conditions

**Table 2 (Continued)** Prior work done in people counting, categories into the application context, nature of illumination variations they can handle, algorithm employed, and the nature of performance achieved.

Work	Context	Illumination	Method	Performance
Mohan, Papageorgiou, and Poggio, 2001 [15]	Horizontal view of walking people	Fixed	Haar wavelet transform, Quadratic SVM for identifying components and SVM for a combination classifier	
Haritaoglu, Flickr, 2001 [7]	Horizontal view of people waiting for cashier	Fixed	Silhouette-based and motion-based people detection. Uses 2 IR light sources for iris detection	39 people, 25-30 fps, 55 minutes of data
Huang, Chow, and Chau, 2002 [9]	View of a station	Fixed	Image segmentation (sub-block), feature extraction (sub-curve) and RBF networks	Up to 92.5% accuracy
Kim et al., 2002 [12]	Top-down view of security gate, moving people	Fixed	Frame differencing and background subtraction, convex hulls of blobs, tracking and counting	10 fps, 6 people
Yang, Gonzalez-Banos, and Guibas, 2003 [25]	8 sensors	Fixed	Background subtraction, planar projection of silhouette, intersection of each projection, remove small polygons and phantoms, report new bounds on the number of objects	15 fps
Ramoser et al., 2003 [17]	Horizontal view surveillance system	Fixed	Standard adaptive background models, CC analysis, blob filtering and model fitting	2 sequences of 500 frames each
Zhao and Nevatia, 2004 [28]	Top-down view of outdoor scene	Varying	Blob extraction, Background/Camera/Human models	99% detection accuracy
Celik et al., 2006 [3]	Top-down view of mall scene	Fixed	Foreground/Background extraction, Gaussian Mixture Model	25 fps and 80% accuracy
Ramanan and Forsyth, 2007 [16]	Indoor and outdoor horizontal views	Varying	Human appearance model	90% average detection on many sequences
Shengsheng et al., 2008 [27]	Overhead camera	Varying	Foreground/Background Edge Model	97.8% accuracy
Fehr et al., 2009 [5]	Top-down view of outdoor scene	Varying	Gaussian Mixtures, Shadow removal	30 fps
Zhao et al., 2009 [29]	Horizontal View of indoor scenes	Small changes	Kalman Filter, Object tracking, Angle histogram, Face Detection	93% accuracy
Ya-Li Hou, and Grantham K.H. Pang, 2011 [8]	Static camera overlooking a large area	Varying	Neural Network, Background estimation using Gaussian Mixture Models, Expectation-Maximization	10 fps, 90% accuracy

### 1.3 Overview of the Approach

Our approach, depicted in Figure 1, relies on the fact that subjects tend to shift or move body parts and are rarely absolutely stationary for an extended period of time. It exploits this observation and first detects frame to frame motion blobs by a combination of frame to frame differencing and motion accumulation. These blobs are then grouped spatiotemporally (in XYT) by treating the image sequence as a three dimensional signal, two image dimensions + time, to form counts of people for each frame. The approach primarily comprises of four levels of processing.



**Fig. 1** Outline of the approach.

1. The first level makes the sequence of images robust to illumination changes. This is achieved by normalizing the images by dividing each pixel by the maximum in its neighborhood. The normalized images are to an extent less susceptible to illumination changes. After obtaining the normalized images, frame to frame differencing is performed and the resulting images are thresholded.
2. Autoregressive (AR) model-based accumulation is used as a second level of processing. The AR model helps to accumulate the temporal and spatial motion to form good 3D blobs that can be clustered by subsequent levels. The output of the AR model is then thresholded.
3. Connected component analysis is the third level of processing and is performed on the threshold-AR images to identify the blobs. This is performed not only spatially, but also temporally. This is done by treating the sequence of images as a block of images. The connected component step is performed to distinguish

blobs from each other and serves as input to the blob grouping step. The first three levels form the *Blob Detection* phase of the approach.

4. Blob grouping is necessary because there are many blobs corresponding to a single person that need to be grouped together to generate an accurate people count.

The approached outlined here is quite robust. Our experiments, which we will present later, show that it is not easily susceptible to lighting changes. Given an empty scene with just lighting change it usually produces a count of zero. It can maintain count under varying illumination conditions. It can count people even if they are partially visible. It is worthwhile mentioning that that counts are generated for any moving objects in the scene. It does not yet try to distinguish between humans and non-humans. Most counting errors are concentrated around frames with large motion events, such as a person moving out from a scene.

## 2 Blob Detection

Blob detection involves intensity image normalization and motion accumulation. The normalizing step is applied to the original images and is important to make the image robust to illumination variations to a certain extent. Once the images are robust to illumination changes, motion accumulation is applied. The motion of each body part is temporary. However, fleeting motion strands will occur spatially close together for the image location occupied by the subject. The motion accumulator should be able to integrate the motion cues that are found by frame to frame differencing both spatially and temporally.

### 2.1 Image Intensity Normalization

The intensity normalization idea relies on the approximate model of the image intensity  $I(x,y,t)$  in an image given by

$$I(x,y,t) = I_0(t)\rho(x,y)\mathbf{n}^T\mathbf{s} \quad (1)$$

where,

- $I_0(t)$ : is the incident intensity,
- $\rho(x,y)$ : is the surface albedo at location  $(x,y)$ ,
- $\mathbf{n}$ : is the surface normal at the location  $(x,y)$ , and
- $\mathbf{s}$ : is the direction to the light source.

Assuming that we are observing mostly static scenes with some degree of motion, the variation in intensity is primarily due to the variation in the intensity of the overall illumination. The ratio of the intensities of two nearby pixels is invariant to the overall intensity. This ratio is expressed mathematically as

$$NI(x_1,y_1,t) = \frac{I(x_1,y_1,t)}{I(x_2,y_2,t)} = \frac{\rho(x_1,y_1)\mathbf{n}_1^T\mathbf{s}}{\rho(x_2,y_2)\mathbf{n}_2^T\mathbf{s}}. \quad (2)$$

This ratio will tend to pick up spatial changes in albedo *and* surface normal, but will be stable with respect to overall illumination changes. It will be one for regions where there is no change in albedo or surface normal. To impart some amount of numerical stability to the computation process, we consider the neighboring pixel to be the one with the maximum intensity in a fixed-size neighborhood around the pixel under consideration. We experimented with 3x3, 5x5, and 7x7 fixed-size neighborhoods. Figure 2 shows some normalized image frames generated using a 7x7 window along with the corresponding original intensity frames. Note how the illumination related shading gradient across the walls are removed. The features of the objects in the third shelf of the book case on the left, which are in the dark in the original image, gain prominence in the normalized images. Figure 3 shows that frame differencing of original intensities detects changes due to illumination variations whereas the frame difference of normalized intensities tends to be robust with respect to illumination changes. The original image difference is not robust to illumination change and this is shown by the red ellipse in the threshold difference of the original image. Corresponding to that region in the threshold difference of the normalized image the illumination change is not captured. Though the normalized image is robust to illumination change to a certain extent, the thresholded difference of the normalized image does introduce some noise. This is shown within the green ellipse in Figure 3. This noise is mostly few pixels in size and can be easily removed by morphological operations.

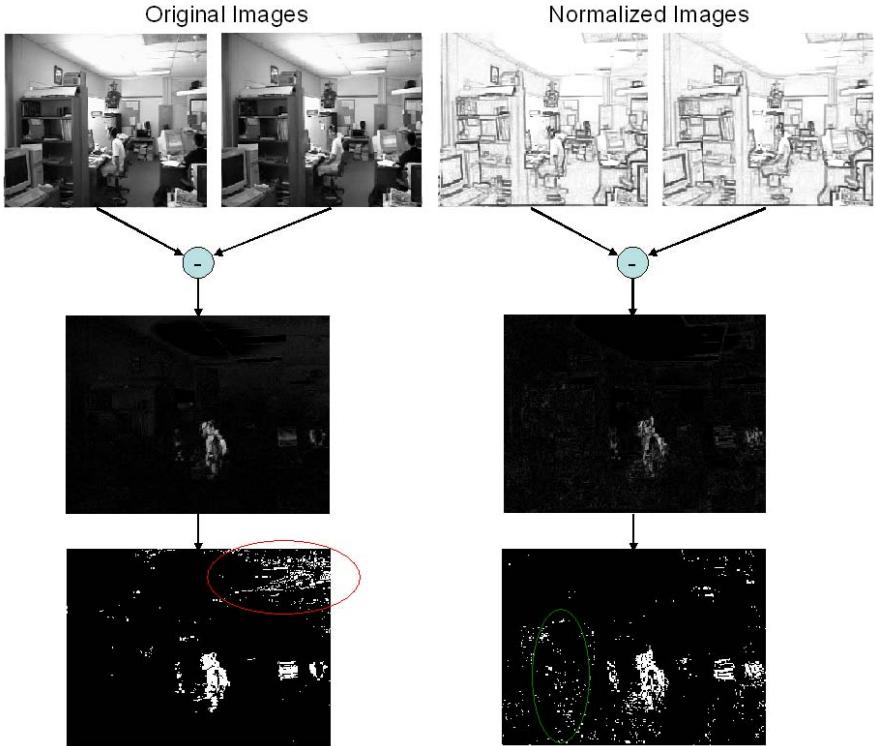


Original images



Normalized images

**Fig. 2** Sample frames showing the locally normalized image intensity representation.



**Fig. 3** Frame difference of original intensities versus frame difference of normalized intensities.

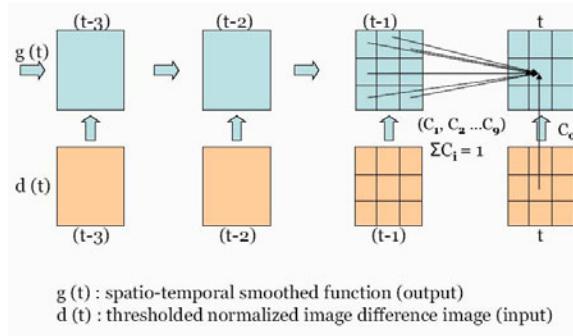
## 2.2 Motion Accumulation Using Autoregressive Models

A mechanism to accumulate intermittent motion that occurs as parts of a subject moves is required. The motion of each body part, such as the hands or the head, will be present temporarily. However, lots of short duration motion strands will occur spatially close together for the image location occupied by the subject. The motion accumulator should be able to integrate, both spatially and temporally, the motion cues that are found by frame to frame differencing. It should possess sufficient “memory” to compensate for the lack of motion at a pixel for a few minutes. It should be able to “forget” short duration motion events that occur only over a few frames. This can be achieved fairly easily using a spatiotemporal autoregressive (AR) filtering of the normalized-image intensity differences. Let  $g(x, y, t)$  denote the corresponding autoregressive (AR) filtered output, which is mathematically expressed in [3] as

$$g(x, y, t) = C_{000} d(x, y, t) + \sum_{k=1}^T \sum_{(i,j) \in R} C_{ijk} g(x+i, y+j, t-k) \quad (3)$$

where,

- $C_{ijk}$ : are AR coefficients,
- $T$ : is the temporal order of the AR process,
- $R$ : represents the spatial order in terms of the local neighborhood considered, and
- $d(x, y, t)$ : denotes a pixel at location  $(x, y)$  in the  $t^{th}$  time frame of the normalized-image difference.



**Fig. 4** Motion history accumulation using the autoregressive (AR) model.

It is important to note that the coefficients should sum to 1 so that the gray level of the pixels are maintained within 0 and 255;  $\sum C_{ijk} = 1$ . In our experiments, we choose all coefficients effect for  $C_{000}$  to be the same. Figure 4 depicts the operations pictorially for  $T = 1$  and  $R = 3 \times 3$  neighborhood around a pixel. Note that the operations involve only local pixels, both spatially and temporally, making it suitable for hardware implementation. Even though the operations are local, the effect of a pixel in  $d(x, y, t)$  persists, mediated through  $g(x, y, t)$ . Using z-transforms, it can be easily shown that the fall in “memory” is exponential, both spatially and temporally. The rate of fall of the exponential function is controlled by the value of the coefficients.

Figure 5 shows a sample output of the AR filtering process. To fully appreciate the effects, one has to view the video sequence. The results reduce the emphasis of changes that happen only in a few frames over time such as those caused by a walking person and enhances statistical changes that are repeated in a local vicinity, such as those caused by a stationary person in the scene.

The memory of the AR process can be a disadvantage when there are random intensity fluctuations, either due to severe illumination change or due to a person walking past the camera. The effect of these events can persist for some frames. To handle such cases, the AR process is reset whenever such drastic changes in the



**Fig. 5** The left column shows the original frames, the middle column shows the output of the frame to frame differencing of the normalized images and the right column shows the output of the autoregressive process.

scene are detected, as reflected in the change of the normalized-intensity histogram of the image. The change in the entropy of the normalized intensity histogram of the frames is monitored. Let  $P_t(k)|k = 0, \dots, 1.0$  represent the histogram for the  $t^{\text{th}}$  frame, normalized to sum to one. The dissimilarity between the distributions from two time instants is quantified by the decrement of entropy that is achieved by combining or merging them as compared to considering them separately. If the probabilities governing the random variables are similar, then the decrement of entropy would be low, implying a minor dissimilarity between the random variables. Mathematically, it is given by

$$\Delta H(t, t+1) = H\left(\frac{1}{2}(P_t(n) + P_{t+1}(n))\right) - \frac{1}{2}(H(P_t(n)) + H(P_{t+1}(n))) \quad (4)$$

where,

- $P_t(n)$ : is the histogram of intensity of normalized image at time ‘t’,
- $P_{t+1}(n)$ : is the histogram of intensity of normalized image at time ‘t+1’,
- $H(P_t(n))$ : is the entropy of intensity of normalized image at time ‘t’:

$$H(P_t(n)) = -\sum_n P_t(n) \log P_t(n),$$

- $H(P_{t+1}(n))$ : is the entropy of intensity of normalized image at time ‘t+1’:

$$H(P_{t+1}(n)) = -\sum_n P_{t+1}(n) \log P_{t+1}(n)$$

- $H(\frac{1}{2}(P_t(n) + P_{t+1}(n)))$ : is the entropy of the “merging” of the histogram of the intensities from the two image frames:

$$H\left(\frac{P_t(n) + P_{t+1}(n)}{2}\right) = -\sum_n \left[ \left(\frac{P_t(n) + P_{t+1}(n)}{2}\right) \log \left(\frac{P_t(n) + P_{t+1}(n)}{2}\right) \right],$$

and

- $n$ : size of the bin of the histogram.

If the intensities in the individual frames have not changed drastically, then the distribution of the merged frames will be the same as that for the individual frames and the decrement of entropy would be zero. Wong and You [23] first used this entropic distance to measure similarity between two random graphs. Note that other measures exist for determining the distance between probability distributions such as the mutual information-based KL measure and the Bhattacharya distance. However, both these measures require that the support set of the two probability measures or the domain of non-zero probabilities be the same. This is not always possible in our context as the same set of intensity values might not exist in two the images.

The blob detection procedure is as: For each frame(k) in a sequence

1. Normalize the image frame ( $N_k$ )
2. Compute difference image:  $D_k = N_k - N_{k-1}$
3. Threshold  $D_k$  to produce  $d_k$
4. Compute AR filtered image  $g(x, y, t)$ :

```

if  $\Delta H(t, t + 1) < H_t$ 
     $g(x, y, t) = C_{000} d(x, y, t) + \sum_{k=1}^T \sum_{(i,j) \in R} C_{ijk} g(x + i, y + j, t - k)$ 
else
     $g(x, y, t) = 0$ 

```

Whenever the change in entropy of the intensity distributions is large, as compared to a threshold ( $H_t$ ), the AR process is reset. This ensures that the effect of drastic changes in a few frames does not persist. Conversely, resetting the AR process can result in the temporal disconnection of genuine motion blobs for people watching television or similar settings. This problem is addressed by the subsequent step of blob grouping which consolidates them.

### 3 Spatiotemporal Grouping into Counts

Following the detection of the motion pixels, a mechanism is required to group the motion pixels corresponding to a single person blobs that form our count. The first

step is to label each XYT blob. This is done by performing connected component analysis on an image sequence by treating it as a block of data. After the blobs are labeled, spatiotemporal grouping is performed to group similar blobs together. Before describing the grouping algorithm we will discuss some of the concepts of perceptual organization. There are three strategies for the spatiotemporal grouping problem [2]:

- *Sequential Approach:* This is the most commonly used method in computer vision. In this method the incoming frames of the video sequence are first processed to extract any spatial grouping. The spatial groupings are then grouped using the temporal information.
- *Interactive Approach:* In this method the temporal grouping performed is used to adjust the spatial grouping.
- *Multidimensional Approach:* In this method the image sequence is treated as a volume and the entire block is processed at one time. This method is by far the best strategy. The amount of computing power and the memory required to work directly on a three-dimensional signal has been a major concern, but this has been alleviated by recent technological advances. We adopt this approach.

**XYT Connected Component Analysis:** After motion accumulation, connected component analysis [6] is performed to identify the 3D blobs. In our method we apply connected component analysis not only in 2D but also in the third dimension of time i.e. we treat the entire image sequence as a block of data and apply connected component analysis to the entire block at once rather than to each image separately. Therefore, the number of neighbors increases to 26 based on 8-connectivity: 9 neighbors each from time  $t-1$  and  $t+1$ , and 8 neighbors from time  $t$ .

**XYT Blob Grouping:** The XYT blobs found by connected component labeling are set up as vertices of the graph and links (edges) are established between blobs. The links possess weights based on the proximity measure. The smaller the distance between the blobs, the more similar they are to each other and vice versa. The proximity factor given below, captures the spatiotemporal proximity of the blobs in distance.

$$A_{ij} = -\exp\left(\frac{(\mathbf{i}-\mathbf{j})^T(\mathbf{i}-\mathbf{j})}{\sqrt[3]{\vartheta_i}\sqrt[3]{\vartheta_j}}\right) \quad (5)$$

where,

- $\mathbf{i} = [x_i, y_i, t_i]$ : Center of XYT Blob  $i$ ,
- $\mathbf{j} = [x_j, y_j, t_j]$ : Center of XYT Blob  $j$ , and
- $\vartheta_i, \vartheta_j$ : Volume of Blobs  $i$  and  $j$ .

Note that the weights are high for blobs that are close together both spatially and temporally. The distance bridged is proportional to the size of the blobs. Larger blobs will be associated with other blobs over a longer distance than smaller blobs. The second step involves the partitioning of this graph into node clusters that are strongly connected to each other.

**Graph Cuts:** Once the links and vertices are established, we partition the graph into clusters based on the disassociation measure. Ideally the graphs are cut along those links with less weight to form a good set of connected components. There are various disassociation measures to cut the graph: minimum cut, normalized cut, average cut, and ratio cut. Consider a graph  $G = (V, E)$  that needs to be partitioned into two sets, A and B such that:  $A \cup B = V$  and  $A \cap B = \emptyset$ , by removing the edges connecting the two sets. This is called the cut [24] and the cut measure is defined as.

$$\text{cut}(a, b) = \sum_{u \in A, v \in B} w(u, v) \quad (6)$$

The optimal way to bipartition the graph is to minimize the cut value. There is a major drawback in using the minimum cut criterion because it has the tendency to cut small sets of isolated nodes in the graph. To overcome this problem the normalized cut [21] is used. The normalized cut computes the cut cost as a fraction of the total edge connections to all the nodes in the graph as opposed to the minimum cut which only computes the value of the total edge weight connecting the two parts. The two most important properties of grouping: minimizing the disassociation between clusters and maximizing the association within cluster are thus satisfied simultaneously. The normalized cut criterion can be approximated efficiently by the following equation [21]:

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y}, \quad (7)$$

where  $\mathbf{D}$  is an  $N \times N$  diagonal matrix with  $\mathbf{d}$  on its diagonal (let  $\mathbf{d}(i) = \sum_j w(i, j)$  be the total connection from node  $i$  to all other nodes), and  $\mathbf{W}$  is an  $N \times N$  symmetrical matrix with  $W(i, j) = \text{weight between nodes } i \text{ and } j$ . This is a generalized eigenvalue equation.

## 4 Experiments and Results

We show results on actual home datasets and laboratory datasets. The data from homes have images in which subjects are watching television and in the laboratory data we have subjects working in front of computers. The frame rate of the video is 4 frames per minute and images have 320x240 resolution. The images have people sitting, walking, talking and entering or exiting. Since the sequences span several days, they also have varying illumination conditions as another factor. In the home dataset we have children playing and also pets, which make counting of people more challenging. We quantified the performance of the people counter by using confusion matrices that uses manually generated people counts as ground-truth.

### 4.1 Result on Home Datasets

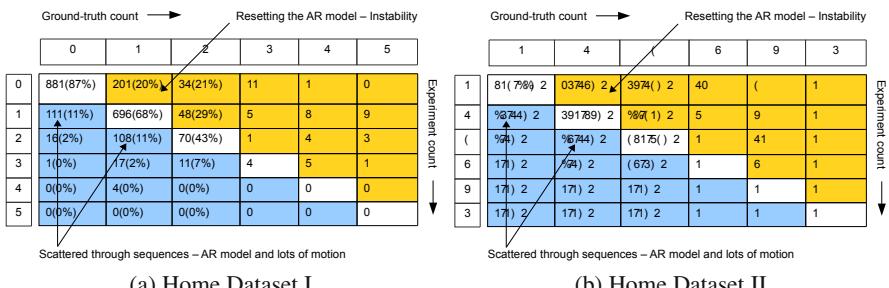
The first set of results we present are on two Nielsen Home datasets. These datasets have very poor quality images, with significant amounts of illumination variations, i.e. lights turned on and off, night vs. day, curtains opened and closed, etc. In some cases, there are individuals walking across the scene. People appear in different

poses in these images including reclining postures. Due to privacy reasons, we cannot show the images or visual results on this data.

In the first dataset (Home Dataset I) we have the images from a single home. The single camera is installed on top of a TV set to get a view of the living room along with the system that grabs and stores the frames. The number of sequences we have processed spans 5 days. In the second dataset (Home Dataset II) we have images from 4 homes. The data was collected using the B/W CCTV high resolution ex-view cube video security camera. It has 600 lines of resolution and uses a 3.7 mm lens. Since the camera used is an analog camera, Sensoray Model 611, a PCI-bus based frame grabber was used to grab and store the images. The size of the images is 320 x 240 and the images are grabbed once every 15 seconds. The data has images from the entire day so it covers various illumination conditions. It has people sleeping, kids playing in the room, windows which enable sunlight to come into the room, lamps, people in the dark watching TV, lights turning on and off.

We selected 45 sequences spread over 5 days from Home Dataset I. Each sequence consisted of 50 frames, resulting in a total of 2250 frames. For each frame, we manually noted the number of persons. From Home Dataset II, we selected 40 sequences obtained from these 4 homes to quantify the performance. Each sequence consisted of 50 frames, resulting in a total of 2000 frames. Table 3 presents the results of comparing the algorithm's counts with the manual counts in the form of a confusion matrix on these two datasets. Each row corresponds to a count produced by the algorithm and the columns correspond to the manually generated ground-truth count. A person was counted in the ground truth if any body part was visible. The entries of the matrix shows the number of frames with the corresponding pairs of counts. The diagonal denotes correct performance. Over-counts are in the lower triangle and the under-counts are in the upper triangle. The percentages are computed over the respective columns and represent the detection rates for scenes with different numbers of people in it. There are more under-counts than over-counts.

**Table 3** Performance on home datasets I and II as captured by confusion matrices that compare automated counts with manually generated ground truth counts.



**Table 4** Performance of the approach without using the autoregressive accumulation process on home datasets I and II as captured by confusion matrices that compare automated counts with manually generated ground truth counts.

**(a) Home Dataset I**

	0	1	2	3	4	5
0	964(96%)	293(29%)	38(23%)	11	1	0
1	42(4%)	626(61%)	39(24%)	5	7	7
2	1(0%)	89(9%)	64(39%)	2	5	6
3	2(0%)	5(5%)	16(10%)	1	2	0
4	0(0%)	3(0%)	5(3%)	2	3	0
5	0(0%)	0(0%)	1(1%)	0	0	0

Unable to group blobs of one person

**(b) Home Dataset II**

	1	3	8	7	6	0
1	964(%E) 2	396(86) 2	41(34) 2	3% 8	1	
3	63(0) 2	678(54) 2	331(80) 2	5 6	1	
8	5(3) 2	08(0) 2	873(07) 2	1 31	1	
7	1(1) 2	4(3) 2	36(8) 2	1 7	1	
6	1(1) 2	1(1) 2	1(1) 2	1 1	1	
0	1(1) 2	1(1) 2	1(1) 2	1 1	1	

Unable to group blobs of one person

In Table 4 we show the corresponding counts without the autoregressive history accumulation process. We see that the percentage of correct detection of empty scenes increased, however, the detection rate for scenes with people is lower. This attests to the importance of the autoregressive motion history accumulation process.

## 4.2 *Laboratory Dataset*

To be able to show examples of images and intermediate processing we also collected data of people sitting and working in the computer vision labs: ENB 218, ENB 348 and ENB 221. The data collection in ENB221 and ENB218 was done using the Canon Optura camera, whereas the data collection in ENB348 was done using the twin cameras used in the home dataset. Although the Canon Optura data collected was in WMV format sampled at 30 fps per seconds, we dropped frames to achieve the same frame rate as of the Home Datasets, i.e. 4 frames per minutes. We collected data of each laboratory for a duration of 5 hours. From this, we used snippets of the data collected based on the number of people, different illumination conditions and different types of activities. The number of people we have in our dataset vary from none to three. We have subjects sitting, working, entering and leaving. We have some illumination changes as the day progresses.

The dataset consists of 25 sequences. The sequences were chosen to capture different illumination conditions, different activities, and different number of subjects. Each sequence has a total of 50 images. Therefore there was a total of 1250 images. In the following sections we will show the results of some sequences covering illumination changes, subjects leaving and entering the laboratory, issues regarding moving objects other than humans, and the number of subjects present in the image.

### 4.2.1 Illumination Changes

Figure 6 shows some images from a sequence that is affected by illumination changes. The illumination change is prominent on the ceiling of the room. Figure 7 presents another example. The algorithm presented in this chapter is able to maintain an accurate people count even under varying illumination conditions. The number of different colors (for the blobs) represent the number of people.

In Figure 8 the subject enters and there are illumination changes. Note that although the blob shape do not correspond exactly to the shape of the person at that time instant. The count is maintained as the shape changes.

Figure 9 shows a sequence in which the light is initially off. The light comes on and a subject is sitting and working. In such a case, if we had used only frame to frame differencing we would have incorrectly acquired the entire image as the difference image. In our case, we also monitor the entropy of each image which helps determine whether to continue or stop the autoregressive process.

### 4.2.2 Different Activity

In Figure 10 a subject enters the room and later leaves. The subject's motion blob persists for a while due to the combined effect of frame to frame differencing and the autoregressive model. This causes over-counting for only a few frames after the subject leaves. This effect can be reduced by adjusting the autoregressive model coefficients.

In Figure 11 one of the subjects is partially visible. We are able to include him in the count because we only use motion as the cue to detect people, we do not detect any specific feature such as the face, hands, etc. Such approaches would have failed in this case.

Figure 12 is an example of three subjects sitting and working. Initially in the sequence there are only two subjects; the third subject enters towards the end of the sequence. The entrance event of the subject is not captured by the low-frame rate video. The subject sitting on the right side is partially visible for some frames in the sequence.

We quantify performance using confusion matrix of counts on a per image frame basis. One axis corresponds to the groundtruth of the number of subjects and the other corresponds to the computer count of the number of subjects. Table 5 shows the confusion matrices for the three laboratory datasets. The parameter values were the same across all datasets. The correct detection rates range from 74% to 100%.

## 5 Parameter Variations

We have experimented with parameter values and evaluated its effect on the people count. There are three major parameters: the window of the AR model used to accumulate motion, the entropy threshold parameter used to reset the AR process for large changes ( $H_t$ ), and the AR coefficient  $C_{000}$ . The rest of the AR coefficients are



**Fig. 6** Two subjects sitting in the laboratory. On left are the original images, and on right are the grouped spatio-temporal blob outputs. The two colors signify the blobs resulting in the count of two.



(a)



(b)



(c)



**Fig. 7** One subject is sitting and the other subject is standing. On left are the original images, and on right are the grouped blob outputs. The two colors signify the blobs resulting in the count of two.



(a)



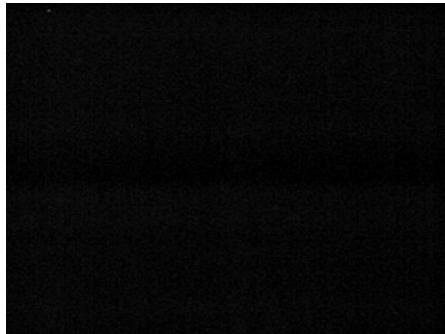
(b)



(c)



**Fig. 8** A subject enters the laboratory and sits down. On left are the original images, and on right are the blob outputs.



(a)



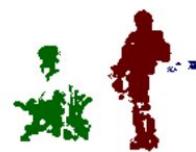
(b)



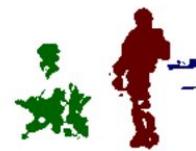
(c)



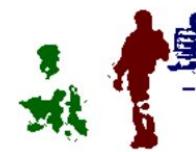
**Fig. 9** Initially it is dark, then the light comes on and a subject enters. On left are the original images, and on right are the outputs.



(a)



(b)



(c)

**Fig. 10** Two subjects are sitting and the third subject enters the laboratory in one frame and leaves in the other frame. On left are the original images, and on right are the outputs. The three colors signify the blobs resulting in the count of three.



(a)



(b)



(c)



**Fig. 11** Two subjects are talking and one is partially visible. On left are the original images, and on right are the outputs. The two colors signify the blobs resulting in the count of two.



**Fig. 12** Three subjects are working in the laboratory. On left are the original images, and on right are the outputs. The three colors signify the blobs resulting in the count of three.

**Table 5** Performance on the three laboratory dataset as captured by confusion matrices that compare automated counts with manually generated ground truth counts.

		Groundtruth Count →							Groundtruth Count →							Groundtruth Count →				
		0	1	2	3	4			0	1	2	3	4			0	1	2	3	4
← Experiment Count	0	14 (87.5%)	23 (10.2%)	0 (0%)	0 (0%)	0 (0%)	← Experiment Count	0	117 (83.5%)	25 (8.4%)	11 (2.4%)	0 (0%)	0 (0%)	← Experiment Count	0	300 (94.3%)	18 (4.85%)	0 (0%)	0 (0%)	0 (0%)
	1	2 (12.5%)	171 (76%)	38 (14.7%)	0 (0%)	0 (0%)		1	20 (14.2%)	220 (74.5%)	55 (12.2%)	0 (0%)	0 (0%)		1	15 (4.7%)	335 (90.2%)	0 (0%)	0 (0%)	0 (0%)
	2	0 (0%)	31 (13.8%)	220 (85.3%)	2 (100%)	0 (0%)		2	3 (2.1%)	47 (15.9%)	330 (73.4%)	3 (17.6%)	0 (0%)		2	3 (0.94%)	18 (4.8%)	62 (100%)	0 (0%)	0 (0%)
	3	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)		3	0 (0%)	3 (1.1%)	52 (11.5%)	14 (82.3%)	0 (0%)		3	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
	4	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)		4	0 (0%)	0 (0%)	1 (0.2%)	0 (0%)	0 (0%)		4	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)

(a) ENB 218

(b) ENB 221

(c) ENB 348

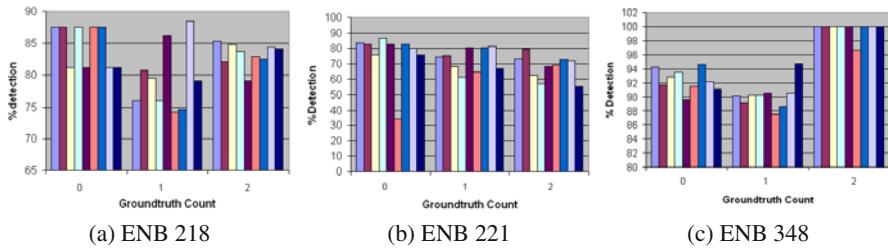
**Table 6** Variation of major parameters used in the people counting algorithm.

AR coefficient $C_{000}$	Entropy threshold ( $H_t$ )	AR window
0.5	0.3	7x7
0.5	0.3	5x5
0.5	0.3	3x3
0.5	0.25	7x7
0.4	0.3	7x7
0.6	0.3	7x7
0.6	0.3	5x5
0.4	0.3	5x5
0.4	0.3	3x3

chosen to be the same with the constraint that all coefficients sum to one. On the three laboratory sequences, we varied these parameter values as listed in Table 6.

Figure 13 plots the different detection rates as the parameters are varied, bunched together for zero, one, and two subjects for each laboratory. The correct detection of an empty room ranges from 81.2% to 87.5% for ENB218, 75.7% to 86.4% for ENB221 and 89.6% to 94.7% for ENB348. The correct detection of one subject in the room ranges from 74% to 88% for ENB218, 61% to 81% for ENB221 and 87% to 94% for ENB348. The correct detection of two subjects in the room ranges from 79.1% to 85.3% for ENB218, 55.4% to 79.5% for ENB221 and 96.7% to 100% for ENB348.

The best detection rates for ENB218 is 88.4% for 1 subject and 84.4% for 2 subjects. In this case the 5x5 window with the coefficient for the current frame 0.4 gave the best detection rates. The best detection rate for ENB221 is 75.5% for 1 subject and 79.5% for 2 subjects. In this case the 5x5 window with the coefficient for the current frame 0.5 gave the best detection rates. The best detection rate for ENB348 is 94.8% for 1 subject and 100% for 2 subjects. In this case the 3x3 window with the coefficient for the current frame 0.4 gave the best detection rates. In the



**Fig. 13** Effects of the variation of parameter values on the correct detection rates for different number of persons in the ground truth.

ENB348 dataset we have only 62 frames with 2 subjects. Hence the 100% detection rate may be lesser than shown if more frames with 2 subjects are included.

## 6 Conclusion

We presented a framework for counting mostly static people in very sparsely-sampled video (4 frames per minutes), spanning days and difficult illumination variations. This is different from most of the work that has been done in people counting, which has focused primarily on pedestrian counting or counting people indoors under static illumination conditions. In the past, the majority of the work in people counting has been done using overhead cameras but in our case we have used cameras that are placed horizontally.

The issue of people counting was handled at both the pixel level using photometry-based normalization, and at the feature level, by exploiting the spatio-temporal coherence that is present in the image sequence. The grouping was conducted using the XYT volume of data to exploit the temporal coherence present in motion. We worked with two types of indoor settings: home and laboratory. The datasets sometimes had drastically varying illumination conditions under which the algorithm presented in this chapter generated the right counts. The results are very encouraging. The algorithm is robust with respect to illumination changes. Counts can be calculated under severe illumination conditions. People counts can be calculated even if they are partially visible. Counts are generated for any moving object in the scene. It does not try to distinguish between humans and non-humans. Counting errors are concentrated around frames with large motion events, such as a person moving out of a scene.

## References

1. Beymer, D.J.: Person counting using stereo. In: Workshop on Human Motion, pp. 127–134 (2000)

2. Boyer, K., Sarkar, S.: *Perceptual Organization for Artificial Vision Systems*. Kluwer (March 2000)
3. Celik, H., Hanjalic, A., Hendriks, E.: Towards a robust solution to people counting. In: IEEE International Conference on Image Processing, pp. 2401–2404 (October 2006)
4. Conrad, G., Johnsonbaugh, R.: A Real-time People Counter. In: ACM Symposium on Applied Computing, pp. 20–24. ACM, New York (1994)
5. Fehr, D., Sivalingam, R., Morellas, V., Papanikopoulos, N., Lotfallah, O., Park, Y.: Counting people in groups. In: IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 152–157 (September 2009)
6. Haralick, M., Shapiro, G.: *Computer and Robot Vision*. Addison-Wesley Longman Publishing Co., Inc. (1992)
7. Haritaoglu, I., Flickner, M.: Attentive billboards. In: Conference on Image Analysis and Processing, pp. 162–167 (2001)
8. Hou, Y.-L., Pang, G.: People counting and human detection in a challenging situation. *IEEE Transactions on Systems, Man and Cybernetics, Part A* 41(1), 24–33 (2011)
9. Huang, D., Tommy, W., Chow, S., Chau, W.: Neural network based system for counting people. In: Industrial Electrical Society, pp. 2197–2201 (2002)
10. Jabri, S., Duric, Z., Wechsler, H., Rosenfeld, A.: Detection and location of people in video images using adaptive fusion of color and edge information. In: International Conference on Pattern Recognition, pp. 627–630 (2000)
11. Kettnaker, V., Zabih, R.: Counting people from multiple cameras. In: IEEE International Conference on Multimedia Computing and Systems, pp. 267–271 (1999)
12. Kim, J., Choi, K., Choi, B., Lee, J., Ko, S.: Real-time system for counting the number of passing people using a single camera., 466–473 (2003)
13. Kuno, Y., Watanabe, T., Shimosakoda, Y., Nakagawa, S.: Automated detection of human for visual surveillance system. In: International Conference on Pattern Recognition, p. C92.2 (1996)
14. Lee, M.: Detecting people in cluttered indoor scenes. In: Computer Vision and Pattern Recognition, pp. 804–809 (2000)
15. Mohan, A., Papageorgiou, C.P., Poggio, T.: Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23(4), 349–361 (2001)
16. Ramanan, D., Forsyth, D., Zisserman, A.: Tracking people by learning their appearance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(1), 65–81 (2007)
17. Ramoser, H., Schlogl, T., Beleznai, C., Winter, M., Bischof, H.: Shape-based detection of humans for video surveillance applications. In: International Conference on Image Processing, pp. III 1013–III 1016 (2003)
18. Rossi, M., Bozzoli, A.: Tracking and counting moving people. In: International Conference on Image Processing, pp. 212–216 (1994)
19. Schofield, A., Mehta, P., Stonham, T.: A system for counting people in video images using neural networks to identify the background scene. In: International Conference on Pattern Recognition, vol. 29(8), pp. 1421–1428 (1996)
20. Segen, J., Pingali, G.: A camera-based system for tracking people in real time. In: International Conference on Pattern Recognition, pp. 63–67 (1996)
21. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8), 888–905 (2000)
22. Stillman, S., Tanawongsuwan, R., Essa, I.: Tracking multiple people with multiple cameras. In: Workshop on Perceptual User Interfaces (1998)
23. Wong, A., You, M.: Entropy and distance of random graphs with application to structural pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 7(5), 599–609 (1985)

24. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(11), 1101–1113 (1993)
25. Yang, D.B., Gonzalez-Banos, H.H.: Counting people in crowds with a real-time network of simple image sensors. In: International Conference on Computer Vision, pp. 122–129 (2003)
26. Yoshida, D., Terada, K., Oe, S., Yamaguchi, J.: A method of counting the passing people by using the stereo images. In: International Conference on Image Processing, p. 26AP3 (1999)
27. Yu, S., Chen, X., Sun, W., Xie, D.: A robust method for detecting and counting people. In: International Conference on Audio, Language and Image Processing, pp. 1545–1549 (July 2008)
28. Zhao, T., Nevatia, R.: Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(9), 1208–1221 (2004)
29. Zhao, X., Delleandrea, E., Chen, L.: A people counting system based on face detection and tracking in a video. In: IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 67–72 (September 2009)

# Scene Invariant Crowd Counting and Crowd Occupancy Analysis

David Ryan, Simon Denman, Sridha Sridharan, and Clinton Fookes

**Abstract.** In public places, crowd size may be an indicator of congestion, delay, instability, or of abnormal events, such as a fight, riot or emergency. Crowd related information can also provide important business intelligence such as the distribution of people throughout spaces, throughput rates, and local densities. A major drawback of many crowd counting approaches is their reliance on large numbers of holistic features, training data requirements of hundreds or thousands of frames per camera, and that each camera must be trained separately. This makes deployment in large multi-camera environments such as shopping centres very costly and difficult. In this chapter, we present a novel scene-invariant crowd counting algorithm that uses local features to monitor crowd size. The use of local features allows the proposed algorithm to calculate local occupancy statistics, scale to conditions which are unseen in the training data, and be trained on significantly less data. Scene invariance is achieved through the use of camera calibration, allowing the system to be trained on one or more viewpoints and then deployed on any number of new cameras for testing without further training. A pre-trained system could then be used as a turn-key solution for crowd counting across a wide range of environments, eliminating many of the costly barriers to deployment which currently exist.

## 1 Introduction

In large public places such as railway stations, airports, shopping centres and sporting events, it is often not possible to monitor every person's individual behaviour due to crowd size. Instead, crowd properties such as the distribution of people throughout the space, throughput rates and local densities can be monitored.

---

David Ryan · Simon Denman · Sridha Sridharan · Clinton Fookes  
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia  
e-mail: david.ryan@qut.edu.au, s.denman@qut.edu.au,  
s.sridharan@qut.edu.au, c.fookes@qut.edu.au

While an individual is capable of causing damage, a criminal will usually choose to do so in an uncrowded environment where he will not be caught. The threats posed in crowded environments are of a different nature, and arise from the crowd's collective properties: "a crowd is something other than the sum of its parts" [12]. These threats include fighting, rioting, violent protest, mass panic and excitement. The most common indicator of such behaviour is crowd size.

Even in peaceful crowds, size may be an indicator of congestion, delay or other abnormality. Crowd related information can also be used to provide important business intelligence. For example, the distribution of crowds throughout a shopping complex or large retail store may be used to analyse consumer shopping patterns, while the overall crowd size may be monitored to assess store performance over time. These measurements are difficult to collect and to quantify without employing considerable manpower, therefore researchers have turned to computer vision based surveillance technologies to collect this data automatically from closed-circuit television footage.

As crowd size is a *holistic* description of the scene, the majority of crowd counting techniques have utilised holistic image features to estimate crowd size. A major drawback of the holistic approach is the large amount of training data required. Due to the wide variability in crowd behaviours, distribution, density and overall size, it becomes necessary to annotate a very large number of frames in order to achieve proper generalisation. In a facility containing numerous cameras, each viewpoint must be trained independently. It is not practical to supply hundreds of frames of ground truth for every viewpoint.

In this chapter, we propose a novel scene-invariant approach that uses *local* features, which are specific to individuals and groups within an image, to estimate the crowd size and its distribution across a scene. While existing techniques have used similar features such as foreground pixels, they are analysed at a holistic level. Local features are used here to estimate the number of people within each *group*, so that the total crowd estimate is the sum of all group sizes. Because local features are used, training data must also be annotated with local information. We propose a unique method of localised ground truth annotation which allows the system to be trained rapidly, and greatly reduces the required training data.

Even though large-scale CCTV networks are becoming increasingly common, automated crowd counting is not widely deployed. One of the largest barriers to full deployment of this technology is the requirement to train each camera independently, which is both time-consuming and expensive. The algorithm proposed in this chapter uses camera calibration to scale features between multiple viewpoints, by taking into account the relative sizes of objects in these scenes. This results in a scene-invariant crowd counting system which may be trained on one camera and then deployed for counting on another. In this chapter, we train our system on a large bank of data from various cameras, before testing it on a new viewpoint. In practice, a system which has been pre-trained on numerous camera viewpoints can operate as a turn-key solution for crowd counting across a wide range of unseen environments. Whenever more training data is annotated, the system may be re-trained on this

larger training database, and then each deployment of the system can be upgraded accordingly.

The proposed system is tested on eight crowd counting datasets, including the UCSD database [7], PETS 2009 [2], PETS 2006 [1], and a custom database collected from our university campus (Figure 1). These datasets feature crowds of size 1-45 people, and capture a wide variation in scene properties, including lighting conditions, lens distortion, camera angle and camera distance. The proposed technique is compared to two holistic techniques, and is shown to outperform holistic techniques in terms of accuracy, scalability and practicality. The system is shown to be highly scalable, as it is capable of extrapolating to crowd sizes which are smaller or larger than those encountered during training; and highly practical, as it can count crowds when trained on as few as 10 frames of training data. Finally, the system is demonstrated to be scene invariant by performing training and testing on different datasets.

The remainder of the chapter is structured as follows: Section 2 provides an overview of existing crowd counting techniques, Section 3 outlines the proposed algorithm, Section 4 presents experimental results and Section 5 presents conclusions and directions for future work.

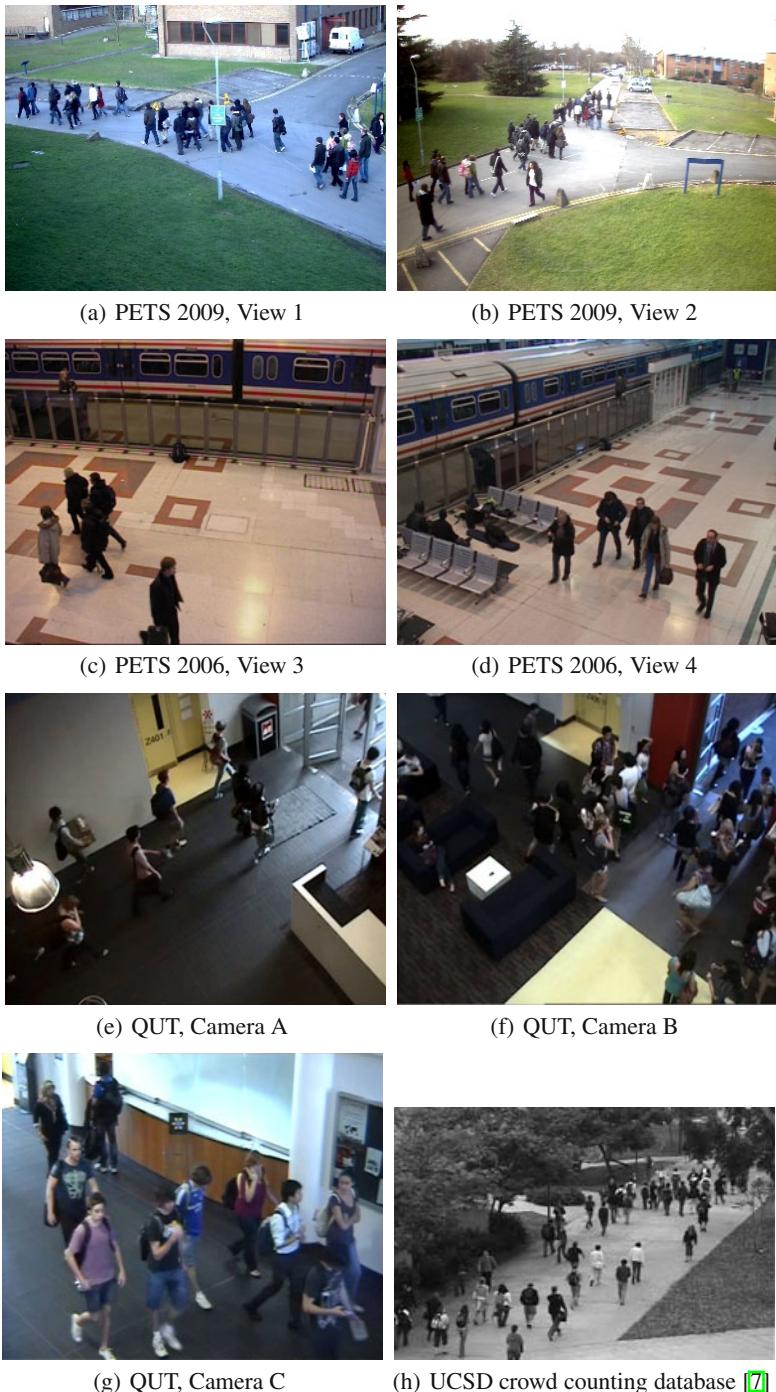
## 2 Background

Computer vision techniques to estimate crowd size generally involve feature extraction followed by a classification or regression stage. Important considerations include the quantity of features extracted, the complexity of the classifier, computation time required, and the size of the training data set. Each new feature introduces an additional level of dimensionality and therefore necessitates a wider set of training data, which reduces a system's practicality.

Background subtraction is a typical first step in automated visual surveillance, and has been used extensively in crowd counting because the foreground pixels in the image usually correspond to humans [12, 10, 30, 29, 25, 20, 6, 19, 37]. A number of other features have been employed such as textural information [35, 28, 27, 32, 42], while some systems use a combination of these features [7]. Earlier systems sought only to classify a crowd's size on a five-point scale (from 'very low' to 'very high' density) whereas recent approaches attempt to estimate the actual number of pedestrians in a scene.

Because crowd size is a holistic description of the scene, a common approach is to extract holistic image features which ideally describe its level of crowdedness. The basis for this approach is summarised by Davies [12]:

Our objective for the models is that they should not involve actual counting of individuals or tracking of the movements of individuals but should be based on a collective description of crowds (e.g. analogous to the ideal-gas theory which ignores individual molecules).



**Fig. 1** Eight datasets were used to evaluate our crowd counting algorithm.

The following review covers two general categories of crowd size monitoring:

1. Holistic approaches which utilise image features to obtain an estimate. These can also be described as “mapping-based” approaches, because they map directly between the feature space and the crowd size estimate. This works best for large crowds, in which the analogy to the ideal-gas theory holds. Holistic approaches are discussed in Section 2.1
2. Local approaches which utilise local image features. These may sometimes be described as “detection-based” approaches, because they detect, track or otherwise classify pedestrians on an individual or group level. Local approaches are discussed in Section 2.2

## 2.1 Holistic Approaches

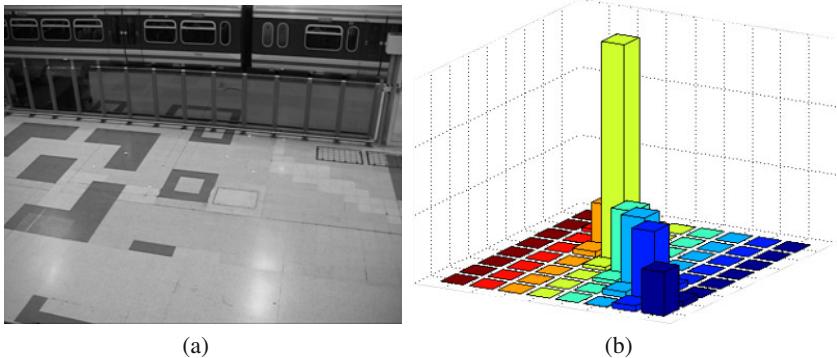
An early system developed by Marana [28, 26] used holistic image features for crowd density estimation, derived from the textures present in the image. The textures present in low density scenes tend to be coarse whereas those in high density scenes are fine. For this algorithm, it is assumed that the scene’s background is relatively smooth compared to the human textures in the foreground. When considered with this assumption in mind, textures are helpful because the introduction of human crowding will disrupt those textures of the background. While textural information will be altered by the presence of crowding, it cannot explicitly segment the foreground from the background.

Haralick [16] proposed a number of well-known textural statistics, derived from the Grey Level Cooccurrence Matrix (GLCM), which measures the quantity of co-occurring pixel values in a greyscale image  $I$ , at a specified offset  $\delta = (\delta_x, \delta_y)$ :

$$G(r, c) = \sum_{(x,y) \in I} \begin{cases} 1 & \text{if } I(x, y) = r \text{ and } I(x + \delta_x, y + \delta_y) = c \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

A three dimensional illustration of the GLCM is depicted in Figure 2. It can be seen that the GLCM is a histogram of grey level cooccurrences, and that for a low frequency image such as the background in Figure 2(a), the histogram bins are greatest along the diagonal because grey levels of equal value frequently occur beside one another. When normalized (by dividing by the number of pixels in the image) the GLCM,  $G$ , becomes a second-order joint conditional probability density function,  $f$ , such that  $f(r, c)$  is the probability of the grey levels  $r$  and  $c$  occurring beside one another.

From the normalized GLCM, holistic textural properties may be calculated. Those proposed by Haralick [16] include: contrast, homogeneity, energy and entropy. Using four different offset directions, Marana [28] applied these properties to provide a total of sixteen holistic features for each image. The mapping from feature space to crowd density was performed using a Kohonen self organising map neural



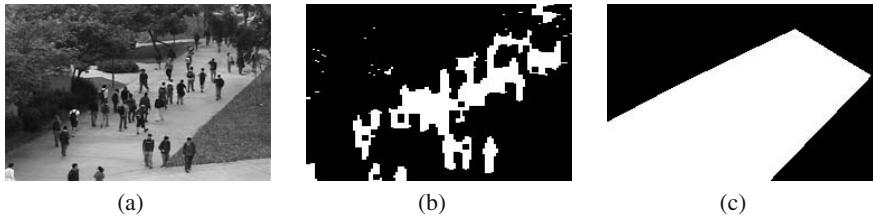
**Fig. 2** (a) Background image from PETS 2006 database [1]. (b) Three dimensional representation of the corresponding GLCM with  $\delta = (1, 0)$ , after the image is quantized to 8 grey levels.

network. Five levels of crowd density were considered, from ‘very low’ to ‘very high’ density, and a correct classification rate of 82% was reported. Other texture-like holistic features include Minkowski Fractal Dimension [27] and Translation Invariant Orthonormal Chebyshev Moments [32]. These achieve similar results using a five-point scale of crowd density, in the range of 70-90% correct classification.

Holistic features such as these are highly sensitive to external changes, such as lighting conditions. Consequently, the system would have to be re-trained after any significant changes in the environment took place. For an indoor environment, it may be desirable to position a large number of cameras throughout a facility. Using holistic features such as these would require that each camera be trained independently, perhaps on hundreds of frames or more. For outdoor environments, the natural fluctuations in lighting between morning and afternoon have been shown to reduce system performance [32].

Xiaohua [43] proposed a pre-processing stage of histogram equalisation to compensate for changing lighting conditions and camera gain. This involves non-linearly mapping the pixel intensities to fully occupy the available range. Xiaohua also proposed the use of the 2D discrete wavelet transform (DWT) as a basis for extracting textural features. Correct classification of 95% was obtained on a testing database of 150 images, demonstrating improved performance compared to previous holistic methods.

A number of algorithms attempt to segment the foreground using background subtraction techniques. Davies [12] utilised a static ‘reference’ background image of a scene in which crowd levels were to be monitored. The reference image was subtracted from each frame before applying a threshold to extract a foreground boolean mask (similar to the one shown in Figure 3(b)). The relationship between the number of people in the scene and the total number of foreground pixels was approximately linear, while the correlation with the total number of edge pixels was not as strong.



**Fig. 3** (a) Frame 1280 of the UCSD crowd counting database [7]. (b) Corresponding motion segmentation (foreground detection), obtained using an adaptive background model [14, 15]. (c) Region of interest for this scene.

The crowd estimate was therefore obtained by using linear regression to model the relationship between foreground pixels and crowd size, and filtering the output in time. The mean relative error was less than 8%. However, the use of a static background image means that the system is sensitive to lighting changes over longer periods of time, whether sudden or gradual. Adaptive background models such as Stauffer-Grimson [39], Zivkovic [46] and Denman [14, 15] are robust against such changes, and have been adopted in more recent crowd counting applications [25, 20, 36, 37].

In practice, the use of foreground detection and linear regression is not sufficient for counting crowds, due to the effects of perspective and occlusion. Paragios [30] introduced the use of a density estimator to account for perspective in an image. The focus of this approach was on change detection rather than crowd counting. Perspective was also considered by Ma [25], who computed a ‘density map’ from four points in an image corresponding to the corners of a rectangle on the ground plane. The density map weighted each pixel according to the area it represented on the ground plane. Hou [17] utilised this approach and estimated the crowd size using a neural network, with the weighted foreground pixel count as an input.

Kong [20] proposed the use of histogram based features to capture the various levels of occlusion present in a scene. The foreground detection result was divided into distinct segments, commonly referred to as ‘blobs’, using a connected components algorithm. A blob size histogram was then populated to capture the range of object sizes in the image. It is expected that individual pedestrians and small groups contribute to the lower histogram bins, while groups of larger sizes contribute to the higher ones. The histogram is a holistic description of the scene, but it captures more directly the range of blob sizes present in the image.

Kong also used the Canny edge detector [5] to extract edge pixels and their angle of orientation. These pixels are masked by the foreground so that those edges in the background are ignored. An edge angle histogram is constructed with eight bins between  $0^\circ$  and  $180^\circ$ . The edge orientation histogram “can distinguish edges caused by pedestrians, which are usually vertical, with other scene structures such as noise, shadows and cars” [20]. There is support for this statement in other visual surveillance research. For example, Dalal [11] described a similar descriptor called

the histogram of oriented gradients (HOG), which uses edge detection to populate a histogram of edge orientations for the explicit purpose of human detection.

Wu [42] used the textures present inside ‘multi-resolution density cells’, spaced across an image at various locations. Within each cell, the textural features used previously by Marana [28] were calculated from the GLCM, enabling the system to estimate crowding for that particular region of the scene. The purpose of this system was to detect local abnormality due to overcrowding or undercrowding. These conditions were detected using a support vector machine (SVM) classifier.

A unique segmentation technique was used by Chan [7, 9] to identify foreground motion. The segmentation is based on dynamic textures (an extension of textures into the temporal domain). For the purpose of crowd counting, Chan considers an outdoor pathway on which pedestrians were classified as walking either toward or away from the camera. Treated as different instances of dynamic textures, these two classes of pedestrian were segmented from the background and from one another.

Chan extracts a large number of holistic image features from the foreground mask for each direction, including foreground area, perimeter pixel count, edge orientation histogram and textural features. In total, 29 features are extracted and Gaussian Process regression is used to predict the number of pedestrians walking in each direction. While this algorithm counts crowds with high accuracy, it comes at the expense of additional training data requirements. The implementation described by Chan utilised 800 frames of training data, which were manually annotated with ground truth (the number of pedestrians moving in each direction). This would be a burdensome task to perform for every camera in a large facility where crowd size monitoring was required.

Also, dynamic textures can only segment moving pedestrians, and not those who have stopped in the middle of the scene. Pedestrians stop frequently in surveillance footage, and this can even be caused by excessive congestion, which is what we seek to detect in the first place. An adaptive background model such as [15] can continue to detect stationary objects for some time after they have come to a stop.

## 2.2 Local Approaches

Local approaches utilise detectors or features which are specific to individuals or groups of people within an image. These groups are independently analysed, so that the total crowd estimate is the sum of its parts.

Lin [23] has proposed the use of head detection for crowd counting. The Haar wavelet transform (HWT) is used in conjunction with the Support Vector Machine (SVM) to classify head-like contours as either a human head or not. This approach may be useful in dense crowds where only the head of each individual is visible.

Dalal [11] introduced the histogram of oriented gradients to represent images, using an SVM classifier to detect humans. Tuzel [41] introduced the use of covariance features as human descriptors, which may be represented as a connected Reimannian manifold. Classification is then performed using a boosting algorithm.

Celik [6] proposes a person-counting algorithm which does not require training. It assumes proportionality between the number of pixels within a blob segment and the number of people represented by that segment, in order to obtain an estimate for each group. Kilambi [18, 19] models a group of pedestrians as an elliptical cylinder, assuming a constant spacing between people within the group. Tracking a large blob over several frames increases the robustness of the group size estimate. It is unclear how the cylinder model assumption would hold up in larger crowds under various configurations, such as that shown in Figure 3(b).

Rabaud [31] used a parallelised LKT tracker [38] to determine partial tracks of interesting feature points across an image. These trajectories are then conditioned and clustered in order to estimate the number of pedestrians walking in a scene. Similarly, Yang [44] used the KLT tracker to establish trajectories of people entering a door from an overhead camera; in this case, counting was performed as a cumulative total over time, rather than in a single image.

Lempitsky [22] proposes an object counting algorithm which sought to estimate a *density function*  $F$ , as a function of the pixel intensities in an image, so that integrating the density over any region would yield the number of objects in that region. This is a localised approach in which every pixel  $\mathbf{p}$  is represented by a feature vector  $\mathbf{x}_\mathbf{p}$ , containing local foreground and gradient information. A linear model is used to obtain the density at each pixel,  $F(\mathbf{p}) = \mathbf{w}^T \mathbf{x}_\mathbf{p}$ , so that the count is obtained across a region of interest  $R$  by integrating over  $F$  as follows:  $\sum_{\mathbf{p} \in R} F(\mathbf{p})$ .

### 3 Scene Invariant Crowd Counting

Existing approaches to crowd counting are scene specific, as they are designed to operate in the same environment that was used to train the system. Because crowd size is a *holistic* description of a scene, the majority of crowd counting techniques have utilised holistic features to estimate crowd size. However, due to the wide variability in crowd behaviours, distribution, density and overall size, holistic systems require a very large training set in order to generalise properly. Indeed, some methods have substantial training requirements of hundreds [12, 20] or even thousands [9] of frames. It would not be practical to annotate this many frames for each installation.

In this chapter, a novel scene-invariant approach is presented which uses *local* features, defined here as features which are specific to an individual or group within an image. While existing techniques have used similar local features such as foreground pixels, they are analysed at a holistic level. Local features are used here to estimate the number of people within each *group*, so that the total crowd estimate is the sum of all group sizes. As local features are used, training data must also be annotated with local information. To provide appropriate training data, a unique method of localised ground truth annotation has been developed which greatly reduces the required training data.

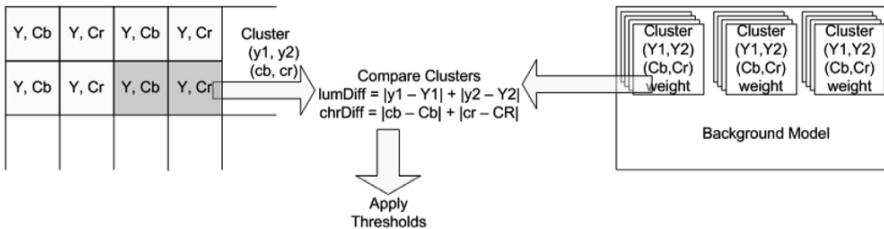
As well as the reduced training requirement, a localised approach also enables the estimation of crowd densities at different locations *within* the scene (unlike holistic systems, which can only provide a density for the whole scene), and allows for a simplistic extension to a multi-camera environment. The ability to determine local crowd densities greatly improves the systems ability to detect abnormalities in a scene. While the overall number of people in a scene may be considered normal, there may be an abnormally high concentration of people in a small area. Holistic systems are unable to detect such an abnormality, however the proposed local approach can detect such an occurrence.

The proposed approach also utilises camera calibration to achieve scene invariance by scaling features appropriately between viewpoints. This enables the system to be deployed on different training and testing sets.

The remainder of this section is structured as follows: Section 3.1 describes the background subtraction and group detection algorithm; Section 3.2 discusses camera calibration, perspective normalisation and how this relates to scene invariance; Section 3.3 explains the feature extraction process; Section 3.4 presents a ground truth annotation strategy that greatly simplifies the training process; Section 3.5 details the chosen regression model; and Section 3.6 proposes an extension to our system which uses group tracking to refine and improve the crowd size estimate.

### 3.1 Background Subtraction and Group Detection

The crowd counting system presented in this chapter has been developed using local rather than holistic features. These features are ‘local’ with respect to the blob segments in a foreground mask, obtained using a foreground segmentation technique proposed by Denman [13, 14, 15].



**Fig. 4** Pixel pairs are grouped into clusters. The adaptive background model contains a group of clusters, each assigned a weight indicating its likelihood, against which incoming clusters are compared.

This background segmentation routine operates in the YCbCr 4:2:2 colour space, which provides some invariance to lighting changes through the separation of colour and intensity. Each pixel in the incoming image,  $I$ , has two values: a luminance and a single chrominance, which alternates between blue chrominance and red chrominance in the horizontal direction (Figure 4). Pixels are paired horizontally so that

for each pair there are four values (two luminance, one blue chrominance and one red chrominance):

$$P(i, j, t) = [y_1, y_2, c_b, c_r] \quad (2)$$

where  $P(i, j, t)$  denotes a pixel pair, or ‘cluster’, formed by grouping the two pixels,  $I(2i, j, t)$  and  $I(2i + 1, j, t)$ . This pairing results in motion detection being effectively performed at half the horizontal resolution of the original image, with the benefit being increased speed.

A multi-modal background model is then constructed, for each pixel pair, by storing a set of possible modes representing the distribution of colours at that location (Figure 10). These are stored as a group of clusters, each accompanied by a weight,  $w_k$ , where  $k$  is used to denote the mode. The weight describes the likelihood of the colour described by that cluster being observed at that position in the image. Each cluster in the background model is represented by:

$$C(i, j, t, k) = [Y_{1k}, Y_{2k}, C_{bk}, C_{rk}, w_k] \quad (3)$$

Clusters in the background model are stored in order of highest to lowest weight. Incoming clusters,  $P(i, j, t)$ , are compared to all possible modes,  $C(i, j, t, k)$ , to determine a match. A match is found by finding the highest-weighted mode which satisfies:

$$|Y_{1k} - y_1| + |Y_{2k} - y_2| < T_{Lum} \quad (4)$$

$$|C_{bk} - c_b| + |C_{rk} - c_r| < T_{Chr} \quad (5)$$

where  $T_{Lum}$  and  $T_{Chr}$  denote the luminance and chrominance thresholds, respectively. Foreground motion is detected if the probability of the matching mode,  $m$ , falls below a threshold,  $T_{fg}$ :

$$F(i, j, t) = \begin{cases} 1 & \text{if } \sum_{k=0}^m w_k < T_{fg} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

The matching cluster in the background model is adjusted to reflect the current pixel colour, and the weights of all clusters in the model at this location are adjusted and normalised to reflect the new state [15]. If no match is found, then the lowest weighted cluster is replaced with a new cluster representing the incoming pixels (and foreground is detected at this location). Clusters are gradually adjusted and removed as required, allowing the system to adapt to slow changes in the background.

In surveillance situations, particularly outdoor scenarios, lighting levels can also change rapidly resulting in large amounts of erroneous motion. When these levels fluctuate, it is the luminance values in an image which undergo significant change, whereas chrominance values remain relatively unchanged. Therefore, to improve performance in real world lighting conditions the luminance threshold  $T_{Lum}$  can be adjusted. It would be ideal to be able to use a single value to adjust for the luminance change in a given frame. However, as the luminance change is not constant across a

scene, images are divided into several small regions, and each is treated separately. (In our application, images are divided into a  $5 \times 5$  grid of sub-regions.)

Thresholds for detection are varied within each region, according to the lighting conditions in that part of the scene. We define the luminance difference,  $\Delta_{Lum}$ , at a cluster to be:

$$\Delta_{Lum} = |Y_{1m} - y_1| + |Y_{2m} - y_2| \quad (7)$$

where  $Y_{1m}$  and  $Y_{2m}$  denote the luminance values of the matching mode  $m$ , and  $y_1$  and  $y_2$  are the luminance values of the incoming cluster. Attaching coordinate and time information, we use  $\Delta_{Lum}(i, j, t)$  to represent the luminance difference at a specific frame and location. Thus the weighted average of luminance changes is calculated across an image region,  $R$ :

$$O_{Lum}(R, t) = \frac{\sum_{(i,j) \in R} \Delta_{Lum}(i, j, t) \times w_m(i, j, t)}{\sum_{(i,j) \in R} w_m(i, j, t)} \quad (8)$$

The use of weighted sum allows pixels that are only recently created, potentially under the present lighting conditions, to be weighted less relative to those that have been present longer. An acceptable range for the luminance offset at time  $t$ , with respect to the previous frame ( $t - 1$ ), is defined:

$$\chi \leq \frac{O_{Lum}(R, t)}{O_{Lum}(R, t - 1)} \leq \frac{1}{\chi} \quad (9)$$

where  $\chi \in [0, 1]$  is the change threshold for the luminance offset. If the change in luminance offset falls outside of this acceptable range, a rapid fluctuation in luminance has been detected across the region, and equation 4 is modified as follows:

$$|Y_{1k} - y_1| + |Y_{2k} - y_2| < T_{Lum} + O_{Lum} \quad (10)$$

where  $O_{Lum}$  is the luminance offset of the region to which the cluster being matched belongs. Loosening the threshold enables improved performance when dealing with both global lighting changes (such as changes in camera gain), or local changes such as variable cloud cover. This approach is robust in various environments, including both indoor and outdoor scenes such as those shown in Figure 1. The full details of this background model are presented in [13, 14, 15].

Following foreground detection, a morphological closing operation is applied to the binary mask in order to obtain ‘cleaner’ and less fragmented blob segments. A connected components algorithm is subsequently run to identify the locations of each blob in an image.

In the proposed crowd counting algorithm, a crowd estimate is obtained for each blob in an image, so that the total estimate for the scene is the sum of the estimates for each individual blob. In order to train the system, ground truth annotation is performed *after* the first stage of image processing, once the foreground is extracted. The group size is explicitly labeled for each blob in an image, therefore each frame

provides several instances of ground truth. The details of this annotation process are discussed in Section 3.4.

This approach is built on the intuition that it is easier for a system to estimate the number of people in each group than to estimate the entire crowd at once. It is possible for a crowd of twenty people to be distributed as two large groups or as ten pairs (for example). Viewed from a holistic perspective, these various crowd distributions can give rise to vastly different image features. Existing techniques cope by extracting a larger quantity of holistic features (for example, 29 features are used by Chan [7]), necessitating more training data and intensive classification strategies. We have found that the relationship between image features and group size is more reliable and consistent when analysed on a local scale. Results comparing the local and holistic approaches are presented in Section 4.1.

### 3.2 Camera Calibration and Perspective Normalisation

Before local features can be used for crowd counting, the effects of perspective and camera distortion must be taken into account. The algorithm presented in this chapter is also designed to operate over multiple viewpoints, therefore the features selected to represent a person or group should be invariant to camera distance and illumination properties. Indeed, these properties are also desirable in a single-viewpoint system for the following reasons:

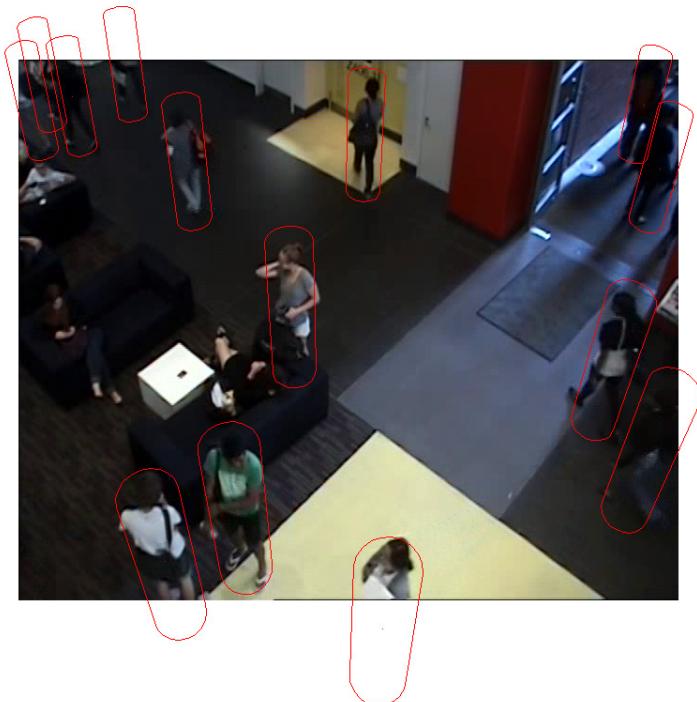
1. Certain objects appear closer to the camera than others.
2. The angle of observation from the camera to an object, with respect to the ground plane, is not constant throughout an image. It can vary greatly, particularly when the camera is placed close to the scene (Figure 5).
3. Illumination can change within a scene over time.

It is therefore important that features are normalised appropriately so that the trained system can count other objects within the scene as well as objects in other scenes. Two methods are proposed to achieve this:

1. Camera calibration is used to compensate for changes in camera positioning.
2. Recent advances in adaptive background modeling are used to compensate for changes in illumination. (Section 3.1)

A number of camera calibration methods have been described [3, 40, 45], although the most popular of these is Tsai's model [40], which is frequently used on visual surveillance databases such as PETS 2006 and PETS 2009 [1, 2]. Tsai's model incorporates camera position, rotation angle, focal length and radial lens distortion parameters to map between the real-world coordinate system ( $x, y, z$ ) and the image plane coordinate system ( $i, j$ ). These parameters are estimated from a manually-specified set of point correspondences between image pixels and real-world locations on the ground plane.

In addition to Tsai's calibration model, a number of automated procedures exist for estimating camera calibration based on human or object tracking [4, 24, 21]. For example, Lv [24] proposes a camera calibration model using vanishing points,



**Fig. 5** Ground truth annotations on Camera B of the QUT dataset. A camera calibration technique [40] is used to project a human-sized cylindrical object into the scene. Perspective normalisation is performed by compensating for the area of these projections at any location in the image.

and describes a self-calibration method based on moving humans where the head and feet positions can be located in multiple frames. Similarly, Krahnstoever [21] presents a Bayesian autocalibration algorithm which includes uncertainty estimates for each of the camera parameters. These approaches could readily be incorporated into our proposed framework to create a truly ‘turn-key’ crowd monitoring system. However, as Tsai calibration parameters are already available for public visual surveillance datasets, and the method is widely used and well understood, we continue to use this model in our experiments.

As outlined in Section 2.1, a common approach to perspective normalisation is to calculate a density map which assigns to each pixel a weight to compensates for perspective [30, 25, 9]. Typically, a reference pixel near the bottom of the image is assigned the value 1.0 and all other pixels are weighted with respect to this reference. For example, pixels higher in the image will be given a larger value because they represent a greater area in the scene.

In the proposed system, a cylinder model to approximate the size of a human, with radius  $r = 0.25$  and height  $h = 1.7$  metres. As depicted in Figure 5, this cylinder may be projected into a scene centered around any pixel  $(i, j)$ . The area of this projected shape in the image plane is denoted  $S(i, j)$ . This procedure is used to generate a density map  $D_2$  which provides a weight to each pixel inversely proportional to the projected area of an object centered at that location:

$$D_2(i, j) = \frac{1}{S(i, j)} \quad (11)$$

This density map,  $D_2$ , provides a weight to each pixel so that an object occupying a set of pixels,  $B$ , has a weighted area of  $\sum_{(i,j) \in B} D_2(i, j)$ . Consequently distant objects occupying fewer pixels are compensated by their larger weights in the density map. The calibrated density map is advantageous because it is defined in terms of real-world objects rather than an arbitrary reference pixel. This approach can scale readily between different camera angles and is inherently scene invariant.

It does not matter that the cylinder model does not match a human size or shape precisely, as its role is only to normalise 2D and 1D features across viewpoints. A number of such features are described in Section 3.3, including the weighted area of each blob. The density map  $D_2$  is suitable for such two-dimensional features as area. However, one-dimensional features such as perimeter and edges are also considered, therefore a density map  $D_1$  for these features is also defined:

$$D_1(i, j) = \sqrt{D_2(i, j)} = \frac{1}{\sqrt{S(i, j)}} \quad (12)$$

If camera calibration is unavailable, a density map can be estimated using alternative approaches such as the method described by Chan [7]. Because this approach utilises a reference pixel instead of a real-world reference (e.g. cylinder model), it is not suitable for performing *scene invariant* crowd counting. However, the system may be trained and tested on the same viewpoint, and retains all of the other benefits of a local features based approach to crowd counting.

### 3.3 Feature Extraction

Several features are extracted from each blob segment to estimate the number of people in the group. The features extracted are similar to those used by Kong [20] and Chan [7], taken at a local level rather than the holistic level. We enumerate all of the blobs in a scene's region of interest using the subscript  $n$ , so that  $B_n$  and  $P_n$  denote the set of pixels inside the  $n$ th blob, and on its perimeter, respectively. The following features are extracted from each blob:

- **Area:** The weighted area of the  $n$ th blob segment is calculated using the two-dimensional density map  $D_2$ :

$$A_n = \sum_{(i,j) \in B_n} D_2(i,j) \quad (13)$$

where  $B_n$  denotes the set of pixels in the blob. The area directly captures the size of the object, normalised for perspective.

- **Perimeter:** The weighted perimeter of the  $n$ th blob segment is calculated using the one-dimensional density map  $D_1$ :

$$L_n = \sum_{(i,j) \in P_n} D_1(i,j) \quad (14)$$

where  $P_n$  denotes the set of pixels on the blob's perimeter. The perimeter is another normalised measure of object size.

- **Histogram of Oriented Gradients:** Edges have been commonly used in previous crowd counting systems. For example, Kong [20] introduced the use of an edge angle histogram on a holistic scale, while Dalal [11] introduced the histogram of oriented gradients (HOG) for person detection. Our system utilises a similar feature, based on the horizontal and vertical derivatives:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \quad (15)$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I \quad (16)$$

where  $*$  denotes convolution between a Sobel kernel and the image,  $I$ , which has been converted to greyscale. The gradient magnitude and orientation at each pixel is therefore:

$$|G(i,j)| = \sqrt{G_x(i,j)^2 + G_y(i,j)^2} \quad (17)$$

$$\angle G(i,j) = \arctan \left( \frac{G_y(i,j)}{G_x(i,j)} \right) \quad (18)$$

For the  $n$ th blob segment, a histogram of oriented gradients  $E_n$  is constructed by allocating each pixel to one of  $H$  histogram channels, based on the pixel's orientation  $\angle G(i,j)$ . The orientation bins are evenly divided over the range  $0^\circ - 180^\circ$ , and a total of  $H = 6$  bins are used. Each pixel within the blob,  $(i,j) \in B_n$ , contributes a weighted vote to a histogram bin. The contribution (or vote) is proportional to the gradient magnitude,  $|G(i,j)|$ ; and it is also weighted by the one-dimensional density estimator  $D_1(i,j)$  to normalise for perspective. If the

value of the  $h$ th histogram bin is denoted  $E_n[h]$ , and the orientation angle for that bin is lower-bounded by  $\theta_h$ :

$$E_n[h] = \sum_{(i,j) \in B_n} \begin{cases} D_1(i,j) \times |G(i,j)| & \text{if } \theta_h \leq \angle G(i,j) < \theta_{h+1} \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

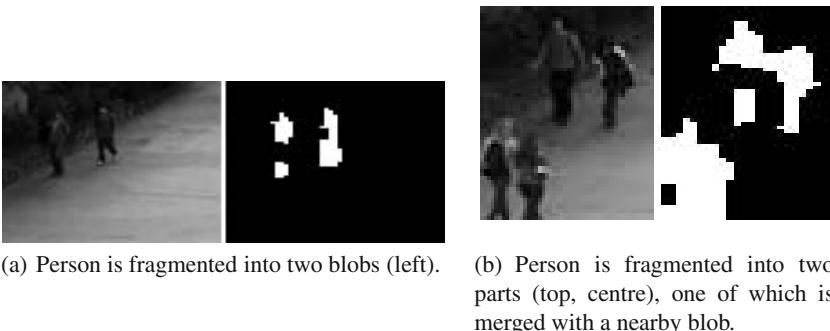
The histogram of oriented gradients is used to help distinguish between humans and other structures in the scene [20]. It also helps to identify occlusions when multiple pedestrians partially block one another from view. Although the blob's area and perimeter are reduced by such occlusions, the image gradients become stronger due to the overlapping body parts, differing skin tones and conflicting clothing.

The full feature vector for the  $n$ th blob is therefore:

$$\mathbf{x}_n = [A_n, L_n, E_n[1], \dots, E_n[H]] \quad (20)$$

### 3.4 System Training

As the proposed algorithm calculates crowd size by determining the number of people in each blob, training is performed on the local level and ground truth annotation must specify a person count for each blob. Due to imperfect foreground segmentation, some blobs are prone to errors such as splitting, fading and noise. This makes annotation difficult and tedious when attempting to allocate fractional counts (as depicted in Figure 6).

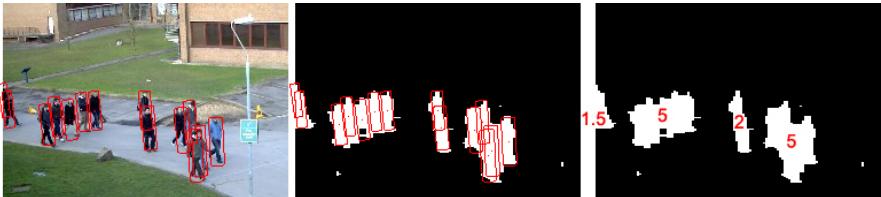


**Fig. 6** Typical errors in foreground extraction.

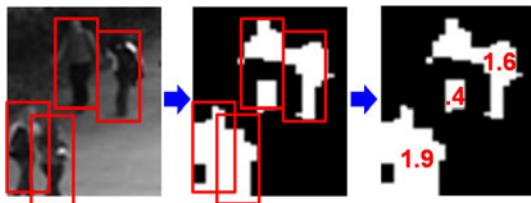
It is desirable for the ground truth to be annotated independently of the processing stage. This is done in a more conventional manner, by simply identifying the image coordinates of each person in the scene. This process is referred to as 'dotting' by Lempitsky [22] because it only requires the user to click once on the centre of each

object in the scene, thereby providing a ‘dot’ annotation. The surrounding region of a person is then approximated by the outline of a cylinder model (Figures 5–7).

The *blob* annotations are then performed automatically by the system, by assigning the annotated pedestrians to their corresponding foreground blobs. This is done by considering the overlap between foreground blobs and the pedestrian bounding regions. For example, in the case of large groups, multiple bounding regions will overlap the same blob (Figure 7(a)). On the other hand, when blob fragmentation occurs, multiple blobs will overlap a single bounding region (Figure 7(b)).



(a) Annotations on the PETS 2009 dataset, View 1 [2].



(b) Annotations on the UCSD dataset [7], demonstrating how incorrect foreground segmentation is handled.

**Fig. 7** The ground truth annotation process. Manual annotations (left) are overlaid on the foreground segmentation results (centre), and the region overlaps are used to automatically determine ground truth counts for each blob (right). Tiny blobs (noise) are assigned zero.

Using set notation, we define a number of regions as sets of pixels in Table 1. The region of interest mask is denoted  $M$ , while the foreground detection result is denoted  $F$ , such that their intersection  $B = M \cap F$  contains the set of blobs  $\{B_n\}$ . Each annotated person has a surrounding region  $R_i$  approximated by a cylinder model, from which we calculate the following values:

- $Q_i$ : the ‘quantity’ of person  $i$  within the scene’s region of interest:

$$Q_i = \frac{|M \cap R_i|}{|R_i|} \quad (21)$$

- $C_{in}$ : the ‘contribution’ of person  $i$  to blob  $n$ :

$$C_{in} = \frac{|R_i \cap B_n|}{|R_i \cap B|} \times Q_i \quad (22)$$

**Table 1** Various regions in an image. Regions are treated as sets of pixels, and set notation is used.

Notation	Description
$M$	Mask of scene (region of interest/ROI).
$F$	Foreground pixels detected using an adaptive background model [14].
$B$	Foreground pixels within the ROI mask, i.e. $B = M \cap F$ . Consists of blobs $\{B_n\}$ .
$B_n$	Blob $n$ within $B$ , where $B = \bigcup_n B_n$ .
$R_i$	Bounding region of person $i$ . (This may be inside the ROI, partially inside at the edge, or outside.)
$R_i \cap B_n$	The foreground pixels inside $R_i$ belonging to blob $B_n$ , of which there are $ R_i \cap B_n $ .
$R_i \cap B$	The foreground pixels inside $R_i$ , of which there are $ R_i \cap B  = \sum_n  R_i \cap B_n $ .

- $f_n$ : the total number of people represented by blob  $n$ . This is the sum of ‘contributions’ from all pedestrians to blob  $n$ :

$$f_n = \sum_i C_{in} \quad (23)$$

Thus  $\{f_n\}$  are the target counts for the blobs in the scene, computed automatically from the pedestrian coordinates (‘dot’ annotations). This procedure simplifies the annotation process (as the user merely need to click once on each person in a GUI); and separates the annotation stage from the segmentation stage. A graphical depiction of this ground truth annotation process is displayed in Figure 7.

An advantage of this methodology is that small blobs generated by noise are assigned an annotation of zero, while fragmented blobs are assigned fractional counts in proportion to their size. This allows some tolerance for errors in the foreground detection result.

The holistic ground truth can be measured in two ways. We consider ‘hard’ and ‘soft’ values. Hard ground truth,  $Q_h$ , is the number of pedestrians whose manual dot annotations lie within the region of interest. This measurement introduces an ambiguity when classifying a pedestrian as either ‘in’ or ‘out’ of a region. Soft ground truth,  $Q_s$ , assigns fractional values to those pedestrians lying on the perimeter of the region of interest:

$$Q_s = \sum_i Q_i \quad (24)$$

This allows the holistic ground truth to be temporarily fractional as pedestrians enter or exit the scene’s boundary. Both measures are considered when presenting results in Section 4.

### 3.5 Regression

Section 3.3 defines a vector of features to capture the number of people within a group, while Section 3.4 describes a localised ground truth annotation methodology. To train the proposed system, a regression function must be learned using a training data set to count the number of people present in each group.

Existing approaches use linear regression [12] [20] [37], neural networks [28] [20] [36] and Gaussian Process regression [2]. Although the linear model has demonstrated good performance on single datasets, it is not clear that the relationship between the image features and crowd size is indeed linear across all operating conditions and viewpoints.

We adopt Gaussian Process regression (GPR) because it does not place any prior assumptions on the functional relationship between the features and the crowd size. Instead, GPR may be thought of as defining a distribution over functions, where inference takes place in the space of functions [33] [34].

The Gaussian Process is defined as a collection of random variables, any finite subset of which have a joint Gaussian distribution. In regression problems, we observe  $N$  samples from a training set, consisting of feature vectors  $\mathbf{X} = \{\mathbf{x}_n\}$  and targets  $\mathbf{f} = \{f_n\}$ . These terms correspond to those in Equations 20 and 23, however, in this case we use  $n$  to enumerate all of the blobs observed in the entire training dataset, rather than just one frame.

In GPR these targets are imagined as a sample from some multivariate Gaussian distribution, whose mean is typically taken to be zero:

$$\mathbf{f} | \mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \quad (25)$$

The covariance matrix,  $\mathbf{K} \in \mathbb{R}^{N \times N}$ , is obtained from the covariance function  $k(\mathbf{x}_n, \mathbf{x}_m)$ , such that  $\mathbf{K}_{nm} = k(\mathbf{x}_n, \mathbf{x}_m)$ . A Gaussian process is fully specified by its covariance function,  $k(\mathbf{x}_n, \mathbf{x}_m)$ , and the mean function  $m(\mathbf{x}) = 0$ . The covariance function expresses the covariance of outputs as a function of inputs. For example, a typical covariance function is the squared exponential:

$$k_{\text{SE}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\text{SE}}^2 \exp\left(-\frac{1}{2\ell^2} |\mathbf{x}_n - \mathbf{x}_m|^2\right) \quad (26)$$

The closer the inputs,  $\mathbf{x}_n$  and  $\mathbf{x}_m$ , to one another, the more correlated their outputs will be. The hyperparameter  $\ell$  is a characteristic length scale which represents the distance one would expect to move in the input space to produce a significant change in the output space.

Given  $N^*$  test inputs,  $\mathbf{X}^* = \{\mathbf{x}_n^*\}$ , we wish to obtain the predictive outputs  $\mathbf{f}^* = \{f_n^*\}$ . In this case,  $\mathbf{X}^*$  refers to the feature vectors of the blobs present in an image during testing. Let  $\mathbf{K}^*$  denote the  $N \times N^*$  train-test covariance matrix with  $\mathbf{K}_{nm}^* = k(\mathbf{x}_n, \mathbf{x}_m^*)$ . Similarly, let  $\mathbf{K}^{**}$  denote the  $N^* \times N^*$  test set covariance with  $\mathbf{K}_{nm}^{**} = k(\mathbf{x}_n^*, \mathbf{x}_m^*)$ .

As all variables in a Gaussian Process are normally distributed, the training and testing data sets are jointly Gaussian [34], with the following distribution:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}^* \\ \mathbf{K}^{*T} & \mathbf{K}^{**} \end{bmatrix}\right) \quad (27)$$

Each subset of these random variables also follows a joint Gaussian distribution:

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \quad (28)$$

$$\mathbf{f}^* \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^{**}) \quad (29)$$

Prediction using Gaussian Process regression is performed by conditioning the predictive outputs on the training data, with the following posterior distribution obtained for  $\mathbf{f}^*$ :

$$\mathbf{f}^* | \mathbf{f}, \mathbf{X}, \mathbf{X}^{**} \sim \mathcal{N}(\mu, \Sigma) \quad (30)$$

$$\mu = \mathbf{K}^{*T} \mathbf{K}^{-1} \mathbf{f} \quad (31)$$

$$\Sigma = \mathbf{K}^{**} - \mathbf{K}^{*T} \mathbf{K}^{-1} \mathbf{K}^* \quad (32)$$

This method provides not only point estimates,  $\mu$ , but also a matrix  $\Sigma$  of covariances for the test outputs. The diagonal elements of  $\Sigma$  can thus be used to obtain pointwise error bars.

$$\sigma_n^2 = \Sigma_{nn} \quad (33)$$

For example, setting a 95% confidence interval, the estimate for test sample  $n$  would be  $\mu_n \pm 1.96\sigma_n$ . One advantage of using error bars becomes clear when group tracking is employed in Section 3.6 as a group of people is tracked over time, the confidence of each estimate can be used to weight predictions accordingly.

For each group, the crowd size estimate is a predictive distribution. To obtain a holistic estimate, these distributions must be combined to get the total number of people in the scene. By calculating the sum of  $N^*$  Gaussian random variables, an overall prediction and variance is obtained for the scene:

$$\mu_{hol} = \sum_{n=1}^{N^*} \mu_n \quad (34)$$

$$\sigma_{hol}^2 = \sum_{n=1}^{N^*} \sigma_n^2 \quad (35)$$

The covariance function  $k(\mathbf{x}_n, \mathbf{x}_m)$  used in our system is designed to capture both short-range and long-range trends in the data. For example, the squared exponential (equation 26) captures the intuitive notion that similar inputs should produce similar outputs. In order to extrapolate the longer trends beyond the training range, the dot product covariance function [34] is also used:

$$k_{\text{DP}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\text{DP}}^2 (1 + \mathbf{x}_n^T \mathbf{x}_m) \quad (36)$$

Combining these terms results in a regression model that preserves the nonlinearities within the training range while extrapolating outside of the training range in a predominantly linear fashion. Finally, independent Gaussian noise is modeled using the term:

$$k_{\text{GN}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\text{GN}}^2 \delta(n, m) \quad (37)$$

where  $\delta$  denotes Kronecker's delta function, and contributes only to the diagonals of  $\mathbf{K}$  and  $\mathbf{K}^{**}$ . The final covariance function is therefore:

$$k(\mathbf{x}_n, \mathbf{x}_m) = k_{\text{SE}}(\mathbf{x}_n, \mathbf{x}_m) + k_{\text{DP}}(\mathbf{x}_n, \mathbf{x}_m) + k_{\text{GN}}(\mathbf{x}_n, \mathbf{x}_m) \quad (38)$$

$$= \sigma_{\text{SE}}^2 \exp\left(-\frac{1}{2\ell^2} |\mathbf{x}_n - \mathbf{x}_m|^2\right) + \sigma_{\text{DP}}^2 (1 + \mathbf{x}_n^T \mathbf{x}_m) + \sigma_{\text{GN}}^2 \delta(n, m) \quad (39)$$

The GPR is ‘trained’ by choosing the hyperparameters,  $\{\sigma_{\text{SE}}, \ell, \sigma_{\text{DP}}, \sigma_{\text{GN}}\}$ , so as to maximise the likelihood of the observed training data  $p(\mathbf{f}|\mathbf{X})$ . (Good predictive performance can still be achieved with reasonable estimates for these hyperparameters.) Following from equation [25],

$$\log p(\mathbf{f}|\mathbf{X}) = -\frac{1}{2} \mathbf{f}^T \mathbf{K}^{-1} \mathbf{f} - \frac{1}{2} \log |\mathbf{K}| - \frac{N}{2} \log 2\pi \quad (40)$$

This term is maximised using an optimisation algorithm such as conjugate-gradients, provided that  $k(\mathbf{x}_n, \mathbf{x}_m)$  is differentiable with respect to each of the hyperparameters. Once optimised, prediction is then performed using equations [30][32].

### 3.6 Tracking Module

Crowd counting algorithms have typically analysed each frame independently of one another, estimating the crowd size based on the features extracted from that frame alone. Although a temporal smoothing may be applied to the holistic count to reduce outliers [12, 36], we propose a local method which employs blob-level tracking to improve each group's estimate.

When two or more groups merge to form a larger group, for example, occlusions often occur that obscure the crowd size estimate. By tracking and counting these groups before they merge, their prior estimates can be used to anticipate the size of the newly formed group. Because occlusions are usually temporary (consider two pedestrians passing by one another on a walkway), this prior information can be used to prevent the estimate from being degraded.

Blobs are tracked as they move through a scene by detecting direct correspondences, splits and merges. This is formulated as an optimisation problem by Mansoud [29], however in this section we describe a simple set of heuristics based on

blob overlap criteria. As we are not concerned with ensuring consistent labeling of objects throughout the sequence, as is required in object tracking, a heuristic based approach that can model the merges and splits of blobs is adequate.

Denoting the  $m$ th blob in frame  $t$  as  $B_{t,m}$ , we define the overlap of two blobs in consecutive frames as the number of pixels belonging to both:

$$O_t(m, n) = |B_{t,m} \cap B_{t+1,n}| \quad (41)$$

Using this notation we track groups throughout a sequence by determining direct matches, merges and splits as follows.

- 1. Direct Match:** The first step in comparing consecutive frames is to detect direct matches between overlapping blobs. Any blob pair,  $B_{t,m}$  and  $B_{t+1,n}$ , which satisfies the following conditions is deemed a match:

$$O_t(m, n) > 0 \quad (42)$$

$$O_t(i, n) = 0 \quad \forall i \neq m \quad (43)$$

$$O_t(m, j) = 0 \quad \forall j \neq n \quad (44)$$

These criteria simply require both blobs to overlap one another exclusively.

- 2. Merging:** After direct matches have been determined, the matched blobs are removed from consideration. The system then detects P:1 merges and 1:Q splits by combining the remaining blobs as follows. A set of  $P$  blobs,  $\{B_{t,M_1}, B_{t,M_2}, \dots, B_{t,M_P}\}$ , are deemed to have merged to form the blob,  $B_{t+1,n}$ , when the following conditions are met:

$$O_t(M_p, n) > 0 \quad \forall p \in [1, P] \quad (45)$$

$$O_t(i, n) = 0 \quad \forall i \notin \{M_0, M_1, \dots, M_P\} \quad (46)$$

$$O_t(M_p, j) = 0 \quad \forall p \in [1, P], \quad \forall j \neq n \quad (47)$$

- 3. Splitting:** Similarly, a split occurs when blob  $B_{t,m}$  is divided into the set of blobs:  $\{B_{t+1,S_1}, B_{t+1,S_2}, \dots, B_{t+1,S_Q}\}$ . A split is determined when the following conditions are met:

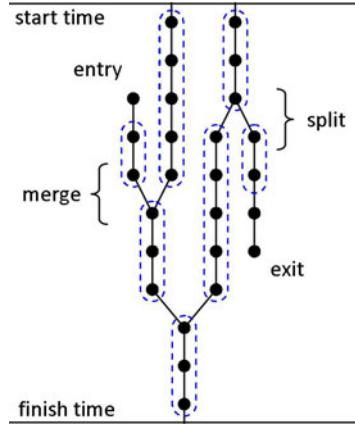
$$O_t(m, S_j) > 0 \quad \forall j \in [1, Q] \quad (48)$$

$$O_t(i, S_j) = 0 \quad \forall j \in [1, Q], \quad \forall i \neq m \quad (49)$$

$$O_t(m, j) = 0 \quad \forall j \notin \{S_0, S_1, \dots, S_Q\} \quad (50)$$

The crowd counting estimates obtained for each blob can then be improved by taking advantage of the detected tracks. The splitting and merging of blobs may be visualised using a graph structure as shown in Figure 8. As blobs enter and exit the

scene, the number of persons that they represent may change while in contact with the perimeter of the scene. Once fully inside the region of interest, however, we assume that directly-matched blobs represent a constant number of people, while merged blobs represent the sum of their constituents' group sizes.



**Fig. 8** Visualisation of blob tracking results. Groups of constant size are circled.

The estimate for the  $m$ th blob in frame  $t$  is denoted  $\mu_{t,m}$  with variance  $\sigma_{t,m}^2$ . Estimates are obtained using Equations [30][33]. A group which has been tracked across  $N$  frames, from time  $t = t_1$  to  $t = t_N$ , and containing the blobs with indices  $\{m_t\}_{t=t_1}^{t_N}$ , has an associated set of group size predictions:  $\{\mu_{t,m_t}, \sigma_{t,m_t}^2\}_{t=t_1}^{t_N}$ . It is reasonable to expect that the number of people contained in this group is constant, if it is fully contained within the region of interest. We therefore seek to obtain an improved estimate for this group size,  $\mu'_{t_N, m_{t_N}}$ , by incorporating the tracking history.

Previous experiments [37] used the mean or median value of the group's historical list of estimates, while Kilambi [19] rounded each to an integer and then took the mode. These approaches assume each estimate to be equally valid, and therefore assign an equal weighting to each. However, in practice some frames may be less reliable due to changing environmental conditions or noise, which contribute to uncertainty in the predicted group size.

The variance provides a measurement of the system's uncertainty in the group size prediction, therefore each estimate within a track is weighted by the inverse of its variance. The improved estimate for the most recent blob in a track is thus the weighted average:

$$\mu'_{t_N, m_{t_N}} = \frac{\sum_{t=t_1}^{t_N} \mu_{t,m_t} / \sigma_{t,m_t}^2}{\sum_{t=t_1}^{t_N} 1 / \sigma_{t,m_t}^2} \quad (51)$$

When two or more groups merge to form a new group, each contains a historical list of estimates and variances. A new list is formed by summing their corresponding elements and truncating the new list's length to the shortest of those being merged. The merged group adopts this list and then appends to it any subsequent estimates while it continues to be tracked. Consequently, a tracked person who is temporarily occluded from view by another group may still be represented in the crowd size estimate due to the weight of its prior history.

The tracking procedure described in this section effectively filters the group size estimates over time as the blobs are tracked. As such, it may be expected to produce a modest improvement over the underlying ‘raw’ estimates obtained using the procedure in Section 3.5. Experimental results in Section 4 indicate that a 4-8% improvement in the mean absolute error are observed.

## 4 Results

This section presents experimental results of the proposed algorithm. Section 4.1 compares the use of local features to holistic methods using the same training and testing viewpoint, and Section 4.2 presents results for scene invariant crowd counting. Eight datasets were used to evaluate our algorithm, and these are summarised in Table 2. Images from each of these datasets is shown in Figure 11.

**Table 2** Eight datasets were used to evaluate our crowd counting algorithm. The total number of frames is listed, and a subset of these frames have been annotated at regular intervals with ground truth. The interval column indicates the spacing between annotated frames.

Data set	# Frames	# Annotated	Interval	Max crowd	Calibration
PETS 2009, View 1 (13-57, 13-59)	220 + 240	46	10	32	Y
PETS 2009, View 2 (13-57, 13-59)	220 + 240	46	10	32	Y
PETS 2006, View 3 (S1)	3000	120	25	5	Y
PETS 2006, View 4 (S1)	3000	120	25	6	Y
QUT, Camera A	10400	50	200	8	Y
QUT, Camera B	5300	50	100	23	Y
QUT, Camera C	5300	50	100	10	Y
UCSD	2000	2000	1	45	N

### 4.1 Comparison of Local and Holistic Features

In this section we assess the performance of a system using local features compared to holistic methods. The proposed algorithm is trained and tested on the UCSD pedestrian database [7]. This database contains 2000 annotated frames of pedestrian traffic moving in two directions. The video has been downsampled to  $238 \times 158$  pixels and 10 fps, grayscale. An example frame is shown in Figure 3.

The performance of the proposed system is assessed using three criteria:

1. **Accuracy:** Although this system is trained on the basis of individual blobs, the testing still takes place on a holistic level. The accuracy of a system is a measure of how closely the estimate follows the ground truth. For testing purposes we consider the mean absolute error and the mean squared error. In order to obtain a direct comparison with competing algorithms, we use ‘hard’ ground truth (Section 3.4), defined as the number of pedestrians whose manually-annotated  $(x, y)$  centroids lie within the region of interest.
2. **Scalability:** Ideally, the training data should cover a wide range of scenarios, similar to those which are expected to be found during operation. In the case of crowd counting, however, we may not have access to video footage of all possible scenarios. Excessive levels of over or under crowding may not be present in the training data because these events are abnormal, and this is the reason we wish to detect them. A system which can extrapolate outside of the ranges found in the training data is of greater practical use.
3. **Practicality:** For a crowd counting system to be practical, it must be relatively easy to deploy. For real world deployment where the algorithm may be required run on several hundred different cameras within a single installation, being able to use a reduced training set is highly desirable. When training crowd counting algorithms, each training frame requires ground truth to be supplied. If several hundred training frames are needed for each camera, then the process of training becomes very tedious and time consuming. To assess practicality, systems are evaluated using reduced training sets.

The following crowd counting techniques are evaluated:

- **Proposed:** The proposed system as described in this chapter is evaluated, in which local features are extracted for each blob and ground truth annotation is performed on a local level. Because camera calibration is not included with the UCSD dataset, we approximate the density map using the relative sizes of reference pedestrians, as described in [7]. A fully calibrated setup is tested in Section 4.2.
- **Kong:** Blobs are sorted into six blob size histograms of bin width 500, as described in [20]. An edge angle histogram is also calculated, for which we use eight histogram bins between  $0^\circ$  and  $180^\circ$ . Regression is performed using a linear model and a neural network. This is a holistic system.
- **Chan:** Segmentation is performed using a mixture of dynamic textures [8], and Gaussian Process regression is used to predict the crowd size moving in each direction (away or towards), from a bank of 29 holistic features [7]. The number of pedestrians moving in each direction is then summed to get the overall crowd count. This system is referred to as ‘Chan: away+towards’. Additionally, the segmentation result for each moving class can be taken together to obtain a full foreground mask, which is then used to train the system to obtain the overall crowd count directly. This system is referred to as ‘Chan: all’. Both implementations are examples of a holistic system.

- **Lempitsky:** Each pixel is represented by a feature vector  $\mathbf{x}_p$  and a density function  $F(\mathbf{p})$  is learned so that integrating the density over any region will yield the number of objects in the region [22]. This is a local approach because dot annotations are used to train the system on a local level.

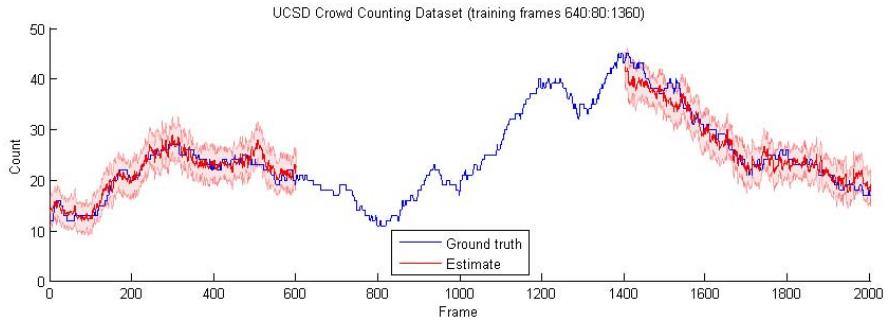
To assess the accuracy of these systems, the testing protocol of Chan [7] was adhered to. Following this protocol, frames 601-1400 of the data set were set aside for training, while the remaining 1200 frames were used for testing. Because the proposed system is trained using local feature vectors obtained from each group, rather than from each frame, only a subset of the 800 training frames were required to train the system. Lempitsky used two subsets of the training data, described using Matlab notation: 605:5:1400 and 640:80:1360.

In order to compare to Lempitsky, the proposed system is also trained using 10 frames. The proposed system is trained using the same subset used by Lempitsky, 640:80:1360, as well as two other subsets, 610:80:630 and 670:80:1390. We obtained results using these additional training subsets because it provides a more representative picture of how the proposed system performs on this dataset.

These results are all tabulated in Table 3. The mean absolute error of the proposed system is less than 2.0, performing competitively with the local approach proposed by Lempitsky [22]. By each measure of accuracy the proposed approach significantly outperforms the holistic systems, Kong [20] and Chan [7]. Incorporating tracking, as described in Section 3.6, provides a further improvement in performance. Mean absolute error is reduced by 4-8% by the inclusion of the tracking module, and mean square error is improved by 6-14%. The results of the proposed system are plotted in Figure 9.

**Table 3** Testing results on the UCSD data set. Frames 601-1400 were set aside for training, and frames 1-600 and 1401-2000 were used for testing. Mean and standard deviation are reported for the neural network based on five runs.

System	Training subset	Mean abs. Error	Mean Square Error
Kong, linear	all	1.92	5.60
Kong, neural network (5 runs)	all	$2.47 \pm 0.41$	$9.53 \pm 3.01$
Chan, away+towards	all	1.95	5.75
Chan, all	all	1.95	6.06
Lempitsky	605:5:1400	1.70	-
Lempitsky	640:80:1360	2.02	-
Proposed, no tracking	610:80:1330	1.79	4.95
Proposed, with tracking	610:80:1330	1.72	4.50
Proposed, no tracking	640:80:1360	1.33	2.91
Proposed, with tracking	640:80:1360	1.28	2.74
Proposed, no tracking	670:80:1390	1.57	3.94
Proposed, with tracking	670:80:1390	1.45	3.39



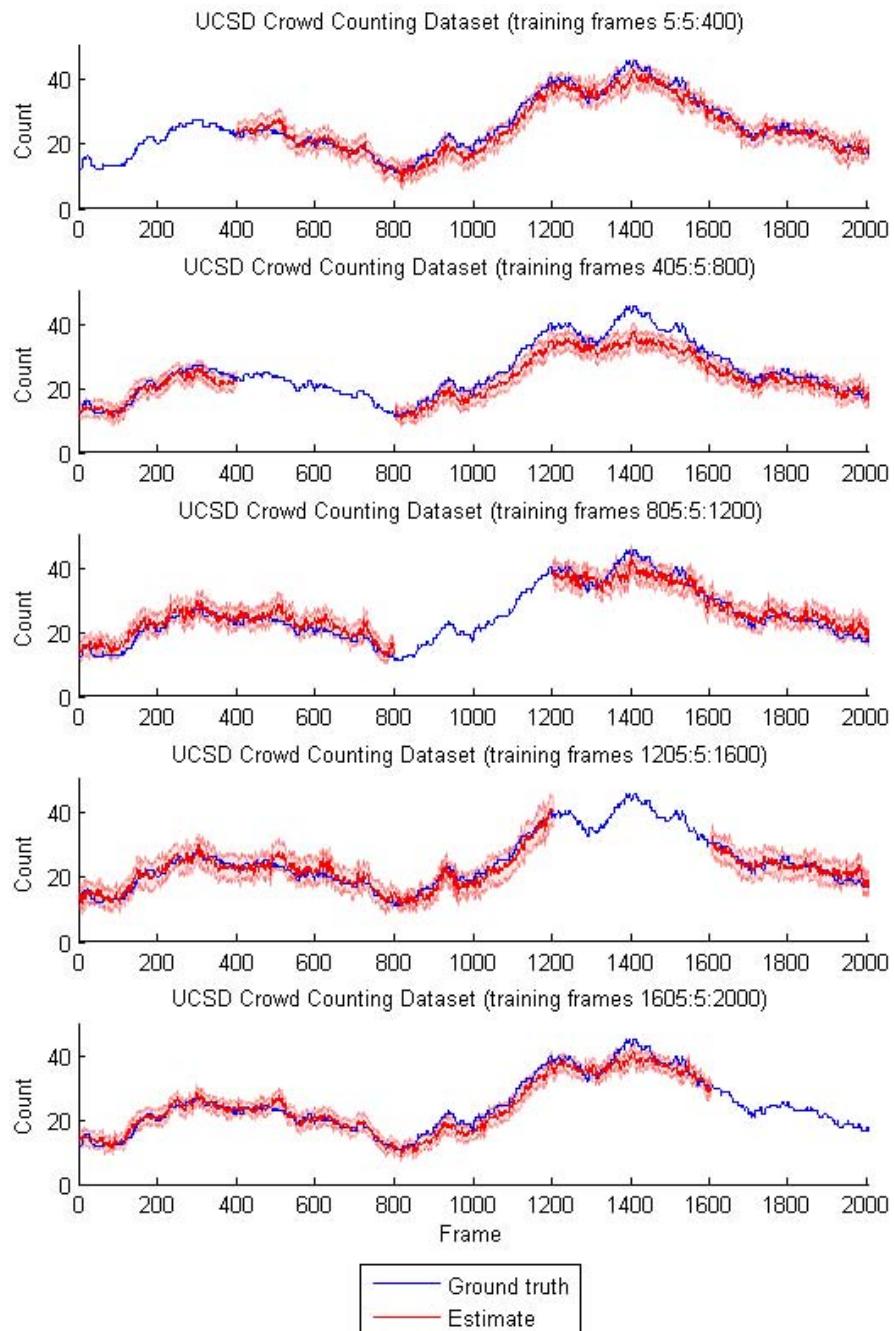
**Fig. 9** Testing results on the UCSD data set. Frames 640:80:1360 were set aside for training, and frames 1–600 and 1401–2000 were used for testing. A 95% confidence interval is drawn either side of the estimate (equations 34[35].

The ability of the system to extrapolate outside the range of crowd sizes seen in the training data set (scalability) is examined by allocating five different subsets for training: {5:5:400}, {405:5:800}, {805:5:1200}, {1205:5:1600} and {1605:5:2000}. Each training set contains 80 frames. Following the experimental protocol of Lempitsky [22], the remaining frames of the dataset were reserved for testing in each case. The results of this experiment are tabulated in Table 4. The proposed system outperforms the holistic method of Kong and performs competitively with Lempitsky. The results are plotted in Figure 10, indicating the ability of the system to scale to larger or smaller crowd sizes than those encountered in the training range.

**Table 4** Scalability testing results on the UCSD data set. Five different training ranges were used, while the remaining frames were withheld for testing. Mean and standard deviation are reported for these five tests.

System	Mean abs. Error
Kong, linear	$2.33 \pm 0.64$
Lempitsky	$1.78 \pm 0.39$
Proposed, no tracking	$1.95 \pm 0.62$
Proposed, with tracking	$1.89 \pm 0.64$

Finally, the practicality of the proposed system was evaluated by repeating the scalability experiments on more sparse training sets: {20:40:380}, {420:40:780}, {820:40:1180}, {1220:40:1580} and {1620:40:1980}. Each training set contained only 10 frames. These training frames contain insufficient data to populate all of the histograms in Kong’s system, prohibiting it from being properly trained. As such the results for Kong are omitted from this experiment. The results for the proposed system are presented against Lempitsky’s approach in Table 5. The proposed system achieves a mean absolute error of 2.18 without tracking and 2.14 when tracking is incorporated; Lempitsky’s approach performs slightly better with a mean absolute error of 2.06.



**Fig. 10** Scalability testing results on the UCSD data set. Five different training ranges were used, while the remaining frames were withheld for testing.

**Table 5** Practicality testing results on the UCSD data set. Five sparse training ranges were used containing only 10 frames, while the remaining frames were withheld for testing. Mean and standard deviation are reported for these five tests.

System	Mean abs. Error
Kong	Insufficient training data
Lempitsky	$2.06 \pm 0.59$
Proposed, no tracking	$2.18 \pm 0.76$
Proposed, with tracking	$2.14 \pm 0.79$

These results indicate that both local approaches are capable of accurate crowd counting, on smaller and larger crowd sizes than those encountered during training, even when a reduced training set is used (10 frames). By contrast, holistic systems appear to require a greater training set (ideally more than one hundred frames). This is most likely because local approaches have access to a wealth of data in every training frame, rather than treating each frame as a single instance.

The training requirements of a local system (Section 3.4) are slightly more involved, as they require a dot to be placed on the centre of each person, whereas a holistic system only requires the overall count. However, as a holistic count requires each individual to be located anyway, it could be argued that annotating each person with a dot involves little additional effort, resulting in a substantially greater performance benefit.

## 4.2 Scene Invariance

In this section we assess the performance of the proposed system when trained and tested on different viewpoints. The purpose of this experiment is to emulate the scenario in which a ‘plug-and-play’ or turn-key system has been pre-trained on a variety of different viewpoints before being deployed on a new scene. Section 4.2.1 introduces a new crowd counting database that has been prepared for this chapter, which will be made available to the computer vision community. Section 4.2.2 presents the scene invariant crowd counting results of the proposed system.

### 4.2.1 QUT Dataset

Seven data sets with camera calibration are listed in Table 2, containing a total of 27,920 frames and more than sixteen minutes of annotated crowd footage. To supplement the existing public data sets, such as PETS 2006 [1] and PETS 2009 [2], a new database has been developed containing footage obtained from our university campus. This database is referred to as ‘QUT’ and will be made available to the computer vision community for experimentation.<sup>1</sup>

---

<sup>1</sup> Please contact David Ryan (david.ryan@qut.edu.au) for details on obtaining this database.

This database contains three challenging viewpoints, which are referred to as Camera A (Figure I(e)), Camera B (Figure I(f)) and Camera C (Figure I(g)). The sequences contain reflections, shadows and difficult lighting fluctuations, which makes crowd counting difficult. Furthermore, Camera C is positioned at a particularly low camera angle, leading to stronger occlusion than is present in other datasets.

Previous crowd counting datasets have been substantially shorter in length than those included in the QUT database. For example, PETS 2009 contains two crowd counting sequences of length 220 and 240 frames, while the UCSD dataset contains 2000 consecutively annotated frames. Although these resources are extremely valuable for testing crowd counting algorithms, they do not adequately capture the long-term performance of a system over varying conditions. For example, if a system performs poorly on one particular frame, it is likely that the preceding and subsequent frames will suffer from the same vulnerability. On shorter sequences such as the PETS 2009 datasets, this may lead to biased results that do not adequately describe a system's true performance capabilities.

In order to combat this potential problem, the QUT datasets are annotated at more sparse intervals: every 100 frames for cameras B and C, and every 200 frames for camera A as this is a longer sequence. Testing is then performed by comparing the crowd size estimate to the ground truth at these sparse intervals, rather than at every frame. This closely resembles the intended real-world application of this technology, where an operator may periodically 'query' the system for a crowd count. Although the human operator does not require this information from *every* frame, the system should at least provide accurate results whenever it is requested.

Due to the difficulty of the environmental conditions in these scenes, the first 400-500 frames of each sequence is set aside for learning the background model. This is a requirement for proper operation of many multi-modal algorithms such as Denman [14, 15], Stauffer-Grimson [39] and Zivkovic [46], which are used very widely in the computer vision field. Existing databases generally do not provide time to learn the background, and although PETS 2009 provides some detached background sequences, they do not immediately precede the crowd counting sequences to be tested, limiting their usefulness.

#### 4.2.2 Results

In this section scene invariance is tested using the seven calibrated datasets from Table 2. In each experiment one viewpoint was withheld for testing, and the remaining six viewpoints were used for training. Ten frames from each training viewpoint were selected, so that a total of sixty training frames were used to train the system in each experiment. Testing was then performed on the remaining viewpoint, using all of the annotated ground truth frames to calculate the mean absolute error and the mean square error.

Because the number of people in each scene was often fractional, we use the 'soft' ground truth defined in equation 24. This makes sense when evaluating our

algorithm because it makes use of local features and has been annotated with occasionally fractional counts (Section 3.4). A blob's ground truth does not jump directly from 0 to 1 (or vice versa) when entering or exiting a scene, for example.

Results for these experiments are tabulated in Table 6. Across all experiments, weighted equally, the mean absolute error was  $1.21 \pm 0.58$ , and in most cases a modest improvement was observed by incorporating the tracking procedure of Section 3.6. The crowd counting results for each sequence are plotted in Figure 11.

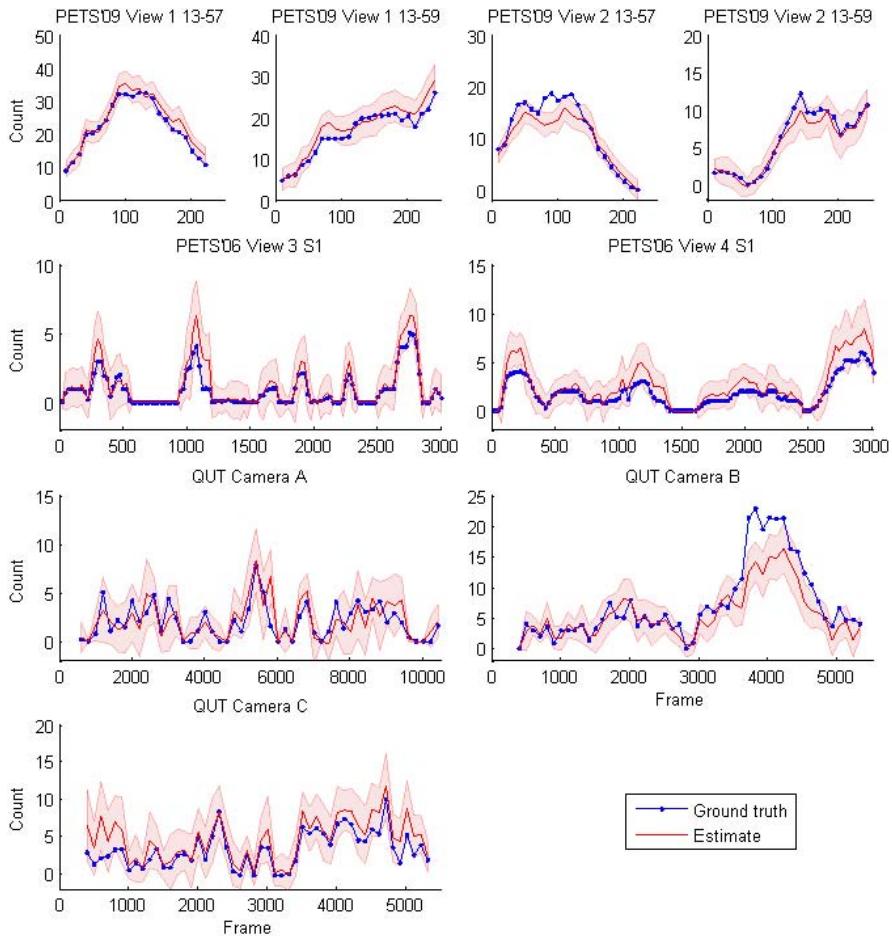
**Table 6** Scene invariant testing results on the seven calibrated data sets of Table 2. When testing each viewpoint, the system is trained on the six other viewpoints.

<b>Test Set</b>	<b>No tracking</b>		<b>With tracking</b>	
	Mean abs. error	Mean square error	Mean abs. error	Mean square error
PETS 2009, View 1	1.70	4.12	1.65	3.91
PETS 2009, View 2	1.24	3.25	1.23	3.31
PETS 2006, View 3	0.34	0.39	0.34	0.39
PETS 2006, View 4	0.79	1.15	0.79	1.15
QUT, Camera A	0.92	1.62	0.92	1.56
QUT, Camera B	2.09	9.49	2.06	9.37
QUT, Camera C	1.36	3.20	1.22	2.42
All tests	$1.21 \pm 0.58$	$3.32 \pm 3.02$	$1.17 \pm 0.57$	$3.16 \pm 3.00$

Screenshots from the system during its operation are shown in Figure 12. Blob perimeters are drawn in red and the group size estimates are written on the centroid of each blob, rounded to the nearest integer. In most cases the group estimate is correct within 1 of the ground truth. An advantage of the local features based approach is that the system can provide a crowding estimate not just for the holistic level, but for the regions occupied by each group within the image. This could be used by a system to detect abnormal crowd distribution patterns or local overcrowding situations, even when the holistic crowd size is within normal ranges.

Figure 12 also includes some false positives in the foreground segmentation (PETS 2009, View 1) and a missed detection (QUT, Camera B). The background subtraction on QUT Camera B is particularly challenging due to the darkness of the scene and the background. This is the main source of error in our experiments, and it accounts for the under-estimation observed for QUT Camera B, which is seen in Figure 11. Conversely, lighting fluctuations in PETS 2009 View 1 resulted in some false positives, accounting for the slight over-estimation also observed in Figure 11.

Background modeling and foreground segmentation continue to remain amongst the major challenges in visual surveillance and the state of the art is continually evolving. Nevertheless the proposed system demonstrates accurate performance on these datasets and in most cases handles noise and blob fragmentation quite well: small instances of noise are disregarded, because they were learned during training with annotations of zero, while fragmented blobs are assigned fractional counts where necessary.



**Fig. 11** Scene invariant testing results on the seven calibrated data sets of Table 2. When testing each viewpoint, the system is trained on the six other viewpoints.

These results support the ability of the proposed system to perform scene invariant crowd counting when trained and tested on different viewpoints, and provides a baseline methodology and database for future scene-invariant experiments.

## 5 Conclusion

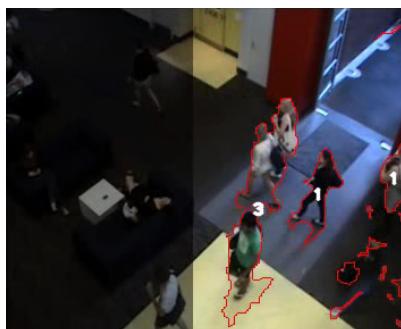
This chapter proposed a novel scene-invariant crowd counting algorithm based on local features, specific to groups and individuals in an image, to estimate the crowd size and its distribution across a scene. Unlike previous systems that have typically employed holistic features, the proposed approach is annotated, trained and tested at a local level. Camera calibration is incorporated into the system to scale features



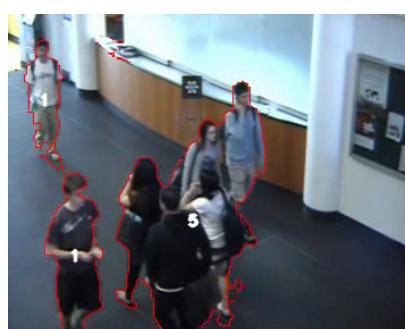
(a) PETS 2009, View 1. Groups of size 15, 6, 4 and 3 are detected.



(b) PETS 2006, View 3. Groups of size 1 and 4 are detected.



(c) QUT, Camera B. Groups of size 3, 1 and 1 are detected.



(d) QUT, Camera C. Groups of size 1, 1 and 5 are detected.



(e) UCSD dataset, smaller groups



(f) UCSD dataset, larger groups

**Fig. 12** Screenshots of the proposed algorithm operating on five viewpoints. The region of interest is highlighted.

between viewpoints, and a tracking algorithm was described to further improve the system's performance.

The proposed approach outperforms the baseline holistic methods of Kong [20] and Chan [7], and performs competitively with the local approach of Lempitsky [22], when trained and tested on the same viewpoint. The proposed system was demonstrated to be highly accurate, scalable and practical, with very minimal training requirements. Accurate test results were obtained from as few as ten training frames of data.

Scene invariance was also demonstrated by training the system on multiple cameras and then testing it on a new, unseen viewpoint. Accurate crowd counting results were obtained for seven calibrated sequences, including a new QUT dataset designed to help evaluate the performance of crowd counting systems in difficult real-world conditions.

The proposed system does not require any additional training when deployed for crowd counting on a new camera. This brings the computer vision field one step closer toward a truly 'plug-and-play' system which is pre-trained on a large bank of data from a variety of cameras. This technology has many potential applications, including automatic gathering of business intelligence, crowd safety monitoring and abnormality detection.

Future research into scene invariant crowd counting will continue to investigate background modeling techniques, scene invariant feature extraction, autocalibration methods, and improved tracking algorithms that can be readily incorporated into framework already proposed in this chapter.

## References

1. Ninth ieee international workshop on performance evaluation of tracking and surveillance (2006), <http://www.cvg.rdg.ac.uk/PETS2006/>
2. Eleventh ieee international workshop on performance evaluation of tracking and surveillance (2009), <http://www.cvg.rdg.ac.uk/PETS2009/>
3. Abdel-Aziz, Y., Karara, H.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In: Proceedings of the Symposium on Close-Range Photogrammetry, pp. 1–18 (1971)
4. Bose, B., Grimson, E.: Ground plane rectification by tracking moving objects. In: IEEE International Workshop on Visual Surveillance and PETS (2004)
5. Canny, J.: A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 8(6), 679–698 (1986)
6. Celik, H., Hanjalic, A., Hendriks, E.: Towards a robust solution to people counting. In: 2006 IEEE International Conference on Image Processing, pp. 2401–2404 (2006)
7. Chan, A., Liang, Z.S., Vasconcelos, N.: Privacy preserving crowd monitoring: Counting people without people models or tracking. In: IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2008, pp. 1–7 (2008)
8. Chan, A., Vasconcelos, N.: Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(5), 909–926 (2008)

9. Chan, A.B., Morrow, M., Vasconcelos, N.: Analysis of crowded scenes using holistic properties. In: Performance Evaluation of Tracking and Surveillance Workshop at CVPR 2009, Miami, Florida, pp. 101–108 (2009)
10. Cho, S.Y., Chow, T., Leung, C.T.: A neural-based crowd estimation by hybrid global learning algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 29(4), 535–541 (1999)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, vol. 1, pp. 886–893 (2005)
12. Davies, A., Yin, J.H., Velastin, S.: Crowd monitoring using image processing. *Electronics & Communication Engineering Journal* 7(1), 37–47 (1995)
13. Denman, S.: Improved detection and tracking of objects in surveillance video. Ph.D. thesis, Queensland University of Technology (2009),  
<http://eprints.qut.edu.au/29328/>
14. Denman, S., Chandran, V., Sridharan, S.: An adaptive optical flow technique for person tracking systems. *Pattern Recognition Letters* 28(10), 1232–1239 (2007)
15. Denman, S., Fookes, C., Sridharan, S.: Improved simultaneous computation of motion detection and optical flow for object tracking. In: Digital Image Computing: Techniques and Applications, DICTA 2009, pp. 175–182 (2009)
16. Haralick, R.: Statistical and structural approaches to texture. *Proceedings of the IEEE* 67(5), 786–804 (1979)
17. Hou, Y.L., Pang, G.: People counting and human detection in a challenging situation. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 41(1), 24–33 (2011)
18. Kilambi, P., Masoud, O., Papanikolopoulos, N.: Crowd analysis at mass transit sites. In: Intelligent Transportation Systems Conference, ITSC 2006, pp. 753–758. IEEE (2006)
19. Kilambi, P., Ribnick, E., Joshi, A.J., Masoud, O., Papanikolopoulos, N.: Estimating pedestrian counts in groups. *Computer Vision and Image Understanding* 110(1), 43–59 (2008)
20. Kong, D., Gray, D., Tao, H.: A viewpoint invariant approach for crowd counting. In: 18th International Conference on Pattern Recognition, ICPR 2006, vol. 3, pp. 1187–1190 (2006)
21. Krahnstoever, N., Mendonca, P.R.S.: Bayesian autocalibration for surveillance. In: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV), pp. 1858–1865. IEEE Computer Society, Washington, DC (2005)
22. Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: Advances in Neural Information Processing Systems, NIPS (2010)
23. Lin, S.F., Chen, J.Y., Chao, H.X.: Estimation of number of people in crowded scenes using perspective transformation. *IEEE Transactions on Systems, Man and Cybernetics, Part A* 31(6), 645–654 (2001)
24. Lv, F., Zhao, T., Nevatia, R.: Self-calibration of a camera from video of a walking human. In: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR 2002), vol. 1, p. 10562. IEEE Computer Society, Washington, DC, USA (2002)
25. Ma, R., Li, L., Huang, W., Tian, Q.: On pixel count based crowd density estimation for visual surveillance. In: 2004 IEEE Conference on Cybernetics and Intelligent Systems, vol. 1, pp. 170–173 (2004)
26. Marana, A.N., Cavenaghi, M.A., Ulson, R.S., Drumond, F.L.: Real-Time Crowd Density Estimation Using Images. In: Bebis, G., Boyle, R., Koracin, D., Parvin, B. (eds.) ISVC 2005. LNCS, vol. 3804, pp. 355–362. Springer, Heidelberg (2005)

27. Marana, A., Da Fontoura Costa, L., Lotufo, R., Velastin, S.: Estimating crowd density with minkowski fractal dimension. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1999. Proceedings, vol. 6, pp. 3521–3524 (1999)
28. Marana, A., Velastin, S., Costa, L., Lotufo, R.: Estimation of crowd density using image processing. In: IEE Colloquium on Image Processing for Security Applications (Digest No.: 1997/074), pp. 11/1–11/8 (1997)
29. Masoud, O., Papanikolopoulos, N.: A novel method for tracking and counting pedestrians in real-time using a single camera. IEEE Transactions on Vehicular Technology 50(5), 1267–1278 (2001)
30. Paragios, N., Ramesh, V.: A mrf-based approach for real-time subway monitoring. In: 2001 Conference on Computer Vision and Pattern Recognition (CVPR 2001), pp. 1034–1040 (2001)
31. Rabaud, V., Belongie, S.: Counting crowded moving objects. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 705–711 (2006)
32. Rahmalan, H., Nixon, M.S., Carter, J.N.: On crowd density estimation for surveillance. In: The Institution of Engineering and Technology Conference on Crime and Security 2006, pp. 540–545 (2006)
33. Rasmussen, C.E.: Gaussian Processes in Machine Learning. In: Bousquet, O., von Luxburg, U., Rätsch, G. (eds.) Machine Learning 2003. LNCS (LNAI), vol. 3176, pp. 63–71. Springer, Heidelberg (2004)
34. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. MIT Press (2006)
35. Regazzoni, C., Tesei, A., Murino, V.: A real-time vision system for crowding monitoring. In: Proceedings of the IECON 1993, International Conference on Industrial Electronics, Control, and Instrumentation, vol. 3, pp. 1860–1864 (1993)
36. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using multiple local features. In: Digital Image Computing: Techniques and Applications. DICTA 2009, pp. 81–88 (2009)
37. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using group tracking and local features. In: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 218–224 (2010)
38. Shi, C.J., Tomasi: Good features to track. In: 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Proceedings CVPR 1994, pp. 593–600 (1994)
39. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999, vol. 2(xxiii+637+663) (1999)
40. Tsai, R.: An efficient and accurate camera calibration technique for 3d machine vision. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1986), pp. 364–374 (1986)
41. Tuzel, O., Porikli, F., Meer, P.: Human detection via classification on riemannian manifolds. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8 (2007)

42. Wu, X., Liang, G., Lee, K.K., Xu, Y.: Crowd density estimation using texture analysis and learning. In: IEEE International Conference on Robotics and Biomimetics, ROBIO 2006, pp. 214–219 (2006)
43. Xiaohua, L., Lansun, S., Huanqin, L.: Estimation of Crowd Density Based on Wavelet and Support Vector Machine. *Transactions of the Institute of Measurement and Control* 28(3), 299–308 (2006)
44. Yang, T., Zhang, Y., Shao, D., Li, Y.: Clustering method for counting passengers getting in a bus with single camera. *Optical Engineering* 49(3) (2010)
45. Zhang, Z.: A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(11), 1330–1334 (2000)
46. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: 17th International Conference on Proceedings of the Pattern Recognition (ICPR 2004), vol. 2, pp. 28–31. IEEE Computer Society, Washington, DC, USA (2004)

# Identifying Customer Behaviour and Dwell Time Using Soft Biometrics

Simon Denman, Alina Bialkowski, Clinton Fookes, and Sridha Sridharan

**Abstract.** In a commercial environment, it is advantageous to know how long it takes customers to move between different regions, how long they spend in each region, and where they are likely to go as they move from one location to another. Presently, these measures can only be determined manually, or through the use of hardware tags (i.e. RFID). Soft biometrics are characteristics that can be used to describe, but not uniquely identify an individual. They include traits such as height, weight, gender, hair, skin and clothing colour. Unlike traditional biometrics, soft biometrics can be acquired by surveillance cameras at range without any user co-operation. While these traits cannot provide robust authentication, they can be used to provide identification at long range, and aid in object tracking and detection in disjoint camera networks. In this chapter we propose using colour, height and luggage soft biometrics to determine operational statistics relating to how people move through a space. A novel average soft biometric is used to locate people who look distinct, and these people are then detected at various locations within a disjoint camera network to gradually obtain operational statistics.

## 1 Introduction

For operators of large, public facilities such as airports, transport hubs and shopping centres, it is important to understand how people move through the environment and how long it takes to move between key points. At present, only manual methods are available to collect such data, typically achieved by a staff member handing a customer a piece of time stamped paper which they hand to a second staff member at a later point in the process.

---

Simon Denman · Alina Bialkowski · Clinton Fookes · Sridha Sridharan  
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia  
e-mail: s.denman@qut.edu.au, alina.bialkowski@qut.edu.au,  
c.fookes@qut.edu.au, s.sridharan@qut.edu.au

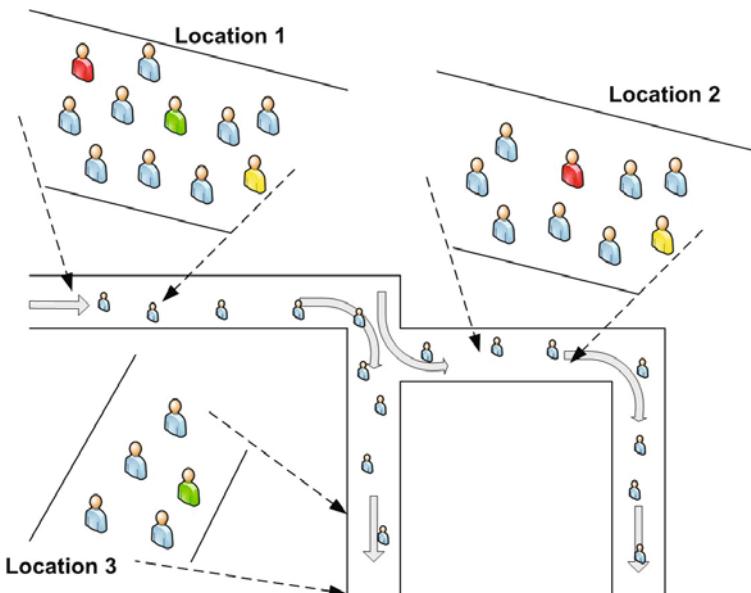
While object tracking systems [20, 6, 54, 17, 18] offer a possible solution for small environments and situations where there is continuous camera coverage, they are not practical for deployment in locations such as an airport, where there are large and frequent gaps in the camera network and a large number of possible paths through the environment (i.e. it is unclear which camera a person would next appear in). While techniques such as crowd counting [55] are designed to operate in such an environment, they only extract information about the crowd as a whole, and disregard the behaviour of individuals within it.

Within a large, disjoint, surveillance environment, soft biometrics [15, 12, 53] offer a means to continuously recognise people as they move through an environment. However, in a crowded space where there are hundreds, or possibly thousands of people present, there are likely to be a large number of people who have a similar appearance making accurate matching across views challenging and error prone. For example, in a typical airport there are likely to be a large number of business men wearing dark suits. However, while it would be difficult to match these similar looking people, a person who stands out from the crowd would be comparatively easy to follow through the environment, even when the path they take and speed which they move through the environment could not be anticipated.

In this chapter, we propose using soft biometrics to model a persons appearance, and from observations of multiple people calculate an average soft biometric. By comparing people to the average biometric, we can then determine how unique their appearance is, allowing us to automatically select only individuals who stand out from the crowd. This sub-set of people can then be re-detected throughout a disjoint camera network using their soft biometrics (see Figure 1), to obtain operational measures such as the time taken to travel from point to point.

We demonstrate the proposed system on a small database of up to four cameras collected in-house, and show that the average soft biometrics can be used to locate individuals with a distinct appearance, and then match these people across disjoint camera views. We show that this can be used to obtain an accurate estimate of the time it takes for people to move through the environment.

The remainder of this paper is structured as follows: Section 2 presents an overview of soft biometrics in surveillance imagery and other related literature; Section 3 outlines the soft biometric models used in this work; Section 4 describes how the average biometrics are calculated and how these are used to detect distinct individuals and calculate operational statistics; Section 5 presents and evaluation of both the soft biometric models (Section 5.1) and the estimation of operational measures through the detection of unique individuals (Section 5.2); and Section 6 concludes the chapter and outlines possible directions for future research.



**Fig. 1** Detecting distinct people - At location 1, three distinct looking people can be detected (yellow, green, and red). As the people move through the environment, two of these distinct people can be re-detected at location 2 (yellow and red), and the third can be re-detected at location 3 (green).

## 2 Background

Soft biometrics are traits that describe a person, but cannot uniquely identify them. They include information such as gender, ethnicity, height and build, and clothing colour. To date, soft biometrics have had two main uses: as a means to improve the performance of traditional biometrics systems by incorporating soft biometrics [33, 1, 44, 46]; or as a way to recognise people in surveillance footage [15, 12, 53].

The initial uses of soft biometrics focused on improving the accuracy of existing ‘hard’ biometric systems. Jain et al. [33] proposed the incorporation of gender, ethnicity, and height information of a person in fingerprint and face recognition using a Bayesian framework for fusion; Ailisto et al. [1] combined body weight and fat measurements to improve the performance of fingerprint biometrics; Marcialis et al. [44] used hair colour and ethnicity to complement face as the primary biometric; while Park et al. [47] proposed incorporating gender and facial marks (i.e. scars, freckles, moles, etc.) to improve face recognition. Niinuma et al. [46] proposed using soft biometrics for continuous authentication of a user at a computer, combining face and clothing colour with traditional face recognition and the users password. In all situations, soft biometrics were able to improve the recognition performance of the combined system.

In addition to identification improvements, soft biometrics can be used to more quickly and efficiently filter a large biometric database. For example, Wayman [65] proposed the use of gender and age for filtering a large biometric database. This limits the number of entries in a database to be searched, and hence significantly reduces the time required for identification. However, if a person is incorrectly classified, recognition performance may be degraded.

Although these systems have successfully used soft biometrics, the focus is on complementing primary biometric systems such as fingerprint and face. In such situations, the soft biometric features are trained and classified on high resolution still images, which are well aligned and often captured under controlled lighting conditions. For soft biometrics in video surveillance, the extraction of soft biometric features is a more challenging task due to lower quality data and uncontrolled environment, such as changes in illumination, resolution, the pose of a person, camera view angle, as well as the presence of occlusions and shadows. In such conditions, extraction of some soft biometrics used in these systems, such as facial marks, is likely to be extremely difficult.

In a surveillance environment, traits that can be observed at a greater range are desirable, as are traits that are invariant to view and to lighting conditions. Cognitive experiments examining suitable traits for describing individuals have shown that individuals recognise other people by assigning them into categories including the soft biometric traits of gender, age, race and build or physique [43]. Samangooei et al. [56] quantified how accurate and useful human-provided descriptions of people are for recognising a person at a distance. Physical traits were categorised into semantic terms for humans to label video sequences of different people, and race and gender were found to be the most effective measures. Recent research has attempted automatic extraction of such traits from video. For example, the extraction of gender from gait [14, 69, 59], the estimation of age from gait [41], and determining ethnicity from faces [42]. While all these techniques show promise, they are difficult to apply in an unconstrained surveillance environment, and on their own do not yield sufficiently unique descriptions when dealing with a large population.

Appearance modelling techniques used in object tracking and person re-detection systems can also be viewed as a form of soft biometric. Appearance models are typically designed to be view and illumination invariant so that they may be used to aid in tracking handover between different camera views, and to aid in tracking during or after occlusions. Haritaoglu et al. [29] proposed a method where data pertaining to the average texture and silhouette of the subject is recorded over a period of time as the object is tracked. This model can be used to re-detect a person if they had been lost for several frames due to occlusion, or had left and re-entered the scene.

Whilst the approach proposed in [29] is effective for a single view, texture information less suitable for transferring from one view to another. Hu et al. [31] extracted three histograms from each person, one each for the head, torso and legs, to not only allow for matching based on colour, but also on the distribution of colour. Chien et al. [7] proposed a colour model (Human Colour Structure Descriptor - HCSD) that aims to capture the distribution of colours in a human body. Three

colours are used to represent the colour of the body, legs and shoes, and positions are defined to describe the position of body and legs relative to the shoes.

Nakajima et al. [45] and Hahnel et al. [25] each proposed techniques to model and recognise people based on their whole body. Nakajima et al. [45] extracts normalised colour histograms and local shape features for detected people and trains SVM classifiers for each person and pose, and the approach is shown to be accurate on a limited data set. Hahnel et al. [25] extends on [45] by extracting additional colour, shape and texture features.

To better evaluate appearance modelling and person re-detection techniques, Gray et al. [24] proposed the VIPeR (Viewpoint invariant Pedestrian Recognition) database, consisting of 632 image pairs, where each pair contains wide and varying changes in viewpoint, illumination and pose. Using the proposed data set, an evaluation of a variety of existing techniques such as 1D and 3D histograms, 3D correlograms and various template matching approaches was performed, and it was shown that a simple 1D histogram outperforms the more complex techniques.

Many recent approaches have sought to combine colour and texture features, and extract texture features which are less view dependant. Farenzena et al. [19] proposed an appearance-based method for person re-identification using symmetry-based features consisting of the overall chromatic content, the spatial arrangement of colours into stable regions (through the extraction of MSCRs [21]), and recurrent local motifs with high entropy (i.e. recurring textures). Symmetry is used to build the model through the use of weighted colour histograms computed along the symmetric axes, and by the sampling of patches to locate local motifs along the axes of symmetry; while the axes of asymmetry are used to segment the person into head, torso and legs. Bak et al. [2] proposed appearance models based on Haar-like features and dominant colour descriptors. The most invariant and discriminative signature was extracted using the AdaBoost algorithm.

Bazzani et al. [4] proposed a person descriptor that incorporates a global feature, in the form of a HSV histogram, and local features, determined through epitomic analysis [34]. A generic epitome (approximately equivalent to an ‘average’ texture of the person), and a set of local epitome (a set of recurring motifs within the subject) are extracted to represent the texture. As with the global feature, the epitomes are represented by a HSV histogram.

Schwartz et al. [57] proposed using a co-occurrence matrix to extract a dense texture representation, as well as extracting edge and colour features for subjects. Features are extracted for a set of overlapping blocks within the subject region, leading to a very large initial feature vector. Dimension reduction is carried out using partial least squares (PLS), and a PLS model is learned for each subject in the gallery by using all other subjects as counter examples. This approach results in a model that highlights the most discriminative features for each subject. When classifying a probe image, the probe is projected into the subspace of each gallery subject, and the minimum Euclidean distance between the probe and gallery vectors is used to classify the subject. The PLS scheme is shown to outperform a PCA and SVM approaches.

Other approaches have sought to improve person classification by incorporating the structure of the human body into the model. Bak et al. [3] proposed using body part detectors to locate individual body parts (head and shoulders, torso, left arm, right arm, legs) as well as the full body, and extracted features for each region. The covariance of features extracted from each region (location, intensity, and gradient) is used to match subjects within a spatial pyramid matching framework.

Xiaogang et al. [67] also sought to model the relationship between different parts of the human body and proposed the idea and shape and appearance context, which models the appearance of an object relative to its own components. From the input image shape and appearance labelled images are generated. The shape image describes the structure of the person (i.e. identifying body regions such as arms and legs), while the appearance image quantises the input image into a set of labelled regions. The co-occurrence relationship between features within the different regions is extracted and is used to describe the appearance of a person.

Zheng et al. [71] extended the idea of appearance context to groups of people, observing that people often move in groups, and the appearance of the group provides additional cues to a person's identity. Two ratio-occurrence measures for comparing groups are developed, one using rectangular rings around the image, and one using features within small image blocks, and it is shown that this combination of feature is effective at recognising groups of people.

Group context is also applied to an individual within the group, by extracting features in rings around the person of interest (providing a group context for the individual), in combination with features for the individual. It is shown that incorporating group context results in a significant performance improvement when identifying individuals.

Gheissari et al. [22] propose a multi-part person model, segmenting the person into 6 horizontal stripes, based on a decomposable triangulated graph model. A feature vector, composed of HSV colour information and edgels (where each edge encodes the edge orientation, and the colour change across the edge), is extracted for each of the horizontal stripes. To compensate for the dynamic properties of clothing which can result in edges changing rapidly, a spatio-temporal graph based segmentation approach to suppress edges is applied. This results in edges that occur within an item of clothing being ignored, while retaining those that relate to the boundary between different clothing items.

An alternative approach to crafting features specific to modelling an individual is to extract very high dimensionality features, and use a classifier to determine the optimal combination of these features to separate the classes. Gray et al. [23] proposed learning a similarity function using a combination of colour and texture (obtained by using the response to Schmid and Gabor filters) information. Localised features in horizontal stripes are computed for each possible feature (i.e. colour channel, or texture representation) and adaboost is used to learn the optimal combination of these features. The proposed approach is evaluated on the VIPeR data set [24] and is shown to outperform the algorithms of Park et al. [48] and a principle axis histogram motivated by [32], as well as simple histogram and template representations.

Prosser et al. [51] and Zheng et al. [72] focused on finding an optimal ranking strategy to improve identification performance. Prosser et al. [51] formulated an alternative ranking strategy that does not rely on the direct computation of distances between gallery and probe models. A modification to the Rank SVM is proposed (termed an Ensemble Rank SVM) that aims to reduce the computational burden of Rank SVM by using a group of ‘weak rankers’. A set of ‘weak rankers’ are learned on subsets of the overall data set, and the weak classifiers are recombined using boosting. The resultant system performs similarly to the traditional Rank SVM, with a significant reduction in computations requirements, using a set of features inspired by [23]. The proposed system also outperforms the ELF model of [23], as well as similar features matched using the Bhattacharyya and L1-norm distance measures.

Zheng et al. [72] proposed the probabilistic relative distance comparison (PRDC), that aims to learn the distance metric that maximises accuracy, to improve person re-identification. The proposed model is formulated to maximise the probability that the correct match for a given image will have a smaller distance to the probe image than an incorrect match (unlike traditional distance learning which seeks to minimise intra-class and maximise inter-class variation). Like other trained classifiers [23, 51], a large suite of features is used, and a set similar to [23] is chosen. The proposed approach is shown to outperform the learned classifier techniques for person re-identification of [23, 51], and is also shown to be more robust when the size of the training set is small.

An alternate matching strategy is proposed by Lin et al. [39], who outlined a pairwise matching approach, where the distance between each pair of gallery subjects is learned, and the match profiles that describe the similarity of the pairs are stored. For a given probe image, the match profile to all subjects in the gallery is computed and the matching subject is determined by comparing the set of profiles. Lin et al. [39] model appearance using normalised kernel density estimation, exploiting the non-parametric nature of kernel density estimation to model the complex colour distributions that people exhibit.

While a large amount of effort has been made to model colour and appearance, they are limited by changes in view. From different angles, people have vastly different appearances, and variations in the appearance of colours, both within a single camera and across multiple cameras within an environment, present further challenges. While some techniques such as [19, 4] aim to improve view invariance by extracting representative and recurring textures (rather than a dense texture representation), if the distance between views is sufficiently great (i.e. viewing the subject from the front, and then the back), these textures still may not re-occur and matching will fail. With this in mind, a variety of other features have been proposed to match subjects across multiple cameras.

The presence of luggage can be used as a descriptor, and a suite of techniques have been proposed to locate and model carried luggage. Approaches for detecting luggage or carried items include using contour analysis and SVM classification [52], locating regions of asymmetry and an analysis of the periodicity of these regions [28], and through an analysis of simple motion region statistics in the case of [11], who sought to determine if a person is walking with a bicycle. Damen et al. [10]

proposed a silhouette based technique that extends [28], comparing a time-averaged silhouette with a set of exemplars rather than assuming that a person is symmetric. Regions that protrude from the exemplar are identified, and constraints on the likely location of carried objects are enforced using a Markov random field. The resultant system is able to identify a wide variety of carried items, although it does rely on the carried object protruding from the person.

Various shape and size features have also been proposed to model people in tracking systems. The silhouette of a person obtained from motion detection is a popular shape features and can be used in a variety of ways. Collins et al. [8] has proposed the use of silhouette-based human identification from body shape and gait, and other features such as boundary distance from the centroid [64], and the convex hull can also be extracted.

Gait recognition techniques such as the Gait Energy Image (GEI) [26, 66, 70] can also be thought of as a soft biometric. The GEI is the average silhouette taken over a single gait period, enabling the temporal information of gait to be encoded in a single frame. Unlike gait techniques that rely on model-fitting, the GEI can easily be captured at long range in the same manner as other soft biometrics.

From an unconstrained surveillance standpoint, shape and gait based methods are limited due to their dependency on the viewing angle, and the quality of the extracted silhouette. Segmentation errors caused by complex background conditions and shadows can affect the performance. Whilst these errors can be overcome by averaging over a gait cycle provided that the errors are random and not systematic (shadows are not random and are still a large problem in complex lighting conditions), but errors caused by view mismatch are much more difficult to overcome and are likely to lead to recognition errors.

With the various limitations of the individual features, several approaches have been proposed that utilise multiple soft biometric and appearance features. Denman et al. [15] proposed modelling colour (as a three part head, torso and legs model) and height to describe people and identify them between camera views in a surveillance environment. Dantcheva et al. [12] described a weight modality, as well as a probabilistic colour histogram (also three part) that could be used to identify people in surveillance imagery. Ran et al. [53] proposed a gait signature, consisting of several soft biometrics based on gait features. Stride length, height and gender could all be extracted from a video sequence. Demirkus et al. [13] presented a system in which different biometric features of people are extracted based on the quality of the current frame. For example, in a frame where the whole body of a subject is visible, height and clothing colour features are extracted, while in another frame only the upper body might be visible, but the face may be quite clear and hence face-based features are extracted. All approaches have shown promise for use in recognition within a surveillance environment.

Soft biometrics have also been applied to the task of a visual search, i.e. locating a person in surveillance footage given a description. Park et al. [48] proposed extracting dominant colours, height and build (determined from the silhouette aspect ratio) to represent a subject. A query could then be submitted to the system to locate a person matching a description. Vaquero et al. [62] proposed an attribute

based search, to locate people in surveillance imagery. Various facial features were extracted such as facial hair (beard, mustache, no facial hair), the presence of eye wear (glasses, sunglasses, no glasses) and headwear (hair, hat, bald), as well as full body features such as the colour of the torso and legs. Queries could be formulated as a combination of these features.

Soft biometrics and person re-detection techniques also have applications in customer behaviour analysis, and a variety of computer vision techniques and systems have been proposed to automatically analyse customer behaviour in retail environments. Many techniques have focused on counting people, with solutions proposed to count both everyone in an area [55, 38], and to count the number of people past a point (i.e. entering a shop) [35, 5]. Senior et al. [58] proposed a system that goes beyond simple counting, and developed a video analytics system for retail which counts the number of customers entering a store, and monitors where they go within the store.

Other vision systems have been developed to determine the interactions between people and products. Haritaoglu et al. [30] developed a body pose detection system using stereo cameras to automatically detect reaching actions of shoppers. Krahnstoever et al. [36] detected head and hand locations with stereo vision to automatically detect the interaction between the shopper and product, and RFID tags were used to track the location and motion of products.

Dwell time in front of products and billboards has also been explored. Haritaoglu et al. [27] developed a system to detect, track and count the number of people standing in front of billboards and extract their gender. Infrared illumination-based pupil detection was used to determine whether people were looking at the billboards, and for how long. Liu et al. [40] used Active Appearance Models (AAMs) to monitor the face for better accuracy in determining customer gaze.

Popa et al. [49] describes a surveillance system which analyses the shopping behaviour of customers in relation to products. A Bayesian Network architecture is employed that consists of three levels: sensor level, observed features level, and the customer's shopping behaviour (i.e. the assigned semantic label to the customer's actions). Through the use of motion detection, trajectory analysis, face localisation and tracking eight behaviours could be described: walking direction, walking speed, stopping, looking at products, facial expressions (for positive/negative appreciation), gestures, and speech. Experimental results on a data set of 10 subjects shows that by clustering the sensed features, and in particular the motion energy and walking trajectories, the approach can distinguish between customers behaviours as being either 'goal oriented' or 'disoriented'.

Popa et al. [50] also described a facial recognition system which analyses a customer's appreciation of a product based on their expression. Active Appearance Models are used to track the face and extract key features, and a Hidden Markov Models is used to classify the expressions into 21 product emotions, obtaining a recognition accuracy of 93%. By automatically detecting and analysing a customer's expression, retailers can better fit products to the customers' needs, enabling more efficient marketing strategies.

These systems provide useful information on customer behaviour on a small scale, but do not scale to large environments. In large environments it may not be feasible to have cameras to monitor every space, and may not provide sufficient resolution for unique identification and so robust algorithms need to be developed to model individuals to enable tracking of people across disjoint camera views.

### 3 Soft Biometrics

Soft biometrics are features that can be easily extracted from a distance. Ideally, for use in an unconstrained surveillance environment any features should also be view invariant. In this work, we consider colour, height and luggage models for a person. We select these modalities as we consider them to be moderately view invariant and easy to extract in difficult conditions (i.e. colour and height can be determined from any angle, while texture, hair and skin features cannot be extracted consistently as the view changes). Furthermore, despite not being as discriminative as other recent approaches (for example [19, 3, 71]), both colour and height modalities are well suited to locating people who have an unusual aspect to their appearance, for instance people who are unusually tall or short, or who are dressed strangely. It should be noted however that other modalities (such as ethnicity, or more advanced colour and texture features) could also be used within the proposed framework.

The colour model is a three part model (head, torso and legs), and each section is modelled separately (see Section 3.1). The height of a person can be extracted using camera calibration, and is outlined in Section 3.2. Luggage can be detected by analysing the symmetry of the silhouette and locating regions of asymmetry. The detection and modelling of luggage is outlined in Section 3.3.

For all soft biometrics outlined within this Section, it is expected that the person of interest has been located within the scene (i.e. through object detection [9, 63] and/or object tracking [17, 6]), and a motion segmentation algorithm [16] is used to separate the subject from the background.

#### 3.1 Colour

A 3D colour soft histogram is computed for each of the head, torso and leg sections ( $C_{head}$ ,  $C_{torso}$  and  $C_{legs}$  respectively). The colour and motion image are used to generate histograms, such that only pixels that are in motion (i.e. part of the person) are included in the histogram. In order to separate the person into these three components, it is assumed that the person has been correctly segmented within the motion image (i.e. the motion regions relevant to the target person have been identified and all other motion is disregarded), and that the person is predominately vertical within the image.

Segmentation of the person into head, torso and legs is performed in two stages:

1. Analysis of the horizontal projection to determine likely regions for the neck and waist;
2. Analysis of the gradient of the regions of interest to locate the neck and waste contour.

The horizontal projection,

$$P_{horiz}(j) = \sum_{i=0}^W M(i, j), \quad (1)$$

where  $P_{horiz}(j)$  is the horizontal projection for row  $j$ ,  $W$  is the image width, and  $M$  is the motion image; is computed for the region containing the person, and it is used to estimate the likely neck and waist positions. Given correct segmentation of the person in the motion image, the location of the neck will coincide with an increase in the horizontal projection ( $v_{neck}$ ), and the waist ( $v_{waist}$ ) will correspond with a gradual decrease in the projection. Figure 2 shows an example of this.

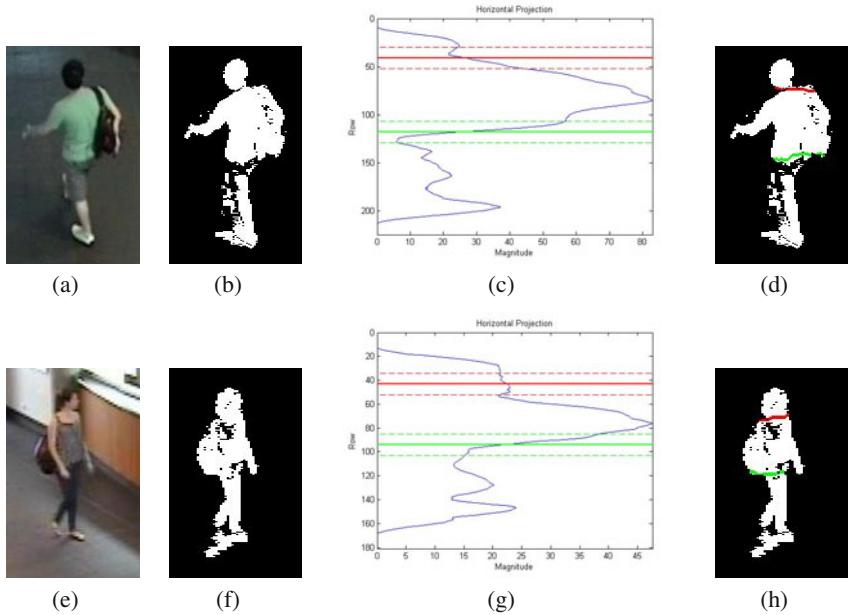
Typically, the neck and waist correspond with a change in appearance, and a local maximum in the image gradient should be observed at this transition. A search region of 10% of the person's height is established around each of these points to determine the actual neck and waist boundaries (see Figure 2 (c) and (g)). Within each search region, each column is searched in turn for the location with the maximum vertical gradient. The resultant vector describes the contour that separates the two regions,

$$V_b(i) = \arg \max_{n=v_1}^{v_2} \delta I(i, n), \quad (2)$$

where  $V_b$  is the contour for the boundary  $b$ ;  $v_1$  and  $v_2$  are the search bounds for the contour, set to 10% of the person's height either side of  $v_{neck}$  and  $v_{waist}$ ; and  $\delta I(i, n)$  is the gradient of the image  $I$  at location  $i, n$ . The computed contour is smoothed using a mean filter. Contours are computed for the neck and waist ( $V_{neck}$  and  $V_{waist}$  respectively), and these are used to divide the regions. Example output from this process is shown in Figure 3. This segmentation process assumes that the region being segmented is a single person, and the approach will produce unexpected results if a region consisting of multiple people is used as input. Within the proposed system, it is the responsibility of a prior process that locates the objects (such as an object detector or object tracker) to ensure that this segmentation is correct.

For each of the three regions, a 3D soft histogram is calculated to represent the appearance. Due to variations in colour across the different cameras, as well as possible errors in segmentation, there are a number of sources of error when computing and comparing histograms. In an effort to minimise the impact of these errors, we use a soft histogram. Soft histograms have previously been applied in a variety of areas, including activity modelling [68] and texture representation [37].

The soft histogram divides the contribution of each sample (pixel) across multiple bins. Weights are assigned based on the proximity of the sample to the centre of the bins. A sample that lies at the centre of a bin will have the majority of its contribution

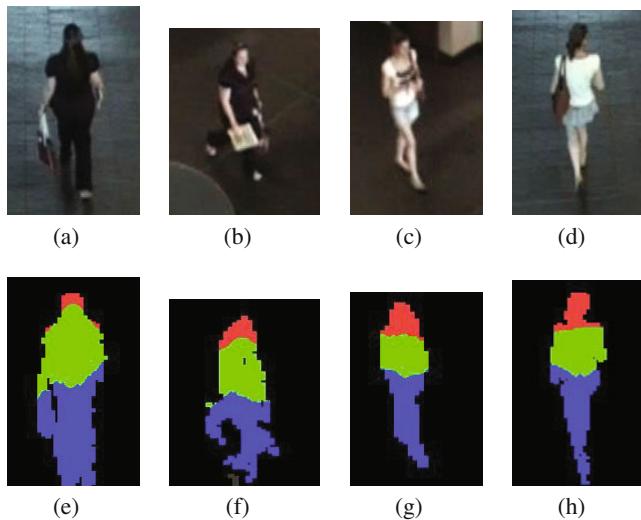


**Fig. 2** Locating neck and waist boundaries - The top and bottom rows show two examples of extracting the neck and waist boundaries. Images (a) and (e) are the input colour images and (b) and (f) are the silhouettes computed through foreground segmentation. (c) and (g) show the smoothed horizontal projection, with the approximate neck ( $v_{neck}$ , red) and waist ( $v_{waist}$ , green) locations, and the search space for the boundary is shown by the dotted lines. (d) and (h) show the final boundary for the neck (red) and waist (green).

assigned to that bin, while a sample that lies on a bin boundary will have its contribution divided across multiple bins. For simplicity, we only consider neighbouring bins within the same dimension (i.e. we do not consider diagonal neighbours). This means that for a three dimensional histogram (i.e. where the bin a sample belongs to is a function of three colour channels: red, green and blue), a single pixel will have its weight divided across seven bins: the centre bin, and the two neighbouring bins in each dimension. The distance a sample is from each bin, and the corresponding weight, is calculated as follows,

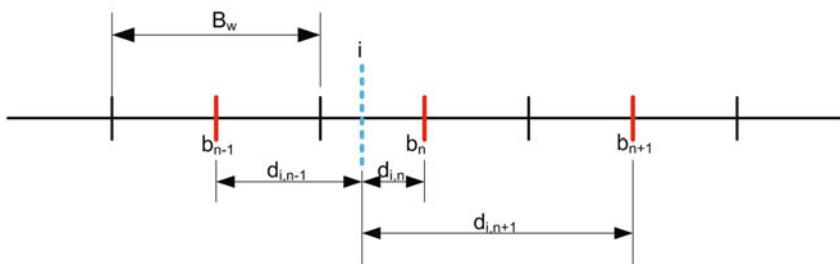
$$d_{i,n} = \frac{3}{2}B_w - |b_n - i|, \quad (3)$$

$$w_{i,n} = \frac{d_{i,n}}{\sum_{m=1}^M d_{i,m}}, \quad (4)$$



**Fig. 3** Segmenting a person into head, torso and leg regions. The top row shows input colour images, the bottom row shows the segmented silhouettes. Red regions are designated as being the head, green as the torso and blue as the legs.

where  $i$  is the input value,  $B_w$  is the width of a single bin,  $b_n$  is the centre of bin  $n$ ,  $d_{i,n}$  is the distance between the input value  $i$  and the centre of bin  $n$ , and  $w_{i,n}$  is the weight that is added to bin  $n$  in relation to sample  $i$ .  $M$  is the total number of bins that the sample is being split between, and  $m$  is an index into this set. This process of assigning weights is also illustrated in Figure 4.



**Fig. 4** Assigning weights to neighbouring bins in a soft histogram - The sample,  $i$  (blue dotted line), lies within bin  $n$ . Weight is assigned to bins  $n$ ,  $n - 1$  and  $n + 1$  based on its proximity to the centre of each bin (red line). It can be seen that the maximum distance possible between a sample and the centre of an adjacent bin is  $\frac{3}{2}B_w$ .

A moving average of the histogram is calculated such that,

$$C'(t) = \frac{L-1}{L} \times C'(t-1) + \frac{C(t)}{L}, \quad (5)$$

where  $C'(t)$  is the value of the average histogram at time  $t$ ,  $C(t)$  is the histogram computed for the frame at time  $t$ , and  $L$  is the learning rate.  $L$  is set according to

$$L = \frac{1}{T}; \text{for } T < W, \quad (6)$$

$$L = \frac{1}{W}; \text{for } W \geq T, \quad (7)$$

where  $W$  is the number of frames used in the model, and  $T$  is the number of updates performed on the model. This ensures that the image that is used to initialise the model does not dominate the model for a significant number of frames. Instead, new information is incorporated quickly when the model is new to provide a better representation of the object being modelled.

Histograms are compared using the Bhattacharyya coefficient,

$$B(C^i, C^j) = \sum_1^N \sqrt{C^i(n) \times C^j(n)}, \quad (8)$$

where  $B(C^i, C^j)$  is the Bhattacharyya coefficient for the comparison of the histograms  $C^i$  and  $C^j$ ,  $C^i(n)$  is the  $n$ th bin for the histogram  $C^i$ , and  $N$  is the total number of bins in the histogram. The histogram comparison is performed using histograms with their bin weights normalised such that they sum to 1. This is done to ensure size invariance. The comparison will return 1 for a perfect match, and 0 for no match.

When comparing colour models for two people, the similarity score is taken as the average of the three histogram comparisons,

$$P_{colour}(i, j) = \frac{(B(C_{Head}^i, C_{Head}^j) + B(C_{Torso}^i, C_{Torso}^j) + B(C_{Legs}^i, C_{Legs}^j))}{3}, \quad (9)$$

where  $P_{colour}(i, j)$  is the similarity score between models  $i$  and  $j$ .

### 3.2 Height

The height of the person is used as a simple descriptor. The height is view invariant, whilst other dimensions (width and depth/thickness) are dependent on the camera angle as well as the persons pose (i.e. as a person walks their width changes as their legs move). Heights are stored for the head, torso and legs.

To determine the height of the person, the head and feet must be located in the image. Top and bottom contours of the motion image are extracted and smoothed. The top of the head is located by searching the top contour to determine the highest

point,  $x_h, y_h$ . The feet positions,  $x_{lf}, y_{lf}, x_{rf}, y_{rf}$ , are determined by finding the lowest point on the bottom contour either side of the head position. A mean foot position ( $x_f, y_f$ ) is computed from these two detected feet locations.

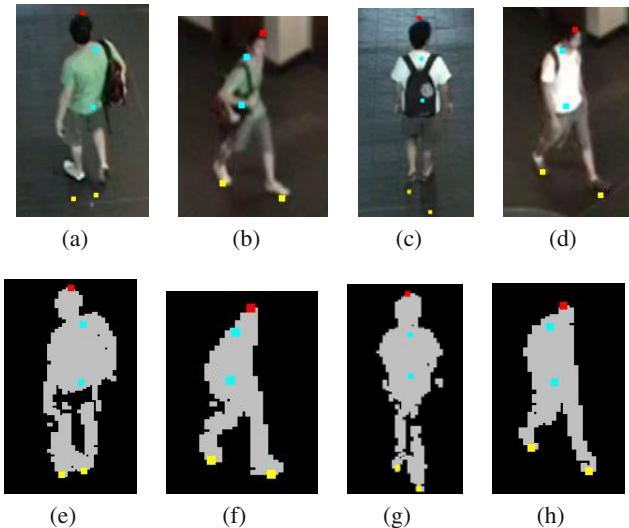
The subject is separated into head, torso and legs as outlined in Section 3.1. The mean points along the two dividing contours,

$$x_{neck}, y_{neck} = \frac{\sum V_{neck}(x)}{N_{neck}}, \frac{\sum V_{neck}(y)}{N_{neck}}, \quad (10)$$

$$x_{waist}, y_{waist} = \frac{\sum V_{waist}(x)}{N_{waist}}, \frac{\sum V_{waist}(y)}{N_{waist}}, \quad (11)$$

where  $x_{neck}, y_{neck}$  and  $x_{waist}, y_{waist}$  are the mean points of the neck and waist contours; and  $V_{neck}$  and  $V_{waist}$  are the contours that define the neck and waist boundaries (see Equation 2); are located and used to divide the region for the height calculations.

Figure 5 shows an example of the located head and feet points, and the points used to divide the subject into head, torso and legs.



**Fig. 5** Detecting the head and feet, and dividing the subject into three regions - The top row shows the colour input image and the bottom row shows the corresponding silhouette. Head, feet, waist and neck points are overlayed on both images. The head points are shown in red, feet shown in yellow, and median position of the waist and neck divisions shown in cyan. It can be seen that in some instances, shadows result in the feet being incorrectly located.

Using camera calibration, the image coordinates of the points can be transferred into a real world coordinate scheme.  $x_f, y_f$  is projected at an assumed  $z$  coordinate (height about the ground plane) of 0. The real world location of the feet can then be

projected back into the image plane at various heights (values of  $z$ ) to determine the height above the ground plane of the waist, neck and head. Heights for the individual components can then be determined,

$$H_{head} = z_{head}^w - z_{neck}^w, \quad (12)$$

$$H_{torso} = z_{neck}^w - z_{waist}^w, \quad (13)$$

$$H_{legs} = z_{waist}^w - z_{feet}^w, \quad (14)$$

where  $H_{head}$ ,  $H_{torso}$  and  $H_{legs}$  are the head, torso and legs heights in world coordinates, and  $z_{head}^w$ ,  $z_{neck}^w$ ,  $z_{waist}^w$ ,  $z_{feet}^w$  are the world coordinates (height off the ground plane) of the head, neck, waist and feet.  $z_{feet}^w$  is always set to 0 (i.e. the person's feet are on the ground).

Heights are progressively updated over multiple observations,

$$H'_{head}(t) = \frac{L-1}{L} \times H'_{head}(t-1) + \frac{H_{head}(t)}{L}, \quad (15)$$

$$H'_{torso}(t) = \frac{L-1}{L} \times H'_{torso}(t-1) + \frac{H_{torso}(t)}{L}, \quad (16)$$

$$H'_{legs}(t) = \frac{L-1}{L} \times H'_{legs}(t-1) + \frac{H_{legs}(t)}{L}, \quad (17)$$

where  $H'_{head}(t)$ ,  $H'_{torso}(t)$  and  $H'_{legs}(t)$  are the average head, torso and leg heights for the model at time  $t$ ;  $H_{head}(t)$ ,  $H_{torso}(t)$  and  $H_{legs}(t)$  are the heights for the image at the current time step computed as described in Equations [12] to [14], and  $L$  is the learning rate.  $L$  is defined in the same manner as the colour model (see Equations [6] and [7]).

For each update, an error is calculated between the average of the model and the new observation,

$$F_{head}^e(t) = |H'_{head}(t) - H_{head}(t)|, \quad (18)$$

$$F_{torso}^e(t) = |H'_{torso}(t) - H_{torso}(t)|, \quad (19)$$

$$F_{legs}^e(t) = |H'_{legs}(t) - H_{legs}(t)|, \quad (20)$$

where  $F_{head}^e(t)$ ,  $F_{torso}^e(t)$  and  $F_{legs}^e(t)$  are the frame errors for the head, torso and leg heights. Over time, an average error for each component ( $E_{head}$ ,  $E_{torso}$  and  $E_{legs}$ , for the head, torso and legs respectively) can be computed from these frame errors ( $F_{head}^e(t)$ ,  $F_{torso}^e(t)$  and  $F_{legs}^e(t)$ ) using the update method of the colour and height models (see Equations [5], [6] and [7]). The cumulative error is used as an approximation to the standard deviation (it is assumed that the observations over time form a Gaussian distribution) of the error, as it is not practical to re-compute the standard deviation each frame, and not ideal to assume a fixed standard deviation. Given that the standard deviation for a sample set is defined as,

$$\sigma = \sqrt{\frac{1}{N} \sum_{n=1}^N (\mu - s_n)^2}, \quad (21)$$

and in the proposed model, for each measure there is one observation at each time step ( $N = 1$ ), so the standard deviation at a given time step is,

$$\sigma = \sqrt{(\mu - s)^2} = |H'(t) - H(t)|, \quad (22)$$

which is the proposed error measure.

When comparing two size models, the mean heights and approximated standard deviations are used to determine the probability of a match. The probability for head, torso and legs heights are defined as,

$$P_{head}(i, j) = \Phi_{0, E_{head}(i)}(|H'_{head}(i) - H'_{head}(j)|), \quad (23)$$

$$P_{torso}(i, j) = \Phi_{0, E_{torso}(i)}(|H'_{torso}(i) - H'_{torso}(j)|), \quad (24)$$

$$P_{legs}(i, j) = \Phi_{0, E_{legs}(i)}(|H'_{legs}(i) - H'_{legs}(j)|), \quad (25)$$

where  $P_{head}(i, j)$  is match between the head component two models,  $i$  and  $j$ ,  $E_{head}(i)$  is the approximated standard deviation of the head height for model  $i$ ,  $H'_{head}(i)$  is the mean head height for model  $i$ , and  $\Phi_{\mu, \sigma}$  is the cumulative density function for the Gaussian distribution. The average of these scores,

$$P_{height}(i, j) = \frac{P_{head}(i, j) + P_{torso}(i, j) + P_{legs}(i, j)}{3}, \quad (26)$$

is taken as the match between models  $i$  and  $j$ .

### 3.3 Luggage

Luggage can be detected by observing symmetry in a person's silhouette (see [29]). From all angles, a person's silhouette is approximately symmetrical when they are walking or running, however objects they may be carrying lead to regions of asymmetry.

The head is located in the same manner as described in Section 3.2, and is used to divide the subject into two non-equal halves. Any region directly either side of the silhouette that has a height with 20% of the head height is not considered when calculating symmetry. We reason that (assuming the head is correctly detected) this portion of the silhouette will always contain the head and torso and some amount of the legs (depending on pose). Any luggage that is being carried will not clearly be visible, any asymmetry that is detected in this region is far more likely to arise from errors in the motion segmentation. Thus, this region is ignored.

This leaves us with two image regions, one either side of the centre column about the head, which are compared for symmetry. We define these image regions as being all pixels in the range  $1 : Left, 1 : Y$  for the left portion, and  $Right : X, 1 : Y$  for the right portion, where  $X$  and  $Y$  are the horizontal and vertical dimensions, and  $Left$  and  $Right$  are the left and right edges of the centre region. Asymmetrical

motion regions are located by searching for differences between the left portion and the mirror image of the right portion. Algorithmically, this is achieved as follows:

```

1: for  $i = Left; i >= 1; i = i - 1$  do
2:   for  $j = 1; j <= Y; j = j + 1$  do
3:     if  $M(i, j) = M(Right + (Left - i), j)$  then
4:        $A(i, j) = A(Right + (Left - i)) = False$ 
5:     else
6:        $A(i, j) = M(i, j)$ 
7:        $A(Right + (Left - i)) = M(i, j)$ 
8:     end if
9:   end for
10: end for
```

where  $A$  is a boolean image indicating regions of asymmetry and  $M$  is the motion image. It should be noted that the left and right portions that we are comparing are not necessarily the same size, and additional bounds checks that are not shown above need to be performed.

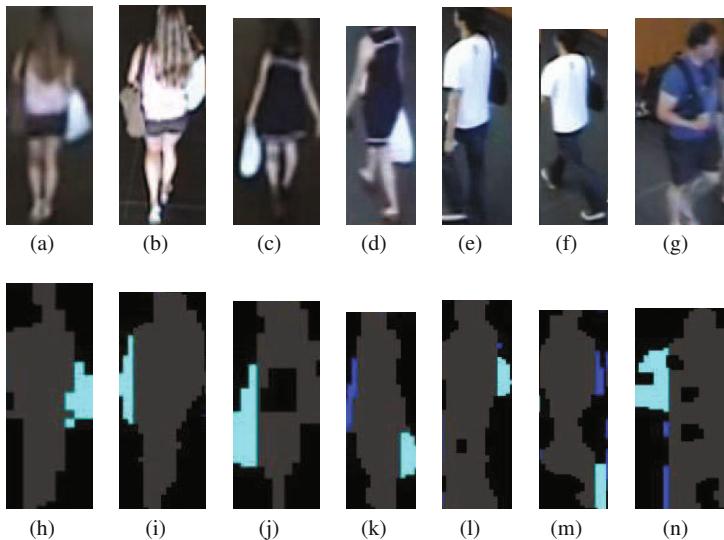
Connected components within  $A$  are detected, and the largest component is selected. If the size of this object is above a threshold (greater than 5% of the pixel count of the entire silhouette), it is accepted as a valid item of luggage. Examples of detected luggage items are shown in Figure 6. While the proposed approach is effective when the motion mask used is free of errors, it can be seen that when segmentation errors such as shadows are present (image pair (f)-(m)), additional regions of asymmetry result which can be incorrectly classified as an item of luggage. We also observe, that when a subject carries luggage on both sides of the body (such as image pairs (a)-(g) and (b)-(h)), the detector will not consistently detect the same item of luggage.

For the detected luggage items, we seek to model the colour of the item and the location of the luggage relative to the person. Two histograms are computed that correspond to the portion of the luggage item above the waist of the person,  $C_a$ , and the portion below the waist,  $C_b$ . The average amount of the luggage in each half is also stored ( $w_a$  and  $w_b$ ). We do not consider whether the luggage is carried on the left or right side of the person as this is dependant on the angle of camera used. Soft histograms as used in the colour model (see Section 5.1) are used, and the model is updated in the same manner (see Equations 5, 6 and 7).

Luggage models are compared by performing a weighted sum of the histogram matches, with weights determined according to the amount of the luggage that exists in each region,

$$P_{luggage}(i, j) = \min(w_a^i, w_a^j)B(C_a^i, C_a^j) + \min(w_b^i, w_b^j)B(C_b^i, C_b^j), \quad (27)$$

where  $i$  and  $j$  are the two luggage models we are comparing, and  $\min$  is an operator that determines the lowest of two values. In the event that one of models  $i$  and  $j$  has not detected a luggage item (and thus does not have a luggage model),  $P_{luggage}(i, j) = 1$ .



**Fig. 6** Detecting luggage - Top row shows input colour image, bottom row shows silhouette images with detected luggage objects marked in Cyan. Areas of asymmetry that are not detected as luggage objects are shown in blue. Errors in detection tend to occur when a subject is carrying multiple items (such as (a)-(g) and (b)-(h)), or when segmentation errors are present (see (f)-(m)).

## 4 Soft Biometrics for Operational Analytics

The soft biometrics outlined in Section 3 can be used to match people across a disjoint camera network, however it is not practical or possible to match every person within the network. However as our aim is to determine operational statistics, we only require information from a representative subset of the population. As such, we propose a system that automatically selects a suitable subset of the population to match across a disjoint camera network.

The proposed soft biometrics require a series of observations to build accurate models. We use the object tracking system described in [17] to track objects within each camera view, and build soft biometric models (see Section 3). Objects are not tracked between views, and the relationship between cameras (i.e. distance between cameras) is unknown.

It should be noted that the performance of the soft biometrics is impacted by the performance of the object tracking system. Errors in the object tracking, such as incorrect segmentation will result in errors in the soft biometric models. In this situation, the part-based segmentation (see Section 3.1) may fail, or produce incorrect results. To help alleviate this, objects that are occluded (as determined by the object tracker) are not considered when building soft biometric models, however it

is expected that some erroneous models will still result due to errors in the object tracking.

The proposed system operates in two phases: a training phase and a testing phase. In the training phase, the soft biometrics that are built can be combined to construct a set of average soft biometrics. During operation, the proposed system then uses the average soft biometrics to automatically select subjects to attempt to match across the disjoint camera views. As the objects are matched, operational statistics, such as the time taken to traverse the environment, are gathered. The average biometrics are outlined in Section 4.1, and the proposed system for calculating operational statistics is presented in Section 4.2.

## 4.1 Average Soft Biometrics

We propose average soft biometrics for colour (see Section 4.1.1) and size (see Section 4.1.2). An average for luggage is not computed, as not all people carry luggage, and an item of luggage is not visible from all camera angles (i.e. a backpack can be seen from side on, but not from a front or rear view).

These average biometrics can then be used to determine how distinct a person is. When multiple soft biometrics are used, the most distinct mode is taken. We argue that as long as a person has one trait that is distinct, then that person is distinct (i.e. a person may be wearing clothes similar to many others, but if they are an unusual height, they are distinct).

### 4.1.1 Colour

The colour soft biometric consists of three colour histograms. The average biometric is simply the average histogram for each component,

$$A_c^{\text{colour}} = \frac{\sum_i^N C_{c,i}}{N}, \quad (28)$$

where  $A_c^{\text{colour}}$  is the average biometrics for component  $c$ ,  $C_{c,i}$  is the colour histogram of component  $c$  for subject  $i$ , and  $N$  is the number of subjects used to build the model.

The distinctiveness,  $D_i$ , of a given subject  $i$ , is determined in the same manner as two subjects are compared,

$$D_i = \frac{B(A_{\text{head}}^{\text{colour}}, C_{\text{head},i}) + B(A_{\text{torso}}^{\text{colour}}, C_{\text{torso},i}) + B(A_{\text{legs}}^{\text{colour}}, C_{\text{legs},i})}{3}, \quad (29)$$

where  $B(a, b)$  is the Bhattacharyya coefficient (see Equation 8) for the histograms  $a$  and  $b$ .

#### 4.1.2 Size

Like the colour model, the size model consists of three parts: individual heights for the head, torso and legs. Each component is considered separately within the average model. Three histograms ( $A_{head}^{size}$ ,  $A_{torso}^{size}$  and  $A_{legs}^{size}$ ) are built that describe the likelihood that a given height for a given component will occur. Each histogram is normalised so that its bins sums to 1.

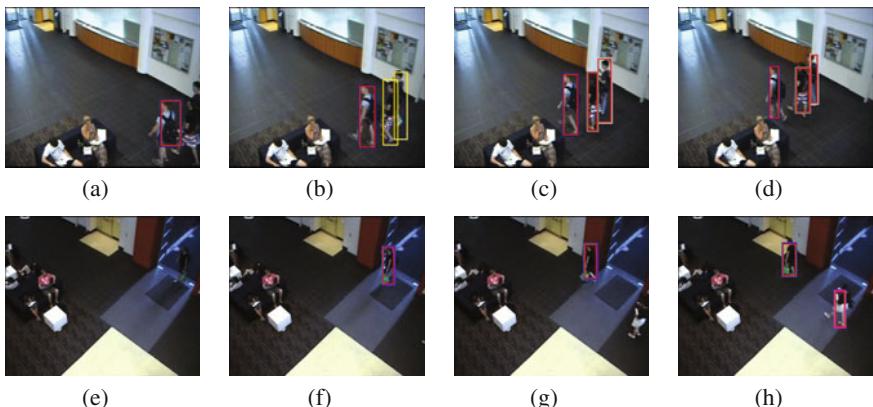
The distinctiveness of a subject can then be calculated as,

$$D_i = \frac{A_{head}^{size}(b_h) + A_{torso}^{size}(b_t) + A_{legs}^{size}(b_l)}{3}, \quad (30)$$

where  $b_h$ ,  $b_t$  and  $b_l$  are the bins that correspond to the heights of the for the head, torso and legs in the subject's model.

#### 4.2 Locating and Detecting Distinct People

The proposed system is intended for situations where there is a sparse disjoint camera network, and/or tracking throughout the complete environment is not possible. The tracking system outlined in [17] is used to track people within the individual cameras to build soft biometrics. An example of the tracking system output can be seen in Figure 7.

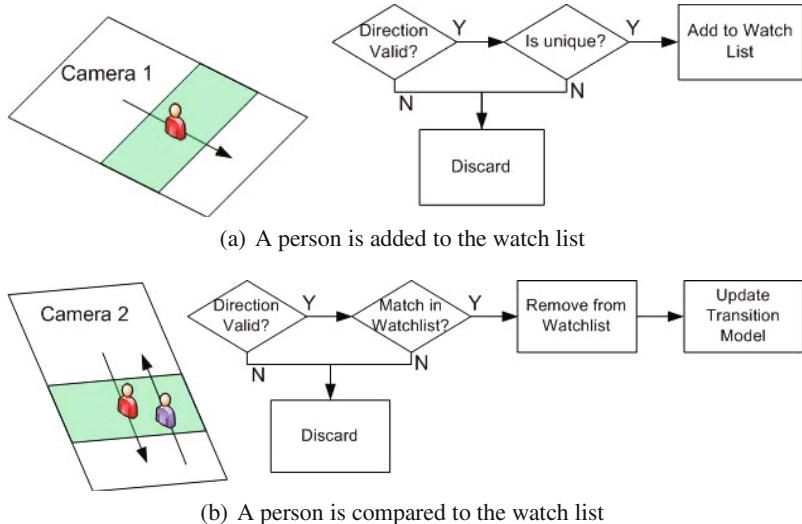


**Fig. 7** Example tracking output - Top and bottom rows show two examples of output from the tracking system. Tracking output is not completely accurate, and errors in the tracking filter through to the construction of the soft biometrics.

The proposed system operates in two stages:

1. Detecting unique people;
2. Detecting matches for the unique people.

These two situations are shown in Figure 8



**Fig. 8** Proposed system - The object tracking algorithm operates within the shaded regions of the camera views to build soft biometrics. People who are classified as unique are added to a watch list, which other people detected elsewhere in the network are compared to.

In the first stage, we seek to identify people who are distinct, and who can potentially be re-detected elsewhere in the camera network. Subjects are tracked as they move through the area of interest, and if they can be tracked for long enough to build a sufficient model (20 frames in the proposed system), they are compared to the average model to determine if they are unique. As outlined in Section 4.1, if any one mode is deemed to be unique, the person is considered unique. Unique people are added to a watch list so that they can be re-detected elsewhere. A minimum frame limit is imposed to reduce errors in the soft biometric models caused by tracking or segmentation errors, as we assume that the majority of frames will be correctly segmented.

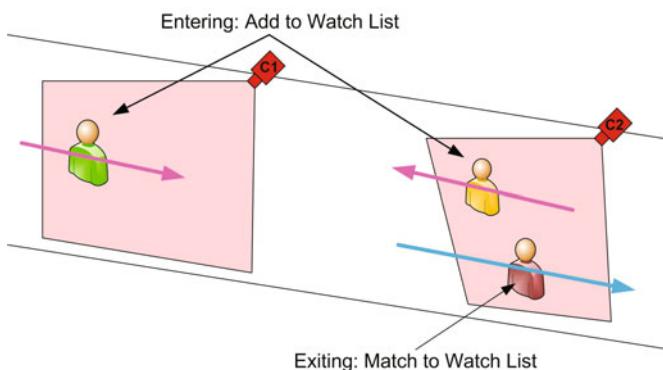
In the second stage, we aim to match people to those previously detected. Like the first stage, people are tracked through the region of interest and once a model has been built, they are compared to all subjects in the watch list by comparing the soft biometric models. For two objects to be matched, the colour, height and luggage soft biometrics must all have a similarity above a threshold,

$$P_{\text{colour}}(i, j) \geq T_{\text{colour}} \& P_{\text{height}}(i, j) \geq T_{\text{height}} \& P_{\text{luggage}}(i, j) \geq T_{\text{luggage}}, \quad (31)$$

where  $i$  and  $j$  are the source and target objects being matched;  $P_{colour}(i, j)$ ,  $P_{height}(i, j)$  and  $P_{luggage}(i, j)$  are the matches between the two models as defined in Equations 9 [26] and 27 respectively; and  $T_{colour}$ ,  $T_{height}$  and  $T_{luggage}$  are thresholds for the three soft biometrics. We argue that as each of these soft biometrics is essentially a weak classifier, all three must be satisfied for a match to occur. Thresholds are selected according to the performance of each classifier, with  $T_{height}$  and  $T_{luggage}$  typically set lower than  $T_{colour}$  (more false positives, fewer false negatives) as they are less reliable. In the event of multiple targets satisfying Equation 31, the target with the highest value for  $P_{colour}(i, j)$  is selected as the match, as the colour modality is the most reliable.

The first valid match that an object receives is taken to be correct (i.e. the system does not wait to see if there is a better match later on). This can potentially lead to errors if people of a similar appearance are present at the same time. However the requirement for people to be ‘distinct’ should reduce the likelihood of such an occurrence. As matches are made, the transition information (mean and standard deviation of time taken) is calculated.

In both stages of the system, the direction the subject is moving is monitored to determine if the person is valid for a comparison, or to be added to the watch list. For example, whilst a person may be tracked through a region where unique people are being detected, if they are moving in a direction that will exit the environment they will not be compared to the average model, or added to a watch list (see Figure 9).



**Fig. 9** Entering and exiting objects - In the scene shown either end of the corridor is an entrance and exit. Objects entering the scene are checked for uniqueness, and possibly added to the watch list. Objects exiting the scene are simply compared to the contents of the watch list.

When training an average biometric model, the system operates in the first mode only and disregards direction. Every object that can be tracked for a sufficient period to build a model is included in the average model.

## 5 Evaluation

We perform two evaluations:

1. Evaluate the performance of the soft biometrics themselves for a identification task;
2. Evaluate the performance of using soft biometrics to identify distinct looking people, and match these people across a disjoint camera network.

These evaluations are presented in Sections 5.1 and 5.2 respectively. Databases have been captured in house for each evaluation, and these are outlined in Sections 5.1.1 and 5.2.1 respectively.

### 5.1 Soft Biometric Performance

#### 5.1.1 Evaluation Data

To evaluate the performance of the soft biometrics for recognising people in a surveillance environment, a database has been captured that contains people moving through a building environment by up to 8 surveillance cameras. In total, 80 people are present in the database.

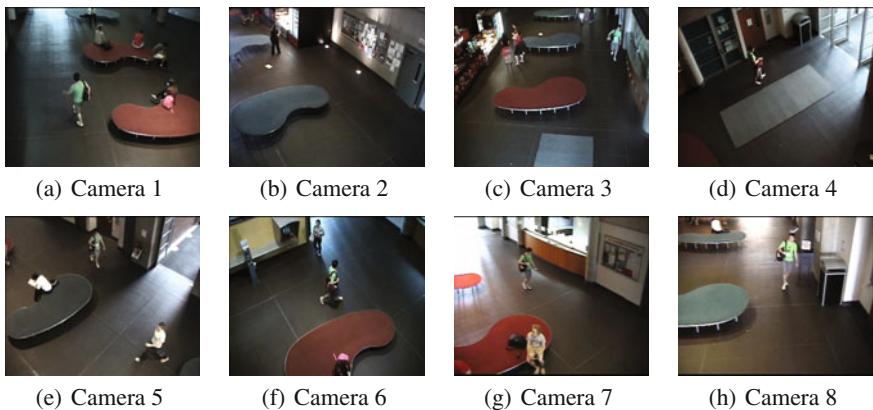
The vast majority of subjects will only pass through a subset of the camera network, and that subset varies from person to person. This provides a highly unconstrained environment in which to test the proposed soft biometrics. The cameras have all been calibrated using Tsai's method [61], and an example image from each camera is shown in Figure 10.

From Figure 10, it can be seen that there is varied lighting across the different camera views, and that subjects will be observed from different angles as they move through the network.

Subject locations in the network are hand annotated (every 10th frame is annotated and intermediate frame locations are interpolated). The annotated locations are coarse and the subject location is refined using motion segmentation (person detection [9, 63] could also be used), thus there are also potential errors that can be caused by incorrect motion segmentation.

In our evaluation, we consider the accuracy of the size and colour soft biometrics. We do not consider luggage as it is not carried by all subjects, and not visible from all camera angles. Cumulative match characteristic (CMC) and synthetic recognition rate (SRR) [24] curves are plotted for both the size and colour biometric as well as an equivalent colour biometric using a hard histogram, and for a fused system that uses a weighted sum to combine the colour (soft histogram), size and luggage biometrics. The following three configurations of soft biometric models are evaluated:

1. Soft biometric models trained on a single camera view;
2. Soft biometric models trained on two camera views;
3. Soft biometric models trained on three camera views.



**Fig. 10** Example images from the 8 cameras used to capture the soft biometric evaluation database. Note that there are different lighting conditions in each camera, and that as person moves through the network they are viewed from several different angles.

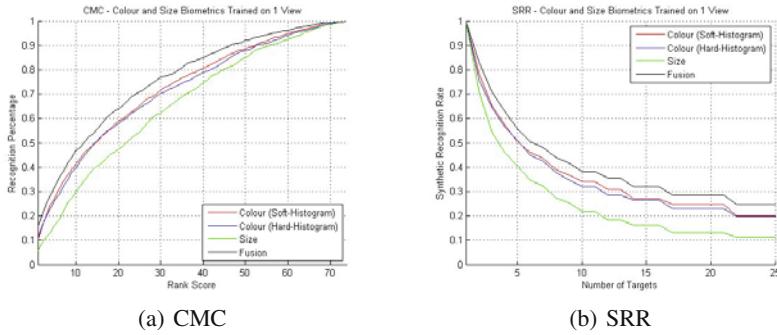
Models are trained and tested on separate views, and 20 images are used from each camera view. It should be noted that as we increase the number of views required for training models, fewer subjects from the database are included as subjects with insufficient data are omitted. As subjects within the database take widely varying paths through the camera network and pass through different numbers of cameras, not all subjects can be used in all cases.

### 5.1.2 Results

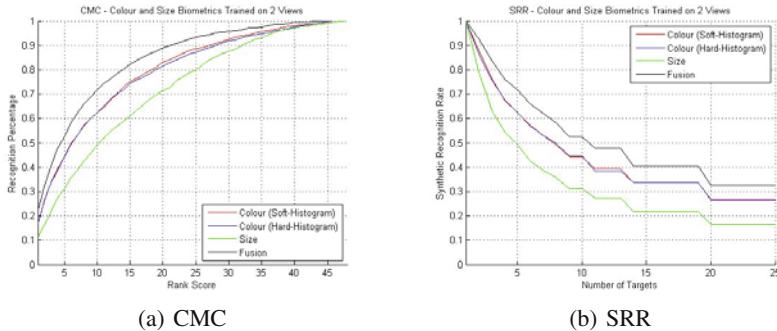
Cumulative match characteristic (CMC) and synthetic recognition rate (SRR) plots are shown for the colour soft biometric (hard and soft histogram variants), size biometric, and a fused system trained on one, two and three camera views are shown in Figures 11, 12 and 13 respectively.

It can be seen that the colour model consistently outperforms the height model, and both models improve as more views are used to train the models. The soft histogram variant of the colour model offers a small but consistent performance advantage over the hard histogram equivalent.

The superior performance of the colour model is to be expected, as there is a far greater variation in colour than size. A large number of people have heights that are within a few centimetres of each other, meaning there is much less variation in this trait. The height biometric is also less accurate to extract. Segmentation errors in the silhouette will result in errors in the detected height. An error of only a few pixels can result in a difference of a few centimetres or more, depending on where in the image the subject appears. While the colour biometric is also susceptible to segmentation errors, these do not result in a significantly different observation unless the segmentation errors result in large portions of the person not being visible.



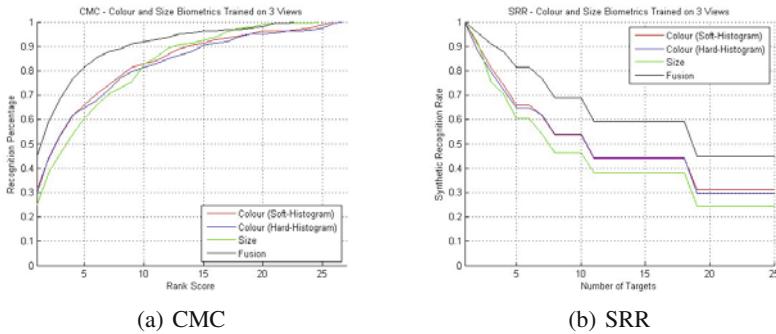
**Fig. 11** Cumulative match characteristic (CMC) and synthetic recognition rate (SRR) plots for models trained on a single camera view.



**Fig. 12** Cumulative match characteristic (CMC) and synthetic recognition rate (SRR) plots for models trained on two camera views.

(i.e. their legs or torso are not detected), or a large portion of the background being included in the model. In cases such as these, it is likely that the division of the person into head, torso and legs would fail (see Section 3.1) and a model would not be built for that frame.

It can also be seen that the soft histogram colour model consistently performs as well as, or better than, the hard histogram equivalent. While using a soft histogram only offers a small performance improvement, this is not surprising as the underlying model based on the division of the subject into three parts is unchanged and the scope for improvement using the soft histogram is limited. The nature of the soft histogram though, which allows ambiguous samples to be spread across multiple bins, does offer some help when dealing with challenging lighting conditions. Colours which lie close to a bin boundary are distributed across neighbouring bins, providing a limited amount of invariance to changes in colour balance between cameras. However, if the change in a colour between cameras is too large (greater than



**Fig. 13** Cumulative match characteristic (CMC) and synthetic recognition rate (SRR) plots for models trained on three camera views.

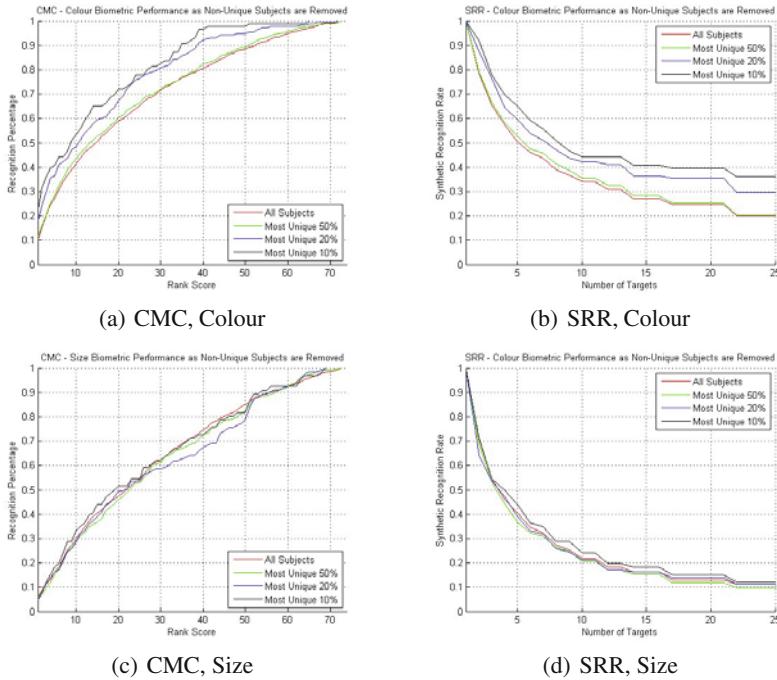
1.5 bin widths in the proposed system, see Section 3.1), the soft histogram will offer no support in dealing with colour changes.

We also observe that performance of the soft biometrics improve as more views are used to build the models. This is to be expected, as the use of multiple views ultimately leads to a more complete and accurate model. In the case of the colour biometric, a person may appear different from the front or back due to an item they are carrying, or an item of clothing such as a jacket that is open at the front. Including data from multiple viewpoints allows for a model that better represents the persons overall appearance.

From Figures 11, 12 and 13 it can also be seen that a simple weighted summation of the modalities results in a noticeable improvement in performance.

Figure 14 shows CMC and SRR plots for both the colour and size soft biometrics as non-unique subjects are removed from the probe set (note that the non-unique subjects are still contained in the gallery set). Subjects who have a uniqueness (determined using an average biometric, see Section 4.1 for further details) below a threshold are removed from the probe set. In Figure 14, soft biometric models are trained and tested on data captured from a single camera view.

For the colour modality, it can be seen that as the less distinct subjects are removed from the probe set, the matching accuracy improves and the performance gain becomes more pronounced as an increasing number of probe samples are excluded. The size modality however does not offer the same improvement, and at some operating points performance of the more distinct probe set is less than that of the unfiltered set. We attribute this to the inherent inaccuracy in the size models, and the low variation in peoples heights. As non-distinct subjects are removed, some of the more ‘unique’ subjects that remain are likely to have heights that are the result or poor segmentation or other errors, thus making them more difficult, rather than easier, to match correctly. However, we do note that at low ranks a performance improvement (while not as large as that in the colour domain) is observed. As our target application is intended to operate with smaller number of subjects (i.e. 20 or



**Fig. 14** CMC and SRR plots for the colour and size soft biometrics as non-unique subjects are removed from the gallery set.

less) in the gallery at any given time, we argue there is still value in using the height modality.

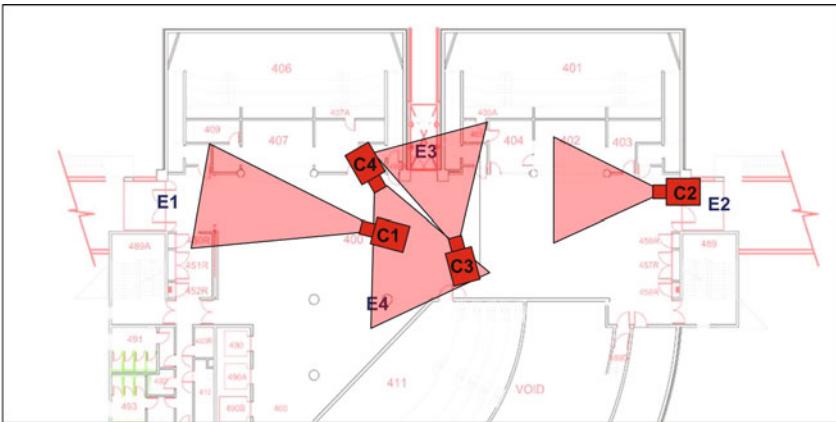
## 5.2 Soft Biometrics for Operations Tasks

### 5.2.1 Evaluation Data

A small database is captured in house, consisting of data captured from up to four cameras. Figure 15 shows the approximate layout of the camera network. All cameras are recorded simultaneously at 25 frames per second.

Two data sets are captured:

- Data set 1 uses only cameras 1 and 2, and consists of three 15,000 frame (10 minute, 25fps) sequences captured at different times (0915, 1315, 1615) during a single day. The three data sets have increasing levels of pedestrian traffic (0915 has the fewest people, 1615 the most people). These sequences are stored and processed at 4CIF resolution (704x576). These three sequences will be referred to as DS1-06-0915, DS1-06-1315 and DS1-06-1615.



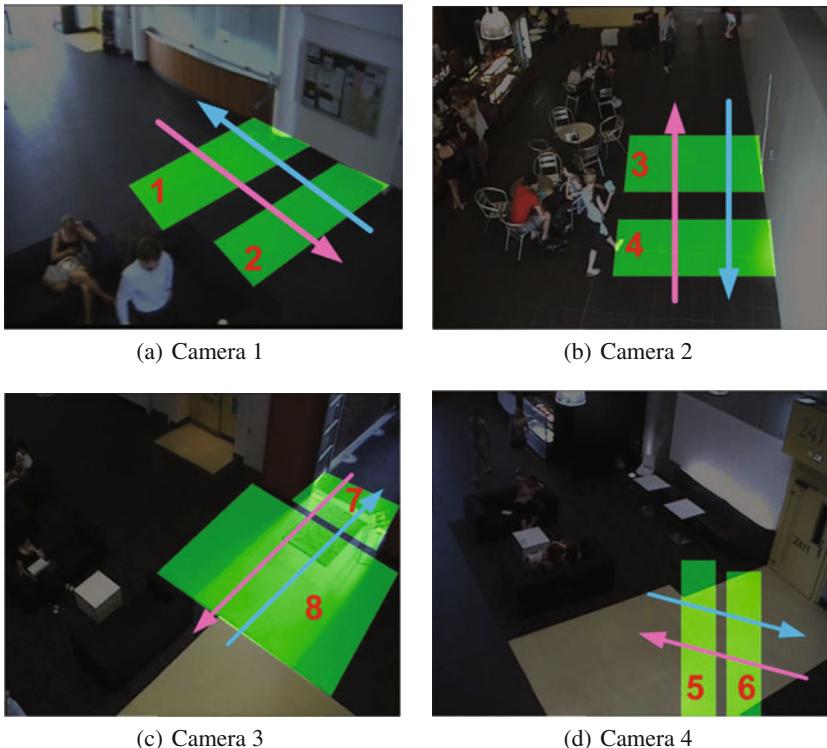
**Fig. 15** Camera network layout - Four cameras (C1, C2 ,C3 and C4; approximate camera views shown in red) observe the three external doorways(E1, E2 and E3), and an internal door. People enter and exit through these doors, as well as through other internal doorways and spaces within the building.

- Data set 2 uses all four cameras, and has two sequences of 30,000 frames each. These sequences have been captured at the same time of day (1710) on consecutive days. Each sequences contains a similar amount of traffic (roughly equivalent to DS1-06-1615). These sequences are stored and processed at CIF resolution (352x288). These two sequences will be referred to as DS2-17-1710 and DS2-18-1710.

Each camera view is configured with entrance and exit directions, as outlined in Section 4.2. Entrances and exits are shown in Figure 16. It should be noted that people who enter through one camera can exit the scene through any other camera, or through an area that is not covered by the cameras (see Figure 15). In this evaluation, we restrict each camera to having a single entrance and exit. We do this to increase the number of samples we observe for each transition, which allows us to better evaluate the accuracy of the estimation. It should be noted though that the proposed framework can support multiple entrances and exits within each camera, and it would also be possible to automatically detect these (see [60]), however this is beyond the scope of this work.

In our evaluation, average biometric models are trained on a single view at a given time period, and the system is then tested using data using captured at the other time period(s). Camera 1 is always used as the training view as it receives the most pedestrian traffic, and so provides the most training data and best average models.

People are detected entering and exiting all views within the test set (see Figure 16), resulting in estimates for every possible path through the camera network. In our evaluation we consider the accuracy of these estimates, and the accuracy with which matches are made between the camera views.



**Fig. 16** Camera configuration - Entrances and exits over which people are monitored are shown. Purple arrows indicate an entrance, blue arrows indicate an exit.

### 5.2.2 Results

Evaluation results for the proposed system using data set 1 are shown in Table II.

It can be seen from Table II that the proposed approach is able to accurately estimate the travel time in both directions in the majority of cases, particularly when sufficient samples are observed. Due to the fact that DS1-06-0915 contains the fewest people, estimates based on this data set are the least accurate. Similarly, estimates obtained using DS1-06-1615 are closest to the ground truth due to the increased number of people to detect and match.

Examples of correct and incorrect matches made using data set 1 are shown in Figures II and III respectively. It can be seen that the proposed soft biometrics are able to cope with variation in the viewing angle, as well as some variation in lighting conditions. The incorrect matches (see Figure III) are caused by either the source and target having a similar appearance; or by poor segmentation, either from the object tracking or motion segmentation (or both).



**Fig. 17** Examples of correctly matched people, all for  $T_m = 0.6$ . Top row shows an example of the ‘source’ image, bottom row the ‘target’ image. Image pairs (a)-(g) and (b)-(h) are from DS1-06-0915, (c)-(i) and (d)-(j) from DS1-06-1315, and (e)-(k) and (f)-(l) from DS1-06-1615. People are correctly matched despite changes in pose, and changes in colour caused by the differing lighting conditions.



**Fig. 18** Examples of incorrectly matched people, all for  $T_m = 0.6$ . Top row shows an example of the ‘source’ image, bottom row the ‘target’. Image pairs (a)-(g) and (b)-(h) are from DS1-06-0915, (c)-(i) and (d)-(j) from DS1-06-1315, and (e)-(k) and (f)-(l) from DS1-06-1615. Errors are made either due to poor image segmentation ((a)-(g), (c)-(i), (d)-(j)), or due to the people having a similar appearance ((b)-(h), (d)-(j), (e)-(k), (f)-(l)).

**Table 1** Evaluation results for data set 1 (DS1-06-0915 is abbreviated to 0915, DS1-06-1315 to 1315 and DS1-06-1615 to 1615) - Three different thresholds for selecting distinct people are used. Lower thresholds result in a higher number of people being added to the watch list and matched, however there are more incorrect matches made between people with a similar appearance. Ground truth times (measured in frames) taken to move through the network are  $\mu = 419, \sigma = 62$  for Left to Right, and  $\mu = 407, \sigma = 51$  for Right to Left.

Distinct Threshold	Training Data set	Testing Data set	C1⇒C2		C2⇒C1	
			Time Taken (frames)	Match Rate	Time Taken (frames)	Match Rate
$T_m = 0.5$	0915	1315	$\mu = 471, \sigma = 216$	60% (6/10)	$\mu = 431, \sigma = 135$	75% (9/12)
	0915	1615	$\mu = 342, \sigma = 112$	86% (6/7)	$\mu = 410, \sigma = 146$	57% (12/21)
	1315	0915	$\mu = 269, \sigma = 167$	33% (1/3)	$\mu = 342, \sigma = 137$	80% (4/5)
	1315	1615	$\mu = 342, \sigma = 112$	86% (6/7)	$\mu = 410, \sigma = 146$	57% (12/21)
	1615	0915	$\mu = 269, \sigma = 167$	33% (1/3)	$\mu = 342, \sigma = 137$	80% (4/5)
	1615	1315	$\mu = 486, \sigma = 239$	63% (5/8)	$\mu = 388, \sigma = 41$	75% (9/12)
$T_m = 0.6$	0915	1315	$\mu = 601, \sigma = 271$	66% (2/3)	$\mu = 395, \sigma = 44$	73% (8/11)
	0915	1615	$\mu = 340, \sigma = 121$	83% (5/6)	$\mu = 365, \sigma = 122$	61% (11/18)
	1315	0915	$\mu = 269, \sigma = 167$	33% (1/3)	$\mu = 338, \sigma = 153$	75% (3/4)
	1315	1615	$\mu = 340, \sigma = 121$	83% (5/6)	$\mu = 404, \sigma = 153$	61% (11/18)
	1615	0915	$\mu = 269, \sigma = 167$	33% (1/3)	$\mu = 338, \sigma = 153$	75% (3/4)
	1615	1315	$\mu = 580, \sigma = 237$	75% (3/4)	$\mu = 379, \sigma = 62$	73% (8/11)
$T_m = 0.7$	0915	1315	$\mu = 674, \sigma = 307$	50% (1/2)	$\mu = 387, \sigma = 47$	50% (2/4)
	0915	1615	$\mu = 420, \sigma = 48$	100% (2/2)	$\mu = 420, \sigma = 118$	78% (7/9)
	1315	0915	$\mu = 106, \sigma = 0$	0% (0/1)	$\mu = 421, \sigma = 247$	33% (1/3)
	1315	1615	$\mu = 399, \sigma = 49$	100% (3/3)	$\mu = 430, \sigma = 170$	69% (9/13)
	1615	0915	$\mu = 106, \sigma = 0$	0% (0/1)	$\mu = 587, \sigma = 97$	50% (1/2)
	1615	1315	$\mu = 674, \sigma = 307$	50% (1/2)	$\mu = 377, \sigma = 46$	60% (3/5)

Results for data set 2 are shown in Table 2. For simplicity, we consider transitions between the same cameras but in different directions as the same (i.e. the transitions C1⇒C2 and C2⇒C1 are considered together as C1↔C2).

Examples of correct and incorrect matches obtained using data set 2 are shown in Figures 19 and 20 respectively. As was observed in data set 1, the soft biometrics are able to cope with variations in viewing angle and colour balance between the camera views. This is even more apparent in data set 2 as cameras 3 and 4 have a greater variation in viewing angle relative to cameras 1 and 2, then cameras 1 and 2 do to each other. Within data set 2, this problem is further exacerbated by the reduction in resolution from 4CIF to CIF.

As was observed in data set 1, a significant portion of the errors are due to the tracking errors (see Figure 7 for an example of the tracking errors present) resulting in soft biometric models that are actually a combination of one or more people. Examples of this can be seen in Figure 20 (see image pairs (e)-(l), (f)-(m) and (g)-(n)). Despite these incorrect matches, accurate estimates of travel times between the camera views can still be obtained provided a sufficient number of samples can be observed. For transitions such as camera 1 to camera 2 (which is the main

**Table 2** Evaluation results for data set 2 - Three different thresholds for selecting distinct people are used. Like data set 1, lower thresholds result in a higher number of people being added to the watch list and matched. Ground truth transition times are  $\mu = 413, \sigma = 57$  for  $C1 \leftrightarrow C2$ ,  $\mu = 236, \sigma = 233$  for  $C1 \leftrightarrow C3$ ,  $\mu = 242, \sigma = 47$  for  $C1 \leftrightarrow C4$ ,  $\mu = 259, \sigma = 25$  for  $C2 \leftrightarrow C3$  and  $\mu = 290, \sigma = 65$  for  $C2 \leftrightarrow C4$ . No ground truth transition time for  $C3 \leftrightarrow C4$  is calculated as insufficient examples are observed in the data (there is only 1 occurrence).

Distinct Threshold	Transition	Training D2-14-1710		Training D2-15-1710	
		Testing D2-15-1710	Time Taken (frames)	Match Rate	Time Taken (frames)
$T_m = 0.6$	$C1 \leftrightarrow C2$	$\mu = 439, \sigma = 181$	73% (30/41)	$\mu = 459, \sigma = 153$	74% (23/31)
	$C1 \leftrightarrow C3$	$\mu = 451, \sigma = 235$	25% (1/4)	$\mu = 369, \sigma = 226$	50% (1/2)
	$C1 \leftrightarrow C4$	$\mu = 849, \sigma = 0$	0% (0/1)	$\mu = 974, \sigma = 0$	0% (0/1)
	$C2 \leftrightarrow C3$	$\mu = 215, \sigma = 88$	67% (2/3)	$\mu = 302, \sigma = 168$	50% (2/4)
	$C2 \leftrightarrow C4$	$\mu = 376, \sigma = 52$	67% (2/3)	$\mu = 278, \sigma = 52$	50% (3/5)
	$C3 \leftrightarrow C4$	N/A	N/A (0/0)	N/A	N/A (0/0)
$T_m = 0.7$	$C1 \leftrightarrow C2$	$\mu = 423, \sigma = 179$	68% (26/38)	$\mu = 459, \sigma = 153$	76% (22/29)
	$C1 \leftrightarrow C3$	$\mu = 451, \sigma = 235$	25% (1/4)	$\mu = 369, \sigma = 226$	50% (1/2)
	$C1 \leftrightarrow C4$	$\mu = 849, \sigma = 0$	0% (0/1)	$\mu = 974, \sigma = 0$	0% (0/1)
	$C2 \leftrightarrow C3$	$\mu = 215, \sigma = 88$	67% (2/3)	$\mu = 302, \sigma = 168$	50% (2/4)
	$C2 \leftrightarrow C4$	$\mu = 376, \sigma = 52$	67% (2/3)	$\mu = 278, \sigma = 52$	50% (3/5)
	$C3 \leftrightarrow C4$	N/A	N/A (0/0)	N/A	N/A (0/0)
$T_m = 0.8$	$C1 \leftrightarrow C2$	$\mu = 436, \sigma = 121$	75% (6/8)	$\mu = 555, \sigma = 306$	33% (2/6)
	$C1 \leftrightarrow C3$	$\mu = 441, \sigma = 101$	0% (0/2)	$\mu = 369, \sigma = 226$	50% (1/2)
	$C1 \leftrightarrow C4$	N/A	N/A (0/0)	$\mu = 974, \sigma = 0$	0% (0/1)
	$C2 \leftrightarrow C3$	N/A	N/A (0/0)	$\mu = 302, \sigma = 168$	50% (2/4)
	$C2 \leftrightarrow C4$	N/A	N/A (0/0)	$\mu = 487, \sigma = 220$	33% (1/3)
	$C3 \leftrightarrow C4$	N/A	N/A (0/0)	N/A	N/A (0/0)

thoroughfare in this environment) where there is a large number of observations, the overall estimate of the travel time is still accurate despite the errors. For the other less common transitions however, estimates are much less accurate. However if additional training data was available it is reasonable to expect that these estimates would improve over time.

In both data sets, we observe that as expected, increasing  $T_m$  reduces the number of people matched between views. However the overall accuracy of the matches remains consistent in most cases. In data set 1 the overall match rate is 66% for  $T_m = 0.5$ , 67% for  $T_m = 0.6$  and 64% for  $T_m = 0.7$ . In data set 2 we observe that the match rate is 62% for  $T_m = 0.6$ , 66% for  $T_m = 0.7$  and 46% for  $T_m = 0.8$ .

As  $T_m$  increases, the number of erroneous matches made due to people having a non-distinct appearance decreases, however errors caused by poor tracking and segmentation persist. This is particularly apparent in data set 2 when  $T_m = 0.8$ . Figure 21 shows 7 of the 10 errors obtained when training using DS2-15-1710 and testing using DS2-14-1710. It can be seen that in all errors the source is poorly segmented. Either part of the subject has been clipped ((a), (c), (f), (g)), or the subject is actually two people ((b), (d), (e)). Poorly or erroneously segmented people are more readily



**Fig. 19** Examples of correctly matched people, all for  $T_m = 0.6$ . Top row shows an example of the ‘source’ image, bottom row the ‘target’. Image pairs (a)-(g), (b)-(h) and (c)-(i) show correct matches made between cameras 1 and 3; pairs (d)-(j) and (e)-(k) show correct matches made between cameras 2 and 3; and pair (f)-(l) shows a match made between cameras 2 and 4.



**Fig. 20** Examples of incorrectly matched people, all for  $T_m = 0.6$ . Top row shows an example of the ‘source’ image, bottom row the ‘target’ image. Image pairs (a)-(h), (b)-(i), (c)-(j) and (d)-(k) are matched incorrectly due to the source and target being of a similar colour. The erroneous image pairs (e)-(l), (f)-(m) and (g)-(n) are the result of poor tracking leading to poorly trained soft biometric models.

retained as  $T_m$  increases as they simply appear more unique. The poor segmentation of these subject means that the soft biometrics are less accurate and so the subject themselves are more likely to be erroneously matched. This could be overcome either through more accurate tracking that feeds into the soft biometrics (which could possibly be aided by using a technique such as the ‘virtual gate’ [35] to detect people as they cross the region of interest), or by applying more stringent checks to the segmented people to make sure that they do include only a single, complete person.



**Fig. 21** Incorrect matches when training with D2-15-1710 and testing with D2-14-1710 for  $T_m = 0.8$ . The top row shows an example source image for the subject. The bottom row shows an example target image for the subject.

## 6 Conclusion

In this chapter we have demonstrated how soft biometrics can be used to recognise people in a disjoint camera network, and how they can be applied to measure operational information such as the average time taken to travel between two points. We have presented a novel approach that calculates an average soft biometric that can be used to locate distinct or unusual looking people, allowing the system to automatically select an appropriate subset of people within a scene for measurement.

In this chapter we have applied this technique to estimating the travel time between multiple points within an environment. We have shown that using simple descriptors (colour, height and luggage), we are able to obtain good estimates for the transition times provided there are sufficient observations. By filtering the set of people that we seek to re-detect, we effectively simplify the matching problem, enabling these simple descriptors to be applied with good results. It should be noted however that additional soft biometrics (such as hair and skin colour, gender, etc.)

could also be used within this framework, as well as more complex texture based descriptors. The proposed framework also has additional applications, such as to estimating a coarse trajectory through an environment, or estimating dwell time within a space. The underlying soft biometrics also have many other applications within security tasks.

Based on this results contained within this chapter, there are several avenues for future work. These include:

- Extending the approach to an incremental framework, where the requirements on the ‘uniqueness’ of individuals is continuously relaxed as more observations are made, and the layout of the network becomes established.
- The use of a robust estimator to estimate the transitions times.
- Testing the proposed system on larger data sets and more complex camera networks.
- The inclusion of additional soft biometrics and descriptors, and improvements to the accuracy of the soft biometrics, through techniques such as colour normalisation to help overcome errors caused by inconsistent lighting within and between camera views.
- Investigating methods to ensure that only correctly segmented people are included, either through improvement of the underlying tracking system, through automatically detecting tracking errors, or through the use of other techniques to detect people as they enter the environment.

## References

- [1] Ailisto, H., Vildjounaite, E., Lindholm, M., Makela, S., Peltola, J.: Soft biometrics—combining body weight and fat measurements with fingerprint biometrics. *Pattern Recognition Letters* 27(5), 325–334 (2006)
- [2] Bak, S., Corvee, E., Bremond, F., Thonnat, M.: Person re-identification using haar-based and dcd-based signature. In: 2nd Workshop on Activity Monitoring by Multi-Camera Surveillance Systems, AMMCSS 2010, in Conjunction with 7th IEEE International Conference on Advanced Video and Signal-Based Surveillance, AVSS, AVSS (2010)
- [3] Bak, S., Corvee, E., Bremond, F., Thonnat, M.: Person re-identification using spatial covariance regions of human body parts. In: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 435–440 (2010)
- [4] Bazzani, L., Cristani, M., Perina, A., Farenzena, M., Murino, V.: Multiple-shot person re-identification by hpe signature. In: 2010 20th International Conference on Pattern Recognition (ICPR), pp. 1413–1416 (2010)
- [5] Beymer, D.: Person counting using stereo. In: Workshop on Human Motion, vol. 0, p. 127. IEEE Computer Society, Los Alamitos (2000),  
<http://doi.ieee.org/10.1109/HUMO.2000.897382>
- [6] Breitenstein, M.D., Reichlin, F., Leibe, B., Koller-Meier, E., Van Gool, L.: Online multi-person tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 99 (2010)

- [7] Chien, S.Y., Chan, W.K., Cherng, D.C., Chang, J.Y.: Human object tracking algorithm with human color structure descriptor for video surveillance systems. In: 2006 IEEE International Conference on Multimedia and Expo, pp. 2097–2100 (2006)
- [8] Collins, R., Gross, R., Shi, J.: Silhouette-based human identification from body shape and gait. In: Proceedings of IEEE Conference on Face and Gesture Recognition, pp. 351–356 (2002)
- [9] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Schmid, C., Soatto, S., Tomasi, C. (eds.) International Conference on Computer Vision & Pattern Recognition, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, vol. 2, pp. 886–893 (2005)
- [10] Damen, D., Hogg, D.C.: Detecting Carried Objects in Short Video Sequences. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 154–167. Springer, Heidelberg (2008), [http://dx.doi.org/10.1007/978-3-540-88690-7\\_12](http://dx.doi.org/10.1007/978-3-540-88690-7_12)
- [11] Damen, D., Hogg, D.: Recognizing linked events: Searching the space of feasible explanations. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 927–934 (2009)
- [12] Dantcheva, A., Velardo, C., D'Angelo, A., Dugelay, J.L.: Bag of soft biometrics for person identification: New trends and challenges. *Multimedia Tools and Applications* 51(2), 739–777 (2011)
- [13] Demirkus, M., Garg, K., Guler, S.: Automated person categorization for video surveillance using soft biometrics. In: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 7667, p. 22 (2010)
- [14] Demirkus, M., Toews, M., Clark, J.J., Arbel, T.: Gender classification from unconstrained video sequences. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 55–62 (2010)
- [15] Denman, S., Fookes, C., Bialkowski, A., Sridharan, S.: Soft-Biometrics: unconstrained authentication in a surveillance environment. In: Digital Image Computing: Techniques and Applications (DICTA), pp. 196–203 (2009)
- [16] Denman, S., Fookes, C., Sridharan, S.: Improved simultaneous computation of motion detection and optical flow for object tracking. In: Digital Image Computing: Techniques and Applications (DICTA), Melbourne, Australia (2009)
- [17] Denman, S., Fookes, C., Sridharan, S.: Group segmentation during object tracking using optical flow discontinuities. In: The 4th Pacific-Rim Symposium on Image and Video Technology, Singapore, pp. 270–275 (2010)
- [18] Denman, S., Fookes, C., Sridharan, S., Ryan, D.: Multi-modal object tracking using dynamic performance metrics. In: 7th IEEE International Conference on Advanced Video and Signal-Based Surveillance, Boston, USA, pp. 286–293 (2010)
- [19] Farenzena, M., Bazzani, L., Perina, A., Murino, V., Cristani, M.: Person re-identification by symmetry-driven accumulation of local features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2360–2367 (2010)
- [20] Fleuret, F., Berclaz, J., Lengagne, R., Fua, P.A.F.P.: Multicamera people tracking with a probabilistic occupancy map. *Transactions on Pattern Analysis and Machine Intelligence* 30(2), 267–282, 0162-8828 (2008)
- [21] Forssen, P.E.: Maximally stable colour regions for recognition and matching. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8 (June 2007), doi:10.1109/CVPR.2007.383120

- [22] Gheissari, N., Sebastian, T.B., Hartley, R.: Person reidentification using spatiotemporal appearance. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1528–1535 (2006)
- [23] Gray, D., Tao, H.: Viewpoint Invariant Pedestrian Recognition with an Ensemble of Localized Features. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 262–275. Springer, Heidelberg (2008)
- [24] Gray, D., Brennan, S., Tao, H.: Evaluating appearance models for recognition, acquisition and tracking. In: PETS (2007)
- [25] Hahnel, M., Klunder, D., Kraiss, K.: Color and texture features for person recognition. In: IEEE International Joint Conference on Neural Networks, Budapest, Hungary, p. 652 (2004)
- [26] Han, J., Bhanu, B.: Individual recognition using gait energy image. IEEE Transactions on Pattern Analysis and Machine Intelligence 28, 316–322 (2006)
- [27] Haritaoglu, I., Flickner, M.: Attentive billboards: towards to video based customer behavior understanding. In: Proceedings of Sixth IEEE Workshop on Applications of Computer Vision (WACV 2002), pp. 127–131 (2002), doi:10.1109/WACV.2002.1182169
- [28] Haritaoglu, I., Cutler, R., Harwood, D., Davis, L.S.: Backpack: Detection of people carrying objects using silhouettes. In: IEEE International Conference on Computer Vision, vol. 1, p. 102 (1999), doi: doi.ieeecomputersociety.org/10.1109/ICCV.1999.791204
- [29] Haritaoglu, I., Harwood, D., Davis, L.: W4: real-time surveillance of people and their activities. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 809–830 (2000)
- [30] Haritaoglu, I., Beymer, D., Flickner, M.: Ghost3D: detecting body posture and parts using stereo. In: Proceedings of Workshop on Motion and Video Computing, pp. 175–180 (2002)
- [31] Hu, M., Hu, W., Tan, T.: Tracking people through occlusions. In: Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, pp. 724–727 (2004)
- [32] Hu, W., Hu, M., Zhou, X., Tan, T., Lou, J., Maybank, S.: Principal axis-based correspondence between multiple cameras for people tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(4), 663–671 (2006), doi:10.1109/TPAMI.2006.80
- [33] Jain, A.K., Dass, S.C., Nandakumar, K.: Soft biometric traits for personal recognition systems. In: International Conference on Biometric Authentication, Hong Kong, pp. 731–738 (2004)
- [34] Jovic, N., Frey, B., Kannan, A.: Epitomic analysis of appearance and shape. In: Proceedings of Ninth IEEE International Conference on Computer Vision, vol. 1, pp. 34–41 (2003), doi:10.1109/ICCV.2003.1238311
- [35] Kim, B.-S., Lee, G.-G., Yoon, J.-Y., Kim, J.-J., Kim, W.-Y.: A Method of Counting Pedestrians in Crowded Scenes. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 1117–1126. Springer, Heidelberg (2008)
- [36] Krahnstoever, N., Rittscher, J., Tu, P., Chean, K., Tomlinson, T.: Activity recognition using visual tracking and RFID. In: IEEE Workshop on Applications of Computer Vision and the IEEE Workshop on Motion and Video Computing, vol. 1, pp. 494–500. IEEE Computer Society, Los Alamitos (2005), doi: doi.ieeecomputersociety.org/10.1109/ACVMOT.2005.17
- [37] Lazebnik, S., Schmid, C., Ponce, J.: A sparse texture representation using local affine regions. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1265–1278 (2005), doi: doi.ieeecomputersociety.org/10.1109/TPAMI.2005.151

- [38] Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: Advances in Neural Information Processing Systems (2010)
- [39] Lin, Z., Davis, L.S.: Learning pairwise dissimilarity profiles for appearance recognition in visual surveillance. In: ISVC, pp. 23–34 (2008)
- [40] Liu, X., Krahstroever, N., Yu, T., Tu, P.: What are customers looking at? (2007)
- [41] Lu, J., Tan, Y.P.: Gait-based human age estimation. In: 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 1718–1721 (2010)
- [42] Lu, X., Jain, A.K.: Ethnicity identification from face images. In: Proc. SPIE, vol. 5404, pp. 114–123 (2004)
- [43] Macrae, C.N., Bodenhausen, G.V.: Social cognition: Thinking categorically about others. *Annual Review of Psychology* 51(1), 93–120 (2000)
- [44] Marcialis, G.L., Roli, F., Muntoni, D.: Group-specific face verification using soft biometrics. *Journal of Visual Languages and Computing* 20(2), 101–109 (2009)
- [45] Nakajima, C., Pontil, M., Heisele, B., Poggio, T.: Full-body person recognition system. *Pattern Recognition* 36(9), 1997–2006 (2003)
- [46] Niinuma, K., Unsang, P., Jain, A.K.: Soft biometric traits for continuous user authentication. *IEEE Transactions on Information Forensics and Security* 5(4), 771–780 (2010)
- [47] Park, U., Jain, A.: Face matching and retrieval using soft biometrics. *IEEE Transactions on Information Forensics and Security* 5(3), 406–415 (2010), doi:10.1109/TIFS.2010.2049842
- [48] Park, U., Jain, A., Kitahara, I., Kogure, K., Hagita, N.: Vise: Visual search engine using multiple networked cameras. In: 18th International Conference on Pattern Recognition, ICPR 2006, vol. 3, pp. 1204–1207 (2006), doi:10.1109/ICPR.2006.1176
- [49] Popa, M., Rothkrantz, L., Yang, Z., Wiggers, P., Braspenning, R., Shan, C.: Analysis of shopping behavior based on surveillance system. In: 2010 IEEE International Conference on Systems Man and Cybernetics (SMC), pp. 2512–2519. IEEE (2010)
- [50] Popa, M., Rothkrantz, L., Wiggers, P.: Products appreciation by facial expressions analysis. In: Proceedings of the 11th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing on International Conference on Computer Systems and Technologies, pp. 293–298. ACM (2010)
- [51] Prosser, B., Zheng, W.-S., Gong, S., Xiang, T.: Person re-identification by support vector ranking. In: Proceedings of the British Machine Vision Conference, pp. 1–21. BMVA Press (2010), doi:10.5244/C.24.21
- [52] Qi, Y., Huang, G.-C., Wang, Y.-H.: Carrying object detection and tracking based on body main axis. In: International Conference on Wavelet Analysis and Pattern Recognition, ICWAPR 2007, vol. 3, pp. 1237–1240 (2007), doi:10.1109/ICWAPR.2007.4421623
- [53] Ran, Y., Rosenbush, G., Zheng, Q.: Computational approaches for real-time extraction of soft biometrics. In: IEEE Int. Conf. on Pattern Recognition, pp. 1–4 (2008)
- [54] Rodriguez, M., Ali, S., Kanade, T.: Tracking in unstructured crowded scenes. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 1389–1396 (2009)
- [55] Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using group tracking and local features. In: 7th IEEE International Conference on Advanced Video and Signal-Based Surveillance, Boston, USA, pp. 218–224 (2010)
- [56] Samangooei, S., Guo, B., Nixon, M.: The use of semantic human description as a soft biometric. In: 2nd IEEE International Conference on Biometrics: Theory, Applications and Systems, BTAS 2008, pp. 1–7 (2008), doi:10.1109/BTAS.2008.4699354

- [57] Schwartz, W.R., Davis, L.S.: Learning discriminative appearance-based models using partial least squares. In: XXII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), pp. 322–329 (2009)
- [58] Senior, A., Brown, L., Hampapur, A., Shu, C., Zhai, Y., Feris, R., Tian, Y., Borger, S., Carlson, C.: Video analytics for retail. In: IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007, pp. 423–428 (2007), doi:10.1109/AVSS.2007.4425348
- [59] Shan, C., Gong, S., McOwan, P.W.: Learning gender from human gaits and faces. In: IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS 2007), pp. 505–510 (2007), doi:10.1109/AVSS.2007.4425362
- [60] Stauffer, C.: Estimating tracking sources and sinks. In: Event Mining Workshop, Madison, WI (2003)
- [61] Tsai, R.Y.: An efficient and accurate camera calibration technique for 3d machine vision. In: IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, pp. 364–374 (1986)
- [62] Vaquero, D.A., Feris, R.S., Tran, D., Brown, L., Hampapur, A., Turk, M.: Attribute-based people search in surveillance environments. In: 2009 Workshop on Applications of Computer Vision (WACV), pp. 1–8 (2009), doi:10.1109/WACV.2009.5403131
- [63] Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR (2001)
- [64] Wang, L., Tan, T., Hu, W., Ning, H.: Automatic gait recognition based on statistical shape analysis. *IEEE Transactions on Image Processing* 12(9), 1120–1131 (2003)
- [65] Wayman, J.L.: Large-scale civilian biometric Systems-Issues and feasibility. In: Proceedings of Card Tech/Secur Tech ID, vol. 732 (1997)
- [66] Xiang-tao, C., Zhi-hui, F., Hui, W., Zhe-qing, L.: Automatic gait recognition using kernel principal component analysis. In: 2010 International Conference on ICBECS, pp. 1–4 (2010)
- [67] Xiaogang, W., Doretto, G., Sebastian, T., Rittscher, J., Tu, P.: Shape and appearance context modeling. In: IEEE 11th International Conference on Computer Vision, ICCV 2007, pp. 1–8 (2007)
- [68] Xu, J., Denman, S., Fookes, C., Sridaharan, S.: Activity modelling in crowded environments: A soft-decision approach. In: International Conference on Digital Image Computing: Techniques and Applications, Noosa Heads, Queensland, Australia (2011)
- [69] Yu, S., Tan, T., Huang, K., Jia, K., Wu, X.: A study on Gait-Based gender classification. *IEEE Transactions on Image Processing* 18(8), 1905–1910 (2009), doi:10.1109/TIP.2009.2020535
- [70] Zhang, E., Zhao, Y., Xiong, W.: Active energy image plus 2dlpp for gait recognition. *Signal Processing* 90(7), 2295–2302 (2010)
- [71] Zheng, W.-S., Gong, S., Xiang, T.: Associating groups of people. In: BMVC 2009 (2009)
- [72] Zheng, W.-S., Gong, S., Xiang, T.: Person re-identification by probabilistic relative distance comparison. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 649–656 (2011)

# **Part III**

# **Behaviour Analysis**

# Automatic Activity Profile Generation from Detected Functional Regions for Video Scene Analysis

Eran Squires, Matthew Turek, Roderic Collins,  
A.G. Amitha Perera, and Anthony Hoogs

**Abstract.** The potential applications of video surveillance to the Business Intelligence domain continue to grow. For example, automatic computer vision algorithms can provide a fast, efficient process to screen hundreds of hours of video for activity patterns that potentially impact the business. Two such algorithms and their variants are discussed in this chapter. These algorithms analyze surveillance video in order to automatically recognize various functional elements, such as: walkways, roadways, parking-spots, and doorways, through their interactions with pedestrian and vehicle detections. The recognized functional element regions provide a means of capturing statistics related to particular businesses. For example, the owner may be interested in the number of people that enter or exit their business versus the number of people that walk past. Results are shown on functional element recognition and business related activity profiles that demonstrate the effectiveness of these algorithms. Experiments are performed using webcam video of a downtown main street in Ocean City NJ, and surveillance video from the CAVIAR shopping center dataset.

## 1 Introduction

Businesses have been utilizing video surveillance for years to detect theft and to monitor for inappropriate or dangerous behavior. These tasks have been manually performed by security guards, which helps to prevent obvious violations and to preserve inventory. However, this video can provide much more value to the business by also providing data for marketing analysis, which can potentially lead to a significant impact on the bottom line. For example, the video can be used to analyze the behavior of pedestrians around a business, as well as to monitor their interactions

---

Eran Squires · Matthew Turek · Roderic Collins · A.G. Amitha Perera · Anthony Hoogs  
Kitware, 28 Corporate Drive, Clifton Park, NY 12065

e-mail: eran.squires@kitware.com, matt.turek@kitware.com,  
roddy.collins@kitware.com, amitha.perera@kitware.com,  
anthony.hoogs@kitware.com

with functional element regions. A functional element is a location or a static object that is primarily defined by its specific function or purpose, rather than its appearance or shape. See the glossary at the end of the introduction for a summary of terms and their definitions. This method combines techniques from video scene analysis with functional recognition to decompose a video scene into its functional elements such as parking-spots, doorway, roadways and walkways. This can enable a business owner to determine what external factors influence sales and potential sales. For example, the owner may be interested in the number of people that enter or exit their store versus the number of people that walk past. There may also be parking circumstances that reduce the flow of pedestrian traffic near their store, such as vehicles that are parked for long periods in what should be short-term parking spots. This information is contained in the ever increasing amount of surveillance video that businesses are acquiring. However, there is not enough man-power to sift through the data and little guidance as to what information is relevant. To this end, there is clear value in automatic algorithms that provide a fast and efficient process to screen the hours of video for activity patterns that potentially impact the business.

Two approaches for automatically analyzing this video for business related intelligence are discussed here. These approaches use surveillance video to analyze the normal pedestrian and vehicle behaviors around a set of store fronts in order to automatically recognize various functional elements. Activity profiles are then extracted from these functional elements which can be used to analyze the causal relationships between functional elements and a business.

Both approaches are robust against highly fragmented tracks and are capable of differentiating between functional elements. The first of these two approaches tests track detections against four predefined track-type models: pedestrians; driving-vehicle; parking-vehicle; and an “other” model to produce a probability of fit to each. The 2D position estimates for each detection are then used as weighted evidence in a functional activity detector which is essentially a nonparametric kernel density estimation algorithm specific to each functional element. For example, the doorway and parking-spot element uses a novel flashlight model with a beam kernel [1]. However, the doorway functional activity detector weights each kernel by the pedestrian track-type probability, while the parking-spot weights each kernel by the parking-vehicle probability. The kernel density for each element is referred to as a functional likelihood map, which reveals the most likely spatial regions for each functional element. This approach can generate any type of functional likelihood map that has supporting track-type features. For example, the track types can be extended to fast and slow driving vehicles, which can be used to generate main and side road functional likelihood maps, respectively.

The second approach is an unsupervised method that identifies spatial regions with similar behaviors [2]. It starts by clustering a set of track-based features derived from spatial grid cells into a behavioral codebook, where each codeword is a group of statistically similar features (behaviors). Codeword histograms are formed for each cell, which are then clustered to produce unique groups of multi-modal behaviors referred to as functional categories. Each spatial grid cell is assigned the label of its most dominant functional category resulting in a set of functional

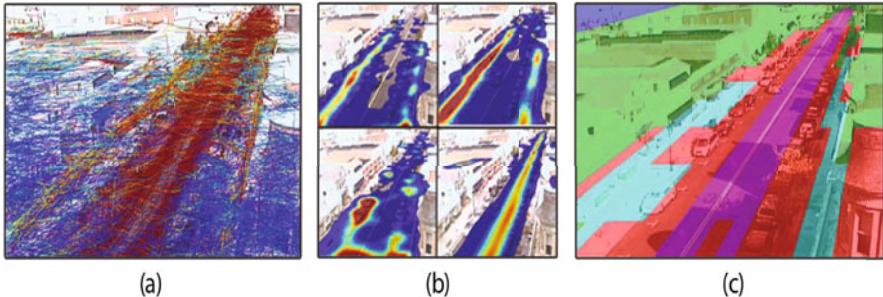
regions. The functional regions produced have the advantage of being able to represent multiple complex modes of behaviors for a single region; however, their semantic meaning is unknown, as this is a fully unsupervised approach. Therefore, these two approaches [1][2] are combined by using features derived from the functional likelihood maps from [1] as inputs to the unsupervised approach [2]. The resulting functional regions are more accurate, and have labels allowing them to be used in the higher level analysis needed for business intelligence studies. In order to simplify referencing these approaches in the remaining sections, the first approach [1] will be referred to as the “Functional Likelihood” method and the second [2] as the “Functional Category” method. Further details on these approaches are discussed in Sects. 3 and 4.



**Fig. 1** (a) Ocean City NJ video, where shops are mostly on the left side of the street (b) CAVIAR Shopping Center in Portugal, where the left wall is lined with shops

These methods are applicable to surveillance cameras as well as webcams. Webcam videos are highly challenging due to their poor quality and low frame rates, both of which lead to highly fragmented and corrupted tracks. Frames from a webcam in Ocean City (OC), NJ and from the CAVIAR shopping center surveillance video are shown in Figs. 1a and 1b respectively. Fig. 1a shows the high level of fragmented tracks from one day of 1Hz OC video along with recognition results for the Functional Likelihood and Category approaches from [1][2]. The complexity of the OC video prevents many approaches from working here [1][2][3]. While recognition methods based solely on appearance might be able to detect the road and perhaps the sidewalks, they would have great difficulty detecting the doorways, and occupied parking spots would be difficult to distinguish from roads. On the other hand, the approaches from [1][2] successfully lead to clusters of grid cells with similar behaviors that yield functional categories, e.g. mixtures of vehicle traffic lanes (roadway), pedestrian traffic lanes (sidewalk), parking spots, and building doorways. The direct results from [1] are in the form of likelihood maps for each functional element (see Fig. 1b), where darker red indicates cells that are more likely to correspond to that functional element and darker blue is less likely. The direct results from [2] are clusters of cells that have similar behaviors, e.g. mixtures of functional elements.

These two methods do have a few key limitations. For example, the Functional Likelihood approach requires the track’s detections to be tested against predefined



**Fig. 2** (a) Heat map showing all tracks for one day of OC webcam video; dark red indicates larger moving object bounding boxes (BB) while dark blue indicates smaller BBs (b) Like-lihood distributions from each functional activity detector [1] shown as heat maps; starting in the top left and going clockwise the functional elements are: doorway, walkway, roadway, and parking-spot; darker red indicates more likely regions, while darker blue is less likely (c) Results of functional category recognition [2]; regions with the same color are the same functional category; [1][2]

track-type models. Similarly, the functional activity models are restricted to the pre-defined functional elements. This approach also does not explicitly recognize functional regions; instead, it produces the likelihood maps for each functional element. On the other hand, the unsupervised approach does not rely on track-type classification or specific activity detectors, making it a very general approach. However, it does rely on features extracted from grid cells, which effectively introduces sensitivity to the resolution of the spatial grid, since smaller grid cells may have insufficient information for clustering and histogram formation. In addition, cell neighborhoods are not directly considered during clustering, although some of the features weakly associate nearby cells. The unsupervised approach is great for generalization, but lacks the semantic meaning behind the classified functional elements. However, using the functional likelihood maps from the Functional Likelihood approach as inputs to the Functional Category approach reduces its sensitivity to the grid cell size while also providing labels for the recognized functional regions.

Most other work on this topic are focused on using tracks of pedestrians and/or vehicles to define normal motion patterns and detect deviations [4][5][6][7][8][9][17]. Makris and Ellis [7][8] describe a method where a mean position trajectory is calculated with linearly connected boundaries that envelop all tracks in the cluster. Similarly, Swears and Hoogs [4] use an online hierarchical agglomerative clustering algorithm to combine similar tracks into Hidden Markov Models (HMMs). These types of algorithms work well for modeling normal paths and detecting deviations, and can also be used to detect entry and exit points in the scene. Another method [10] automatically detects track sources and sinks by clustering the start and end points of tracks, in order to improve track linking between source-sink pairs. While developing normalcy models for paths is a great tool for detecting anomalous behavior, it does not differentiate between parking spots and roadways, or walkways and building entrances. Alternatively, the source and sink model works well for detecting

points where moving objects enter and exit the scene, e.g. doorways and edge of camera field of view, but only when the tracks are continuous. Unfortunately, even the most advanced tracker produces many fragmented tracks making this approach unworkable for real-world data.

Alternative approaches rely on clustering features extracted directly from moving object detections [16, 20], making them independent of tracking errors. The clusters formed from these approaches use absolute position features that tie them to specific locations in the scene. Meanwhile, the two approaches discussed here perform clustering without position information; this allows a single cluster to be spatially disjoint across the scene, i.e. one cluster for building doorway located at the entrance of multiple buildings. Being spatially agnostic also allows the clusters to be transferred to similar scenes for functional region recognition without having to explicitly learn the new scene’s clusters (functional categories), see [2] for details. One drawback is that the some functional regions may be over-segmented, which can be mitigate by performing a smoothing operation on the resulting functional regions with a Markov random field classifier or a Gaussian spatial smoothing operation. The spatiac smoothing approach was used here when extracting activity profiles.

Operating on pure detections is less than ideal given that object Id is not preserved and kinematic information is lost. However, crowded scenes, as in [18], may prevent traditional trackers from producing reliable tracks. Alternative flow field based methods [18] can improve tracking in these scenes, but may be computationally intractable and still produce undesirable errors. Therefore, it may be necessary to directly use moving object detections [16, 20], which limit functional categories types that can be learned but still provide useful information for analysis.

Experiments are shown on two video datasets; the Ocean City webcam video introduced in [1], and the surveillance video from the CAVIAR dataset [3]. An example frame of both datasets is shown in Fig. 1. The experiments show functional element recognition results on both data sets. The first set of experiments show the results directly from the Functional Likelihood and Functional Category approaches. The second set of experiments show how using the functional likelihood maps from [1] as input features to the Functional Category approach improves the functional region recognition results while producing labels for each. Results are also shown for a simplified version of the Functional Category approach that directly clusters features derived from the functional likelihood maps, which demonstrates how the functional categories can be more descriptive, less sensitive to grid cell size, and produce categories with meaningful labels. This latter approach is used to extract statistics related to potential customers activities.

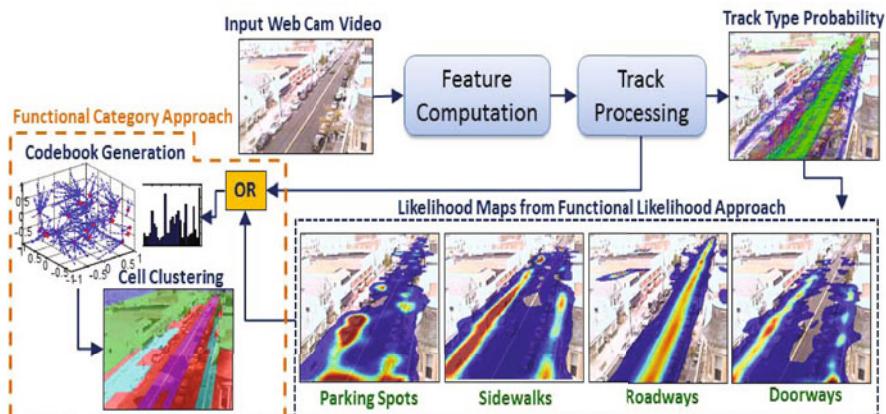
The following is a list of terms that warrant further attention:

- **Track-Type:** Label associated with track, or a portion of it, i.e. pedestrian, vehicle, parking-vehicle, or “other” (clutter) track
- **Functional element:** A location or a static object that is primarily defined by its specific function or purpose, rather than its appearance or shape.
- **Functional likelihood map:** The kernel density for each element, which reveals the most likely spatial regions for each functional element.

- **Function likelihood feature-space:** A set of likelihoods obtained by testing each track's detection against each functional likelihood map.
- **Hierarchical feature-space:** A descriptive set of behavioral features that provide robustness to tracking difficulties and capture multiple levels of behavior details. These include detection, track, and cell-level features that are accumulated on a per grid cell basis.
- **Functional category:** A cluster representing a mixture of behaviors from functional elements or codeword histograms. (Spatially agnostic learned scene model that is transferable to other scenes).
- **Functional region:** Grounding of functional categories to spatial grid cells through a classification step, i.e. most likely assignment.
- **Functional activity:** Pedestrian or vehicle activities that interact with functional elements, i.e. person entering/exiting building interacts with the doorway functional element.

## 2 Overall Approach

The overall approach for the Functional Likelihood and Functional Category methods are summarized in Fig. 3. First, the input video is calibrated to the ground plane where tracking is performed and where kinematic features are less dependent on being in the near or far field of the image. Moving objects are detected and formed into tracks using standard algorithms, and several feature types are computed. Several track processing steps are performed in order to reduce the number of false tracks and to increase the quality of the computed tracks.



**Fig. 3** Overall approaches for both functional element recognition approaches. The OR block in the flow chart indicates that the Functional Category approach can also use features from the Functional Likelihood approach

Both approaches discussed here use these steps to produce tracks and features for their higher level algorithms. However, the Functional Likelihood approach requires an additional step to determine the probability that the track's detections are

a person, vehicle, parking-vehicle, or “other” track-type. This probability is used to weight the contribution of a track’s detection kernel on the density estimate for a particular functional activity detector. The resulting functional likelihood maps have higher intensity regions where the functional element of interest is more likely to exist.

The Functional Category approach produces categorized functional regions, where the colors are used to represent different functional categories as shown in the Fig. 3 cell clustering image. This approach can also use the outputs of the Functional Likelihood approach as input features, as indicated by the OR box in the flow chart. Further details on both approaches are discussed in Sect. 3 and 4.

Both approaches require a minimum of about half hours worth of detection or tracking data for busy scenes, 45 minutes to an hour for moderately busy scenes and several hours to several days for scenes with very little activity. Its important to note that the activity is usually not equally distributed throughout all parts of the scene. For example, some parking-spots or roads may have very high levels of traffic, while others have very little. So, the amount of data required depends on the users preferences for the amount of scene coverage.

These approaches are currently offline algorithms, but future work plans to extend these to learn functional regions incrementally. To make the Functional Likelihood approach an online algorithm, the functional likelihood maps would need to either be continuously updated or updated after a certain amount of data is received, which does not require reprocessing of the entire dataset. The functional likelihood features from the new tracks can then be assigned to their most likely functional category, where high entropy categories can be further segmented. The updated set of categories can then be used to classify the functional regions with updated functional category labels. The Functional category approach can be extended in a similar manner, where the codebook histograms can be incrementally updated and used to update the functional categories and then the functional regions.

### 3 Functional Likelihood Approach

The Functional Likelihood approach utilizes tracks that are assigned track-type probabilities to accumulate evidence for four functional activities, which are then used to produce functional likelihood maps for four functional elements. The Functional elements are: walkway, doorway, roadway, and parking-spot; and are recognized by detecting the functional activities that interact with them: person walking on sidewalk, person entering/exiting a building, vehicle driving on roadway, and vehicle entering/exiting parking-spot. The following section first discusses several track processing steps for improving the quality of the tracks and then the details of the functional activity detectors and functional likelihood maps.

### 3.1 Track Processing

Raw tracks are computed by associating moving object detections across frames to a single object. The association is done using a kinematic model and a cost matrix, which is evaluated with the Hungarian algorithm to determine the optimal detection-to-track associations [14]. There was no appearance based modeling used in this tracker. Considering the complexity and low frame rate of webcam surveillance data, the tracking results are reasonable, but far from perfect. Therefore, online track processing steps are performed to obtain longer consistent tracks and to reduce the number of false alarms. Both improvements will lead to higher track quality for ingest into higher level algorithms.

A simple step for reducing false alarm tracks is to screen out any tracks that have less than a certain number of detections; we retained tracks if they have more than eight detections. Additional techniques, such as removing spurious tracks, breaking tracks at the point of corruption, and linking fragmented tracks are implemented and will either reduce the number of false tracks or improve their quality.

Spurious tracks are false tracks that are generated by significant global changes in the scene’s appearance which can be caused by sensor noise or natural events (e.g. cloud movement which creates moving shadows across the scene). Since spurious tracks tend to occur in large numbers on the same frames, all tracks that start on a frame that has a large spike in moving object detection are simply removed.

Tracks become corrupted when “other” detections appear within their correlation gate and are chosen over the existing valid detection. This “other” detection could be a false detection on the corner of a building, a shadow, or simply a nearby pedestrian or vehicle detection. Detecting track miscorrelations can result in the removal of valid track detections, but the benefit greatly outweighs the risk. Miserrelated detections are identified and removed by calculating the average Euclidian distance,  $\sigma$ , between adjacent position detections of a track and then breaking it at points that are greater than some distance threshold,  $\sigma = \mu + n^* \sigma$ , where  $n$  the number of standard deviations,  $\sigma$ . Breaking corrupted tracks will result in more consistent tracks that are shorter and a higher quality, which can then be recombined using track linking.

Track linking or merging is the process of combining tracks that are continuations of each other into a single track. Track linking significantly improves track quality, especially if all merged measurements are filtered as one long track. Track linking is more beneficial when applied online, as the new improved track can be used to advance the accuracy of the next detection-to-track correlation. Our approach utilizes a simplified version of the Perera et al. [14] algorithm; which shows how to efficiently handle the complexities that can occur during the track linking process.

### 3.2 Track-Type Interfacing

The track-type refers to whether the track, or a portion of it, is generated from a pedestrian, vehicle, parking-vehicle, or “other” (clutter) track. This information is extremely beneficial when detecting activities specific to either a pedestrian or a

vehicle. For example, only tracks with at least a 20% probability of belonging to the track-type of interest are processed through the weak functional activity detectors. The track-type probabilities are determined by testing the track's features against predefined models for each class, and normalizing the resulting likelihoods. Track features and track-type probability calculations are discussed below.

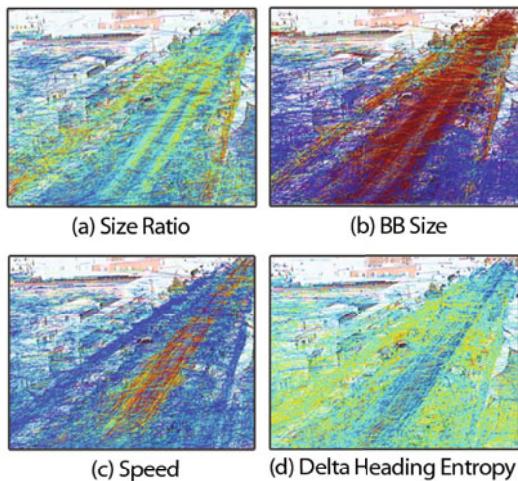
### 3.2.1 Feature Extraction

Any available features can be used to define track-type models. Currently, two track and two high-level image based features are used. One of the track-based features is speed,  $s_k = |\vec{v}_k|$ , where the velocity estimate,  $\vec{v}_k = (v_x, v_y)$ , is output from a Kalman filter on track sample number  $k$ . The other track-based feature is the entropy of the change in heading,  $\epsilon$ , over a sliding window of length,  $L$ :

$$\epsilon(k) = \sum_{i=k-L+1}^k \hat{C}(i,k) \log(\hat{C}(i,k)) \quad (1)$$

where  $\hat{C}(i,k) = C(\Delta_i) / \sum_{j=k-L+1}^k C(\Delta_j)$ ;  $|\theta_i - \theta_{i-1}|$ ,  $\theta = \tan^{-1}(\frac{v_y}{v_x})$ , and  $C(\Delta_i)$  reflects the histogram counts in a particular angular difference bin.

The image pixel features include the track's Bounding Box (BB) size in pixels and its width-to-height ratio. Each track-type has a “known” range of parameters for each feature. Feature ranges can be automatically determined using Support Vector Machines (SVM) or Nonlinear Discriminative Analysis (NDA). However, they are manually defined here by looking at feature heat maps of all the training tracks that are color-coded according to their value, Fig. 4 for the Ocean City data. This has the benefit of not requiring annotated track-types, which are needed by SVM or NDA approaches. Tracks that have larger speeds are darker red and lower speeds are light blue. It is clear from Fig. 4 that the BB size ratio, BB size, speed and entropy can be used to discriminative between pedestrians, vehicles and “other” tracks. The vehiclparking activity detector can recognize parking vehicles by combining these features into the 4D feature space. Features parameters for the four track-types models are derived from these



**Fig. 4** Heat maps of four different track and image features were used to define the parameter ranges for the four different track-types: person, vehicle, vehicle-parking, and “other” 

plots based on their observed maximum and minimum feature values. The data is tested against these models and their probability output is normalized to produce a-posterior probabilities with uniform priors, as discussed in the following section.

### 3.2.2 Probability Calculations

The four track-types form a complete mutually exclusive set, so the sum of their probabilities is equal to one. To simplify notation the track-types will be assigned numbers: 1= pedestrian, 2= vehicle, 3= vehicle-parking and 4= other. The model,  $\lambda_i$ , for each track-type,  $i$ , are formed by assuming that the minimum and maximum values of the feature's range, found in Sect. 3.2.1, are both Gaussian distributed with a mean,  $\vec{\mu}_i$ , and standard deviation,  $\vec{\sigma}_i$ , where  $i = 1, 2, 3, 4$ . The parameters of these distributions can be learned automatically, but this requires manually annotating a large number of track-types, so the heat maps from Fig. 4 were used to manually define them here. These result in a bandpass cumulative distribution function (cdf) that is used to calculate the likelihood of a detection falling within the specified range and thus belong to this model.

Testing a detection,  $\vec{\sigma}_k$ , from track sample number  $k$  against model  $i$  produces a likelihood that the detection was generated by that model,  $P(\vec{\mu}_i^{\min} \leq \vec{\sigma}_k \leq \vec{\mu}_i^{\max} | \lambda_i)$ , which uses the model's bandpass cdf for its calculations. It is assumed that all detections assigned to a given track belong to the same object. The object is also assumed to be of a certain track-type for a period of time. These two assumptions allow us to smooth the probabilities over a window eliminating minor fluctuations in the probabilities. The probabilities are smoothed by taking the geometric mean, over a window of length  $L_2$ , as shown below.

$$P(O_k | \lambda_i) = \prod_{j=k-L_2+1}^k P(\vec{\mu}_i^{\min} \leq \vec{\sigma}_k \leq \vec{\mu}_i^{\max} | \lambda_i)^{\frac{1}{L_2}} \quad (2)$$

for all track-types,  $i = 1, 2, 3, 4$ , and each track sample,  $k$ . The probabilities are then normalized so they sum to one, resulting in the a-posterior probabilities:

$$P(\lambda_i | O_k) = \frac{P(O_k | \lambda_i)}{\sum_j P(O_k | \lambda_j)}; i = 1, 2, 3, 4 \quad (3)$$

These posterior probabilities are then input to the functional activity detectors to generate likelihood maps of the functional regions, Sect. 3.3. Fig. 5 shows the four track-type probabilities for the Ocean City data as heat maps, where high probabilities are darker red and low probabilities are darker blue. Notice that a large portion of the tracks from the 20 hours of video are clutter tracks with a high probability of being an “other” track-type, and are not shown in Figs. 5a and 5c. Also, the parking-spot regions are still not separate parking-spots; however, using the flashlight model with the beam kernel from 11 does provide some separation of the parking-spots, Fig. 8c. Details on the activity detectors are covered in the next section.

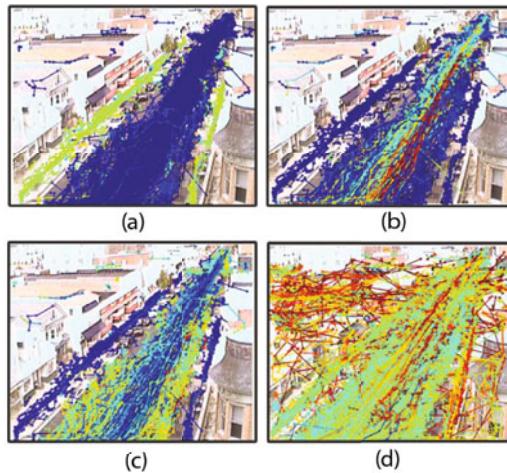
### 3.3 Functional Element Regions

The spatial likelihood maps for each functional element are created in an online manner by spreading spatial kernels over a relatively high resolution 2D grid on the scene’s ground plane,  $\vec{G}_{xv} = (g_x, g_v)$ . Higher likelihoods are produced in regions with consistently high functional activity over time. A functional activity detector can be created for any activity that has features which support its detection and are related to the desired functional elements. For the approaches discussed here they are: person-entering/exiting-building, person-walking-on-sidewalk, vehicle-driving-on-road, and vehicle-entering/exiting-parking-spot. Tracks are tested against these four functional activity detectors where each functional activity detector is made up of a simple set of conditions specific to the activity of interest.

The four functional activities are separated into two sets of dual detectors: low entropy activities such as walking or driving; and high entropy activities such as entering/exiting a doorway or parking-spot. The latter is more difficult than walkways or driveways due to the high level of fragmented tracks. Therefore, we use the flashlight beam kernel [1] to accumulate evidence at entry and exit focal points.

#### 3.3.1 Doorway and Parking-Spot Detector (Flashlight Model)

The flashlight model is based on a kernel in the shape of a flashlight beam,  $K_{beam}$ , projected onto the ground plane as if it were coming from a person holding a flashlight. The beam is intended to “light up” the doorways of buildings by accumulating this weak evidence over time. However, it is also used to emphasize the parking-spots, since the vehicles also enter and exit a functional region. Fig. 6 shows several of these beam kernels spread over the Ocean City scene after a dozen tracks have been created for the doorway detector. The only difference between the doorway and parking-spot detectors is that the doorway kernel is weighted by the probability of the track’s detection being a person and the parking-spot by the probability of being a vehicle. See Szwars et al. [1] for implementation details on the Flashlight Model.



**Fig. 5** Track-type probabilities for 20 hours of Ocean City data **(a)** Probability of pedestrian walking **(b)** Probability of vehicle driving **(c)** Probability of vehicle parking **(d)** Probability of “other”

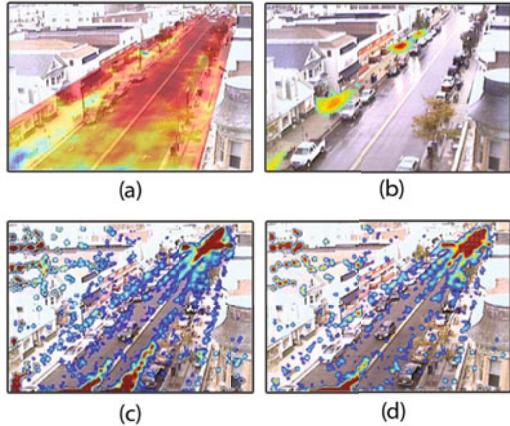


**Fig. 6** Multiple beam kernels from the Flashlight model after 12 tracks; notice the large number of false alarms created by the high level of track fragmentation, even after track processing [11]

Figure 7 shows the resulting functional likelihood map for the doorway detector (a) and the segmented functional doorway regions (b). Figure 7c and (d) show results of comparative approaches where spatial kernels are accumulated into likelihood maps, without the Flashlight model, using sources Fig. 7c and sinks Fig. 7d and then thresholded. All track-types were used in the source and sink likelihood maps, but even if only pedestrian tracks were considered, the regions in the pathway and the doorways are inseparable. Notice there are a very large number of false sources and sinks compared to the Flashlight model.

### 3.3.2 Walkway/Roadway Detectors

Implementation of the walkway and roadway detectors is discussed in Swears et al. [11]. These determine the usual travel path for pedestrians and vehicles. They both have low entropy and larger speed features compared to their counterparts (entering building and parking). Fortunately, these kinematic features have already been incorporated into the track-type probability calculations, Sect. 3.2.2, i.e. the detectors only need to accumulate weighted evidence using the position kernel not the beam kernel, where the roadway has larger kernel spread parameters.



**Fig. 7** (a) The likelihood map for the Building Doorway Detector from one day of OC data; dark blue indicates higher likelihoods and dark red lower (b) doorway regions detected through segmentation of the likelihood via thresholding (c) A likelihood map of track sources without the Flashlight model (d) Likelihood map for the track sinks without the Flashlight model

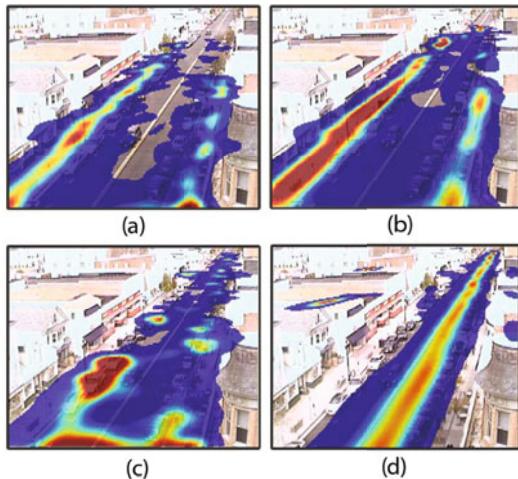
The resulting likelihood maps for all four functional elements are shown in Fig. 8 for high resolution grid cells and are considered the kernel distributions of the four models. Regions with higher likelihoods, dark red, indicate regions that have evidence that fits well with the expected behavior for that functional element. Notice some functional element regions are missed due to low levels of activity.

## 4 Functional Category Approach

This second approach to functional element recognition is inspired by the bag-of-words paradigm for object recognition and is referred to as the Functional Category Approach. It identifies regions with similar behaviors in the same scene and/or across scenes by clustering histograms based on a trajectory-level, behavioral code-book. The regions do not need to be spatially contiguous; in fact, some are expected to be spatially disjoint while having similar functional properties. A cluster of such regions (in feature space, not scene coordinates) corresponds to a functional category that can be assigned a conceptual label.

The track processing and feature extraction steps from the Functional Likelihood method, Sect. 3.1 are also used here. However, there is an additional feature extraction step that creates a common hierarchical feature-space representation for all behavioral scene element categories. At a high level, the remaining steps are: 1) unsupervised learning of behavioral category models that are independent of scene location and transportable to other similar surveillance scenes [2]; and 2) segmenting video scenes into functional categories. This method can represent a variety of multi-modal scene categories that are common in complex outdoor activities.

Both this approach and the Functional Likelihood based approach use grid cells to accumulate evidence. However, this approach uses a much lower resolution for the spatial grid, due to its hierarchical features-space and use of grid cell histograms. Replacing the hierarchical feature-space with the functional likelihood feature-space that is output from the first approach significantly reduces sensitivity to grid cell size, while also improving the recognition results.



**Fig. 8** (a) The likelihood map for the walkway functional element from twenty hours of Ocean city data; dark red indicates higher likelihoods and dark blue lower likelihoods (b) Likelihood map for roadway element (c) Likelihood map of parking-spots using flashlight mode (d) Likelihood map of doorways using flashlight model

**Table 1** Detection, Track, Cell-level Features

Element Feature		Type
$d_{j,k}$	0	Median speed
	1	Median change in Speed
	2	Median change in angle
	3	Median size
	4	Median detection aspect ratio
$f_{j,k}$	0	Track length
	0	Number of track starts in cell
	1	Number of track stops in cell
	2	Number of tracks through cell
	3	Entropy of outgoing heading distribution
	4	Entropy of incoming heading distribution

Detection level features,  $d_{j,k}$ , for track j in cell k; track level features,  $f_{j,k}$ , for track j in cell k; and cell level features for cell k

The following subsections discuss the hierarchical feature-space, behavioral codebook generation, as well as cell histogram clustering. Results using this approach are shown in Fig. 15b, Sect. 6.1

#### 4.1 Hierarchical Feature-Space Computation

The hierarchical feature-space representation is a descriptive set of behavioral features that provide robustness to tracking difficulties and capture multiple levels of behavior details. These include detection, track, and cell-level features that are accumulated on a per grid cell basis. The cell-level features capture the relationship between cells traversed by the same track, along with localized relationships between tracks over time. The feature set characterizes local behaviors the same way patch descriptors characterize local appearance for object recognition.

The detection, track, and cell-level features are listed in Table I. The top block in the table contains the detection-based features, the middle block contains track-level features, and the bottom block contains cell-level features. Detection-level features incorporate information based solely on moving object detections that are within a particular cell. This feature vector  $d_{j,k}$ , is a statistical summary (median) of the feature types for each track over the detections that are assigned to the cell. The listed detection based features were chosen because they are the main attributes for discriminating between vehicles and pedestrians.

Track-level features,  $f_{j,k}$ , are also computed for each track that passes through a cell, where the tracks length is meant to differentiate the roadways more due to their longer durations. Finally, cell-level features  $c_k$  are computed across all tracks that pass through a cell. These were chosen to aid in discriminating between parking vehicles and roadways as well doorways and walkways. Automatic features selection algorithms can be used here, but it requires annotating large amounts of data, so there is still some level of user specification.

The entropy of the heading distribution for incoming/outgoing tracks is computed as in Equation (1.4), with distribution taken over the heading of all start/stop detections in a cell. Note the entropy measure is independent of particular orientations, allowing the feature to be built into a codebook on one scene and applied on another.

$$E(\tau) = -\sum_i p(\theta_i) \log(p(\theta_i)) \quad (4)$$

where  $p(\theta_i)$  is the heading probability of the  $i^{th}$  detection on track  $\tau$ . The histogram of the headings along a track is used to model heading probability.

It is important to choose features judiciously, as additional non-informative features can clutter the feature space and effectively add a noise term to feature distances which leads to misclassifications. An ensemble feature vector  $x_{j,k}$  for each track  $j$  and cell  $k$  is represented as:

$$x_{j,k} = [d_{j,k} \ f_{j,k} \ w_c c_k]^T \quad (5)$$

where  $w_c$  is a weighting for cell features  $c_k$ . The collection of all feature vectors  $F$  in a video sequence is:

$$F = \{x_{0,0}, \dots, x_{j,k}, \dots, x_{n,m}\} \quad (6)$$

for  $n$  tracks and  $m$  cells. An issue arises in simultaneously handling detection, track, and cell-level features as there are more track level feature instances (one per track crossing a cell) than cell level features (one per cell). This discrepancy is handled by down weighting cell features influence through  $w_c$ . This allows a single codebook to be built on a feature space that includes both track and cell feature dimensions. Currently  $w_c = 1/n_{celltracks}$  is used where  $n_{celltracks}$  is the number of tracks per cell.

## 4.2 Behavioral Codebook Generation

The use of codebooks or “visual words” has become very popular in visual recognition tasks, as they effectively and compactly represent high-dimensional, complex feature spaces. Visual words are used here because the behavioral feature space can be complex, but should also contain high-density areas corresponding to common activities. Stauffer and Grimson previously applied a codebook to activity analysis [6], where position was included and tracks were represented by sets of centroid labels. Here, dependence on absolute features such as position and heading are explicitly avoided, since the goal is to learn behavioral categories independent of locations or location-specific behaviors. Also, the codebook histogram is defined on cells, not tracks, which enables use of the hierarchical feature space.

All ensemble feature vectors in the video scenes,  $F$ , are used to create the codebook. Before clustering, each feature is normalized by its standard deviation across the observed data. The codebook is then generated by clustering the ensemble features. Experiments were conducted with both K-means and mean-shift [15] and it was determined that mean-shift is considerably more stable than K-means with

random initialization. Mean-shift also has the advantage that K does not need to be specified, although the bandwidth parameter does impact performance. The mean-shift for feature  $x_i$  with bandwidth parameter  $h$  and kernel  $g()$  is defined in [15]:

$$m_{h,G}(x) = \frac{\sum_{i=1}^n x_i g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)}{\sum_{i=1}^n g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)} \quad (7)$$

Two iterations of mean-shift are performed on the set of features  $F$  to generate representative features, which was experimentally determined to produce sufficient results. An Epanechnikov kernel is used to represent the underlying distribution, yielding a mean-shift implementation with a uniform kernel. The Epanechnikov kernel was chosen due to its optimality properties and because the bandwidth parameters fully define its range without the needing truncation as with Gaussian kernels. The bandwidth for the Fig. 2c result was determined to be 2.0 for codebook generation through experimental cross-validation over a small range of values. Using this spread parameter in the mean-shift algorithm resulted in 80 codewords, which are used in the cell histogram and clustering approach discussed in the next section.

### 4.3 Cell Histogram Representation and Clustering

Once the codebook is defined, a histogram is created for each cell that treats each codeword as a bin. The closest codebook centroid is identified for each integrated feature vector in a cell and the corresponding histogram bin is incremented. To assure statistical meaning, cells with too few feature vectors, < 20, are ignored.

The final segmentation step involves clustering cells with similar histograms. Mean-shift is used for this clustering, as the number of clusters is not known a priori, and it is expected that the clusters are non-Gaussian and generally noisy. Spatial cell position is not considered, as it is desirable to have cells that can be grouped with similar behaviors that are located in different parts of the same scene or in different scenes. Cell neighborhood information is not currently used in the clustering process, although some features are computed across neighboring cells.

Ideally, each of the resulting clusters would correspond to a single functional category such as road or parking spot, and each functional category would have one cluster. In practice, any of the categories may have a multi-modal feature distribution, in which case mean-shift will create multiple modes for one category. Figure 2c shows the four functional categories that are generated from clustering the cell's histograms with mean-shift and a spread parameter of 0.22.

Generally, it is expected that many scene elements will have non-trivial distributions of behavioral features. For simple categories, such as vehicular or pedestrian traffic lanes, it is simple to compute low-level features on individual tracks, and perform clustering on raw features. In these cases, the use of cell-level features and codebook histogram clustering may be more powerful (and complex) than required.

Many other scene categories have more complex and variable behaviors. Building entrances and exits, parking spots and crosswalks, for example, may exhibit a variety of behaviors, even at the same locations. This can be represented with the

cell clustering approach, as long as the mode patterns are similar across the class. For one activity pattern in a multi-modal cell, the tracks will give rise to feature vectors in histogram bin  $h_i$ . For a second activity pattern in the same cell, the tracks will give rise to feature vectors in histogram bin  $h_j$ . In this case, the cell histogram will have two modes and should be clustered with other cells that have the same two modes. This situation should arise for a cell outside a doorway, for example, if some people walk straight past the doorway (bin  $h_i$ ) and others enter (bin  $h_j$ ).

#### **4.4 Functional Element Recognition**

Once the clusters are formed, functional element recognition can be performed on a new video scene, or new parts of an existing scene. The spatial scale and partitioning of the new scene should be comparable to those used in training, and the scene should have comparable behaviors for the same scene categories (this may not be true across different regions of the world).

Each cell is recognized independently. Cell features are computed from its tracks as in training, and the cell histogram is created using the prior codebook. Each cluster is a collection of codebook histograms, and cells are labeled using mean-shift on the cell histogram, and outputting the cluster found by mean-shift.

### **5 Business Intelligence Analysis**

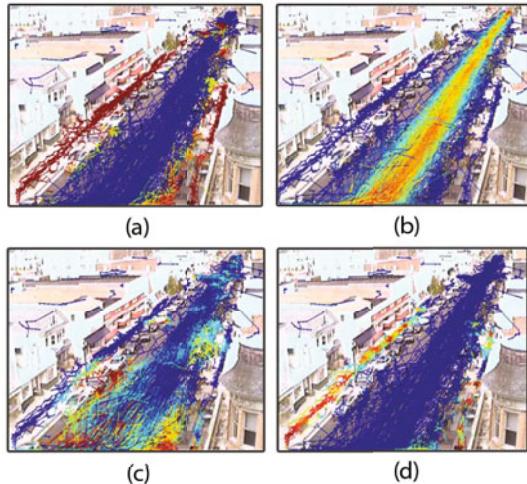
The Functional Likelihood approach, (see Sect. 3) does not directly produce functional regions; rather, it produces a functional likelihood map for each functional element. However, functional regions can be obtained by thresholding or clustering features derived from the likelihood maps (functional likelihood features) followed by functional category assignment. Functional regions produced by the direct clustering method are the source of the activity profiles used for the business intelligence analysis, as they are descriptive and semantically meaningful. Activity profiles are accumulations of the number of moving objects assigned to a particular functional region over time and are used to accumulate the business related statistics.

An alternative to directly clustering the functional likelihood features is to use them as features in the Functional Category approach, Sect. 4 discussed further in Sect. 6. The following sections discuss the functional likelihood feature-space, functional region recognition, activity profile extraction, and potential customer analysis.

#### **5.1 Functional Likelihood Feature-Space**

Functional likelihood maps resulting from the Functional Likelihood approach are essentially kernel distributions that capture the more likely functional element regions and collectively are the functional element models. As models, track detections can be tested against them through their kernel distributions, determining the likelihood that the detections fit the models. The resulting set of likelihoods from all track detections are referred to as the functional likelihood feature-space. This replaces the hierarchical feature-set in the Functional Category approach when

merging the two approaches. Figure 9 shows likelihood results when testing against each functional element model. The resulting feature vector has four independent likelihoods, one per model.

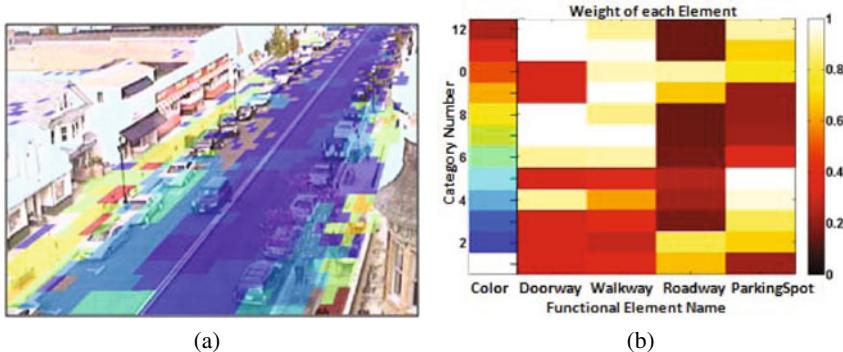


**Fig. 9** (a) Likelihoods when testing detections against walkway model in Fig. 8; dark red indicates higher likelihoods and dark blue lower likelihoods (b) Detection likelihoods relative to roadway model (c) Detection likelihoods relative to parking-spots model (d) Detection likelihoods relative to doorway model

## 5.2 Functional Region Recognition

After the functional likelihood feature-space is created, it is passed into a hierarchical divisive clustering algorithm to produce a behavior codebook similar to that used in the Functional Category approach, except that no grid cells are used to form the codebook and the functional likelihood feature-space is used instead of the hierarchical feature-set. The clusters or codewords are now the functional categories; therefore, all that remains is to assign the category labels to spatial locations. To this end, the track's likelihood features are assigned to the most likely codeword on a per detection basis. Then, a spatial grid is imposed on the ground plane of the scene, and track detections along with their category labels are assigned to their nearest cell. The category labels are accumulated into cell histograms on a per grid cell basis, where each histogram bin is a category label, as with the Functional Category approach. The grid cell locations, functional regions, are then assigned labels of the most frequently occurring category label.

Figure 10a shows the functional categories that are formed after directly clustering the features in Fig. 9 using a hierarchical divisive clustering algorithm. Notice the cell size is larger,  $1.7 \times 0.66$  m, than those in the functional likelihood maps,  $0.5 \times 0.5$  m, but smaller than those from the Functional Category approach in Fig. 2c,  $10.2 \times 2.30$  m. These cell sizes were manually determined by iterating over a coarse range of values for each. The cell size for the likelihood maps needs to be smaller than the cells size used in the functional approaches and was chosen to be relatively small in order to capture more fluctuations in the features. The cell size for the Functional Likelihood approach, with direct clustering, was chosen to be smaller than the size of the smallest desired functional region, i.e. doorway, to allow for



**Fig. 10** (a) Functional element categories formed from directly clustering the detection likelihood's in Fig. 9 overlaid on Ocean City background image (b) Clusters/categories in ground plane coordinates with Id legend

slight over-segmentation. The exact cell size used was determined by experimenting with a small range of values. The cell size for the Functional Category approach has to be larger since the features being clustered are derived directly from the cells instead of each of the track's detections.

The Kmeans and meanshift clustering algorithms were used as well, but the divisive clustering approach segments the doorways and other functional categories more accurately with these continuous features.

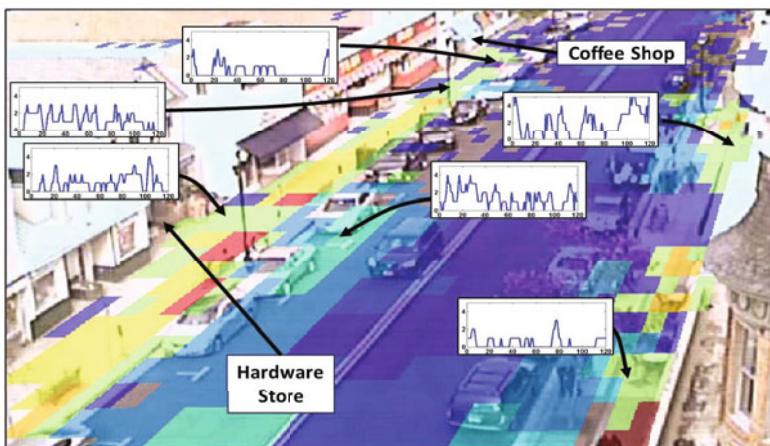
Notice the sidewalks, roadways, and parking spots are segmented into multiple modes, indicating that there are functional regions that are a mixture of functional elements, e.g. high likelihood of being a walkway and an entrance, or a parking-spot and roadway (pulling-out of parking spot). The labels for the functional categories in Fig. 10a are defined using the colors in the left column of Fig. 10b. This figure also shows how a category's  $1 \times 4$  prototype vector can be represented by functional element weights. These weights are determined by comparing the category's prototype vector to its corresponding functional likelihood features from Fig. 9. Note several of the categories have multiple modes indicating that they are not mutually exclusive. For example, categories 7, 8, 11, and 12 have very strong contributions of the doorway as well as the walkway. These modes are differentiated by the varying levels of the likelihoods, Fig. 9, for each element and their combinations. The category labels and their distinct modes are important for extracting activity profiles that can potentially be used for causal analysis. The labels can be used to locate particular regions of interest, i.e. doorways, while the distinct modalities are key to creating unique characteristics for relating regions to, for example, the doorway of a particular building. The functional categories in Fig. 10 also represent the types of functional regions that are produced from the Functional Category approach.

### 5.3 Extracting Activity Profiles

The scene's spatial grid cells are each classified as one of the functional categories as discussed in Sect. 5.2. The activity profiles could be extracted from each of these grid cells; however, this results in sparse profiles that may also be intractable for further processing in higher level algorithms. Therefore, functional regions that are connected and have the same functional categories are merged into one region. Minor fluctuations in the spatial boundary of each merged region are removed by spatially smoothing the labels with a Gaussian kernel and then thresholding the result. All grid cells above the threshold are assigned to the merged functional regions and the activity profile is extracted from this larger region.

Recall that the activity profile is a count of moving objects assigned to functional regions on a given frame number over time. Figure 11 shows the functional regions from Fig. 10, and activity profiles for several detected doorways and a parking-spot over 120 time steps (x-axis) from zero to four (y-axis).

The activity profiles in Fig. 11 are generated by taking the tracks that result from 20 hours of Ocean City video and segmenting them into 120 non-overlapping temporal windows, that accumulate the counts on each frame for all windows. This provides a visual representation of the relative density of activity for the merged functional regions. Notice how some of the functional regions have consistently high activity and others are sparse. This type of temporal information can be used in higher level algorithms for discriminating between different types of the same functional categories (e.g. doorway/building types).

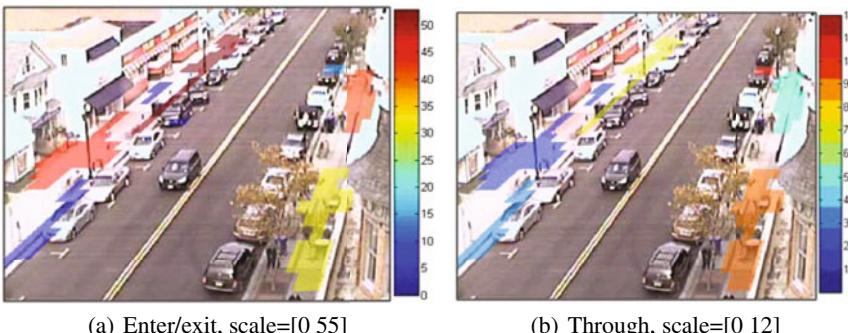


**Fig. 11** Overlaid functional element regions where each color is a functional cluster, e.g. doorway or parking spot; accumulated activity profiles are also shown for doorways (green) and a parking-spot region (blue-green)

## 5.4 Potential Customer Analysis

Business owners can gain a better understanding of their business by studying activity near their storefront and by seeing how their business relates to others nearby. For example, how many pedestrians walk past instead of enter, do parking-spots in front of the business have a high turnover rate, and does the business of interest have more or less activity than nearby stores. These questions can be answered by analyzing the activity in the automatically detected functional regions related to a particular business, e.g. functional doorway. A year-to-year analysis of these activities can also be performed to understand trends and impacts of changes to the scene, e.g. new metered parking or a store opening or closing nearby.

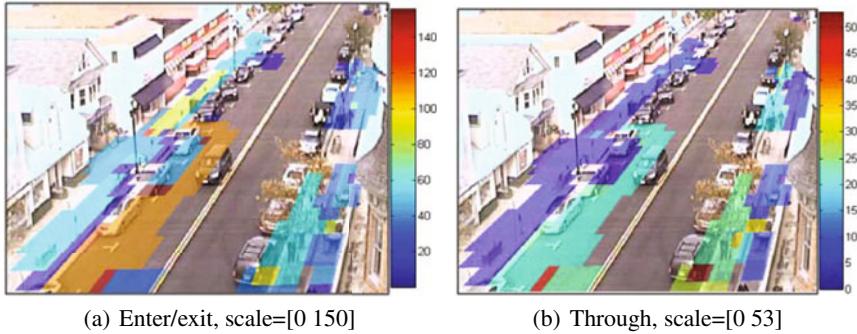
Numbers of pedestrians that enter vs. walk past is determined by automatically counting tracks that start or end in its functional doorway region and compare that to the number that enter the region but continue through it. The results of this comparison for the Ocean City data, are shown in Fig. 12 for the detected doorway regions.



**Fig. 12** Doorway, (a) All merged functional doorway regions, color coded according to the number of pedestrians that enter/exit (b) Doorways color coded according to number of pedestrians that pass through; the legends for each image show their corresponding color-to-count conversion, notice the different count scales

Figure 12a shows the number of pedestrians that enter/exit the doorway regions as a heat map, where darker red indicates higher activity and darker blue is lower activity. Figure 12b shows the activity heat map for the number of pedestrians that pass-through the doorway regions. Notice, the detected doorway regions are not perfect; some doorways that are separate regions in Fig. 10 are merged here due to the spatial smoothing step. Several conclusions can still be made from the doorways that are detected; for example, businesses near the coffee shop (Fig. 11) have more customers than all other businesses, but they also have a high number of costumers that pass through their doorway regions (“lost” customers). Alternatively, the hardware store (Fig. 11) also has a high level of activity, but a low number of “lost” customers.

Figure 13 can also be used to analyze the walkway and parking-spot functional element regions. The image in Fig. 13a shows that the parking-spots in front of the



**Fig. 13** Walkway, (a) All merged functional regions that have a high concentration of walkway or parking-spot elements; color coded according to the number of pedestrians that enter/exit (b) Same functional regions as (a) color coded according to the number of pedestrians that pass through

hardware store have a high level of enter/exit activity while those in front of other businesses are much lower. This can indicate that the vehicles are parked for longer periods of time, for the regions with lower activity, or that the level of activity is truly low. Alternatively, the higher activity parking region is most likely in front of a higher activity business. This is confirmed here by comparing the functional regions in Fig. 13a and seeing that the doorway and parking-spot in front of the Hardware store both have relatively higher levels of enter/exit activity. These types of associations can also be used to characterize a building type and to recognize others that are similar to it in an unlabeled set of functional regions.

Experiments on the Ocean City webcam video dataset [1] and the CAVIAR surveillance dataset [3] are discussed in Sect. 6 and 7, respectively.

## 6 Ocean City Experiments

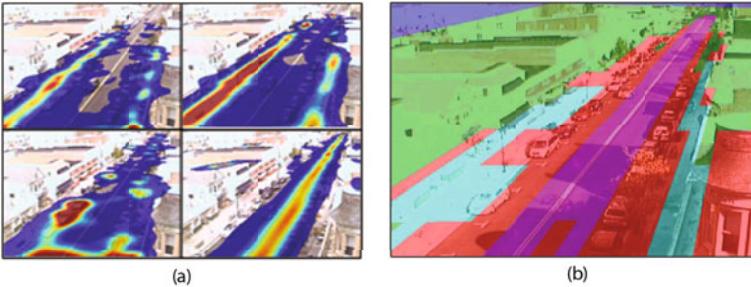
The majority of the Ocean City results were covered in Sect. 5. However, the results of the Functional Likelihood and Functional Category approaches are discussed further along with experiments that use the functional likelihood feature-space as inputs to the Functional Category approach.

### 6.1 Functional Likelihood and Functional Category Results

The Ocean city results from Fig. 2 for the Functional Likelihood approach, Sect. 3, and the Functional Category approach, Sect. 4, are repeated in Fig. 14 for convenience.

Notice the functional likelihood maps from the Functional Likelihood approach, Fig. 14a, have higher resolution that allows it to aid in differentiating between functional elements more easily than the lower resolution Functional Category approach. However, no functional regions with category labels are explicitly produced here as

they are with the Functional Category approach, Fig. 14b. Unfortunately, the grid cell size needs to be larger for the Functional Category approach in order to accurately create the behavior codebook using cells. As mentioned in the previous sections, this can be mitigated by using the functional likelihood features-space instead of its standard hierarchical feature-space, as shown in the Sect. 6.2.



**Fig. 14** (a) Likelihood map results from Functional Likelihood approach, shown as heat maps; starting in the top left and going clockwise the functional elements are: doorway, walkway, roadway, and parking-spot; darker red indicates more likely regions, while darker blue is less likely (b) Results of Functional Category approach; regions with the same color are the same functional category [2]

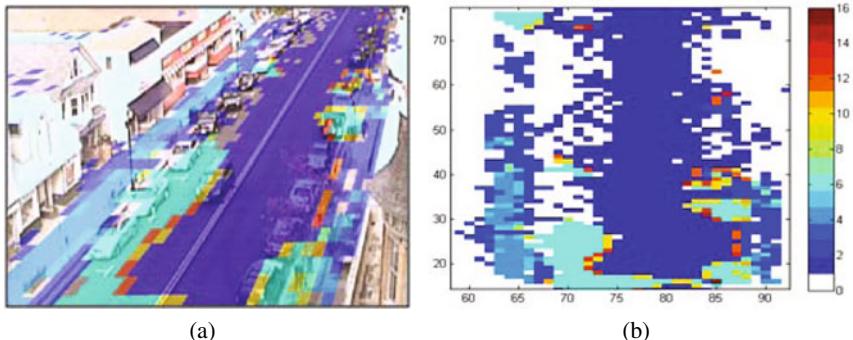
## 6.2 Functional Category Approach with Functional Likelihood Feature-Space

The functional likelihood feature-space can be used to replace the hierarchical feature-space in the Functional Category approach. This reduces its sensitivity to the grid cell size, since the contribution of the track features are already contained in the higher resolution functional likelihood features. Therefore, the grid cell size can now be reduced from the 10.2 x 2.3m in Fig. 14b to the 1.7 x 0.66m in Fig. 16, which is a 95.2% reduction in the cell's area. Figure. 15 shows that the higher resolution leads to more functional modes and a higher accuracy of the functional region recognition. For example, notice the doorways and their different modes are now detected to some degree, when they were completely missed in the Fig. 14b results. Similarly, the parking spots are now segmented.

These results are now comparable to the direct clustering method from Sect. 5.2 that performs clustering on the functional likelihood feature-space without using cells. The benefit of the cell based approach is not demonstrated with this data, but it is ideal for dimension reduction in the larger datasets that results from wide-area surveillance. Analogous results for the CAVIAR dataset are shown in Sect. 7.

## 7 CAVIAR Shopping Center Experiments

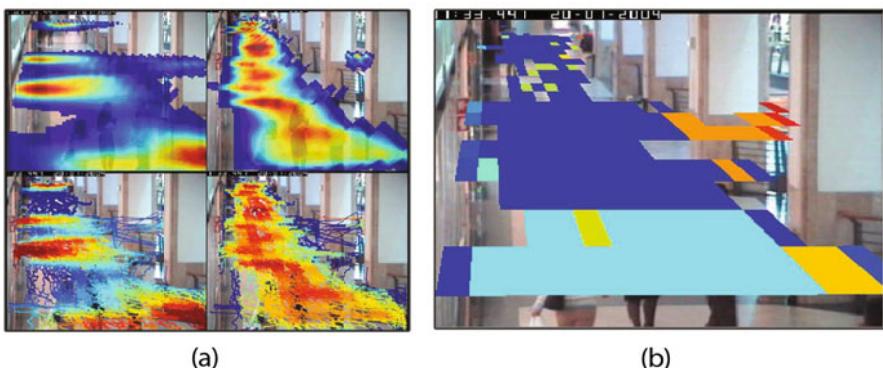
The Caviar dataset is much less complicated than the Ocean City data, as there are no vehicles and manual truth tracks are used instead of automatically computed



**Fig. 15** (a) Functional category results from using the likelihood feature-set in the Functional Category approach overlaid on Ocean City image; each color represents a different functional category (b) Functional Category results image with legend showing category IDs

tracks. This makes the functional region recognition results more accurate, which in turn should produce more accurate activity profiles for business analysis. Even though there are no vehicle elements, business related activities can still be analyzed using the various doorway and walkway functional categories. The following experiments cover the Functional Likelihood and Functional Category approaches using the CAVIAR dataset. Results are shown when the likelihood features-set is used in the Functional Category approach. Results are also shown for the direct clustering of the likelihood feature-set approach and the business analysis experiments.

### 7.1 Functional Likelihood and Functional Category Results



**Fig. 16** (a) Functional likelihood map results from each functional activity detector (top) along with their corresponding likelihood feature-set (bottom). The Left column of the subplots is from the doorway element and the right is from the walkway element; darker red indicates more likely regions, while darker blue is less likely (b) Results of unsupervised functional region recognition; regions with the same color are the same functional category

The analogous version of Fig. 8's functional likelihood maps and Fig. 9's functional likelihood feature-space are shown in Fig. 16a for the Caviar dataset. There are no vehicles in the shopping center data, so only two functional elements are considered: walkway and doorway. Functional regions resulting from using the hierarchical feature-set in the Functional Category approach are also shown in Fig. 16b. Grid cell resolution and mean-shift parameters were experimentally determined here.

The left side of the shopping center is lined with several stores and there is a perpendicular walkway intersecting the main walkway near the middle of the scene going to the right. Both ends of the main walkway also intersect walkways.

The Functional Likelihood approach, Fig. 16a (top-left), has high likelihoods for the two main store doorways and interprets the intersection of the top and bottom walkways with the main walkway as doorways. This is expected, as the tracks start and end at both locations and people enter and exit the scene from different directions. The Functional Category approach, Fig. 16b, recognizes the entrance to the main walkway at the bottom of the scene and the main walkway regions. However, the doorways and walkway are under-segmented in some regions making it difficult to characterize the scene. These results are improved by using the functional likelihood feature-space from Fig. 16a (bottom) as inputs to the Functional Category approach, Fig. 17.

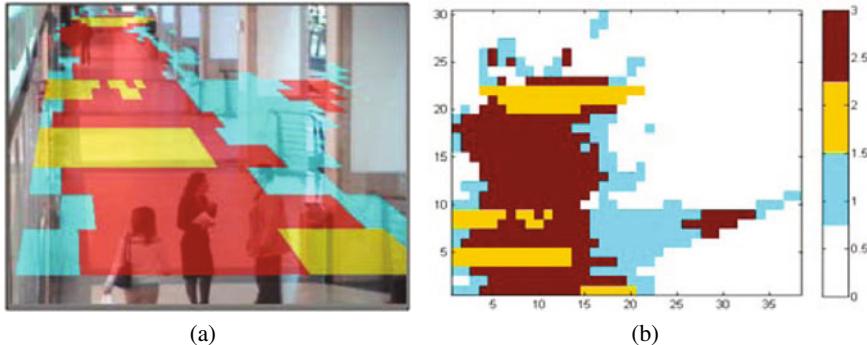
## 7.2 Functional Category Approach with Functional Likelihood Feature-Space

The analogous versions of the Ocean City's Functional Category approach with, Fig. 15, are shown in Fig. 17 for the CAVIAR dataset.

The results shown in Fig. 17 models the main walkway very well and even captures the main doorways well, which is ideal for recognizing contiguous functional elements. However, it is more challenging to characterize each element when there is only one category assigned to it, since all movers will be accumulated into a single profile which removes discriminative spatial context. Adjusting the mean-shift spread parameter does produce more modes here, but regions were over-segmented without clear relationships to the functional elements. The following approach where the likelihood feature-set is directly clustered, produced the more desirable number of categories and segmentation results, as shown in Fig. 18.

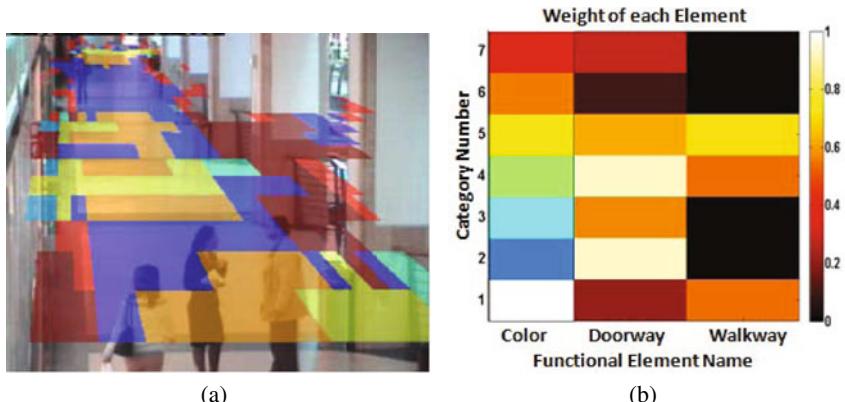
## 7.3 Direct Clustering of Functional Likelihood Feature-Space Results

The direct clustering approach used in Sect. 5.2 on the Ocean City data was applied here to the CAVIAR data, where the functional likelihood feature-space was directly clustered, without cells, using a divisive clustering algorithm. The number of clusters was set to seven, which is about three times the number of functional elements, thus allowing for multiple modalities.



**Fig. 17** (a) Functional category results from using the likelihood feature-set as inputs to the Functional Category approach overlaid on CAVIAR background image; each color represents a different functional category (b) Functional category results image with legend showing the category Ids

Figure 18a shows the resulting functional category regions overlaid on the CAVIAR background image, along with the corresponding functional categories in image coordinates and its legend in Fig. 18b.



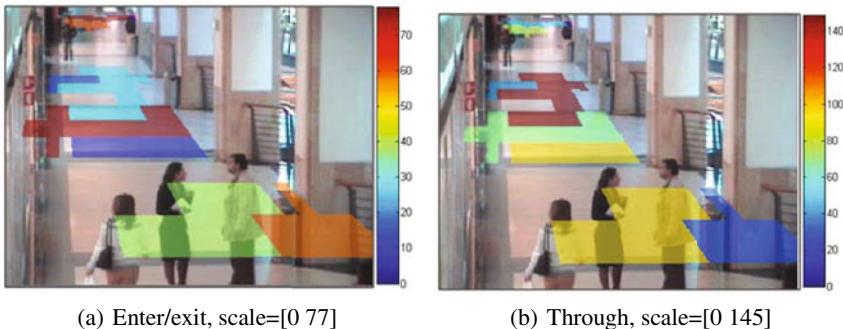
**Fig. 18** (a) Functional element categories formed directly from the likelihood feature-set in Fig. 16 (bottom) overlaid on CAVIAR background image (b) Clusters/categories in image coordinates with legend for category Ids

Notice the functional categories that results from this divisive clustering approach have multiple modes that are consistent throughout the scene and represent a mixture of the two functional elements. Figure 18b shows the functional category IDs (rows), their corresponding color (column 1), and the concentration of their functional elements (columns 2 and 3). This is analogous to Fig. 10b, where the whiter cells in columns two and three indicate a higher concentration of that functional element and darker red a lower concentration.

The yellowish-green functional category (Id=4), Fig. 18a, has a much higher concentration of the doorway element than walkway, as seen in Fig. 18b. However, another related category, orange (Id=5), has relatively high concentrations of both elements and Id=1 is mostly related to the walkway element. These various category regions can be used to characterize the activity of customers and potential customers as related to a particular storefront.

## 7.4 Potential Customer Analysis

The analysis performed in Sect. 5.4 on the Ocean City data is also performed on the CAVIAR data. Numbers of customers entering/exiting and passing through the regions is shown in Fig. 19 for doorway regions and Fig. 20 for walkway regions.

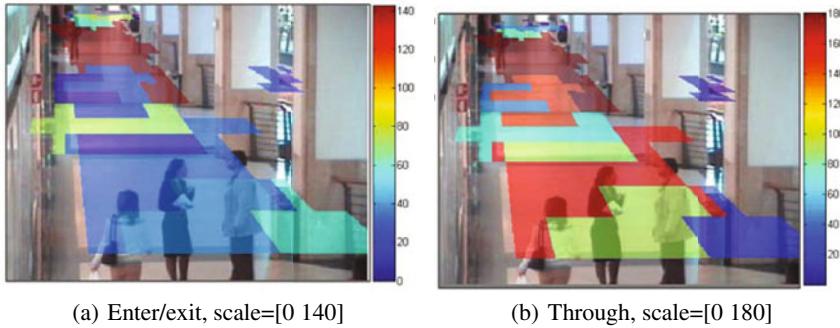


**Fig. 19** Doorway, (a) All functional doorway regions color coded according to the number of pedestrians that enter/exit (b) Doorways are color coded according to number of pedestrians that pass through the region; the legends for each image show their corresponding color to count conversion

The results in Fig. 19a show the number of pedestrians entering or exiting the functional doorway regions. Notice the red doorway region on the left has the highest amount of enter/exit activity (76 movers), while even the combined activity of the blue and light blue doorway regions is only about half of that (41 movers). It can also be seen that most pedestrians enter/exit the scene using the intersecting walkway at the top of the scene.

Figure 20 shows activity regions with a high concentration of the walkway element. Some of these regions are also doorway regions due to the different modes of the functional categories. Also, the colors of each region are based on a different scale, so the colors in Fig. 19 do not correspond to the same amount of activity as that in Fig. 20 or even between the same figure images. This is required in order to make the regions apparent in a single image.

In general, there is more pedestrian traffic in the portion of the main walk-way near the top of the image. This could be useful in deciding the optimal location of a business, since there are more potential customers in the high activity regions.



**Fig. 20** Walkway, (a) All walkway functional regions that are color coded according to the number of pedestrians that enter/exitt (b) Same functional regions as (a), but color coded according to the number of pedestrians that pass through the regions

## 8 Summary

The approaches from [12] were discussed along with extensions of each for recognizing functional regions in webcam and surveillance video. It was shown that activity profiles can be extracted from each region for high level business intelligence analysis. More specifically, results were shown for the Functional Likelihood and Functional Category methods from [12] for the Ocean City and CAVIAR data. It was shown how the results of the Functional Likelihood method could either be directly clustered to form functional regions or used as features in the Functional Category method. Both techniques reduce the sensitivity of the functional regions to the grid cell size. However, the direct clustering approach produces more informative modes to its functional categories, which was shown to be very useful for extracting activity profiles. On the other hand, feeding the Functional likelihood results into the Functional Category algorithm produced more contiguous functional regions than the other methods, which is ideal for scene segmentation. It was also shown how the activity profiles can be used to analyze potential customer behaviors surrounding a business of interest. For example, the location of high activity parking spots relative to business locations, relative activity between businesses, and scene understanding for possible business locations.

**Acknowledgements.** This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. W91CRB-10-C-0098 and is Approved for Public Release, Distribution Unlimited. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

## References

1. Swears, E., Hoogs, A.: Functional Scene Element Recognition for Video Scene Analysis. In: 2009 IEEE Workshop on Motion and Video Computing (2009)

2. Turek, M.W., Hoogs, A., Collins, R.: Unsupervised Learning of Functional Categories in Video Scenes. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 664–677. Springer, Heidelberg (2010)
3. CAVIAR project IST 2001 37540. In: CAVIAR: Context Aware Vision Using Image-Based Active Recognition. EC Information Society Technology's,  
<http://homepages.inf.ed.ac.uk/rbf/CAVIAR/caviar.htm>  
(cited March 29, 2011)
4. Swears, E., Hoogs, A., Perera, A.G.A.: Learning Motion Patterns in Surveillance Video using HMM Clustering. In: 2008 IEEE Workshop on Motion and Video Computing (2008)
5. Grimson, W., Stauffer, C., Romano, R., Lee, L.: Using adaptive tracking to classify and monitor activities in a site. In: 1998 IEEE Conference on Computer Vision and Pattern Recognition (1998)
6. Stauffer, C., Grimson, W.: Learning Patterns of Activity Using Real-Time Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 747–757 (2000)
7. Makris, D., Ellis, T.: Learning semantic scene models from observing activity in visual surveillance. *2005 IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 397–408 (2005)
8. Makris, D., Ellis, T.J.: Path Detection in Video Surveillance. In: *Image and Vision Computing*, vol. 20, pp. 895–903 (2002)
9. Junejo, I., Javed, O., Shah, M.: Multi Feature Path Modeling for Video Surveillance. In: *International Conference on Pattern Recognition*, pp. 716–719 (2004)
10. Stauffer, C.: Estimating tracking sources and sinks. In: *2003 Computer Vision and Pattern Recognition Workshop* (2003)
11. Stark, L., Bowyer, K.: Achieving Generalized Object Recognition through Reasoning about Association of Function to Structure. *IEEE Pattern Analysis and Machine Intelligence* (1991)
12. Gupta, A., Davis, L.: Objects in Action: An Approach for Combining Action Understanding and Object Perception. In: *Computer Vision and Pattern Recognition* (2007)
13. Peursum, P., West, G., Venkatesh, S.: Combining Image Regions and Human Activity for Indirect Object Recognition in Indoor Wide-Angle Video. In: *2005 International Conference on Computer Vision* (2005)
14. Perera, A.G.A., Srinivas, C., Hoogs, A., Brooksby, G., Hu, W.: Multi-object tracking through simultaneous long occlusions and split-merge conditions. In: *Computer Vision and Pattern Recognition* (2006)
15. Comaniciu, D., Meer, P.: Mean Shift: a robust approach toward feature space analysis. *2002 IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 603–619 (2002)
16. Li, J., Gong, S., Xiang, T.: Scene Segmentation for Behaviour Correlation. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part IV*. LNCS, vol. 5305, pp. 383–395. Springer, Heidelberg (2008)
17. Wang, X., Tieu, K., Grimson, W.E.L.: Learning Semantic Scene Models by Trajectory Analysis. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) *ECCV 2006*. LNCS, vol. 3953, pp. 110–123. Springer, Heidelberg (2006)
18. Ali, S., Shah, M.: Floor Fields for Tracking in High Density Crowd Scenes. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part II*. LNCS, vol. 5303, pp. 1–14. Springer, Heidelberg (2008)
19. Yang, Y., Liu, J., Shah, M.: Video Scene Understanding Using Multi-scale Analysis. In: *2009 International Conference on Computer Vision* (2009)
20. Loy, C.C., Xiang, T., Gong, S.: Time Delayed Correlation Analysis for Multi-Camera Activity Understanding. *International Journal of Computer Vision* (2010)

# Analyzing Groups: A Social Signaling Perspective

Loris Bazzani, Marco Cristani, Giulia Paggetti, Diego Tosato, Gloria Menegaz,  
and Vittorio Murino

**Abstract.** This chapter introduces some basic methods to deal with groups of people in surveillance settings. Recently, modeling groups has become a very active trend for video surveillance researchers. Our solution is proper of the recently forged field of social signaling, since it embeds notions of social psychology into computer vision techniques, offering a novel research perspective for the video surveillance community. In particular, we present methods to discover and track groups of people, and to infer what is the focus of attention of each person, that is, we estimate the portion of a scene that is frequently observed by people. Each method we present is evaluated in an experimental section on real scenario, that gives a clear idea of its performance and potentialities.

## 1 Introduction

Recently, researchers in surveillance shifted the attention from the monitoring of a single person in a camera-monitored environment to that of groups: this novel level of abstraction provides event descriptions which are semantically more meaningful, highlighting barely visible relational connections among people. Even if computer vision and pattern recognition supported this new perspective by providing computational models for capturing the whereabouts of groups, such disciplines rarely

---

Loris Bazzani · Giulia Paggetti · Diego Tosato · Gloria Menegaz

University of Verona, Verona, Italy

e-mail: [name.surname@univr.it](mailto:name.surname@univr.it)

Marco Cristani

University of Verona, Verona, Italy, and

Istituto Italiano di Tecnologia (IIT), Genova, Italy

e-mail: [marco.cristani@iit.it](mailto:marco.cristani@iit.it)

Vittorio Murino

Istituto Italiano di Tecnologia (IIT), Genova, Italy

e-mail: [vittorio.murino@iit.it](mailto:vittorio.murino@iit.it)

consider that the basic ingredient of a group is the human being, and that a group is based on interactions among humans. Actually, to the best of our knowledge, all the work that deal with groups assumes that people are simple points on a plane [23, 32, 78, 53, 48, 84, 39] that in some cases may obey to physical laws of attraction and repulsion [54, 66]. None of them considers that working on groups implies to focus on the analysis of the human behavior – a process subject to principles and laws rigorous enough to produce stable and predictable patterns corresponding to social, emotional, and psychological phenomena. On the other hand, these topics are the main subjects of other computing domains, in particular social signaling and affective computing [80], that typically neglect scenarios relevant to surveillance and monitoring.

Social signaling and computer vision are tightly intertwined. In our context, they attempts to discover social interactions using statistical analysis of spatial-orientational arrangements that are relevant in a social psychology sense. Social signals are conveyed, often outside conscious awareness, by nonverbal behavioral cues like facial expressions, gaze, vocalizations (laughter, fillers, back-channel, etc.), gestures and postures. So, there have been identified a large number of behavioral cues carrying social meaning, which are grouped into five classes called codes<sup>1</sup>: physical appearance (attractiveness, clothes, ornaments, somatype, etc.) [59, 86], vocal behavior (everything else than words in speech) [58, 65], face and eyes behavior (expressions, gaze, head pose, etc.) [10, 13], gestures and postures (hand and body movements, conscious and unconscious gestures, orientation with respect to others, etc.) [55, 64], space and environment (mutual distances, spatial organization of people, territoriality, geometric constraints) [30, 59]. In this context, social interactions are here intended as the acts, actions, or practices of two or more peoepl mutually oriented versus each other, that is, every behaviors affecting or considering others' subjective experiences or intentions [61]. For instance, talking is the most common kind of social interaction, but working together, playing chess, eating at a table, and offering a cup of water are social interactions too. In general, any dynamic sequence of social actions among individuals (or groups) that modify their actions and reactions by their interaction partner(s) are social interactions.

The methods presented here take into account these cues in order to give a spectra of algorithms that deal with the *group* entity in a more principled way. In sociology, a group may be defined as a collection of people who share certain aspects, interact with one another, accept rights and obligations as members of the group and share a common identity. We are conscious that identifying such complex relations is a hard task in a typical surveillance scenario, where the input of the method is just a video. For this reason, we set up a definition of group that considers some social signaling cues coming from the code of face and eyes behavior and the space and environment component. In this work, we consider several aspect of groups: 1) the *life of a group*, analyzing how the presence of a group can be detected in crowded situations (i.e., the birth and the death of a group), 2) how a moving group can be tracked (its *evolution*), and 3) which basic activities are carried out by their components in terms

---

<sup>1</sup> For a complete review of social signaling, please read [79].

of interactions between the humans and the environment. In particular, we detect the regions of the environment where the attention of humans is more focused.

The *birth of a group* or its initialization (and, consequently, its break-up) can be performed in two ways. The first assumes that individuals are stationary for a period of time in a given location: for example, in a cocktail party, people may be discussing for a while around a table, before leaving. In a canteen, elements of a group may be clustered around a vending machine. In these cases, social theories help in individuating a group, which can be subsequently followed by a tracking algorithm. In particular, relative positioning and head direction may support this analysis. The second initialization method takes into account situations where people usually move, and no aggregations of stationary people may be observed. In this case, we advocate the use of proxemics, which states that the kind of relation present among the persons depends on the distances they have with respect to each other. Here, we mainly take into account the first scenario, where more interesting and stable (over time) social interactions can be found.

The *evolution of a group* considers a group while it is moving. In this case, we present a tracking approach which embeds the knowledge of the states of the single individuals and the state of a group for providing a robust group localization.

Moreover, we present the idea of how people interact with an environment through an *interest map*, *i.e.*, a map that highlights the part of a scene more considered by a person. For instance, a vending machine in an empty room will surely attract more the attention of people than the peripheral walls. In this direction, we present a system that exploits the use of the head position and orientation for extracting such information.

All these aspects that characterize a group build upon unconventional features that change the perspective followed so far by scholars involved in video surveillance: from the general and unique point of view of a single camera mounted on a wall to a *subjective*, personal, viewpoint, aimed at understanding what is experienced by each single person in the monitored scene. In this context, we propose a general social scenario in which we estimate the position of every person so as to keep track of the related distance among them. This will help in inferring the kind of relation which holds among people in a scene. Another interesting feature we propose is the visual focus of attention, that is, the visual field of view of a person approximated using computational geometry techniques. This helps in estimating the focus of attention of a person while immersed in a whatever scenario.

The rest of the chapter is organized as follows. Section 2 details the basic elements proposed by the computer vision community, and used here both as low-level information and as a basis to understand the techniques illustrated in the subsequent Sections. The core of the chapter is represented by Sections 3, 4 and 5, describing approaches to initialize the groups, to track groups and to create interest maps of a monitored setting, respectively. Finally, conclusions are drawn in Section 6.

## 2 Background

The automatic recognition of social interactions in video recordings is undoubtedly one of the main challenges for a surveillance system. This is usually accomplished using a serial architecture built upon an array of techniques aimed at extracting low-level information, followed by a classification stage. Computer vision techniques are typically exploited in a bottom-up way for extracting low-level features from videos, useful to allow high-level inference. First, all the people in the camera-monitored environment are localized, by exploiting a tracker that provides the trajectory of each person. When the position is estimated, a head orientation method computes the pose of the head of each person, and the subjective field of view (i.e., the view frustum) is initialized by exploiting the calibration of the camera. Thus, the basic components used by the proposed architecture are: a multi-target tracker (see Sec. 2.1), a head pose computation method (see Sec. 2.2), and the subjective view frustum estimation (see Sec. 2.3).

### 2.1 Tracking with Particle Filters

Many multi-target tracking techniques have been successfully proposed in literature from the Kalman filter and its extensions [28, 27] to the more recent Probability Hypothesis Density filter [41] and particle filter (PF) [12, 51]. The success of a tracking algorithm depends on several factors: the strategy used for tracking, the data association approach, the appearance model, occlusion modeling, and so on. Among the realm of the tracking strategies, an important role is played by the particle filtering [12]: the general framework and the simplicity of the method make it one of the most used methods for tracking in the last years.

Particle filtering offers a probabilistic framework for recursive dynamic state estimation. The approach was born originally for single-target tracking [24], then later it was extended to a multi-target tracking scenario [25]. Multi-target particle filters follow different strategies to achieve good tracking performances avoiding huge computational burdens. These are due primarily to the high number of particles required, which is (in general) exponential in the number of targets to track. Recently, an interesting yet general solution has been proposed in [35]. Here, the Hybrid Joint-Separable (HJS) filter is introduced, that maintains a linear relationship between the number of targets and the number of particles. In addition, an occlusion model has been proposed exploiting the camera calibration.

From a Bayesian perspective, the single object tracking problem aims at recursively calculating the posterior distribution  $p(x_t|z_{1:t})$  by exploiting the Chapman-Kolmogorov equation, where  $x_t$  is the current state of the target (e.g., its position and its scale),  $z_t$  is the current measurement (e.g., the current frame), and  $x_{1:t}$  and  $z_{1:t}$  are the states and the measurements up to time  $t$ , respectively. In formulae:

$$p(x_t|z_{1:t}) \propto p(z_t|x_t) \int p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})dx_{t-1} \quad (1)$$

The equation is fully specified by an initial distribution  $p(x_0|z_0) = p(x_0)$ , the dynamical model  $p(x_t|x_{t-1})$ , and the observation model  $p(z_t|x_t)$ . Particle filtering (PF) approximates the posterior distribution by a set of  $N$  weighted particles, *i.e.*,  $\{(x_t^{(n)}, w_t^{(n)})\}_{n=1}^N$ ; a large weight  $w_t^{(n)}$  mirrors a state  $x_t^{(n)}$  with high posterior probability. In this way, the integral in Eq. 1 has not to be analytically solved, and, instead, the posterior at time  $t - 1$  is sampled, defining a set of state hypotheses (the particles) that evolve according to the dynamical model  $p(x_t|x_{t-1})$  (the prediction step), and which is evaluated via  $p(z_t|x_t)$  (the observation step). The parameter  $N$  is manually set here, but there exist techniques for estimating the optimal number of particles that minimizes some measure of tracking distortion [51].

HJS filter [35] is an extension of the PF for multiple targets. Defining  $\mathbf{x}_t = \{x_t^1, x_t^2, \dots, x_t^K\}$  the joint state (the ensemble of the  $K$  individual states), HJS adopts the approximation  $p(\mathbf{x}_t|z_{1:t}) \approx \prod_k p(x_t^k|z_{1:t})$ , that is, the joint posterior could be approximated via the product of its marginal components ( $k$  indexes the individual targets). The dynamics and the observation models of HJS are expressed as follows:

$$p(x_t^k|x_{t-1}^k) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}^{-k}|z_{1:t-1}) d\mathbf{x}_{t-1}^{-k} \quad (2)$$

$$p(z_t|x_t^k) = \int p(z_t|\mathbf{x}_t) p(\mathbf{x}_t^{-k}|z_{1:t-1}) d\mathbf{x}_t^{-k} \quad (3)$$

where the apex  ${}^{-k}$  addresses all the targets but the  $k$ th. These equations encode an intuitive strategy, *i.e.*, that both the dynamics and the observation phases of the  $k$ th target lie upon the consideration of a joint dynamical model  $p(\mathbf{x}_t|\mathbf{x}_{t-1}) \approx p(\mathbf{x}_t) \prod_k q(x_t^k|x_{t-1}^k)$  and observation model  $p(z_t|\mathbf{x}_t)$ . The joint dynamical model, through the prior  $p(\mathbf{x}_t)$ , avoids that multiple targets with single motion described by  $q(x_t^k|x_{t-1}^k)$  collapse in a single location, and the joint observation model considers that the visual appearance of a single target might be occluded by another object, acting as a z-buffer. The two models are weighted by posterior distributions that essentially promote trusted joint objects configurations (not considering the  $k$ th object).

The observational model  $p(z_t|\mathbf{x}_t)$  quantifies the likelihood of the single measure  $z_t$  given the state  $\mathbf{x}_t$ , considering inter-objects occlusions. It is built upon the representation of the targets, that here are constrained to be human beings. The human body is represented by its three components: head, torso and legs. The observational model works by evaluating a separate appearance score for each object (summing then the contribute of the single parts). This score is encoded by a distance between the histograms of the model and the hypothesis (a sample), and it involves also a joint reasoning captured by an *occlusion map*. The occlusion map is a 2D projection of the 3D scene which focuses on the particular object under analysis, giving insight on what are the expected visible portions of that object. This is obtained by exploiting the hybrid particles set  $\{x_p\}_{p=1}^{N \cdot K}$  in an incremental visit procedure on the ground floor. The hypothesis nearest to the camera is evaluated first. Its presence determines

an occluding cone in the scene, with an associated confidence that depends on the observational likelihood achieved. Parts of other objects farther from the camera that fall in the occlusion cone are considered less important in their observational likelihood computation. The process of map building is iterated going deeper and deeper in the scene.

In formulae, the observation model is defined as

$$p(z_t | x_p) \propto \exp\left(-\frac{fc_p + bc_p}{2 \sigma^2}\right), \quad (4)$$

where  $fc_p$  is the foreground term, *i.e.*, the likelihood that an object matches to the model considering the un-occluded parts, and  $bc_p$ , the background term, accounts for the occluded parts of an object. For more details, readers may refer to [35].

## 2.2 Head Orientation Estimation

Head orientation estimation is becoming an important computer vision application. There are several diverse approaches present in the literature: a recent review can be found in [47], where a performance analysis of different methods is presented, and a list of the commonly used dataset for head pose estimation is shown. The CLEAR workshops are important events for the head pose estimation community, and several important approaches can be found in the related proceedings [72, 71]. It is worth noting that most of the approaches are based on classification schemes.

In the multi-faceted ensemble of the classification approaches, boosting-based techniques play a primary role [37, 81, 90, 77, 88, 50]. Boosting [18, 63, 19] is a remarkable, highly customizable way to create strong and fast classifiers, employing various features fed into diverse architectures with *ad-hoc* policies. Among the different features exploited for boosting in surveillance applications (see [89] for an updated list), covariance features [76] have been exploited as powerful descriptors of pedestrians [77, 88], and their effectiveness has been explicitly investigated in a comparative study [50]. When injected in boosting systems [77, 88, 50, 75], covariances provide strong detection performance, encapsulating possible high intra-class variances (due to pose and view changes of an object of interest). They are in general stable under noise, and furnish an elegant way to fuse multiple low-level features as, in fact, they intrinsically exploit possible inter-feature dependencies. In this chapter, we present in details the method proposed in [75].

The tracker provides the location of the head and the feet for each person in each frame. As for the head approximate position, we define a square window  $I$  of size  $r \times r$ , where we run a multi-class algorithm that recovers the head orientation. The size  $r$  is chosen large enough in order to contain a head, considering the experimental physical environment and the camera position.

For the multi-class classification, we boost regression trees [8, 75], because they are the ideal weak learning strategy, since they can tolerate a significant amount of

labeling noise and errors in the training data (which are very likely in low resolution images). Moreover, they are very efficient at runtime, since matching a sample against a tree is logarithmic in the number of leaves.

From the mathematical point of view, they are an alternative approach to nonlinear regression. The principle is to subdivide, or partition, the space in two smaller regions, where the data distribution is more manageable. This partitioning proceeds recursively, as in hierarchical clustering, until the space is so tight that a simple model can be easily fitted. The global model thus has two parts: one is just the recursive partition, the other is a simple model for each cell of the partition. Regression trees are more powerful than global models, like linear or polynomial regression, where a single predictive formula is supposed to hold over the entire data space.

In order to avoid the overtraining of the regression tree, we establish as stopping rule a minimal number  $\tau$  of observations per tree leaf, that is experimentally estimated (see Sec. 3.2).

In our approach, we extract from each image of size  $I$  ( $r \times r$  pixels), a set  $\Phi(I, x, y)$  of dimension  $r \times r \times d$  features where  $d = 12$  and  $x, y$  are the pixel locations, that is defined as follows:

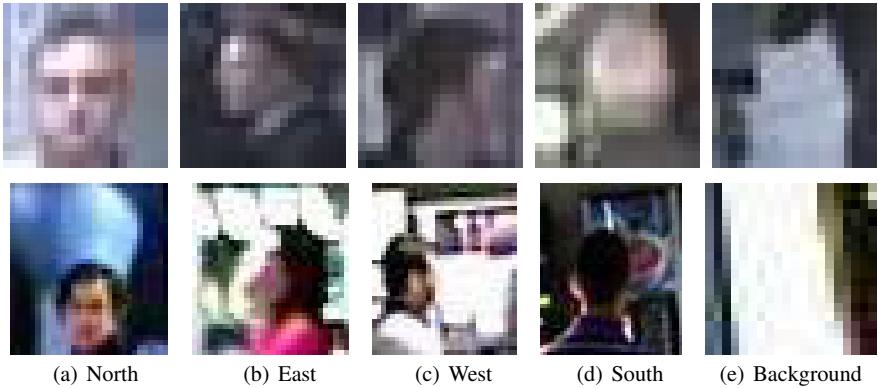
$$\Phi(I, x, y) = [X \ Y \ R \ G \ B \ I_x \ I_y \ O \ \text{Gab}_{\{0, \pi/3, \pi/6, 4\pi/3\}}]. \quad (5)$$

$X, Y$  represent the spatial layout maps in  $I$ , and  $R, G, B$  are the color channels.  $I_x$  and  $I_y$  are the directional derivatives of  $I$ , and  $O$  is the gradient orientation. Finally, Gab is a set of 4 maps containing the results of Gabor filtering, with filters of dimension  $2 \times 4$ , sinusoidal frequency 16, and directions  $\mathcal{D} = \{0, \pi/3, \pi/6, 4\pi/3\}$ . In order to increase the robustness to local illumination variations, we apply the normalization operator introduced in [77] before applying the multi-class framework. First, we estimate the covariance of the image  $I$ , denoted as  $X_I$ . Then, for each element  $X_i$  of the dataset, we apply the following normalization:

$$\hat{X}_i = \text{diag}(X_I)^{-\frac{1}{2}} X_i \text{diag}(X_I)^{-\frac{1}{2}}, \quad (6)$$

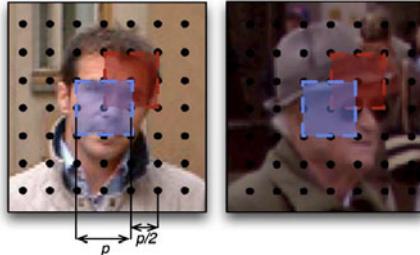
where  $\hat{X}_i$  is the normalized descriptor, and  $\text{diag}(X_I)$  is a square matrix with only the diagonal entries of  $X_I$ .

Our approach takes inspiration from the literature on dense image descriptors (see [11] as an example). We sample the window  $I$  employing an array of uniformly distributed and overlapping patches of the same dimension. For each of the  $N_p = 16$  sampled patches inside the  $r \times r$  region of interest, described by the covariance matrix of a set of  $d$  image features described by the Eq. (5), a multi-class LogitBoost classifier is trained. Each class represent a different head orientation sampled according with a fixed sampling step  $\alpha$  and from an extra class containing all the background examples. We experimentally found that  $\alpha = 90^\circ$  which correspond to the semantic classes North, South, East and West, is enough for our purposes. Fig. 1 shows some training and testing examples for each class. At testing time, each patch of a sample window (Fig. 2) is independently classified. Then, the classification



**Fig. 1** Examples of the 5 semantic classes we defined for the multi-class problem of head pose estimation. a) North, b) East, c) West, d) South, and e) Background. The first row shows some examples of the training set, and the second row shows some sample windows at testing time. Note that the images have very low resolution (min.  $20 \times 20$  pixels).

result is given by a majority criterion across the patches. We name the combination of this patch description that encodes the local shape and appearance and its uniformly distributed architecture *ARray of COvariances* (ARCO, for the sake of brevity).



**Fig. 2** Array of Covariance matrices (ARCO) feature. The image is organized as a grid of uniformly spaced and overlapping patches. The head orientation result of each patch is estimated by a multi-class classifier.

More formally, given a set of patches  $\{P_i\}_{i=1,\dots,N_p}$ , we learn a multi-class classifier for each patch location  $\{F_{P_i}\}_{i=1,\dots,N_p}$  through the multi-class LogitBoost algorithm [19], adapted to work on Riemannian manifolds, as suggested by [77, 75]. This method implies that each covariance matrix must be projected on a proper tangent space (vector space) of the Riemannian manifold to be classified. Since we deal with a multi-class problem, a common tangent space is chosen where all the covariances are projected and discriminated. Computational considerations suggest to use

the identity matrix  $I_d$  as projection point. From a mathematical point of view, the projection is a logarithmic transformation of the (positive) eigenvalues of a covariance matrix; therefore, the computational complexity of each projection is bounded by the eigenvalue decomposition complexity  $O(d^3)$ . Since  $d$ , the number of image features, is small the projection results a fast operation. All the details of the projection operation are contained in [77, 75].

Let  $\Delta_j = \sum_{i=1}^{N_p} (F_{P_i} == j)$  be the number of patches that vote for the class  $j \in \{1, \dots, J\}$ . To assign a class label  $c$  to a new image, we fuse the votes with a majority voting strategy among all the classes:

$$c = \arg \max_j \{\Delta_j\}, \quad j = 1, \dots, J. \quad (7)$$

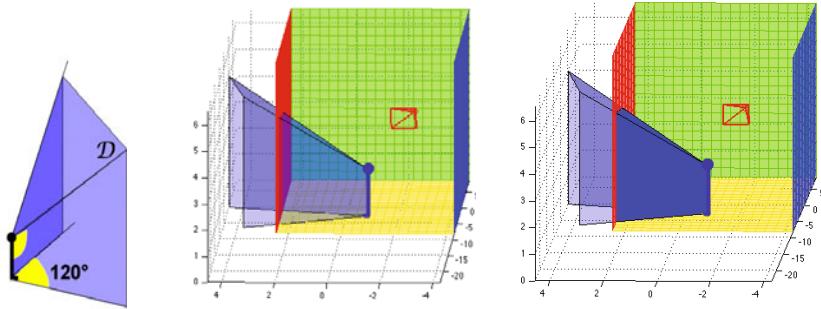
Actually, in our approach, we employ 5 classes mentioned above, i.e., North, South, East, West, and Background. The first four classes indicate the four directions related to the camera orientation. The Background class is introduced to manage cases where the tracker fails to provide a correct head position. We are aware that the use of only four directions may lead to rough estimates, but it should be considered that the resolution of the source video data is very poor. We are also aware that the head orientation could be injected in a tracking framework [67], as an additional state that characterizes the human body in this way smoother results should be obtained. However, this smoothing gives poor results and in some cases a drift of the track when dealing with low frame rate videos. Thus, for the sake of the generality, we prefer to keep person tracking and head orientation estimation separated, so as to minimize the error in low frame rate scenario and also they can be used separately in other applications.

The ARCO representation has several advantages. First, it allows to take into account different features, inheriting their expressivity and exploiting, by definition, possible correlations. Second, due to the use of integral images for the computation of the covariance matrices [77], it is fast, making it suitable for a possible real-time usage.

### 2.3 Subjective View Frustum Estimation

The Visual Focus Of Attention (VFOA) [74, 40, 67] is a very important aspect of non-verbal communication. It is well known that a person's VFOA is determined by his eye gaze. Since objects are foveated for visual acuity, gaze direction generally provides more precise information than other bodily cues regarding the spatial localization of the attentional focus. A detailed overview of gaze-based VFOA detection in meeting scenarios is presented in [1]. However, measuring the VFOA by using eye gaze is often difficult or impossible: either the movement of the subject is constrained or high-resolution images of the eyes are required, which may not be practical [44, 69], and several approximations are considered in many cases. For example, in [74], it is claimed that the VFOA can be reasonably inferred by head pose, and this is the choice made in many works. Following the same hypothesis, in

[67] pan and tilt parameters of the head are estimated, and the VFOA is represented as a vector normal to the person's face, and it is employed to infer whether a walking person is focused on an advertisement located on a vertical glass or not. Since the situation is very constrained, this proposed VFOA model works pretty well, but a more complex model, considering camera position, person's position and scene structure, is required in more general situations. The same considerations hold for the work presented in [40], where Active Appearance Models are fitted on the face of the person in order to discover which portion of a mall-shelf is observed. In [31], the visual field is modeled as a tetrahedron associated with a head pose detector. However, their model fixes the depth of the visual field, and this is quite unrealistic.



**Fig. 3** Left: the SVF model. Center: an example of SVF inside a 3D “box” scene. In red, the surveillance camera position: the SVF orientation is estimated with respect to the principal axes of the camera. Right: the same SVF delimited by the scene constraints (in solid blue).

In cases where the scale of the scene does not allow to capture the eye gaze directly, viewing direction can be reasonably approximated by just measuring the head pose. This assumption has been exploited in several approaches dealing with a meeting scenario [74, 73, 49, 82] or in a smart environment [67, 34]. Following this claim, and considering a general, unrestricted scenario, where people can enter, leave, and move freely, we approximate VFOA as the *Subjective View Frustum* (SVF), first proposed in [16]. This feature represents the three-dimensional (3D) visual field of a human subject in a scene. According to biological evidence [52], the SVF can be modeled as a 3D polyhedron delimiting the portion of the scene that the subject is looking at (see Figure 3).

More in detail, the SVF is defined as the polyhedron  $\mathcal{D}$  depicted in Figure 3. It is composed by three planes that delimit the view angles on the left, right and top sides, in such a way that the angle span is  $120^\circ$  in both directions. The 3D coordinates of the points corresponding to the head and feet of a subject are obtained from a multi-target tracker, while the SVF orientation is obtained by an head pose detector.

The SVF  $\mathcal{D}$  is computed precisely using computational geometry techniques. It can be written as the intersection of three negative half-spaces defined by their supporting planes of the left, right and top sides of the subject. In principle, the SVF

is not bounded in depth, modeling the human capability of focusing possibly on a remote point located at infinite distance. However, in practice, the SVF is limited by the planes that set up the scene, according to the 3D scene (see Figure 3). The scene volume is similarly modeled as intersection of negative half-spaces consequently, the exact SVF inside the scene can be computed solving a simple *vertex enumeration* problem, for which very efficient algorithms exist in literature [57].

### 3 The Birth of a Group

Employing the SVF in conjunction with cues of the *space and environment* category allows to detect signals of the possible people's interest, with respect to both the physical environment [16], and the other participants acting in the scene. More specifically, we present a method to statistically infer if a participant is involved in an interactional exchange. In accordance with cognitive and social signaling studies, we define the birth of a group when multiple and stable relations are detected over time. In particular, it is highly probable that a relation takes place when two persons are closer than 2 meters [79], and looking at each other [87, 33, 26]. We assume that this condition can be reliably inferred by the position and orientation of the SVFs of the people involved. This information can then be gathered in an *Inter-Relation Pattern Matrix* (IRPM), that encodes the social exchanges occurred among all the persons in a scene. The work we present in this section has been published in [4, 15].

Detecting human relations may be useful to instantiate a more robust definition of group in surveillance applications. Actually, in the last few years, several applications focused on group modeling [46, 43], and person re-identification [91] have been proposed. In the former case, a group is defined following physically-driven proximity principles. While in the latter, groups are exploited to improve person re-identification, relying on the fact the people usually stay in the same group when moving in an environment.

Our proposal is a step towards automatic inference and analysis of social interactions in general, unconstrained conditions: it is alternative to the paradigm of wearable computing [56, 9], or smart rooms [83]. In the typical non-cooperative video surveillance context or when a huge amount of data is required, wearable devices are not usable. Moreover, the use of non-invasive technology makes people more prone to act normally.

Considering the literature (except our first work in [16]), the “subjective” point of view for automated surveillance systems was taken into account by [5], taking inspiration from [60], and it represents therefore the most similar approach in the literature to our work. The difference between [5] and our system are that 1) in [5], the gaze is projected on the ground plane, while in our case we embed the 3D subjective view frustum in the 3D scene, employing computational geometry rules, so that the full 3D information allows finer spatial reasoning, needed, for example, to deal with head poses having different tilt angles. 2) They do not perform interaction analysis, and the subjective point of view was functional solely on the estimation of scene interest maps.

Summarizing, we introduce the concept of Inter-Relation Pattern Matrix that exploits the SVF. Its aim is to infer relations among people for detecting groups in a general crowded scenario. This work not only fills a gap in the state of the art of social signaling aimed at understanding social interactions, but also represents a novel research opportunity, alternative to the scenarios considered so far in socially-aware technologies, where automatic analysis techniques for the spatial organization of social encounters are taken into account.

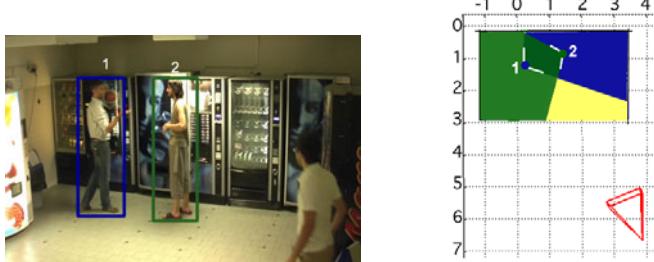
In this section, we consider a scenario where individuals are quasi-stationary for a short period of time in a given location, and we use just simple proxemics cues [79], when dealing with moving people. However, the SVF can also be exploited as a supplementary hint to make more robust the proxemics-based method even in that scenario.

In Sec. 3.1, the method to build the Inter-Relation Pattern Matrix is described. Then, in Sec. 3.2, experiments and results on home-made and public datasets are shown.

### 3.1 The Inter-Relation Pattern Matrix

The SVF can be employed as a tool to discover the visual dynamics of the interactions among two or more people. Such analysis relies on few assumptions with respect to social cues, i.e., that the entities involved in the *social interaction* stand closer than 2 meters (covering thus the *socio-consultive zone* – between 1 and 2 meters – the *casual-personal zone* – between 0.5 and 1.2 meters – and the *intimate zone* – around 0.4-0.5 meters) [79]. Then, it is generally well-accepted that initiators of conversations often wait for visual cues of attention, in particular, the establishment of eye contact, before launching into their conversation during unplanned face-to-face encounters [87, 33, 26]. In this sense, SVF may be employed in order to infer whether an eye contact occurs among close subjects or not. This happens with high probability when the following conditions are satisfied: 1) the subjects are closer than 2 meters; 2) their SVFs overlap, and 3) their heads are positioned inside the reciprocal SVFs (see Figure 4). The Inter-Relation Pattern Matrix (*IRPM*) records when a possible social interaction occurs, and it can be formalized as a three-dimensional matrix [17], where each entry  $IRPM(i, j, t) = IRPM(j, i, t)$  is set to one if subjects  $i$  and  $j$  satisfy the three conditions above, during the  $t$ -th time instant.

The IRPM matrix serves to analyze time intervals in which we look for social interactions. Let us suppose to focus on the time interval  $[t - T + 1, t]$ . In this case we take into account all the IRPM slices that fall in  $[t - T + 1, t]$ , summing them along the  $t$  direction, and obtaining the *condensed* IRPM (*cIRPM*). Intuitively, the higher is the entry  $cIRPM_t(i, j)$ , the stronger is the probability that subjects  $i$  and  $j$  are interacting during the interval  $[t - T + 1, t]$ . Therefore, in order to detect a relation between a pair of individuals  $i, j$  in the interval  $[t - T + 1, t]$ , we check if  $cIRPM_t(i, j) > Th$ , where  $Th$  is a threshold defined *a priori*. This threshold filters out noisy group detection: actually, due to the errors in the tracking and in the head



**Fig. 4** Left: two people are talking each other. Right: top view of their SVFs: the estimated orientation, East for 1 and West for 2, is relative to the camera orientation (the pyramid in red in the picture). The SVFs satisfy the three conditions explained in Section 3.1

pose estimation, the lower the threshold, the higher the possibility of false positives detection. In the experiments, we show how the choice of the parameters  $T$  and  $Th$  impacts on the results, in term of social interaction detection rates.

The cIRPM represents one-to-one exchanges only, but we would like also to capture the presence of *groups* in the scene. Here, we will not use the term group in its sociological meaning, because we are aware that detecting such complex relations using just a video as input is a hard task. For this reasons we consider the group, as an assemblage of people standing near together, and forming a collective unity, a knot of people. The latter meaning is closer to our aims.

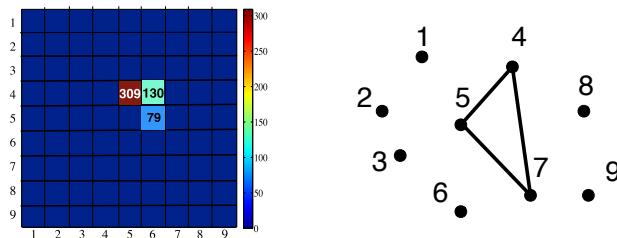
Operationally, we treat the *cIRPM* as the adjacency matrix of an undirected graph, with a vertex  $v_i$  for each people in the scene, and an edge  $e_{ij}$  if  $cIRPM_t(i, j) > Th$ . The *groups* present in the scene are detected by computing the connected components of the graph. Some examples are depicted in Figures 6, 7 and 8.

### 3.2 Experimental Results

These experiments aim at showing the capabilities of the proposed approach. We recorded a video sequence of about 3 hours and a half duration, portraying a vending machines area where students take coffee and discuss. The video footage was acquired with a monocular IP camera, located on an upper angle of the room. The people involved in the experiments were not aware of the aim of the experiments, and behaved naturally. Afterwards, since creating the ground truth by using only the video is a complex task, we asked to some of them to fill a questionnaire inquiring if they talked to someone in the room and to whom. Then, the video was analyzed by a psychologist able to detect the presence of interactions among people. The questionnaires were used as supplementary material to confirm the validity of the generated ground truth. This offers us a more trustworthy set of ground truth data for our experiments.

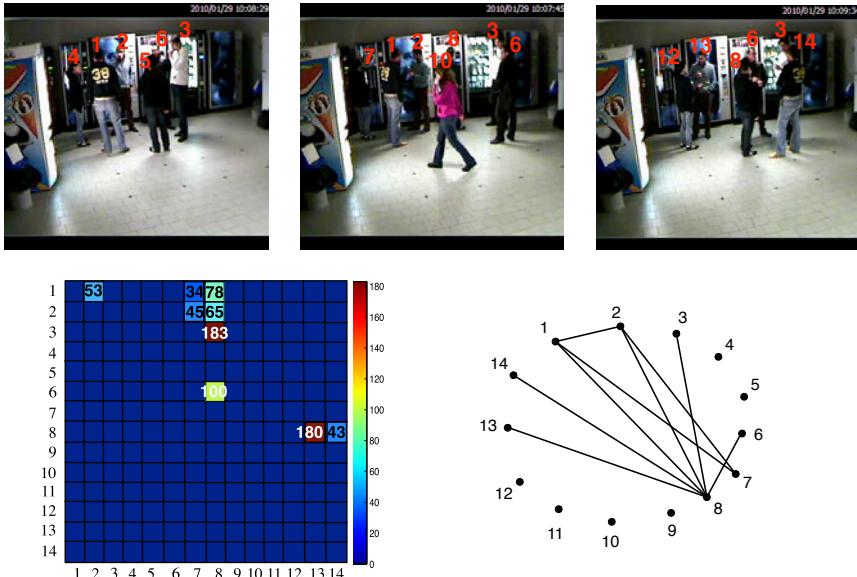


**Fig. 5** Examples of tracking and head orientation classification results. The largest box represents the tracking estimation, the smaller box the area where the head is positioned, and the triangle depicts the estimated head orientation.



**Fig. 6** Example of IRPM analysis of sequence  $S_{04}$ . On the top row, some frames of the sequence. On the bottom row, on the left, the cIRPM matrix. Being the cIRPMs symmetric and having null main diagonals, we report for clarity only its strictly upper triangular part. On the right, the corresponding graph. As one can notice, only one group (composed by people 4, 5 and 7) is detected. This is correct, since the other persons in the sequence were not interacting.

The publicly available dataset, called GDet<sup>2</sup>, is composed of 12 sub-sequences of about 2 minutes each. They are chosen such that to represent different situations, with people talking in groups and other people not interacting with anyone. For each sub-sequence, we performed tracking, head orientation classification (some examples are shown in Figure 5), and construction of the three-dimensional IRPM, indicating which people are potentially interacting at a specific moment.



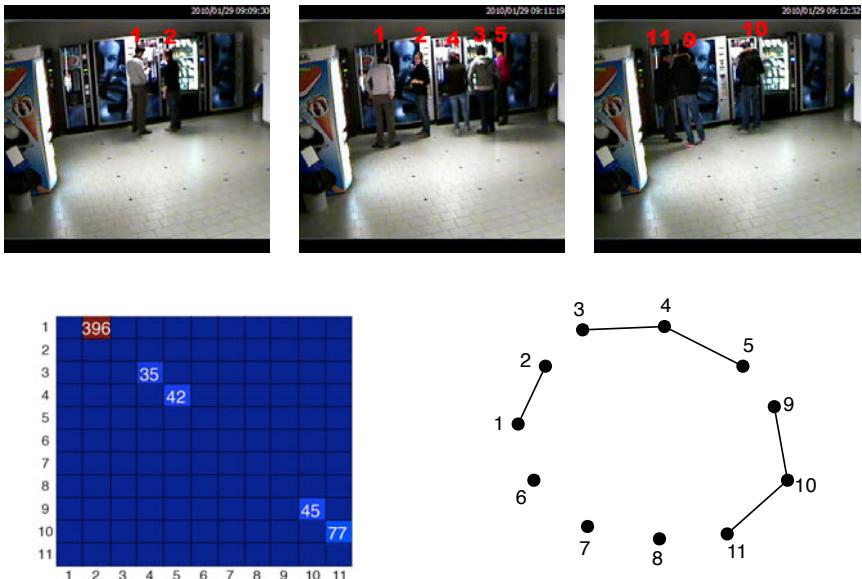
**Fig. 7** Example of cIRPM analysis of sequence  $S_{08}$ . One big group (1,2,3,6,7,8,13,14) is detected. Note that some people are represented by more than one track, since due to severe or complete occlusions the tracks are sometimes lost and need to be reinitialized (see the text for more details). Person 10 that enters in the room is correctly detected as non-interacting by the cIRPM.

The comparison of our results with the ground truth revealed that 8 out of 12 sequences were correctly interpreted by our system. One can be considered wrong, because there are 2 groups in the scene, and our system reveals that they belong all to the same group. In the other three sequences there are some inaccuracies, like a person left out of a group. These inaccuracies are mainly due to error propagation from tracking and head orientation classification, particularly challenging when people are grouped together and frequently intersect. A qualitative analysis of the results is shown in Figures 6, 7 and 8. The first row of each figure depicts three sampled frames from each sequence and contains the identifiers of each person. The second

<sup>2</sup> The dataset is available at

<http://www.lorisbazzani.info/code-datasets/multi-camera-dataset/>

row depicts the *cIRPM* on the left and the graph structure that defines the group interactions on the right. In all the three experiments, all the groups are detected correctly. In particular, Fig. 6 shows the case where a single, small group and other individuals are present in the scene during the recording. In Fig. 7 a more complex situation is analyzed, that is, a big group is in the scene (composed by 6 individuals). One big group (1, 2, 3, 6, 7, 8, 13, 14) is found by our method. Note that some people are represented by more fragments of tracks, because we have tracking failures due to long and complete occlusions (person 10 occludes the group). Thus, the lost tracks are reinitialized with a new ID. The associations between the different track fragments are: (1, 14), (2, 13), (4, 7, 12), and (5, 8). The automatic association between IDs is also possible in such scenarios using re-identification or re-acquisition methods such as [14, 8, 3, 9], but it is out of the scope of this work. Fig. 8 shows that our model is able to detect interactions also when the scene contains multiple groups.



**Fig. 8** Example of cIRPM analysis of sequence  $S_{01}$ . Three groups (1,2),(3,4,5), and (9,10,11) are detected. One can note that some people are represented by more than one track, since due to severe or complete occlusions the tracks are sometimes lost and need to be reinitialized (e.g. 6,7,8 are reinitialized as 9,10,11, respectively).

A more sophisticated analysis of accuracy performances of our method is shown in Fig. 9 and 10. The graphs summarize the group detection accuracy in terms of precision (on the left) and recall (on the right). In the definition of those measurements, we consider as true positive when a group is detected considering all its constitutive members. If a person that belongs to a group is not detected, we have a false negative, and a similar reasoning applies for the false positive.

Fig. 9 depicts the statistics as a function of the size of the time interval  $T$  frames (x-axis) used to accumulate the IRPM. Each curve corresponds to a value of threshold  $Th$  (5, 20, 60 and 100). From this figure, we notice that increasing  $T$  gives worse accuracy. Moreover, the peak of each curve depends on both the threshold and the time interval size. We obtain the best performance by setting the  $Th$  equal to 20; the peak of this curve corresponds to  $T$  equal to 300 frames. Instead, Fig. 10 shows the performances increasing the threshold (x-axis) used to detect the groups. Each curve corresponds to a value of  $T$  (120, 300, 480, 720, 900, and 1200 frames). The common behavior of all the curves is that increasing and decreasing too much the threshold decreases the accuracy. This analysis confirms that the best performances are given by setting the threshold to 20 and the time interval to 300 frames. When  $T$  increases the accuracy drastically decreases and the peak of each curve is shifted, depending on the time interval size.

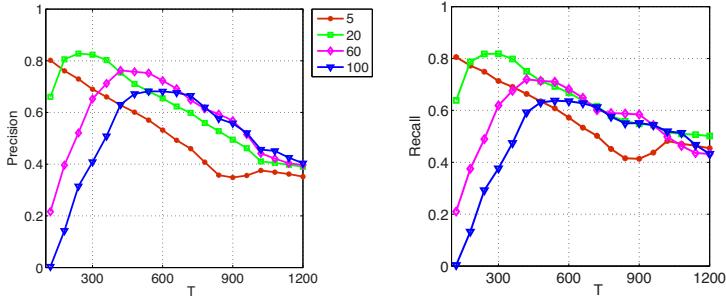
Intuitively, when the threshold is too low and the time window is too small, our method detects interactions that could contain false positives. Increasing the size of the time window and the threshold permits to average out and cancel out these false positive, because the IRPM becomes more stable. On the other hand, when the threshold is too high, our model is not able to detect interactions, because  $cIRPM_t(i, j) > Th$  is zero for each  $(i, j)$ . To deal with this problem, we could fix the time interval larger. However, in this case, a group interaction interval shuld be smaller than the time window, and in any case the threshold would result too high to detect groups. For these reasons, precision and recall in Fig. 9 and Fig. 10, respectively, decrease before and after the optimal setting of the parameters ( $Th = 20$  and  $T = 300$ ).

## 4 The Evolution of a Group

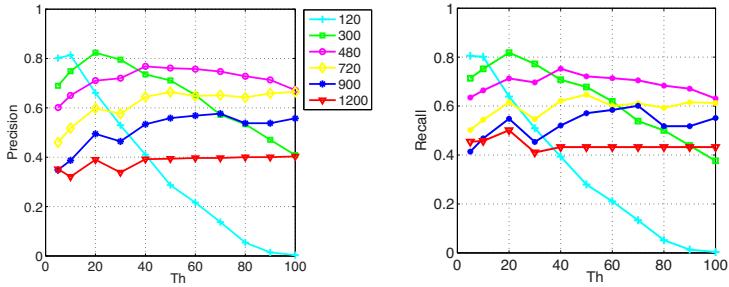
In this section, we consider the evolution of a group as an instance of tracking, called *group tracking*, published in [2]. Once a group is initialized through cIRPM or employing simple proxemics guidelines, a group can move in the scene, and we propose a tracking approach which embeds the knowledge of the states of the single individuals and the state of the group to provide a robust group localization and tracking.

Group tracking (GT) is of high interest for video surveillance purposes as it allows fine scenario descriptions addressing choral actions that may lead to important threats and highlighting social bounds among individuals. At the same time, it represents a challenging issue, since a group of people is a highly structured entity whose dynamics is complex, and whose appearance is erratic, due to intra- and inter-group occlusions phenomena.

There have been few recent attempts to deal with GT problem. The literature on GT could be partitioned in two sets. The first, *top-down GT*, contains techniques that model groups as blob entities found after background subtraction [46, 85, 43]. In [46], a foreground segmentation method classifies the moving regions in people and groups. In [43], a foreground subtraction-based method models the object



**Fig. 9** Evaluation of precision (left) and recall (right) of the proposed method varying the size of the time interval  $T$  (x-axis) used to compute the IRPM. The graph shows one curve for each threshold (5, 20, 60 and 100). The maximum for both the statistics is given by setting  $Th = 20$ .



**Fig. 10** Evaluation of precision (left) and recall (right) of the proposed method varying the threshold  $Th$  (x-axis) used to detect the groups. The graph shows one curve for each time window (120, 300, 480, 720, 900, and 1200). The maximum for both the statistics is given by setting  $T = 300$  and the peak corresponds to  $Th = 20$ .

paths using a Bayesian network. A set of empirical rules are employed to detect the groups, however, intra- and inter-group dynamics are not considered in these methods.

The second, *bottom-up GT*, is formed by algorithms that operate after that the individuals have been individually tracked [20, 45, 38, 36]. A set of empirical merging and splitting rules embedded into a Kalman filter are proposed in [20] to track groups. However, the Kalman filter is not able to deal with non-linear dynamics, if not resorting to more complex variants. In [45], a deterministic mass-spring model interprets the result of a multi-object tracker, joining objects sharing a common behavior. In [38], a lattice-based Markov Random Field combined to a particle filter tracks groups as near-regular textures. A method that tracks a group of highly correlated targets by employing a Markov Chain Monte Carlo particle filter is proposed in [36]. However, the last two approaches deal with very constrained intra-group dynamics because they assume a strong correlation among the targets.

In this section, we present a novel way to track groups, namely Collaborative Particle Filters (Co-PF). The underlying idea consists in designing two tracking processes observing a scene under two different perspectives: a low-level, *multi-object tracker* (MOT) performs tracking of multiple individuals, separately; a high-level, *multi-group tracker* (MGT) focuses on groups, and uses the knowledge acquired by the MOT to refine its estimates. Each process consists in a Hybrid-Joint Separable (HJS) filter (see Sec. 2.1), permitting to track multiple entities dealing with occlusions in a very effective way.

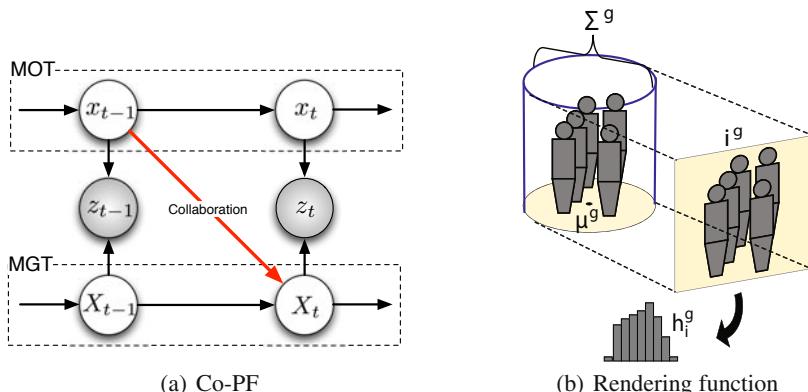
The input given by the MOT flows to the MGT in a principled way, *i.e.*, revising the MGT posterior distribution by marginalizing over the MOT's state space. In this way, the MGT posterior is peaked around group configurations formed by trusted individual estimates. In practice, our framework allows to: i) track *multiple* groups; ii) deal with intra- and iii) inter-group occlusions in a 3D calibrated context. The latter two conditions have never been taken into account jointly, and define a brand new operating context, where we put a solid possible solution. Synthetic and real experiments validate our approach and encourage further developments for Co-PF.

The rest of the Section is organized as follows. In Sec 4.1 the collaborative particle filter framework is described and related experiments are presented in Sec 4.2.

## 4.1 Collaborative Particle Filter

The framework we analyze is sketched in Fig. 11(a): the MOT tracks the individuals in the scene, whereas the MGT tracks groups of individuals. Both the processes share the same observations,  $\{z_t\}$ , and this highlights our key intuition: the two processes evaluate the scene under two different points of view.

The MOT process is modeled by HJS filter [35] (Sec. 2.1). Each individual state is modeled as an elliptical shape on the ground plane, *i.e.*,  $x^k = \langle \mu^k, \Sigma^k \rangle$ , where  $\mu^k$  is



**Fig. 11** Collaborative PF idea and group rendering.

the position of the individual on the ground plane<sup>3</sup>,  $\Sigma^g$  is a covariance that measures the occupancy of the body projected on the ground plane (see [35] for more details).

The MGT process customizes the HJS filter for dealing with groups, and incorporates a fusion component, accepting information from the MOT. We denote the  $g$ th group as  $X_t^g = \langle \mu^g, \Sigma^g \rangle$ , where  $\mu^g$  is the 2D position on the floor of the centroid of the  $g$ th group and  $\Sigma^g$  is the covariance matrix that approximates the projection of its shape on the floor. The choice of an ellipse for modeling the floor projection of a group is motivated from a sociological point of view, exploiting proxemics notions that describe a group as a compact closed entity [22]. The posterior of the MGT of the  $g$ -th group follows the Bayesian recipe (Eq. 1), so that

$$p(X_t^g | z_{1:t}) \propto p(z_t | X_t^g) \int p(X_t^g | X_{t-1}^g) p(X_{t-1}^g | z_{1:t-1}) dX_{t-1}^g. \quad (8)$$

The dynamical model  $p(X_t^g | X_{t-1}^g)$  is derived as in Eq. 2, where the joint dynamical model  $p(\mathbf{X}_t | \mathbf{X}_{t-1}) \approx p(\mathbf{X}_t) \prod_g q(X_t^g | X_{t-1}^g)$  has  $\mathbf{X}_t = \{X_t^1, X_t^2, \dots, X_t^G\}$ , with  $G$  the number of groups in the scene. In this case, the function  $q(X_t^g | X_{t-1}^g)$  is modeled by considering the nature of  $X_t^g = \langle \mu^g, \Sigma^g \rangle$ . For the centroid  $\mu^g$ , we assume a linear motion, perturbed by white noise with parameter  $\sigma_\mu$ . The dynamics of the covariance matrix  $\Sigma^g$  is defined by a perturbation of its principal axes, *i.e.*, by varying its eigenvalues  $\{\lambda_i\}_{i=1,2}$  and eigenvectors  $\{\mathbf{v}_i\}_{i=1,2}$ . In particular, we rotate the principal axes by an angle  $\theta$ , by modifying the eigenvectors:  $V' = [R(\mathcal{N}(\theta, \sigma_\theta)) \mathbf{v}_1, R(\mathcal{N}(\theta, \sigma_\theta)) \mathbf{v}_2]$  and then, we vary the amplitude of the principal axes by modifying the eigenvalues as follows:

$$\Lambda' = \begin{bmatrix} \mathcal{N}(\lambda_1, \sigma_\lambda) & 0 \\ 0 & \mathcal{N}(\lambda_2, \sigma_\lambda) \end{bmatrix} \quad (9)$$

where  $R(\cdot)$  is a rotation matrix and  $\sigma_\theta$  and  $\sigma_\lambda$  are user-defined noise variance values. The matrixes  $V'$  and  $\Lambda'$  are then used to recompose the new hypothesis  $\Sigma' = V' \Lambda' V'^T$ , that will represent a new perturbed elliptical shape. The dynamics prior  $p(X_t^g)$  implements an exclusion principle using Markov Random Fields [35] that cancels out inconsistent hypothesis (*e.g.*, individuals in the same location).

The (single) observation model  $p(z_t | X_t^g)$  is derived from Eq. 3, where we have  $p(z_t | \mathbf{X}_t)$  as joint observation model. In order to easily evaluate an observation  $z_t$ , we employ a *rendering function* that maps a state in a convenient feature space<sup>4</sup>. The idea is depicted in Fig. 11(b): when a new group is detected at time  $t$  in the scene, its centroid  $\mu^g$  and occupancy area  $\Sigma^g$  are robustly estimated, forming the initial state  $X_t^g$ . The rendering function builds a volume of height 1.80m upon the area  $\Sigma^g$ , in order to surround the people of the group. From this volume, the projection  $i^g$  (namely, the model of  $X_t^g$ ) on the image plane is evaluated, and finally, the histogram  $h_i^g$  is computed. This function permits to estimate novel state

<sup>3</sup> Please note that the ground plane position is inferred employing the calibration of the camera.

<sup>4</sup> This is analogue to what was done in [35] for the single individuals.

hypotheses  $X'^g_t$ : given its components  $\langle \mu'^g, \Sigma'^g \rangle$ , the rendering function takes the model  $i^g$  deforming it opportunely (by a re-scaling, considering the  $\mu'^g$ , and by a shearing, taking into account the deformation resulted by the perturbation of the covariance matrix  $\Sigma'^g$ ). This brings to a novel  $h'^g_i$ , which is compared with the observation estimated directly from the scene by the rendering function applied to  $\langle \mu'^g, \Sigma'^g \rangle$ . We use the Bhattacharyya distance as similarity measurement.

The joint observation model  $p(z_t | \mathbf{X}_t)$  mirrors what part of the group  $X'^g_t$  is visible (not occluded) by taking into account the remaining groups  $X'^{-g}_t$ . This encodes at the same time the advantages and limitations of the observation model. Actually, we assume a group as a rigid solid shape (the model  $i^g$ ), and this permits to model inter-group occlusions, but it does not model intra-group occlusions (i.e., persons of a group that mutually occlude each other). This leads to tracking applications where a strong intra-group occlusion causes the loss of that group.

Co-PF solves this problem, and permits a very fine estimation of the whereabouts of a scene, making the group tracking very robust. It basically injects the information collected by the MOT into the MGT. Considering the filtering expression in Eq. 8, the fusion occurs on the posterior at time  $t - 1$ :

$$p(X'^g_{t-1} | z_{1:t-1}) \propto \int p(X'^g_{t-1} | \mathbf{x}_{t-1}, z_{1:t-1}) p(\mathbf{x}_{t-1} | z_{1:t-1}) d\mathbf{x}_{t-1} \quad (10)$$

The first term of Eq. 10 is the core of our approach as it revises the group posterior distribution at time  $t - 1$ , also considering the states of the single individuals. In this way, the second term (the posterior at time  $t - 1$  of the MOT process) may be considered as a weight that mirrors the reliability of the individual states.

A convenient way to model distributions conditioned on multiple events is that of the Mixed-memory Markov Process (MMP) [62], that decomposes a structured conditioned distribution as a convex combination of pairwise conditioned distributions. This leads to:

$$p(X'^g_{t-1} | \mathbf{x}_{t-1}, z_{1:t-1}) \approx \alpha_1 p(X'^g_{t-1} | \mathbf{x}_{t-1}) + \alpha_2 p(X'^g_{t-1} | z_{1:t-1}), \quad (11)$$

where  $\alpha_1, \alpha_2 > 0$  and  $\alpha_1 + \alpha_2 = 1$ . We can now rewrite Eq. 10 as:

$$p(X'^g_{t-1} | z_{1:t-1}) \approx \alpha_1 \int p(\mathbf{x}_{t-1} | z_{1:t-1}) p(X'^g_{t-1} | \mathbf{x}_{t-1}) d\mathbf{x}_{t-1} + \quad (12)$$

$$\alpha_2 p(X'^g_{t-1} | z_{1:t-1}) \underbrace{\int p(\mathbf{x}_{t-1} | z_{1:t-1}) d\mathbf{x}_{t-1}}_{=1}. \quad (13)$$

At this point, it is easy to realize that  $p(X'^g_{t-1} | z_{1:t-1})$  becomes a combination of the natural group posterior and a marginalization of the *linking* probability  $p(X'^g_{t-1} | \mathbf{x}_{t-1})$ , that relates a group to individuals, weighted by the MOT posterior. In other words, the group posterior is revisited by injecting in a principled way the information on the single targets (the MOT posterior), conveyed selectively by  $p(X'^g_{t-1} | \mathbf{x}_{t-1})$ . An example will demonstrate the advantage of this formulation.

The linking probability  $p(X_{t-1}^g | \mathbf{x}_{t-1})$  is factorized as an MMP as follows:

$$p(X_{t-1}^g | \mathbf{x}_{t-1}) \approx \sum_{k=1}^K p(X_{t-1}^g | x_{t-1}^k) \beta^{k,g} \quad (14)$$

$$\approx \sum_{k=1}^K p(x_{t-1}^k | X_{t-1}^g) p(X_{t-1}^g) \beta^{k,g} \quad (15)$$

where  $\beta^{k,g} > 0 \forall k, g$  and  $\sum_k \beta^{k,g} = 1$ . Each term of the sum in Eq. 14 represents the posterior probability that the  $g$ th group  $X_{t-1}^g$  contains the  $k$ th target  $x_{t-1}^k$ .

In Eq. 15 the posterior is modeled employing the Bayes rule, where  $p(x_{t-1}^k | X_{t-1}^g)$  defines the *linking* likelihood that each single individual state  $x_{t-1}^k$  is a subpart of  $X_{t-1}^g$ . Hence, we define a probability model based on three components: 1) appearance similarity, 2) dynamics consistency, and 3) group membership. The appearance similarity is encoded by the Bhattacharyya distance between the HSV histograms of the two entities:  $d_{\text{HSV}}(X_{t-1}^g, x_{t-1}^k)$ . The dynamics consistency rewards the person state whose motion component is similar to that of the group. In practice, we check the 2D displacement on the floor by calculating  $d_{\text{dir}}(X_{t-1}^g, x_{t-1}^k) = |1 - |\text{dir}(X_{t-1}^g) - \text{dir}(x_{t-1}^k)||/\pi|$ , where  $\text{dir}(\cdot)$  gives the direction (an angle) of the person or group. Finally, the group membership evaluates the spatial proximity of the person state and of the group state:

$$d_{\text{mbr}}(X_{t-1}^g, x_{t-1}^k) = \begin{cases} 1 & \text{if } x_{t-1}^k \in X_{t-1}^g \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

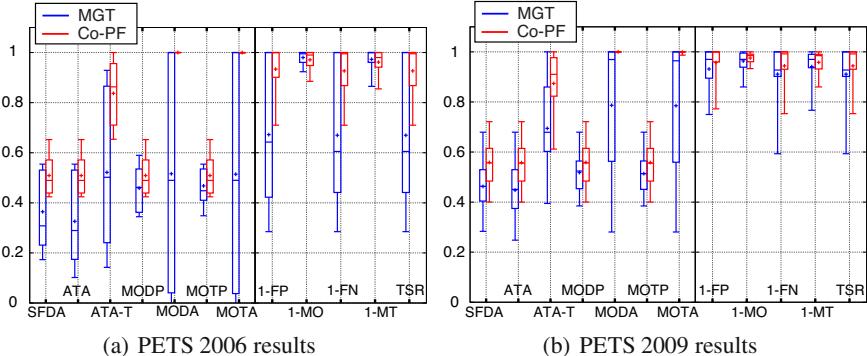
where the membership operator  $\in$  controls if the  $k$ th person position is inside the  $g$ th group ellipse. Therefore,  $p(x_{t-1}^k | X_{t-1}^g) = d_{\text{HSV}}(X_{t-1}^g, x_{t-1}^k) \cdot d_{\text{dir}}(X_{t-1}^g, x_{t-1}^k) \cdot d_{\text{mbr}}(X_{t-1}^g, x_{t-1}^k)$ . The coefficients  $\beta^{k,g}$  express a linking preference that an object belongs to a group, and are left here as uniform, *i.e.*,  $\beta^{k,g} = 1/G$ .

Finally, the prior  $p(X_{t-1}^g)$  discards the biggest and the smallest group hypotheses, rejecting the particles in which the size of the group is below a threshold  $\tau_b$  or above a threshold  $\tau_a$ .

An example that explains the strength of our formulation can be represented by an intra-group occlusion in the  $g$ th group at time  $t-1$ , which is very common due to the dynamical nature of a group of moving people. Let  $x_{t-1}^k$  a target of the group  $X_{t-1}^g$  that vanishes as occluded by the remaining individuals of that group. The group posterior  $p(X_{t-1}^g | z_{1:t-1})$  will not be very high, for the limits of the visual, rigid, group representation. However, the MOT process, dealing with single objects and managing their occlusions, will “understand” the fact that  $x_{t-1}^k$  is occluded, producing a high  $p(x_{t-1}^k | z_{1:t-1})$ . This probability value will flow through  $p(X_{t-1}^g | \mathbf{x}_{t-1})$ , which is high because, even if occluded, the position and the velocity of  $x_{t-1}^k$  are correctly estimated by the MOT process, and will give a high linking likelihood. This will reinforce the final estimation of the hybrid posterior for  $X_{t-1}^g$ , thus permitting to estimate the subsequent group sample set in a more correct way.

## 4.2 Experimental Results

Our approach has been evaluated on synthetic data and publicly available datasets (PETS 2006<sup>5</sup> and PETS 2009<sup>6</sup>). We carried out a comparative analysis with respect to the MGT (without the proposed collaboration stage), highlighting that Co-PF is more able to deal with intra- and inter-group occlusion. Other approaches have not been taken into account because of the lack of: 1) on-line available code for any of the approaches in the state of the art 2) a shared, labelled, dataset.



**Fig. 12** Statistics on the synthetic test set.

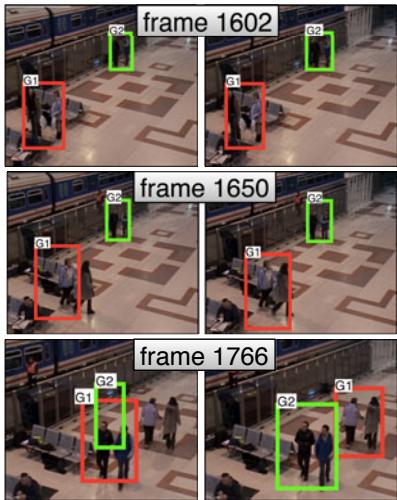
The simulations on the synthetic test set are carried out in order to build statistics on ground-truthed sequences. The test set is built to emulate the scenarios in PETS dataset by using the same background and the same calibration data. Each sequence contains static images of people walking in the environment and forming groups. We artificially create a set of 26 sequences (13 for each dataset), choosing two different points of view in order to deal with variably scaled people: the first camera is closed to the people, while the second one is far. The number of people and the number of groups vary in different sequences from 3 to 20 and from 1 to 5, respectively. The number of person in a group varies from 2 to 6. The parameters are set as follows:  $\sigma_\mu = 0.05$ ,  $\sigma_\lambda = 0.05$ ,  $\sigma_\theta = \pi/40$ , 256 bin are used for the HSV histogram,  $\alpha_1 = \alpha_2 = 0.5$ ,  $\tau_b = 0.5$ ,  $\tau_a = 2.5$ .

A comparison has been done between the Co-PF with  $N = 50$  and  $N_g = 50$  (the number of particles for each group) and MGT with  $N'_g \approx N_g + N \cdot \frac{K^2}{G^2 \cdot C}$ , where  $C = 5$  has been empirically chosen,  $K$  and  $G$  are the number of people and groups, respectively. In this way, the computational burden of the two methods is similar. To evaluate the performance on the synthetic test set, we adopt the follow measures: Average Tracking Accuracy (ATA), Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), False Positive (FP), Multiple Objects

<sup>5</sup> <http://www.cvg.rdg.ac.uk/PETS2006/>

<sup>6</sup> <http://www.cvg.rdg.ac.uk/PETS2009/>

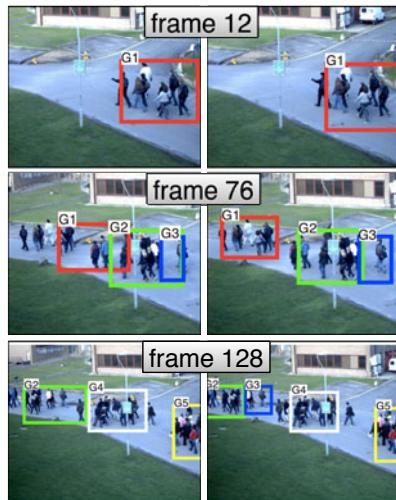
PETS 2006: sequence S6-T3-H  
view 4, frames 1600-1830



(a) MGT

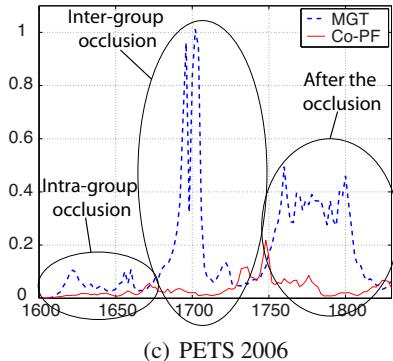
(b) Co-PF

PETS 2009: sequence S1-L1  
Time13-45 view 1, frames 1-221

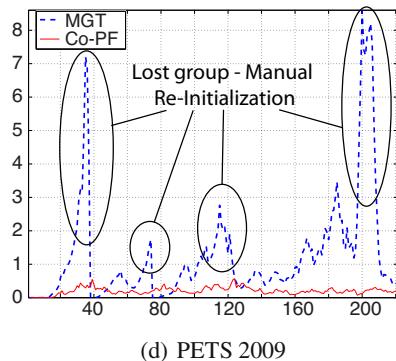


(a) MGT

(b) Co-PF



(c) PETS 2006



(d) PETS 2009

**Fig. 13** Comparison of MGT (first and third column) and Co-PF (second and fourth column) on PETS 2006 and PETS 2009. The second row compares the PF uncertainty [42] in the two experiments.

(MO), False Negative (FN), Tracking Success Rate (TSR) and so on (further details in [29, 68]). For each measure, a *boxplot* representation is given [29], where the box is defined by the 1st quartile, median, and the 3rd quartile; the extremities outside the box are the smallest and largest value, and the "+" is the mean value. The comparison (Fig. 12(a) and Fig. 12(b)) shows that in the PETS2006 synthetic dataset our Co-PF strongly outperforms the MGT in terms of all the measures. Even though the PETS2009 sequences are slightly harder, Co-PF often succeeds where MGT fails, yielding to higher performances.

Moreover, we perform the test on portions of the PETS datasets, using the same settings. We consider sequences where the groups were not subjected to splits or merges, in order to stress the capability of tracking group entities with intra- and inter-group occlusions. Initialization of groups has been done by fitting the  $\mu^g$  and  $\Sigma^g$  to the projections of the individuals new entries on the ground plane. If lost, a group is manually reinitialized. Note that group split and merge is not modeled in this probabilistic framework. It is an hard problem that has to be handled as future work. We show here two representative examples. In real scenarios, MGT is not able to deal completely with the intra- and inter-group dynamics (Fig. 13(a)). On the other hand, Co-PF exploits the MOT results, enriching the posterior knowledge given by the MGT (Fig. 13(b)).

To give further support to our Co-PF, we evaluate the uncertainty of the particle filters [42]. Fig. 13(c) depicts that the MGT uncertainty is peaked when an intra- and inter-group occlusion occurs. After the occlusion the uncertainty is high because the track is erroneously lost (two tracks on a single group). Fig. 13(d) shows a similar behavior of Fig. 13(c), highlighting that the MGT loses the tracks several times.

## 5 Human-Environment Interactions: Interest Maps

The contribution of this Section is a visualization application of the SFV-based framework (see Sec. 2.3), called the *Interest Map*, published in [16]. Since the part of a scene that intersects the SVF is the area observed by the SVF owner, we collect this information for each subject, over a given time interval. This permits to infer which are the parts of the scene that are more observed, thus, where human attention is more plausibly focused. The gathered information is visualized as a suitable color map, in which “hot” colors represent the areas more frequently observed, and the opposite for the “cold” areas. This kind of inference is highly informative at least for two reasons. The first one is diagnostics, in the sense that it gives us the possibility to observe which are the areas of a scene that arouse more attention by the people. The other one is prognostics, since it enables us to devise the parts of the scene that are naturally more observed, because for example they are the natural front of view in a narrow transit area, or for other reasons that this method cannot guess (the interest map only highlights the tangible effects). In a museum, for example, one may be interested in understanding which artworks receive more attention, or in a market which areas attract more the customers. In a prognostic sense, it may be useful for marketing purposes, such as for example decide where to hang an advertisement.

Section 5.1 describes how a 3D map of the monitored environment is created. Since an accurate head pose estimation is not always possible, for example, because of low resolution, an alternative way to describe the pose is the motion orientation of a person (described in Section 5.2). The interest map generation process is presented in Section 5.3 and experiments, reported in Section 5.4, show qualitative results of the interest map given the monitored environment.

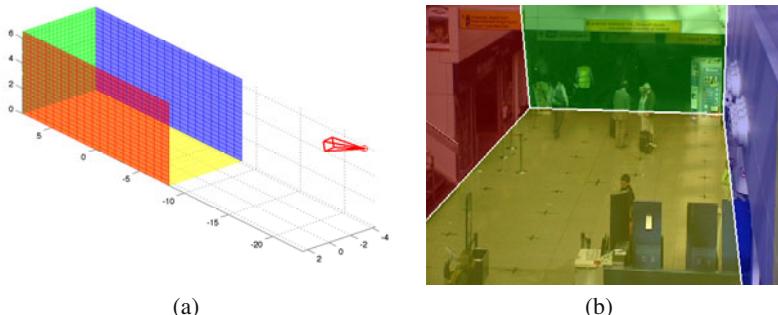
### 5.1 3D Map Estimation

Let's suppose that the camera monitoring the area is fully calibrated, *i.e.*, both internal parameters and camera position and orientation are known. For convenience, the world reference system is fixed on the ground floor, with the  $z$ -axis pointing upwards. This permits to obtain the 3D coordinates of a point in the image if the elevation from the ground floor is known. In fact, if  $P$  is the camera projection matrix and  $\mathbf{M} = (M_x, M_y, M_z)$  the coordinates of a 3D point, the projection of  $\mathbf{M}$  through  $P$  is given by two equations:

$$u = \frac{\mathbf{p}_1^T \mathbf{M}}{\mathbf{p}_3^T \mathbf{M}}, \quad v = \frac{\mathbf{p}_2^T \mathbf{M}}{\mathbf{p}_3^T \mathbf{M}}, \quad \text{with } P = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix}. \quad (17)$$

$(u, v)$  are the coordinates of the image point. Thus, knowing  $(u, v)$  and  $M_z$  it is possible to estimate the position of  $\mathbf{M}$  in the 3D space.

Therefore, a rough reconstruction of the area, made up of the principal planes present in the scene, can be carried out (see an example in Figure 14). These planes represent the areas of the scene that are interesting to analyze, and the Interest Map will possibly be estimated on them only. Nevertheless, in principle, a more detailed 3D map can be considered, which can be obtained in two ways: first, a manual modeling of the scenario through Computer-Aided Design technologies, and, second and more interestingly, using Structure-from-Motion algorithms [70, 21, 7].



**Fig. 14** 3D reconstruction of the area being monitored. (a) The 3D map of the principal planes. The red cone represents the camera. (b) The planes are projected through the camera and superimposed on one image.

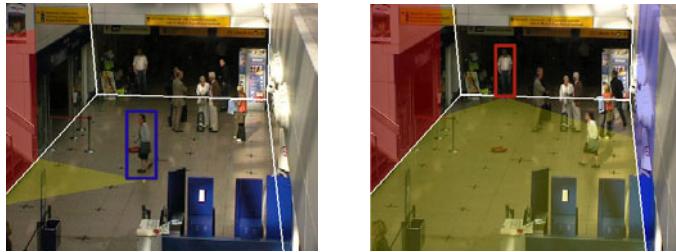
## 5.2 Motion Orientation Estimation

The tracking algorithm of Sec. 2.1 provides the position of each person  $i$  present in the scene at a certain moment  $t$ . When it is not possible to apply an head pose estimation algorithm, a simpler pose estimation method is required. In this case, the motion vector can provide the orientation  $\theta_{i,t}$  where people are watching. This is a reasonable assumption in a dynamic scenario, because when people walk, they usually look at the direction where they are suppose to go, and therefore they tend to keep the head lined up with the body most of the time. We calculate the angle between the motion direction, given by the tracker, and the camera orientation, using the camera calibration parameters. Therefore, this approach can be seen as an alternative, yet simpler solution to the method proposed in Sec. 2.2 that could be useful in specific cases. Moreover, the two approaches could be fused in order to rule out the disadvantages and for making the pose estimation more robust when dealing with both static and moving people.

## 5.3 Interest Map Generation

Once we have estimated the ground floor position and orientation of each individual  $(x_{i,t}, y_{i,t}, \theta_{i,t})$ , we instantiate a SVF for each person. The SVF  $\mathcal{D}_{i,t}$  represents the portion of 3D space seen by the  $i$ -th subject and it is constrained to the main planes of the scene described in Sec. 5.1. A full volumetric reasoning could be considered too, but this would capture other kinds of information, such as people interactions.

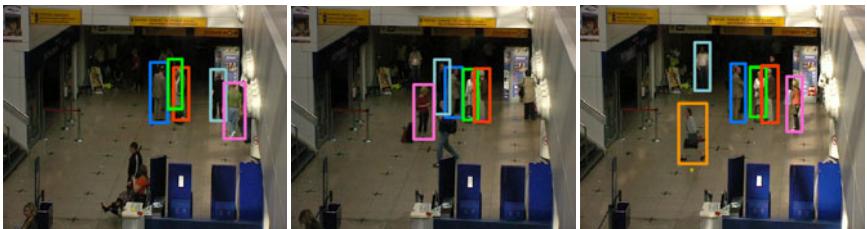
Each SVF  $\mathcal{D}_{i,t}$  at current time is projected on each scene plane. This is equivalent to estimate the vertices of  $\mathcal{D}_{i,t}$  lying on each plane, project these vertices onto the image and select those pixels that lie inside the convex hull of the projected vertices. In this way, the selected pixels represent the projection of each SVF in the image plane. Two examples of the projected SVF are shown in Figure 15. The projections of the SVFs of all the subjects present at the current time-step are then accumulated in a instantaneous map  $M_t$  (2D matrix of the same size of the camera frames). We define the *interest map* as the accumulation over time of these instantaneous maps, *i.e.*,  $IM = \sum_{t=1}^T M_t$ . Note that the interest map  $IM$  can be computed also in a time window (sum from  $T - \tau$  to  $T$ , where  $\tau$  is the size of time window) when the sequences are very long, like in real scenarios. The contributions provided by all tracked people in the sequence, or a set of sequences, are conveyed in the same interest map. Using a similar procedure, a subjective interest map (one independent map for each subject) could easily be computed, but here we restrict the analysis to the interest map for all the subjects. Note that the values of the interest maps vary in the range  $[0, K \cdot \tau]$  where  $K$  is the number of total tracks and  $\tau$  is the chosen size of the time window.



**Fig. 15** Two examples of projection of the SVF on the scene’s main planes. The 3D map permits to suitably model the interactions of the SVF with the scene.

#### 5.4 Experimental Results

We perform some tests over the publicly available PETS 2007 sequence sets<sup>7</sup>, aiming at showing the expressiveness of our framework on widely known and used datasets. Two sequences are taken into account for the experimental validation, both belong to the S07 dataset depicting an airport area monitoring. The first sequence is captured by Camera 2, the second one is captured by Camera 4.

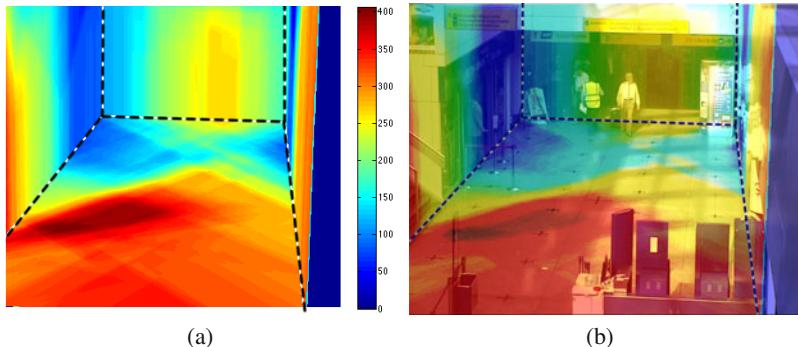


**Fig. 16** Some frames of the sequence from camera 2. The bounding boxes highlight the tracking results.

In Figure 16 we show the tracking results (bounding-boxes) of three frames of the first considered sequence. Totally, 1 minute of activity has been monitored, tracking continuously 5 people at a time in average. The resulting Interest Map is depicted in Figure 17 superimposed as transparency mask to a frame of the video. The “hottest” area is the one closest to the camera, in the direction of the stairs on the left. Indeed, in the sequence, many people cross that area from right to left. Another interesting area is at the end of the corridor, while the entrance on the left end has never been watched. Finally, the other people detected throughout the sequence are on the right end, going North.

For the second sequence, captured by Camera 4, 1 minute has been monitored, tracking 4 people at a time in average. The SVF analysis produces the results shown in Figure 18. In this case, the most seen areas of the parallelepiped (the 3D map)

<sup>7</sup> <http://www.cvg.rdg.ac.uk/PETS2007/>

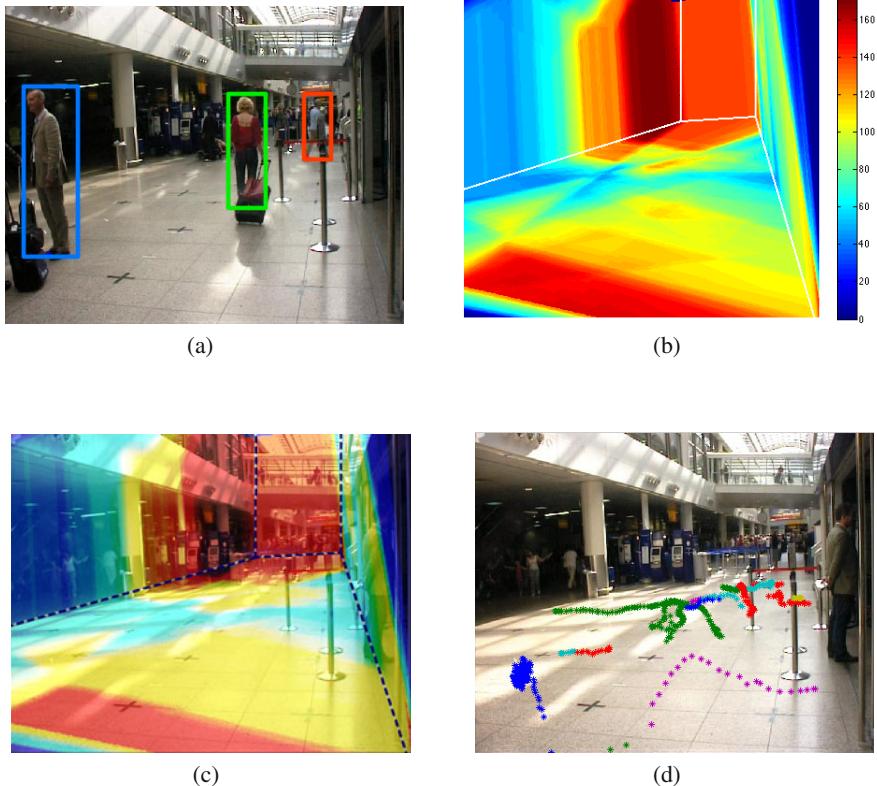


**Fig. 17** (a) Interest Map for S07 sequence from camera 2 (“hot” colors represent the areas more frequently observed, and the opposite for the “cold” areas). (b) The same Interest Map superimposed on one frame of the sequence.

are two (Fig. 18(b)-(c)). The left corner of the parallelepiped is “hot” because most of the people go towards that region of the corridor. The second “hot” area is the area in front of the camera, due to a person loitering there most of the time interval considered. As a comparison we plot together (Fig. 18(d)) the tracking results. This representation is less meaningful from the point of view of the analysis of the people attention. Our information visualization technique is instead intuitive and it captures in a very simple and richer way where people attention is focused.

## 6 Conclusions

This chapter presents a set of techniques for managing groups and group activities, taking into account social psychology aspects that define the human’s acting. In this way, we moved from the un-personal objective point of view of the video camera capturing people as they were abstract entities, to a new perspective where a subjective viewpoint of the individuals is taken into account. In this scenario, the position of a person is linked with the relative location (and orientation) he/she has with respect to all the other subjects in the scene: actually, what is sensed by the single persons helps more strongly in assessing what he/she is doing with respect to the sterile point of view of a video camera mounted on a wall. This chapter is one of the early example of how computer vision and social signaling may collaborate for a new level of the video surveillance research.



**Fig. 18** (a) One frame of sequence S07 camera 4, with the tracking results. (b) The obtained Interest Map (“hot” colors represent the areas more frequently observed, and the opposite for the “cold” areas). (c) The same Interest Map superimposed on one frame. (d) The tracks of the 4 people estimated throughout the sequence displayed in the same frame.

## References

- [1] Ba, S.O., Odobez, J.-M.: A Study on Visual Focus of Attention Recognition from Head Pose in a Meeting Room. In: Renals, S., Bengio, S., Fiscus, J.G. (eds.) MLMI 2006. LNCS, vol. 4299, pp. 75–87. Springer, Heidelberg (2006)
- [2] Bazzani, L., Cristani, M., Murino, V.: Collaborative particle filters for group tracking. In: IEEE International Conference on Image Processing (2010)
- [3] Bazzani, L., Cristani, M., Perina, A., Farenzena, M., Murino, V.: Multiple-shot person re-identification by hpe signature. In: 20th International Conference on Pattern Recognition (ICPR), pp. 1413–1416 (August 2010)
- [4] Bazzani, L., Tosato, D., Cristani, M., Farenzena, M., Pagetti, G., Menegaz, G., Murino, V.: Social interactions by visual focus of attention in a three-dimensional environment. In: Expert Systems (2011) (in Print)
- [5] Benfold, B., Reid, I.: Guiding visual surveillance by tracking human attention. In: Proceedings of the 20th British Machine Vision Conference (September 2009)

- [6] Breiman, L., Friedman, J.H., Olshen, R., Stone, C.J.: Classification and Regression Trees. *Ann. Math. Statist.* 19, 293–325 (1984)
- [7] Brown, M., Lowe, D.G.: Unsupervised 3d object recognition and reconstruction in un-ordered datasets. In: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling, pp. 56–63. IEEE Computer Society, Washington, DC (2005)
- [8] Cheng, D.S., Cristani, M., Stoppa, M., Bazzani, L., Murino, V.: Custom pictorial structures for re-identification. In: British Machine Vision Conference, BMVC (2011) (in Print)
- [9] Choudhury, T., Pentland, A.: The sociometer: A wearable device for understanding hu-man networks. In: CSCW - Workshop on ACCUCE (2002)
- [10] Cohn, J.F.: Foundations of human computing: facial expression and emotion. In: Pro-ceedings of the 8th International Conference on Multimodal Interfaces, pp. 233–238. ACM (2006)
- [11] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Pro-ceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 886–893 (2005)
- [12] Doucet, A., de Freitas, N., Gordon, N. (eds.): Sequential Monte Carlo methods in prac-tice. Springer (2001)
- [13] Ekman, P.: Facial expression and emotion. *American Psychologist* 48(4), 384 (1993)
- [14] Farenzena, M., Bazzani, L., Perina, A., Murino, V., Cristani, M.: Person re-identification by symmetry-driven accumulation of local features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2360–2367 (June 2010)
- [15] Farenzena, M., Tavano, A., Bazzani, L., Tosato, D., Pagetti, G., Menegaz, G., Murino, V., Cristani, M.: Social interaction by visual focus of attention in a three-dimensional environment. In: Workshop on Pattern Recognition and Artificial Intelligence for Hu-man Behavior Analysis at AI\*IA (2009)
- [16] Farenzena, M., Bazzani, L., Murino, V., Cristani, M.: Towards a Subject-Centered Anal-yysis for Automated Video Surveillance. In: Foggia, P., Sansone, C., Vento, M. (eds.) ICIAP 2009. LNCS, vol. 5716, pp. 481–489. Springer, Heidelberg (2009)
- [17] Freeman, L.: Social networks and the structure experiment. In: Research Methods in Social Network Analysis, pp. 11–40 (1989)
- [18] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55(1), 119–139 (1997)
- [19] Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: A statistical view of boosting. *The Annals of Statistics* 28(2), 337–374 (2000)
- [20] Gennari, G., Hager, G.D.: Probabilistic data association methods in visual tracking of groups. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)
- [21] Gherardi, R., Farenzena, M., Fusillo, A.: Improving the efficiency of hierarchical structure-and-motion. In: IEEE Conference on Computer Vision and Pattern Recog-nition, pp. 1594–1600 (June 2010)
- [22] Hall, E.T.: The hidden dimension, vol. 6. Doubleday, New York (1966)
- [23] Hongeng, S., Nevatia, R.: Large-scale event detection using semi-hidden markov mod-els. In: IEEE International Conference on Computer Vision, vol. 2 (2003)
- [24] Isard, M., Blake, A.: Condensation: Conditional density propagation for visual tracking. *International Journal of Computer Vision* 29, 5–28 (1998)
- [25] Isard, M., MacCormick, J.: BraMBLe: a bayesian multiple-blob tracker. In: Int. Conf Computer Vision, vol. 2, pp. 34–41 (2001)

- [26] Jabarin, B., Wu, J., Vertegaal, R., Grigorov, L.: Establishing remote conversations through eye contact with physical awareness proxies. In: CHI 2003 Extended Abstracts (2003)
- [27] Julier, S., Uhlmann, J.: A new extension of the kalman filter to nonlinear systems. In: Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Orlando, FL (1997)
- [28] Kalman, R.E.: A new approach to linear filtering and prediction problems. Tran. of the ASME Journal of Basic Engineering (82), 35–45 (1960)
- [29] Kasturi, R., Goldgof, D., Soundararajan, P., Manohar, V., Garofolo, J., Bowers, R., Boonstra, M., Korzhova, V., Zhang, J.: Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. IEEE Transactions on Pattern Analysis and Machine Intelligence, 319–336 (2009)
- [30] Knapp, M.L., Hall, J.A.: Nonverbal communication in human interaction. Wadsworth Pub. Co. (2009)
- [31] Lablack, A., Djeraba, C.: Analysis of human behaviour in front of a target scene. In: IEEE International Conference on Pattern Recognition, pp. 1–4 (2008)
- [32] Lan, T., Wang, Y., Yang, W., Mori, G.: Beyond actions: Discriminative models for contextual group activities. In: Advances in Neural Information Processing Systems, NIPS (2010)
- [33] Langton, S.H.R., Watt, R.J., Bruce, V.: Do the eyes have it? cues to the direction of social attention. Trends in Cognitive Neuroscience 4(2), 50–58 (2000)
- [34] Lanz, O., Brunelli, R., Chippindale, P., Voit, M., Stiefelhagen, R.: Extracting Interaction Cues: Focus of Attention, Body Pose, and Gestures, pp. 87–93. Springer (2009)
- [35] Lanz, O.: Approximate bayesian multibody tracking. IEEE Trans. Pattern Anal. Mach. Intell. 28, 1436–1449 (2006)
- [36] Lao, Y., Zheng, F.: Tracking a group of highly correlated targets. In: IEEE International Conference on Image Processing (2009)
- [37] Li, S.Z., Zhu, L., Zhang, Z., Blake, A., Zhang, H., Shum, H.-Y.: Statistical Learning of Multi-view Face Detection. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 67–81. Springer, Heidelberg (2002)
- [38] Lin, W.C., Liu, Y.: A lattice-based mrf model for dynamic near-regular texture tracking. IEEE Trans. Pattern Anal. Mach. Intell. 29(5), 777–792 (2007)
- [39] Lin, W., Sun, M.-T., Poovendran, R., Zhang, Z.: Group event detection with a varying number of group members for video surveillance. IEEE Transactions on Circuits and Systems for Video Technology 20(8), 1057–1067 (2010)
- [40] Liu, X., Krahnstoever, N., Ting, Y., Tu, P.: What are customers looking at? Advanced Video and Signal Based Surveillance, 405–410 (2007)
- [41] Maggio, E., Piccardo, E., Regazzoni, C., Cavallaro, A.: Particle phd filtering for multi-target visual tracking. In: IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 1101–1104 (2007)
- [42] Maggio, E., Smeraldi, F., Cavallaro, A.: Combining colour and orientation for adaptive particle filter-based tracking. In: British Machine Vision Conference (2005)
- [43] Marques, J.S., Jorge, P.M., Abrantes, A.J., Lemos, J.M.: Tracking groups of pedestrians in video sequences. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop, vol. 9, pp. 101–101 (June 2003)
- [44] Matsumoto, Y., Ogasawara, T., Zelinsky, A.: Behavior recognition based on head-pose and gaze direction measurement. In: Proc. Int'l Conf. Intelligent Robots and Systems, vol. 4, pp. 2127–2132 (2002)
- [45] Mauthner, T., Donoser, M., Bischof, H.: Robust tracking of spatial related components. In: IEEE International Conference on Pattern Recognition, pp. 1–4 (December 2008)

- [46] Mckenna, S.J., Jabri, S., Duric, Z., Wechsler, H., Rosenfeld, A.: Tracking groups of people. Computer Vision and Image Understanding (2000)
- [47] Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation in computer vision: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 31, 607–626 (2009)
- [48] Ni, B., Yan, S., Kassim, A.A.: Recognizing human group activities with localized causalities. In: CVPR 2009, pp. 1470–1477 (2009)
- [49] Otsuka, K., Yamato, J., Takemae, Y., Murase, H.: Quantifying interpersonal influence in face-to-face conversations based on visual attention patterns. In: Proceedings of the Conference on Human Factors in Computing Systems, pp. 1175–1180. ACM, New York (2006)
- [50] Paisitkriangkrai, S., Shen, C.H., Zhang, J.: Performance evaluation of local features in human classification and detection. *Computer Vision, Institution of Engineering and Technology* 2(4), 236–246 (2008)
- [51] Pan, P., Schonfeld, D.: Dynamic proposal variance and optimal particle allocation in particle filtering for video tracking. *IEEE Transactions on Circuits and Systems for Video Technology* 18(9), 1268–1279 (2008)
- [52] Panero, J., Zelnik, M.: Human dimension & interior space: a source book of design reference standards. Whitney Library of Design (1979)
- [53] Park, S., Trivedi, M.M.: Multi-person interaction and activity analysis: a synergistic track- and body-level analysis framework. *Mach. Vision Appl.* 18, 151–166 (2007)
- [54] Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: modeling social behavior for multi-target tracking. In: Proc. 12th International Conference on Computer Vision, Kyoto, Japan (2009)
- [55] Pentland, A., Pentland, S.: Honest signals: how they shape our world. The MIT Press (2008)
- [56] Pentland, A.: Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Trans. Pattern Anal. Mach. Intell.* 22, 107–119 (2000)
- [57] Preparata, F.P., Shamos, M.I.: Computational geometry: an introduction. Springer (1985)
- [58] Psathas, G.: Conversation analysis: The study of talk-in-interaction. Sage Publications, Inc. (1995)
- [59] Richmond, V.P., McCroskey, J.C., Payne, S.K.: Nonverbal behavior in interpersonal relations. Allyn and Bacon (2000)
- [60] Robertson, N., Reid, I.: Estimating Gaze Direction from Low-Resolution Faces in Video (2006)
- [61] Rummel, R.J.: Understanding conflict and war. Sage Publications (1981)
- [62] Saul, L.K., Jordan, M.I.: Mixed memory markov models: Decomposing complex stochastic processes as mixtures of simpler ones. *Machine Learning* 37(1), 75–87 (1999)
- [63] Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.* 37, 297–336 (1999)
- [64] Schefflen, A.E.: The significance of posture in communication systems. *Communication Theory*, 293 (2007)
- [65] Scherer, K.R.: Personality markers in speech. Cambridge Univ. Press (1979)
- [66] Scovanner, P., Tappen, M.F.: Learning pedestrian dynamics from the real world. In: IEEE International Conference on Computer Vision, pp. 381–388 (2009)
- [67] Smith, K., Ba, S., Odobez, J., Gatica-Perez, D.: Tracking the visual focus of attention for a varying number of wandering people. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(7), 1–18 (2008)

- [68] Smith, K., Gatica-Perez, D., Odobezi, J., Ba, S.: Evaluating multi-object tracking. In: IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 36–43 (2005)
- [69] Smith, P., Shah, M., da Vitoria Lobo, N.: Determining driver visual attention with one camera. *IEEE Transactions on Intelligent Transportation Systems* 4(4), 205–218 (2003)
- [70] Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: ACM Transactions on Graphics, vol. 25, pp. 835–846. ACM (2006)
- [71] Stiefelhagen, R., Bowers, R., Fiscus, J. (eds.): Multimodal Technologies for Perception of Humans: International Evaluation Workshops on Classification of Events, Activities and Relationships 2007. Springer, Heidelberg (2008)
- [72] Stiefelhagen, R., Garofolo, J. (eds.): Multimodal Technologies for Perception of Humans: First International Evaluation Workshop on Classification of Events, Activities and Relationships 2006. Springer, New York Inc. (2007)
- [73] Stiefelhagen, R., Yang, J., Waibel, A.: Modeling focus of attention for meeting indexing based on multiple cues. *IEEE Transactions on Neural Networks* 13, 928–938 (2002)
- [74] Stiefelhagen, R., Finke, M., Yang, J., Waibel, A.: From gaze to focus of attention. *Visual Information and Information Systems*, 761–768 (1999)
- [75] Tosato, D., Farenzena, M., Spera, M., Murino, V., Cristani, M.: Multi-Class Classification on Riemannian Manifolds for Video Surveillance. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 378–391. Springer, Heidelberg (2010)
- [76] Tuzel, O., Porikli, F., Meer, P.: Region Covariance: A Fast Descriptor for Detection and Classification. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 589–600. Springer, Heidelberg (2006)
- [77] Tuzel, O., Porikli, F., Meer, P.: Pedestrian detection via classification on riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.* 30, 1713–1727 (2008)
- [78] Vaswani, N., Chowdhury, A.R., Chellappa, R.: Activity recognition using the dynamics of the configuration of interacting objects. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 633–640 (2003)
- [79] Vinciarelli, A., Pantic, M., Bourlard, H.: Social Signal Processing: Survey of an emerging domain. *Image and Vision Computing Journal* 27(12), 1743–1759 (2009)
- [80] Vinciarelli, A., Pantic, M., Bourlard, H., Pentland, A.: Social signals, their function, and automatic analysis: a survey. In: Proceedings of the 10th International Conference on Multimodal Interfaces, pp. 61–68. ACM, New York (2008)
- [81] Viola, M., Jones, M.J., Viola, P.: Fast multi-view face detection. In: Proc. of Computer Vision and Pattern Recognition, Citeseer (2003)
- [82] Voit, M., Stiefelhagen, R.: Deducing the visual focus of attention from head pose estimation in dynamic multi-view meeting scenarios. In: Proceedings of the 10th International Conference on Multimodal Interfaces, ICMI 2008, pp. 173–180. ACM, New York (2008)
- [83] Waibel, A., Schultz, T., Bett, M., Denecke, M., Malkin, R., Rogina, I., Stiefelhagen, R.: SMaRT: the Smart Meeting Room task at ISL. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 752–755 (2003)
- [84] Wang, X., Ma, X., Grimson, W.E.L.: Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Trans. Pattern Anal. Mach. Intell.* 31, 539–555 (2009)
- [85] Wang, Y.-D., Wu, J.-K., Kassim, A.A., Huang, W.-M.: Tracking a variable number of human groups in video using probability hypothesis density. In: IEEE International Conference on Pattern Recognition (2006)

- [86] Warner, R.M., Sugarman, D.B.: Attributions of personality based on physical appearance, speech, and handwriting. *Journal of Personality and Social Psychology* 50(4), 792 (1986)
- [87] Whittaker, S., Frohlich, D., Daly-Jones, O.: Informal workplace communication: what is it like and how might we support it? In: CHI 1994, p. 208 (1994)
- [88] Wu, B., Nevatia, R.: Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection. In: Proceedings of the International Conference of Computer Vision and Pattern Recognition (2008)
- [89] Wu, B., Nevatia, R.: Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. *Internation Journal of Computer Vision* 82(2) (April 2009)
- [90] Wu, B., Ai, H., Huang, C., Lao, S.: Fast rotation invariant multi-view face detection based on real adaboost. In: Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition, FGR 2004, pp. 79–84. IEEE Computer Society, Washington, DC (2004)
- [91] Zheng, W., Gong, S., Xiang, T.: Associating groups of people. In: Proceedings of the British Machine Vision Conference (2009)

# **Part IV**

# **Systems**

# Video Analytics for Business Intelligence

Asaad Hakeem, Himaanshu Gupta, Atul Kanaujia, Tae Eun Choe,  
Kiran Gunda, Andrew Scanlon, Li Yu, Zhong Zhang, Peter Venetianer,  
Zeeshan Rasheed, and Niels Haering

**Abstract.** This chapter focuses on various algorithms and techniques in video analytics that can be applied to the business intelligence domain. The goal is to provide the reader with an overview of the state of the art approaches in the field of video analytics, and also describe the various applications where these technologies can be applied. We describe existing algorithms for extraction and processing of target and scene information, multi-sensor cross camera analysis, inferencing of simple, complex and abnormal video events, data mining, image search and retrieval, intuitive UIs for efficient customer experience, and text summarization of visual data. We have also presented the evaluation results of each of these technology components using in-house and other publicly available datasets.

**Keywords:** Automated video analysis, context, business intelligence, search, retrieval, 3D object modeling, calibration, geo-registration, pixel-to-text, image and video understanding, complex event detection.

## 1 Introduction

Automated video analysis has made rapid progress in the security domain. The motivation is simple. Labor intensive security is expensive - an army of security guards subtracts from the bottom line. Cameras help reduce cost, but who is watching the video feeds? Labor based security is also ineffective – dedicated and motivated personnel cannot stay focused on surveillance and monitoring tasks for more than 20 minutes [42]. Automated video analysis offers an ever-vigilant alternative for these mundane monitoring tasks. Initially focused on moving objects, automated analysis largely ignored all other contextual information. Today’s state-of-the-art automated

---

Asaad Hakeem · Himaanshu Gupta · Atul Kanaujia · Tae Eun Choe ·  
Kiran Gunda · Andrew Scanlon · Li Yu · Zhong Zhang · Peter Venetianer ·  
Zeeshan Rasheed · Niels Haering  
ObjectVideo, Inc., 11600 Sunrise Valley Dr. Reston, VA 20191, USA  
e-mail: {ahakeem, hgupta, akanaujia, techoe, kgunda, ascanlon, liyu,  
zhang, pvenetianer, zrasheed, nhaering}@objectvideo.com

video analysis systems integrate and learn from all available sensors and from any and all prior information about the scene, targets and expected behaviors to further improve effectiveness.

This enables automated video analysis solutions for new domains, including Business Intelligence (BI). In some ways BI applications are less challenging than security applications. For instance, controlled lighting conditions, better knowledge of the target types of interest, and the predominant use of static cameras in BI applications simplify background modeling, target detection and classification. On the other hand BI applications tend to require better segmentation and tracking methods to handle the higher target densities in typical indoor applications.

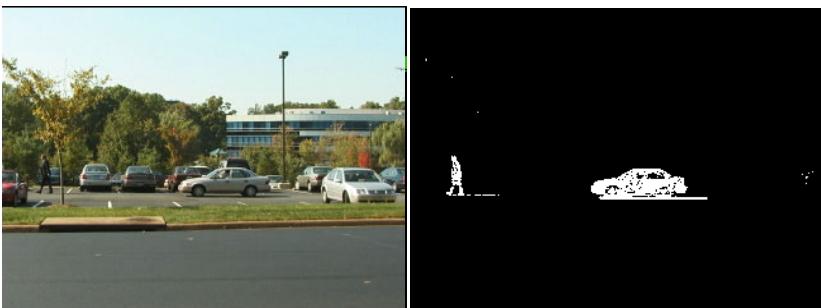
As in other domains, good automated video analysis solutions for BI

- Combine video sensors and non-video sensors to provide robust solutions. For instance, point-of-sales applications correlate information from bar code readers, light barriers, scales and Radio Frequency ID (RFID) tags with visual information to detect theft and some forms of “sweet hearting” (cashier moves merchandise past a scanner without recording the item).
- Exploit scene knowledge. For instance, good person and event counting applications take advantage of existing choke points, camera locations and calibration.
- Make use of known information about target types, IDs and expected movement patterns. For instance, a shopping cart model can help improve customer count accuracy, vehicle and person models can improve behavior analysis, the expectation that people and children wait in checkout lines together with their shopping carts and baskets can improve checkout line length estimation, customers’ likely movement direction coupled with camera placement can improve the quality of customer face libraries.

The rest of the chapter is divided into the following sections. Section 2 discusses target and scene analysis where we describe target detection, segmentation, and tracking in Section 2.1, target property extraction for humans and vehicles in Sections 2.2 and 2.3 respectively, target best shot estimation in Section 2.4, and concluding with scene content extraction in Section 2.5. Section 3 details multi-sensor cross camera analysis where we describe target geo-registration in Section 3.1 and cross camera tracking in Section 3.2. We further detail video event detection in Section 4, where we describe atomic event detection in Section 4.1, complex event detection in Section 4.2, and abnormal event detection in Section 4.3. Section 5 describes the video content management and retrieval techniques, where we discuss the video data management system in Section 5.1, pixel-to-text report generation in Section 5.2, and video event and query retrieval in Section 5.3. Section 6 depicts the user interfaces and customer experience, where the site wide geo-browser based awareness is detailed in Section 6.1, activity specification framework in Section 6.2, interactive search in Section 6.3, query UI on the smart phone in Section 6.4, face browser in Section 6.5, and event/person counting suite in 6.6. We conclude the chapter with some applications of the presented methods in Section 7.

## 2 Target and Scene Analysis

The first step in performing video analytics for business intelligence involves target and scene analysis. Target analysis provides detailed information about the target movement patterns, appearance and other characteristics which can be used for enhanced activity recognition and target identification. Scene analysis provides information about the content of the scene, which can further be used to accurately model the target to scene element interaction which is necessary for advanced activity recognition. The key goal behind the various algorithms and techniques discussed in this section is to automatically learn various target and scene attributes from video for later use in advanced activity recognition and intelligent high-level inference.



**Fig. 1** Moving object detection via background subtraction.

### 2.1 Target Detection, Segmentation and Tracking

Target detection is the task of grouping foreground pixels (corresponding to a given object in the scene) into distinct, labeled entities in the current image. We utilized background modeling techniques to calculate an initial pixel-level indication of foreground and background areas (see Figure 1). Changed pixels are those that are inconsistent with the background model. Traditional connected components using 4- or 8-connectedness are used to group pixels into distinct labeled objects in a given pixel image. We used an improved computational approach based on Boult's Quasi-connected-components algorithm [1].

Target tracking is the task of temporally associating target detections, maintaining measurements on a target, and treating it as a single entity throughout the time it is visible in the sensor. Tracking algorithms are central to any video analysis system deployed for business intelligence. Automated tracking typically takes two steps: first, targets from the current image are matched with objects that we are already tracking, and then we filter and predict the location of tracked objects for matching during the following frame. Matching is complicated by the fact that objects can become occluded, and potential one-to-many or many-to-one matches can make corresponding current targets to old targets somewhat ambiguous. There are two major categories of filters extensively used for the visual tracking: Kalman

filters [2,4,5] and particle filters [3,6]. We have developed specific algorithmic approaches to handle multiple target tracking and occlusion management.

Given the target detections, as well as calibrated cameras, we can further refine the target segmentations which can be very useful for applications such as person counting. First, a target prediction module is used to create the predicted human target model for each tracked target. A Kalman filter is used to estimate the 3D target footprint location. The corresponding 2D convex model is computed based on the camera calibration. Next, these predicted targets are matched with the foreground blobs detected in the current frame. Next, human targets that have matching blobs are updated using these matching blobs as new observations. This updating combines the current observation and the prediction. An important problem is how to weigh these two against each other? The system uses two internal states, the target motion and the target stability states to determine the weighting. The target motion state reflects the moving status of a target and can be represented by a 3 state finite state automaton (FSA): “*Moving*” state: the centroid of the target is changing; “*Stopped*” state: the image centroid of the target is stable, but motion is detected within the target (internal motion); “*Stationary*” state: the target is completely motionless. The target stability state reflects the confidence of the matching between the prediction and observation and can be represented by a 4 state FSA: “*Appearing*” state: the human visibility ratio or the observed human recall ratio is increasing. This represents that the target is entering the camera view or coming out from behind some occluding background object.; “*Disappearing*” state: the human visibility ratio or the observed human recall ratio is decreasing.. It indicates that the target is leaving the camera view or becoming occluded by a background object; “*Occlusion*” state: multiple targets match to the same blobs. This indicates that the target is occluding or occluded by other targets; “*Stable*” state: consistent matching between observation and prediction.

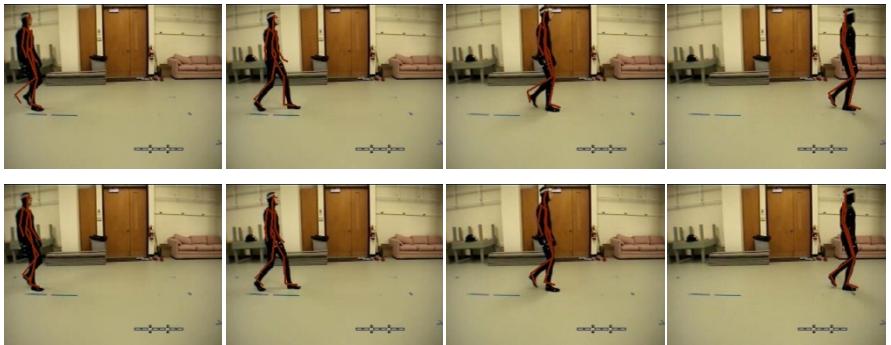
Furthermore, the “Merge/split targets” module corrects target segmentation errors by splitting up blobs that are too big and merging the ones that are too small. A target will be split into two if both of the two new targets satisfy the new target criteria and the matching score between the blob and the targets improves significantly. For simplicity, the system never splits a target into more than two pieces in a single step. But big blobs will continue to be split into smaller pieces in subsequent frames. Two targets with high *overlap* value are candidates for merging. They are merged if the merge increases the matching score. To prevent multiple people in occlusion from merging together, both targets must be in moving motion state for a certain duration. This is based on the observation that the likelihood of a moving human target consistently be fully occluded by another moving human target is very small.

## 2.2 Target Property Extraction for Humans

Behavior analysis is limited by what we know about the moving objects. For instance, information about an object’s broad classification and location enables us to analyze the interaction of the observed objects with image regions. Even this level of an analysis can be operationally relevant for perimeter and access control,

etc. In this section we discuss how the extraction and modeling of target attributes vastly increases our vocabulary to analyze detailed actions.

We model the human target using a 3D pose, which is estimated using a combined framework of discriminative and generative methods. The *discriminative framework* for human pose estimation is composed of the following key components: (1) person localization; (2) feature extraction and; (3) learning of mappings from image descriptor space to 3D joint angles space. Firstly, Human targets are localized using background subtraction described in Section 2.1. Secondly, the features are extracted from the masked image regions as fixed dimensional image descriptors. The image descriptors compactly encode the shape and appearance of the human targets in the image and are fed to the learned predictor models to directly infer the 3D joint angles. Thirdly, for modeling multi-valued 2D-to-3D mappings, we use multiple “experts” (regression functions) [9,10] to map similar 2D visual inputs to distant 3D outputs. We employ the *Bayesian Mixture of Experts (BME)* framework to learn the relation between the 2D image descriptors and the human pose.



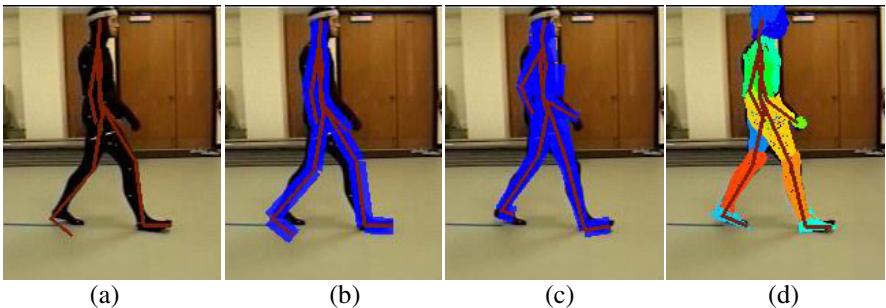
**Fig. 2** Pose refinement with skeleton model. Top row: The initial pose estimates obtained from the discriminative framework. Bottom row: The corresponding refined poses obtained from top-down framework based on MCMC.

The pose estimated from bottom-up framework is further refined using *generative algorithms*. We adopted the Markov chain Monte Carlo (MCMC) framework [11,12,13] for pose estimation. Pose estimation is formulated as the problem of estimating the state of a system. Denoting  $X$  as the vector of model parameters and  $Y$  as the image observations, pose estimation is formulated as a Bayesian inference for estimating the posterior distribution. An MCMC based approach approximates the posterior using a proposal distribution  $q(X' | X_{t-1})$ . The samples are drawn  $\{X^1, \dots, X^i, \dots\}$  from the proposal distribution using a Markov chain based on the Metropolis-Hastings algorithm [11]. At the  $t$ -th iteration, a candidate  $X'$  is

sampled from a proposal distribution  $q(X^t | X_{t-1})$  and accepted as the new state  $X_t$ , with a probability  $a(X_{t-1} \rightarrow X^t)$  where:

$$a(X_{t-1} \rightarrow X^t) = \min \left\{ 1, \frac{p(X^t | Y)q(X_{t-1} | X^t)}{p(X_{t-1} | Y)q(X^t | X_{t-1})} \right\}.$$

In the data-driven MCMC paradigm, image observations are used for generating additional proposals, and enhancing the convergence of the MCMC. The joint angles having very low variances for all the Gaussian components are ignored. The sampling from mixture of Gaussian is a 2-stage process: (1) Select a Gaussian mixture component based on the gate weights; (2) Sample from the selected Gaussian distribution component. Figure 2 shows the initial estimates and the refined results for the MCMC optimization.



**Fig. 3** Coarse-to-fine shape fitting. (a) Initial pose estimate. (b) Initial cylindrical model. (c) Refined cylindrical model. (d) Detailed pose and shape fitting using CAESAR model.

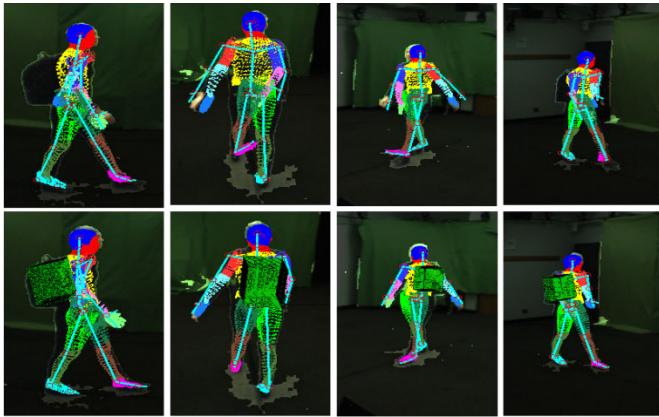
The estimated 3D pose is used to initialize search in the parameter space of human shapes for detailed 3D shape estimation of the human target. We model 3D shape of humans using polygonal 3D mesh surfaces skinned to an underlying skeleton. We assume that the 3D mesh surface undergoes deformation only under the influence of the skeleton attached to it. Shape of human body can vary both due to anthropometry and the pose of the target. Anthropometric variability is modeled by the learned 3D shape prior for humans from the registered CAESAR data[14]. The shape deformation due to pose is obtained by first skinning the 3D mesh to the skeleton and transforming the vertices under the influence of associated skeletal joints. We use Linear Blend Skinning (LBS) for efficient non-rigid deformation of skin as a function of underlying skeleton. LBS is achieved by associating the vertices to  $n$  nearest joints. The transformation is computed as weighted sum of the transformation where weights are computed as inverse distance from the  $n$  joints. Although rich in terms of representation, global 3D human shape representation cannot model 3D shapes with disproportionately sized body parts. In order to support rich set of human shapes we use a combined local part-based and global

optimization scheme that first searches in the local subspace of human body parts to match the observation, followed by constraining the whole shape using global human shape model. Fitting body parts independently causes discontinuities along the joints and generates unrealistic shapes. Constraining the shape to lie in the global shape space therefore ensures it to be a valid shape. For linear PCA based shape models, this is efficiently done by ensuring the PCA coefficients of the shape (when projected to the subspace) to lie within a range of variance. Our algorithm does alternate search in the parameter space of 3D human pose ( $P$ ) and shape ( $S$ ) to simultaneously refine the pose and fit detailed 3D shape to the observation. The search is performed using Data Driven MCMC with metropolis-hastings method wherein the proposal map does not use the predictive distribution obtained from bottom-up methods but rather is modeled as Gaussian distribution conditioned on the current state  $q(x'|x_{t-1})$  where  $x_{t-1} = \{P_{t-1}, S_{t-1}\}$ . The likelihood distribution is modeled as symmetrical chamfer distance map to match the 2D projection of the model to the observed image silhouettes. For optimizing the 3D pose, we use the current 3D shape to search in the parameter space of articulated human pose. Plausible 3D shapes are sampled from the Gaussian distributions that the PCA based subspace represents for each of the body parts. The search is performed by alternately fitting the 3D pose first, followed by optimization of the shape parameters of the individual body parts. At each iteration the 3D shape of human body is constrained using global shape model to ensure a valid shape, as shown in Figure 3.

### 2.2.1 Concealed Object and Backpack Detection

Given the automated 3D pose estimation of the human model, our system supports automatic estimation of size of an accessory bag carried or concealed by humans. Backpacks are modeled as a trapezoidal shape and are rigidly attached to the torso so the translation and orientation of the backpack can be directly computed using that of torso. The two parameters of the trapezoid (thickness and orientation of non-perpendicular face) are iteratively estimated during the 3D shape fitting. The shape of the accessory is initialized using an average thickness of the torso. The framework functions as a generative classifier to identify whether a human is carrying backpack or not. Improvement in the likelihood of fit for the model with the attached accessory implies presence of backpack. This is illustrated in the Figure 4 whereby use of model with an attached accessory improved the likelihood of fit. Generative models provide a flexible classification framework for identifying whether a human target is carrying a backpack or not. We fit the two human 3D shape models to the observed 3D data: model without a backpack and a model with backpack attached to the torso. The model with the best likelihood (or minimum matching cost to observation) indicates whether the target is carrying a backpack or not.

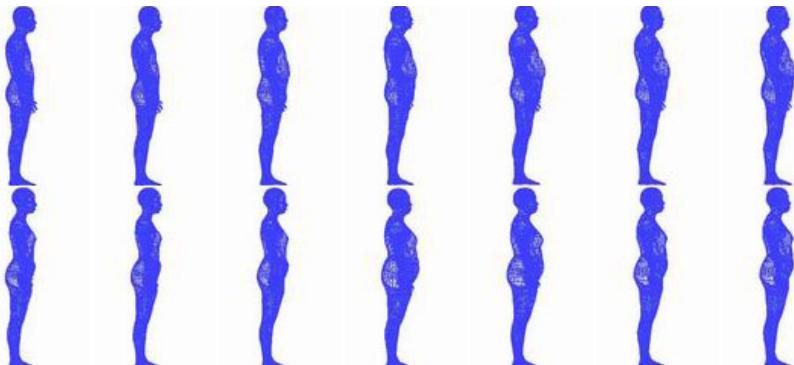
The algorithm can be applied to detect any accessory with the assumption that difference in the silhouettes of the person with and without the accessory is noticeable. It also assumes regular fitted clothes worn by the human subjects as loose clothing may cause unnatural change in the shape of the silhouettes that may be falsely fitted by the accessory model.



**Fig. 4** Cost of fitting a human 3D shape model with and without backpack. This is a generative classification framework for recognizing whether a subject is carrying a backpack or not. In the above example containing human carrying a backpack, human shape model with backpack achieves lower matching cost.

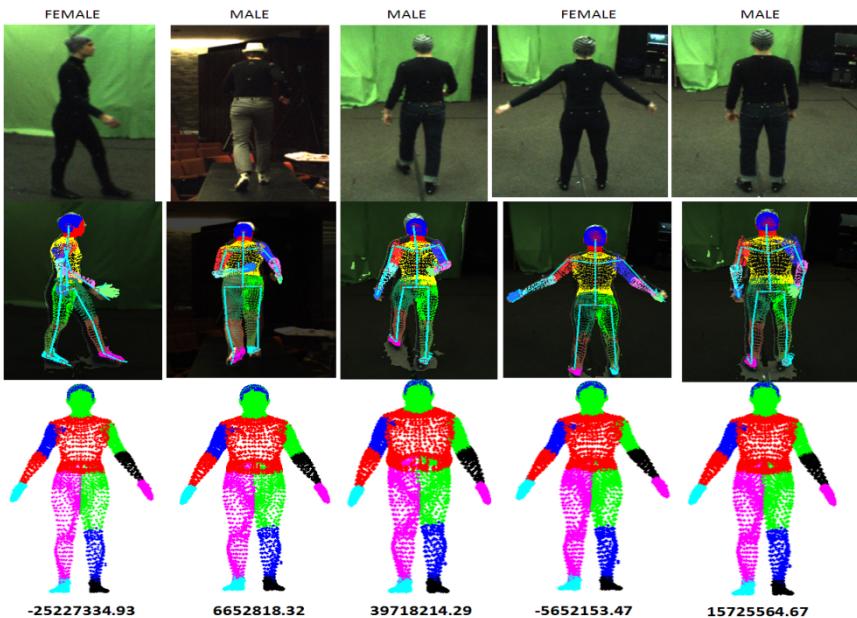
### 2.2.2 Gender and Age Classification

The estimated 3D shape of the human target can be used for inference of a variety of human attributes such as gender and age. Demographic features such as gender, ethnicity, physical attributes such as height, weight and facial and body appearance can be inferred either by computing local spatial statistics of different regions of the fitted 3D shape or by determining global anthropometric variations that characterizes these features. In order to directly infer various attributes from the fitted shape, we first learn representative 3D shape models of human targets. We use the registered CAESAR [14] scan data for multiple persons, containing equal number of vertices and polygons. We used Principal Component Analysis (PCA) to compute global shape subspace to capture whole body shape variability. Figure 5 show the shape deformation due to gender.



**Fig. 5** Shape deformation with increasing age, from 20 years (furthest left) to 75 years (furthest right). Top: male. Bottom: female.

Using age information from the CAESAR dataset, we also modeled the shape transformation across the person's age. Using *k-nearest neighbor* regression algorithm (KNN), the system learns exemplary body shapes at varying ages (from 20 to 75 years old) for male and female, as shown in Figure 5. For gender-based shape analysis, we used linear discriminant analysis (LDA) to find the shape feature projection that best separates the shape profile of the two gender classes. Figure 6 shows the projection of the 3D shape vertices on the LDA vector. Also shown are the projection coefficients of the female and male body 3D shapes on the LDA vector. Here the threshold for gender classification is set to 0 and negative LDA coefficients denote female shapes.



**Fig. 6** Gender classification using Linear Discriminant Analysis (LDA). Top row shows one of the images, second and third row shows the 3D shape fitting results and the projected LDA vector respectively.

Our 3D human pose estimation framework was evaluated on six different sequences from CMU motion capture database. Table 1 shows the human pose estimation accuracy as the average joint angle error in degrees per frame for different sequences.

**Table 1** Pose estimation results for various activities using the our framework

Activity	No. of training exemplars	No. of testing exemplars	Average joint angle error (in degrees)
<b>Anger walk</b>	1553	1564	12.8161
<b>Heavysteet walk</b>	1508	1526	8.9617
<b>Jauntily walk</b>	1549	1586	14.8653
<b>Stealthily walk</b>	1569	1576	13.2827
<b>Boxing</b>	2780	1362	5.7725
<b>Normal Walk</b>	1020	350	4.7287

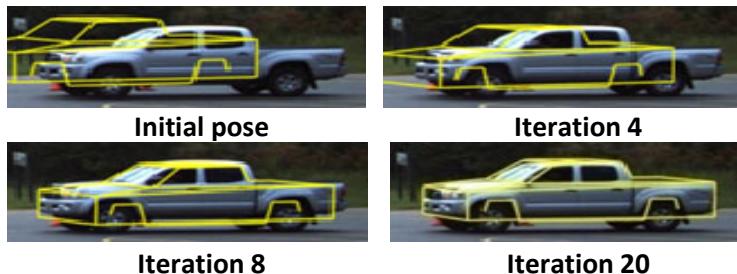
The extraction of this information enables numerous new capabilities, such as the detection of a) person-to-person interactions: greeting, hugging, fighting, signaling, etc. b) person-to-object interactions: taking objects from a store shelf, holding a phone, carrying a bag or backpack, etc., c) gait, gestures and attributes: person calling for help, aggressive poses, age, gender, race, build, etc.

These capabilities facilitate applications and operationally relevant information for market research, retail, health care, education and many other domains.

### 2.3 Target Property Extraction for Vehicles

Vehicles are an important class of objects. They are by far the most popular mode of transport, they can cause traffic congestion, they can carry dangerous cargo in large quantities, and people share a lot of common space with them every day. For all these reasons many applications derive significant additional value from vehicle attributes, beyond their location. In order to extract target properties from vehicles we have developed a vehicle fitting and classification method using a 3D vehicle model. Detected vehicles are classified according to class (passenger car, SUV or pickup truck) using a 3D model-based fitting method. Approaches exist to fit and classify vehicles [18, 19, 20], however, few of them are real-time methods. The novel real-time vehicle classification algorithm uses simplified 3D models learned containing 16 vertices and 28 facets that best approximate the mean shape of multiple 3D models in each class, as shown in Figure 7.

**Fig. 7** Simplified models of the three vehicle classes



**Fig. 8** Iterative vehicle fitting.

To classify a target, the above vehicle models are fitted to the silhouette edges, obtained by refining the results of foreground detection using morphological filtering and Gaussian smoothing. The initial vehicle location and heading is estimated by back-projecting the detected target onto the road according to a calibrated projection matrix. The simple calibration procedure involves manually aligning 3D vehicle models to a few exemplar views and produces a projection matrix that encodes both the camera orientation and ground plane location. Figure 8 illustrates several iterations of the vehicle model fitting procedure. To determine the class of a detected vehicle, the vehicle is fitted with all three models and the model with lowest matching error is selected as the vehicle class. Model fitting is based on minimizing the Chamfer distance between the silhouette edges and projected model edges [18]. To improve robustness, boundary edge pixels are classified into four groups based on edge orientation. A corresponding Chamfer distance map is constructed using pixels in each group. At each iteration of the fitting procedure, the vehicle model is projected onto the image in the current estimated pose, and the matching error is computed as the sum of the Chamfer distances along each projected edge from the map corresponding to its orientation. The estimated pose is updated for the next iteration based on the derivative of the matching error.

The vehicle fitting algorithm described above has been quantitatively evaluated on three video sequences of public roads with a total of 611 vehicles in varying orientations including frontal, profile and turning. Table 2 provides the confusion matrix for vehicle classification based on manually observed ground truth. The results indicate that the proposed model-based fitting method is capable of classifying the vehicle class with a precision of 95%.

**Table 2** Confusion Matrix of Vehicle Classification

<b>Detection Ground truth</b>	<b>Passenger Car</b>	<b>SUV</b>	<b>Pick-up Truck</b>
<b>Passenger Car</b>	414	9	5
<b>SUV</b>	1	103	2
<b>Pick-up Truck</b>	11	4	62

Knowledge of vehicle types facilitates applications such as traffic monitoring (vehicle density by vehicle type, vehicle speed by vehicle type, etc.), traffic control (are truck-free zone rules being violated, toll-calculation, etc.), security (is the driver of a parked car on the phone, requesting help, or abandoning a vehicle?), person-vehicle-interactions (is a person getting into the driver or passenger side of a vehicle?, using the rear or front door?, is a person getting something out of the trunk of the car or did they open the hood ?, etc.). In these and other scenarios detailed object information enables many applications that require fine grained behavior analysis.

## 2.4 Best Shot Estimation

Target best shot is a high resolution frontal or profile face shot obtained once the target detection and tracking is performed. A high-resolution snapshot of a target can be of immense value for various purposes. The target best shots can be submitted to a face recognition software to compare against a watch-list of people and automatically alert if there is a hit. Also, once an intrusion has already occurred, a target best shot can be of great importance for forensic purposes. In order to obtain the best shot, we utilize the leader-follower architecture described conceptually in Figure 9. Leader-Follower uses one camera, called the ‘leader’, to provide an overall picture of the scene, and another camera, called the ‘follower’, to provide high-resolution images of targets of interest. Upon detecting a suspicious activity, the leader can order a follower to follow the target using a pan, tilt, and zoom (PTZ) camera. The follower receives a stream of location data about targets from the leader, filters it, and translates the stream into pan, tilt, and zoom signals for a robotic PTZ camera unit. The resulting system is one in which one camera detects threats, and the other PTZ camera obtains high-resolution images of the threatening targets.

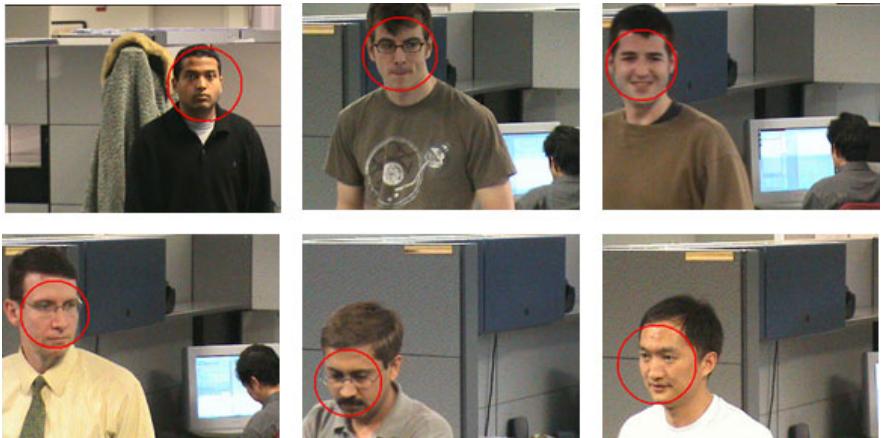


**Fig. 9** Leader-Follower architecture conceptual diagram.

The above system can be easily extended to include multiple leader and follower cameras in the framework. For instance, one may add multiple followers to a given leader. One may have multiple leaders commanding a single follower. Also, one may use different kinds of cameras for the leader or for the follower. For example, a normal, perspective camera or an omni-camera may be used as cameras for the leader. One could also use thermal, near-IR, color, black-and-white, fisheye, telephoto, zoom and other camera/lens combinations as the leader or follower camera.

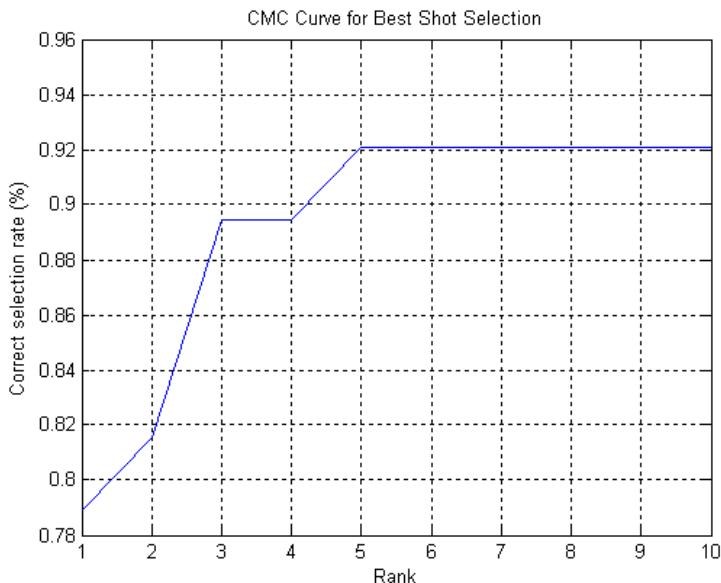
The high resolution tracked target in PTZ camera is further processed to detect faces using Viola-Jones [8], which involves building several boosted classifiers  $F_k$  with constantly increasing complexity and chaining them into a cascade with the simpler classifiers going first. During the detection stage, the current search window is analyzed subsequently by each of the  $F_k$  classifiers that may reject it or let it go through. That is,  $F_k$  ( $k=1..N$ ) are subsequently applied to the face candidate until it gets rejected by one of them or until it passes them all. Another advantage is that each of the stages need not be perfect; in fact, the stages are usually biased toward higher hit-rates rather than towards small false-alarm rates. By choosing the desired hit-rate and false-alarm rate at every stage and by choosing the number of stages accurately, it is possible to achieve very good detection performance.

Once a face has been successfully detected in the frame, the next step in the best shot selection algorithm is to determine the Best Shot Analysis (BSA) score, which is used to determine the quality of the target's face in the image. The best shot analysis module computes a weighted score based on the following metrics: Face size – large face region implies more pixels on the face; and Skin tone ratio – quality of the face shot is directly proportional to the percentage of skin-tone pixels in the face region. Figure 10 shows the best shot frames extracted by the best shot analysis module.



**Fig. 10** Best shot estimation Top: Frontal face detection, Bottom: Mild profile face detection.

In order to measure the effective of the best shot detection algorithm, we evaluated our system on a total on 38 indoor PTZ videos involving 19 unique subjects. The ground-truth best shot frames were selected based on inputs from three human observers. From the Cumulative Match Characteristics (CMC) plot shown in Figure 11, it can be inferred that if the ground-truth best shot frame was present in the set of selected best shots (with 92% probability), it was always among the top 5.

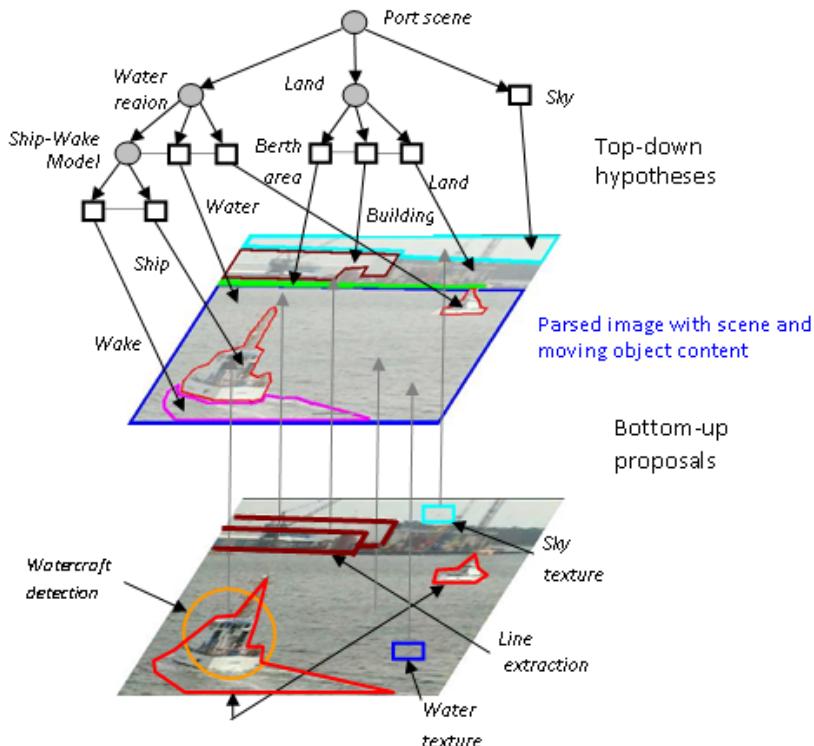


**Fig. 11** CMC curve of best shot selection algorithm.

## 2.5 Scene Content Extraction

Scene analysis provides information about the content of the scene, which can further be used to accurately model the target to scene element interaction which is necessary for advanced activity recognition. Scene content extraction is the process of classifying the imagery into scene elements and objects. We utilize a stochastic attribute image grammar [21] that serves as a unified methodology for analysis, extraction, and representation of the visual elements and structure of the scene, such as the roads, sky, vehicles, and humans. At the lowest level of the grammar graph are the image primitives such as image patches, lines or color blobs. Serving as basic cues, these primitives are combined to form larger objects and scene structure. The production rules realize the composition of the image elements with attributes. As illustrated in Figure 12, graphs are used to represent the grammars where the nodes represent the visual elements and the edges depict the rules defining the relations between the elements. Under the stochastic attribute image grammar methodology, the image content extraction is formulated as a graph parsing process to find a specific configuration produced by the grammar that best describes the image. The inference algorithm finds the best configuration by integrating bottom-up detections and top-down hypotheses. As illustrated in Figure 12, bottom-up detection includes classification of image patches as regions of sky, water, and watercraft, which generate data-driven candidates for the scene content. Top-down hypotheses, on the other hand, are driven by scene models and contextual relations represented by the attribute grammar.

A parsed image is an instance or configuration of the attribute graph grammar that aims to find a parse-graph that maximizes the posterior probability, under the Bayesian framework. The objective is to find the graph configuration that best represents the input image. The process is initiated with the bottom-up approach that generates candidate proposals by changing the labeling of scene elements based on local features. These proposals are used, in a stochastic manner, to activate the instantiation of production rules in the attribute grammar, which in turn generate top-down hypotheses. The top-down inference guides the search based on domain knowledge and contextual relations between objects and scene elements. These rules specify how attributes of elements are estimated and passed along the parse-graph through the use of constraint equations. Data-Driven Markov Chain Monte Carlo (DDMCMC) [22] is used to maximize the posterior probability for scene content inference.



**Fig. 12** Image content inference with bottom-up proposals and top-down hypotheses. Bottom-up detection, such as moving blobs and line extraction, are used to generate bottom-up proposals for image content. Top-down hypotheses are based on contextual relationships between image elements described in the attribute graph. The fusion of both approaches enables a more accurate and robust inference of image content.

For evaluation, a dataset of 90 different scenes is collected, of which 45 is used for training and the remaining for testing. These include maritime and urban scenes. The overall classification accuracy is 73.6%. Table 3 shows the confusion matrix at pixel level and Table 4 shows the breakdown of *recall* and *precision* for overall scene element classification. The performance is reasonable for *sky*, *water* and *road*, while there is some confusion between *land* and *road*, as well as *land* and *vegetation*, which have similar appearance. In future work, we plan to investigate the use of global feature models to improve the overall classification.

**Table 3** Scene Element Classification Confusion Matrix

	Road	Veg.	Water	Land	Sky
Road	0.781	0.086	0.026	0.098	0.009
Veg.	0.069	0.677	0.001	0.240	0.013
Water	0.126	0.008	0.824	0.018	0.023
Land	0.223	0.171	0.025	0.527	0.053
Sky	0.025	0.056	0.021	0.132	0.766

**Table 4** Overall Scene Element Classification Accuracy

Stages	Objective function	Classification Accuracy (%)
1	Bottom-up stage: max local likelihood	60.4
2	Local likelihood with affinity measure	67.5
3	Global optimization: Max. joint posterior	70.3

### 3 Multi-sensor Cross Camera Analysis

The second step in performing video analytics for business intelligence involves data fusion from multiple modalities such as RFID tags attached to items, GPS, video input from multiple heterogeneous EO and IR sensors, active PTZ camera best shot information, Geographic Information System (GIS) road and building information, as well as other sources of information. Fusion of multi-modal data provides an integrated view of information, essential in performing video analysis for business intelligence. We now describe the technical details for achieving multi-modal data fusion in the following sections.

#### 3.1 Target Geo-registration

Wide area situational awareness requires fusion and coordination among the multi-modal data analysis systems and sources. For that purpose, we perform automated multi-sensor geo-registration. Such a mapping not only enables critical

operational tasks that include data integration and inter-sensor tracking, but in case of a geodetic coordinate system, also provides metric information, such as size, distance, speed, and heading of the detected objects for high level inference. The following explains automatic camera geo-registration method.

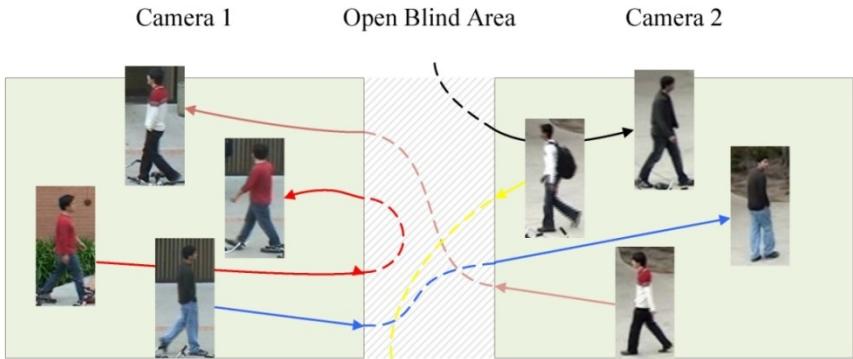
Assuming that the area where targets move in the scene is planar (a fair assumption for most business intelligence scenarios), the relationship between geodesic coordinates and image coordinates can be represented in terms of a homography (given by a  $3 \times 3$  matrix, say  $\mathbf{H}$ ). If  $\mathbf{p}$  is the projection of a geodesic point  $\mathbf{P}$  in the image plane, then  $\mathbf{p} = \mathbf{H}\mathbf{P}$ , where both  $\mathbf{p}$  and  $\mathbf{P}$  are in a homogeneous coordinate system. The homography  $\mathbf{H}$  can be obtained by solving a linear system obtained by identifying four or more corresponding points in the image and the world that lie on the ground-plane and do not form degenerate cases, such as, co-linearity.

Our approach exploits the geo-positioning information broadcast by mobile transponders. The transponders can be dedicated mobile units (boats, people, and vehicles) for periodic geo-registration of the system, or they can be part of the usual traffic that happens to broadcast this information. The examples of such traffic include ships and boats that broadcast AIS information (in some cases mandated by port authorities), and vehicles and people or items with RFID tags. The data fusion service receives the time-stamped geo-positional data from GPS/AIS data receivers along with the target metadata being streamed from individual sensors via video analysis services. The fusion sensor geo-registers each sensor by employing a RANSAC-based method to reject outliers and estimate the parameters of a model that optimally fits the dataset.

### **3.2 Cross Camera Tracking**

Wide area situational awareness requires coordination among the multi-sensor analysis systems, so as to perform multiple camera tracking. A successful system for multi-target tracking across multi-camera consists of two main parts: (1) intra-camera tracking, i.e., tracking multiple targets within a camera; (2) inter-camera association, i.e., "handover" of tracked targets from one camera to another. Although there have been significant improvements in intra-camera tracking, inter-camera track association when cameras have non-overlapping fields of view (FOVs) remains a less-explored topic.

An illustration for inter-camera association of multiple tracks is shown in Figure 13. Compared to intra-camera tracking, inter-camera association is more challenging because (1) the appearance of a target in different cameras may not be consistent due to different sensor characteristics, lighting conditions, and viewpoints; (2) the spatio-temporal information of tracked objects between cameras becomes less reliable. Besides, the open blind area significantly increases the complexity of the inter-camera track association problem.

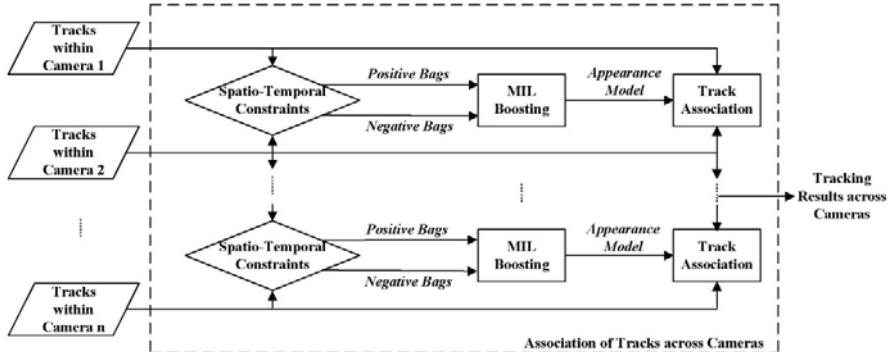


**Fig. 13** Illustration of inter-camera association between two non-overlapping cameras.

Associating multiple tracks in different cameras can be formulated as a correspondence problem. Given the observations of tracked targets, the goal is to find the associated pairs of tracks which maximize a joint linking probability, in which the key component is the affinity between tracks. For the affinity score, there are two main cues to be considered: the spatio-temporal information, and appearance relationships between two non-overlapping cameras. Compared to spatio-temporal information, the appearance cues are more reliable for distinguishing different targets especially in cases where FOVs are disjointed. Between a set of tracks among two non-overlapping cameras, the aim of the affinity model is to distinguish the tracks which belong to the same target from those which belong to different targets. Previous methods [24,26,27] mostly focused on learning the appearance models or mapping functions based on the correct matches, but no negative information is considered in their learning procedure.

Collecting positive and negative training samples on-line is difficult since no hand-labeled correspondence is available at runtime. However, by observing spatio-temporal constraints of tracks between two cameras, some potentially associated pairs of tracks and some impossible pairs are formed as “weakly labeled samples”. We propose to adopt the Multiple Instance Learning (MIL) [23,25,28] to accommodate the ambiguity of labeling during the model learning process. Then the learned discriminative appearance affinity model is combined with spatio-temporal information to compute the crucial affinities in the track association framework, achieving a robust inter-camera association system.

Our system contains three main components: the method of collecting online training samples, the discriminative appearance affinity model, and track association framework. The system block diagram of our method is shown in Figure 14. Negative samples are collected by extracting pairs of tracks in two cameras which overlap in time. It is based on the assumption that one object cannot appear in two non-overlapping cameras at the same time. Instead of labeling each sample, several potentially linked pairs of tracks constitute one positive “bag”, which is suitable for the Multiple Instance Learning (MIL) algorithm.



**Fig. 14** The block diagram of our system for associating multiple tracked targets from multiple non-overlapping cameras.

The learning of appearance affinity model is to determine whether two tracks from different cameras belong to the same target or not according to their appearance descriptors and similarity measurements. Instead of using only color information as in previous work, appearance descriptors consisting of the color histogram, the covariance matrix, and the HOG feature, are computed at multiple locations to increase the power of description. Similarity measurements based on those features among the training samples establish the feature pool. Once the training samples are collected in a time sliding window, a MIL boosting algorithm is applied to select discriminative features from this pool and their corresponding weighted coefficients, and combines them into a strong classifier in the same time sliding window so that the learned models are adapted to the current scenario. The prediction confidence output by this classifier is transformed to a probability space, which cooperates with other cues (e.g. spatial correspondence and time interval) to compute the affinity between tracks for association.

To perform track association across multiple cameras, we firstly focus on the track association between two cameras and then extend it to the case of multiple cameras. Assuming that the task of a single camera tracking has been already solved; there are  $m$  tracks in camera  $C^a$  and  $n$  tracks in camera  $C^b$  respectively. We define an extended correspondence matrix of size  $(2m+2n) \times (2m+2n)$  as follows:

$$H = \left[ \begin{array}{cc|cc} A_{m \times m} & B_{m \times n} & F_{m \times m} & -\infty_{m \times n} \\ D_{n \times m} & E_{n \times n} & -\infty_{n \times m} & G_{n \times n} \\ \hline J_{m \times m} & -\infty_{m \times n} & & 0_{(m+n) \times (m+n)} \\ -\infty_{n \times m} & K_{n \times n} & & \end{array} \right]$$

The components of each sub-matrix, i.e. the liking probability or affinity between two tracks, is defined as the product of three important cues (appearance, space, time). By applying the Hungarian algorithm [40] to  $\mathbf{H}$ , the optimal assignment of association is obtained efficiently. A summary of each sub-matrix in is given in Figure 15.

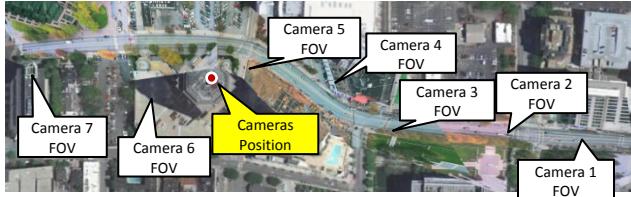
matrix	description	element
<b>A</b>	the target leaves and returns to $C^a$	$A_{ij} = -\infty$ if $i = j$
<b>B</b>	the target leaves $C^a$ and enters $C^b$	$B_{ij}$ is a full matrix
<b>D</b>	the target leaves $C^b$ and enters $C^a$	$D_{ij}$ is a full matrix
<b>E</b>	the target leaves and returns to $C^b$	$E_{ij} = -\infty$ if $i = j$
<b>F</b>	the target terminates in $C^a$	$F_{ij} = -\infty$ if $i \neq j$
<b>G</b>	the target terminates in $C^b$	$G_{ij} = -\infty$ if $i \neq j$
<b>J</b>	the target is initialized in $C^a$	$J_{ij} = -\infty$ if $i \neq j$
<b>K</b>	the target is initialized in $C^b$	$K_{ij} = -\infty$ if $i \neq j$

**Fig. 15** A short summary of the elements in each sub-matrix in  $\mathbf{H}$ , which models all possible situations between the tracks of two non-overlapping cameras.

We tested the system on multi-camera videos in the NGSIM Peachtree dataset, which contains 7 videos taken on top of a building (see Figure 16). We selected the first 5 minutes of video as the test data. These test videos look like a relatively easier example, since camera FOVs are overlapping each other and all video frames are time-synchronized with the same frame rate. However, it still contains multiple targets in a complex situation. The targets frequently stop at the traffic signal and vehicles and pedestrians are often occluded by buildings and trees. In Camera 5, 6, and 7, vehicles are heavily occluded by trees.



**Fig. 16** The snapshot of 6 out of 7 NGSIM videos (taken at Peachtree Dr, Atlanta, GA). The order of cameras is reversed considering the spatial arrangement of cameras.



**Fig. 17** The projected images of NGSIM Peachtree videos on the map. Homographies for each camera are manually annotated. All 7 cameras are located on top of a building.

For camera calibration, correspondences between points on each image and lat-long coordinates of a geo-browser (e.g. GoogleEarth) are manually annotated to estimate the image-to-ground homography. The projected images are shown in Figure 17. For this data set, time, location, and size are used as similarity measures.

For evaluation, 21 targets shown from all 7 cameras are randomly selected as groundtruth. Evaluation results across multiple cameras are shown in Table 5. In the table, *GT* indicates the number of groundtruth targets, and *Detection Recall* is the detection rate within a camera. Precision could not be calculated since not all targets are annotated as groundtruth. When a target is tracked more than 80% compared to the corresponding groundtruth, it is considered as *Mostly Tracked*. *Mostly Lost* is when the target is tracked less than 20%. *Partially Tracked* is in the between. *Frag* indicates the number of fragments of a target in a single camera and *X-Frag* is average number of fragments per target across multiple cameras.

**Table 5** Cross Camera Tracking For NGSIM Dataset

Video Number	GT	Detection Recall	Mostly Tracked	Partially Tracked	Mostly Lost	Frag	X-Frag
1	21	94%	85.7%	14.3%	0	2	
2	21	95%	95.2%	4.8%	0	0	
3	21	97%	90.5%	9.5%	0	0	
4	21	97%	95.2%	4.8%	0	0	
5	21	85%	61.9%	38.1%	0	0	
6	21	90%	85.7%	14.3%	0	0	
7	19	89%	78.9%	21.1%	0	0	
Average	<b>21</b>	<b>93%</b>	<b>84.7%</b>	<b>15.3%</b>	<b>0%</b>	<b>0.29</b>	<b>2.14</b>

## 4 Video Event Detection

The third step in performing video analytics for business intelligence involves automatically detecting user-specified events of interest and other abnormal events from

video. Once we have ascertained the location and time of these events, the system can provide useful actionable information and situation awareness for numerous applications. We now describe the technical details in the following sections.

## 4.1 Atomic Event Detection

For event detection in video, we leverage the existing state-of-the-art in extracting descriptive information about visual events, including semantic and contextual information, as well as, relationship between activities performed by different agents. We have developed rule-based algorithms for the detection of atomic actions from video. These algorithms take the image meta-data, i.e., detection, tracking and classification information of targets, as input. Some of the atomic actions that are automatically detected by our system are:

**Tripwire Crossing Detection:** Detects when the specified object moving in a specified direction crosses over a line (tripwire) drawn within the camera's field of view. For example, if the object is a person, the person's feet must cross over the line in the specified direction to trigger an alert. Tripwires can be uni-directional or bi-directional, and may indicate perimeter breaches.

**“Enters” Action Detection:** Detects when the specified object type enters an Area of Interest (AOI) such as a sensitive installation, from any direction within the camera's field of view.

**“Exits” Action Detection:** Detects when the specified object type exits an AOI from any direction within the camera's field of view.

**“Appears” Action Detection:** Detects when the specified object appears in an AOI without appearing within the camera's field of view previously. In other words, the first time the object appears within the field of view is when it appears in the AOI (for example, by walking through a doorway that exists inside the AOI).

**“Loiters” Action Detection:** Detects when the specified object stays inside of a designated AOI for the specified time within the camera's field of view.

**“Leave Behind” Action Detection:** Detects when an object has been left behind or inserted in the full view of a camera, or a designated AOI. For example, a Leave Behind rule will trigger an alert when a suspicious object is left on the ground, indicating an explosive device threat.

**“Take Away” Action Detection:** Detects when an object has been removed from the full view of a camera, or from a designated AOI. For example, a Take Away rule will trigger an alert when a picture is removed from a wall, indicating theft from an art gallery.

**Object Size Filters:** Filter out objects that are too large or too small to be objects of interest that can trigger alerts. For example, size filters can be used to distinguish between a passenger vehicle and a transport truck.

**Object Speed Filters:** Filter out objects that are too fast or too slow to be objects of interest that can trigger alerts. For example, speed filters can be used to distinguish between a normally moving passenger vehicle and a fast moving vehicle that might be carrying explosive devices and approaching a military installation.

**Object Class Filters:** Filter out objects based on its operator’s category of interest that can trigger alerts. For example, class filters can be used to trigger an alert when a large truck is approaching a check post, where as the usual traffic consists of sedans passing the check post.

## 4.2 Complex Event Detection

Once the system has detected simple atomic events as described in the previous section, the next step is to enable the system to detect complex events which are formulated as a set of multiple atomic events connected by spatial and temporal relationships. Complex events are detected using a *stochastic context-free grammar* (SCFG) [29]. The grammar is used to represent activities where production rules describe how activities can be broken down into sub-activities or atomic actions (described in Section 4.1). In the stochastic approach, a probability is attached to each production rule for the sequence of atomic actions, depicting the likelihood of a particular production rule to be utilized in the generation of the activity.

Given an input sequence of detected actions, the *Earley-Stolcke Parsing algorithm* [30] is used to parse the sequence based on the SCFG. The parsing algorithm is an iterative process for each input sequence of actions. Each iteration consists of three steps: *prediction*, *scanning*, and *completion*, in a manner similar to the top-down bottom-up approach of the image attribute grammar. The parsed result defines the event that occurred. To overcome the presence of unrelated concurrent actions, we use semantics and contextual information to identify *associations* between visual elements and actions based on factors such as target type, time, location, behavior and functional roles. This process removes unrelated data for event detection, thereby reducing computational complexity.

## 4.3 Detecting Abnormal Events

In the previous sections, we have described methods to detect events of interest specified by the user. The onus is on the user to specify exactly what they are looking for, formulate that information as a query and then feed it to the system. But, in a real-world environment, sometimes it is not apparent what constitutes an event of interest and the user might really care about detecting anything that is out of the ordinary. In order to detect abnormal events, we model normalcy using target property maps. Modeling of what constitutes “normal” in a given scenario requires deeper understanding and careful modeling of the underlying process, its variables and associated parameters. We argue that, over the period of its operation, an intelligent system should be able to learn a model of the scene parameters from the observables and be able to make inferences about the observables based on this model. The normalcy model of the scene behavior will be developed based

on the spatial, temporal and feature based representation of the scene context using multi-sensor data. Formally, a scene behavior can be seen as a stochastic process, which is described by a set of random variables  $\mathbf{z}$ , some of which are observable, say  $\mathbf{z}_o$ , and describe the observed signal whereas the other are hidden, say  $\mathbf{z}_h$ , and describe the events in the world that are causing the observed signal.

If  $\boldsymbol{\theta}$  is a set of the model parameters, then the problem is to learn the probability model  $p(\mathbf{z}_o | \boldsymbol{\theta}) = p(\mathbf{z}_o | \mathbf{z}_h, \boldsymbol{\theta})p(\mathbf{z}_h | \boldsymbol{\theta})$ , and to estimate the model parameters that maximize the likelihood given the measurements. Here, we present a framework to learn this scene model in the form of a multivariate non-parametric probability density function of spatial, temporal and feature based representation of scene context, background, and events/behavior. These features or the target properties include geo-spatial location of objects, transition times from one point to another, time/day/season, speed, and heading, size, sighting frequency, etc.

Using target property maps we learn more complex single and multi-sensor (site-wide) target behaviors. Each property map is learned separately as a 2-dimensional density of observations on the image lattice. The technology performs well in detecting single-property based anomalies, for example, detecting a person in the restricted area (using human sighting frequency map), detecting a speeding vehicle (using speed map), or detecting a vehicle on the wrong side of the road (using direction map). However, anomalies involving more than one property are not modeled by the existing method. For example, consider a path where the traffic density in one direction is higher than the density in other direction. If this fact is not captured by the sighting frequency map of that path, then if a higher traffic density is observed in the wrong direction, it will not register as an anomaly in the current system. Detection of such an anomaly requires joint probability density function of both sighting frequency and heading. Thus, detailed modeling of normal behavior in the scene requires joint modeling of target properties and scene context.

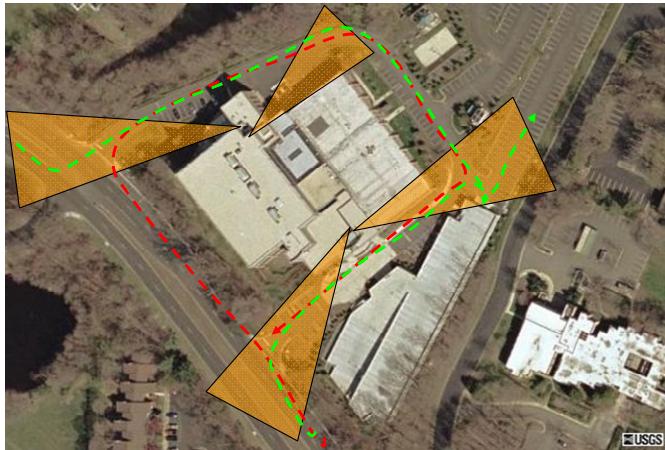
We extend the model by using a joint probability model that was learnt in an unsupervised fashion by observing the features during the course of operation of the system and obtaining Parzen window estimates from these observations. The model encodes the probabilistic nature of the behavior of moving objects in the scene, and different properties of the scene can be directly inferred from it, for example, the exit and entry locations, areas of occlusion, traffic patterns, preferable paths, and correlation among different observations. When the targets in the field of view do not exhibit the expected behavioral pattern according to the learned model, it results in a lower probability for the observed features, and an alarm for suspicious/anomalous behavior is generated. The proposed model is generative in nature and hence can be used to generate likely behaviors by using importance sampling techniques, such as Markov Chain Monte Carlo methods. The model can also be used as a prediction tool that can predict the next link of the event chain as the event unfolds. To decide whether a complex chain of events constitutes an anomaly to the learned model, the predicted behavior can be matched with the observed one.

The knowledge of target's common entry and exit locations and their paths provides a symbolic representation of target movement patterns. It also provides an efficient means for compressing the trajectory data for logging purposes and semantic scene analysis. Furthermore, it enables indexing and retrieval of targets by their origin or destination and natural language event reporting.



**Fig. 18** Normal path detection for vehicles in a parking lot. Green represents the entry regions, red represents the exit region and blue represents the path.

Given a learned model of spatial positions and transition times, each track on the image lattice can be seen as a random walk where the probabilities of transition at each state of the walk are given by the learned spatio-temporal kernel density estimate. A Markov Chain Monte Carlo sampling-based scheme can then be used to generate the most likely paths in the scene and to compute a probabilistic prediction of the target's location (probability density of the next state of the random walk) ahead of the time given its previous state. It also aids the recognition of unusual behavior identified as atypical object motion. For example, if a target strays from its prescribed path in a high security area, the event can be described as a target originating at an entry point and straying from its path. Figure 18 depicts some vehicle-paths detected in a parking lot scene. The path model is also used to predict a target's most probable destination given the origin position and the path traversed. It is also used to identify targets that stray from their expected paths or travel in an abnormal direction. Figure 19 shows site-wide paths and anomalies, where the observations are provided in map-based coordinate system.



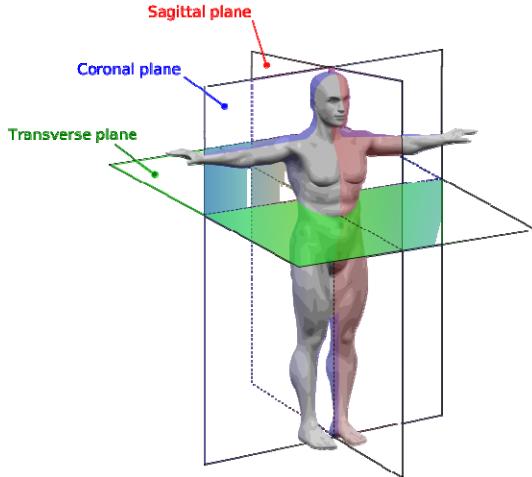
**Fig. 19** A site being observed with four cameras. The orange-overlay represents the camera FOVs. The green and red dotted lines show tracks of normal and suspicious targets respectively.

#### 4.3.1 Abnormal Gait Recognition

Given the automated 3D pose estimation of the human model (discussed in Section 2.2), our system automatically detects abnormal gait based on sagittal elevation angles [15]. The sagittal elevation angles are invariant across subjects of different heights and weights, and across different stride velocities. Elevation angles measure the orientation of a limb segment with respect to a vertical line in the world. *Sagittal elevation angles* are a projection of elevation angles of different limb segments onto the sagittal plane (vertical plane which bisects the figure into left and right halves as shown in Figure 20). Motivation for this choice stems from biomechanical research [16, 17] that describes interesting properties of this angle which makes it useful for recognition of abnormal gaits in humans. In our model, we measure the elevation angles of four limb segments of the lower body: the foot, the lower leg, the upper leg, and the pelvis.

For recognizing a sequence we used the generative classification based on Hidden Markov Models (HMM) in our framework. The *Hidden Markov Model (HMM)* has been very popular in the machine learning community and widely applied to speech recognition, part-of-speech tagging, handwriting recognition and bioinformatics. HMM assumes that a walking sequence is a Markov process describing how the pose changes during walking. A number of states are used to represent different poses during the walking action. An observation likelihood distribution models the relationship between the states and the observation. This likelihood distribution is represented by a mixture-of-*Gaussian* density functions, which is a combination of several Gaussian distribution components. Based on the previous state and the current observation, HMM may switch from one state to another. During training, the number of states and the number of components in the mixture-of-Gaussian likelihood distribution are chosen using the Bayesian Information Criterion (BIC). BIC model selection method selects the optimal

model that best describes the statistics of the training data while avoiding over fitting. Therefore, it generalizes better to previously unseen test dataset. We experimented two types of HMM: (i) a *left-right HMM* which assumes that the action is composed of a specific sequence of poses, and (ii) an *ergodic HMM* which allows all possible transitions between poses.



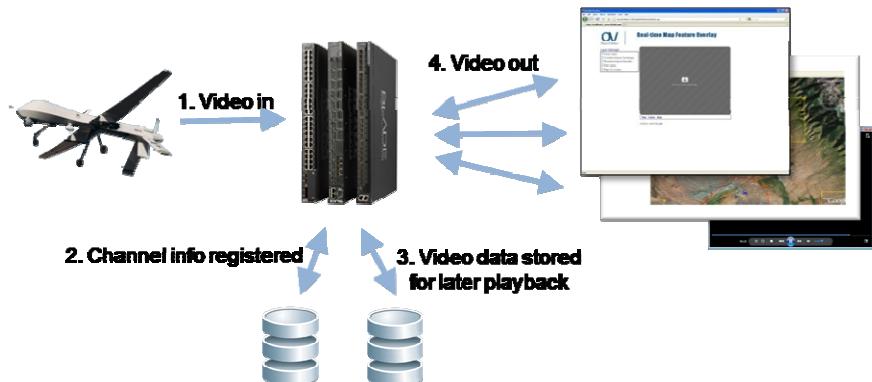
**Fig. 20** Various imaginary planes dividing the human body; Sagittal plane is a vertical plane along Y-axis that divides the body into left and right portions (Copyright: Yassine Mrabet)

## 5 Video Content Management and Retrieval

The fourth step in performing video analytics for business intelligence involves indexing the high level extracted information from the various algorithms for video retrieval. We now describe the technical details in the following sections.

### 5.1 Video Data Management System

The Video and Data Management System (VDMS) is a software-based digital media recorder (shown in Figure 21) that is capable of reliably and simultaneously ingesting and disseminating streams of data from numerous input sources in a variety of formats including video, audio, and analytics-based metadata. The system was developed to augment and enhance the current capabilities of commercial digital video recorders (DVRs) and video management systems (VMSs). It provides robust and flexible handling of large volumes of non-standard data. It was specifically created to handle unusual data types such as JPEG 2000 (JP2) streams and non-multi-media streams such as detected target snapshots or Moving Target Indicators (MTI). The system was also designed to be adaptable to a variety of uses by enabling dynamic creation and configuration of data channels at runtime, enabling scalability through the addition of COTS hardware, and providing an interoperable API facilitating third-party integration.



**Fig. 21** High-level VDMS architecture. 1) Video and sensor metadata is streamed live from UAV or from captured data stored to one of multiple load-balanced VDMS servers. 2) Basic channel information is stored in a SQL Server database to be shared across each running instance of the VDMS. 3) Video and metadata is written in blocks to network accessible storage in flat files optimized for fast retrieval. 4) Live and/or archive video is streamed to one or more subscribers on demand.

The VDMS is a Windows Service that manages connections between independent media sources and subscribers, handles data storage and retrieval, and provides an integrated framework for custom processing of streamed data. A lightweight database is used to store meta-information about each active media channel that may be shared across load-balanced instances of the VDMS running on the network. Recorded data is stored in a network accessible location, such as a network-attached storage (NAS) server or a distributed file system, providing each server on the network with access to the recorded data. Data streams are sent from data source to servers and from servers to subscribers over a network socket. Features of interest to the video management system include:

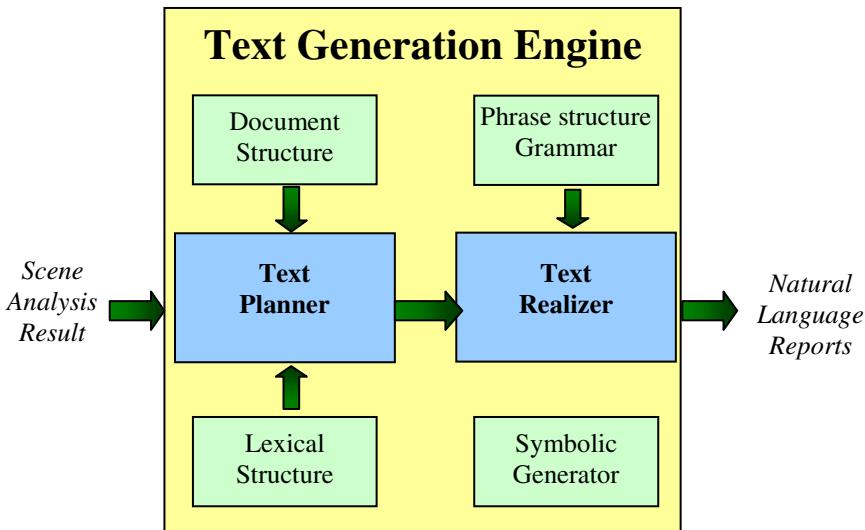
*Distributed & scalable architecture:* The VDMS features a distributive and modular architecture that is COTS hardware scalable. Internal load-balancing logic automatically distributes the load of input and output streams across all the servers configured on the network. Internal algorithms calculate the server weight according to identified key machine-specific performance metrics and route incoming search and streaming requests to the lightest weight servers. As the requirements for data throughput increases, additional standard, server-grade hardware may simply be dropped into the system, rapidly increasing the maximum capacity of the system.

*Custom data processing framework:* The VDMS features a plug-in framework enabling custom processing on media streams. A simple application programming interface (API) is provided to integrators to execute such actions as format mediation, analytics, and rendering. Plug-in processing may be applied to input streams (where results are commonly shared across all clients) or to output streams (where results are typically tailored to a single subscriber). The plug-in framework also

includes processing of archived data which enables forensic analysis. Custom modules developed by integrators may be dropped into a running VDMS at any time to near instantly expand the functional capabilities or implement needed software fixes to the system without costly or time consuming system redeployment.

## 5.2 Pixel-to-Text Report Generation

After the video events are recognized through video analysis process, those events are transformed to a text report in natural language. The generated text description of scene content allows the information to be easily documented, stored, retrieved, transmitted, and read by human analysts. In addition, text reports can be easily transformed to speech (discussed in Section 6.4). The text generation process is a pipeline of two distinct tasks (1) text planning, and (2) text realization, as shown in Figure 22. The text planner selects the content to be expressed, specifies hard sentence boundary, and provides information about the content. Based on this formation, the text realizer generates the sentences by determining grammatical form and performing word substitution. The *Text Planner* module translates the semantic representation to a representation that can readily be used by the *Text Realizer* to generate text. This immediate step is useful because it converts a representation that is semantic and ontology-based to a representation based on lexical structure.



**Fig. 22** Pixel-to-Text report generation framework.

The output of *Text Planner* is based on text generator input representation known as a *functional description* (FD) which has a *feature-value pair* structure. This structure is commonly used in text generation input schemes such as the *HALogen* [32] and *Sentence Planning Language* (SPL) [31]. The output is stored in a file called *text-planner-xml*. We use XML as the output format so that standard xml parsing tool can be used, but it essentially follows the same *labeled feature-value* structure of the logic notation used in *HALogen* [32].

<b>Scene Understanding</b> <i>Report generated on &lt;date and time&gt;</i>  <b>Source Information</b> <i>Information on video source ...</i>  <b>Scene Context</b> <i>Scene context description ...</i>  <b>Object Summary</b> <i>Brief description of objects in the scene ...</i>  <b>Events of Significance</b> <i>Significant events detected ...</i>  <b>Detailed Object Information</b> <i>Detailed events description ...</i>	<pre>&lt;document&gt;   &lt;title&gt;Scene   Understanding&lt;/title&gt;   &lt;section&gt;     &lt;title&gt;Source     Information&lt;/title&gt;     &lt;paragraph&gt;       &lt;sentence&gt;         ...       &lt;/sentence&gt;       ...     &lt;/paragraph&gt;     ...   &lt;/section&gt;   &lt;section&gt;     &lt;title&gt;Scene Context &lt;/title&gt;     ...   &lt;/section&gt;   ... &lt;/document&gt;</pre>
(a) Text report layout	(b) Corresponding text-planner-xml structure

**Fig. 23** An example of document structure for text report.

The document structure is dependent on application. As an example, a simple text report document structure was designed, as shown in Figure 23. Other document structures can be designed for other applications, such as email-notification of events, video data summarization, etc. For each sentence, the *functional description* language specifies the details of the text we want to generate, such as the *process* (or *event*), *actor*, *agent*, *predicates*, and other functional properties. The Text Realizer module utilizes a simplified *head-driven phrase structure grammar* (HPSG) [34] to generate text sentences. HPSG consists of two main components (i) a highly-structured representation of grammatical categories, (ii) a set of descriptive constraints for phrasal construction.

As described earlier, the input is the *functional description* with a collection of features. HPSG consists of a generation grammar representing the structure of features with production rules. For scene understanding applications, generated texts

are mostly *indicative* or *declarative* sentences and this simplifies the grammar structure significantly. The grammatical categories in HPSG consist of:

- S, sentences
- NP, noun phrase
- VP, verb phrase
- PP, preposition phrase
- N, noun, e.g. *car, road*
- V, verb, e.g. *move, turn,*
- A, adjective, e.g. *red, fast*
- DET, determiner, e.g. *the, a, an*
- Pron, pronoun, e.g. *he, she, he/she, it*
- P Preposition, e.g. *to, in*
- ADV Adverb, e.g. *left, up*

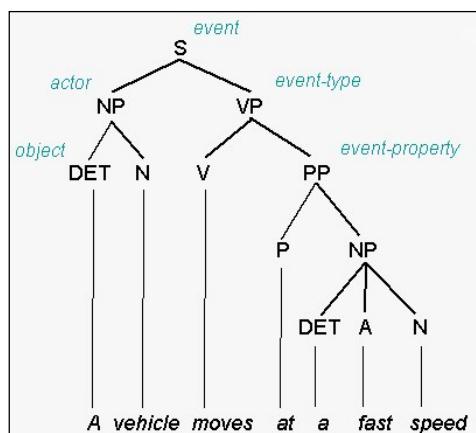
The set of production rules or descriptive constraints include:

- $S \rightarrow NP\ VP$
- $NP \rightarrow DET\ (A)\ N$
- $VP \rightarrow V\ NP$
- $VP \rightarrow V\ PP$
- $VP \rightarrow V\ ADV$
- $PP \rightarrow P\ NP$

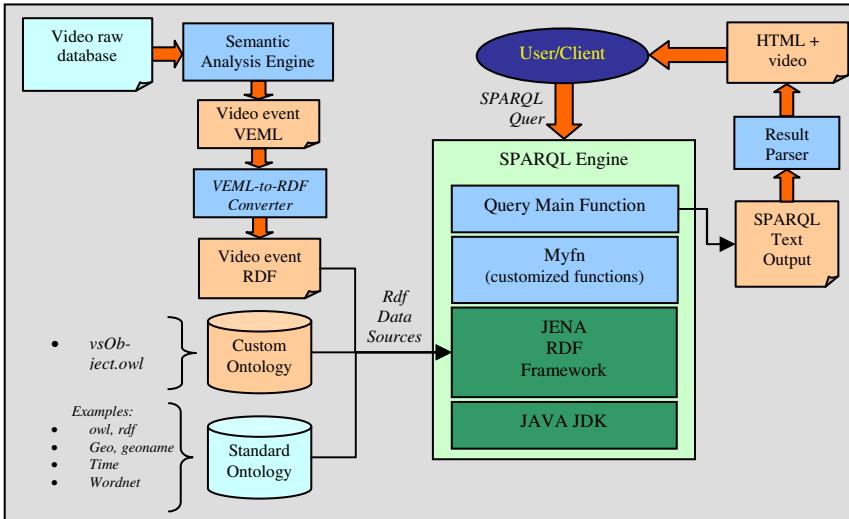
Rules with features are used to capture lexical or semantic properties and attributes. For example, to achieve *person-number agreement*, the production rules include variables so that information is shared across phrases in a sentence:

$$S \rightarrow NP(\text{per,num})\ VP(\text{per,num})$$

A unification process [33] matches the input features with the grammar in a recursively manner. The derived lexical tree is then linearized to form sentence output, as illustrated in Figure 24.



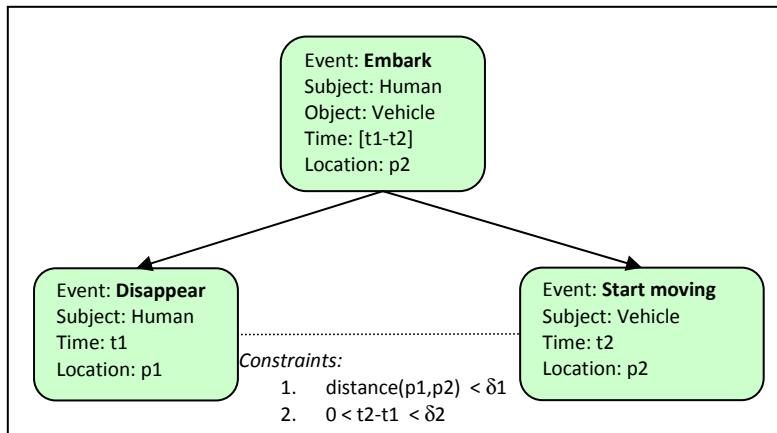
**Fig. 24** An example of sentence generation using phrase structure grammar.



**Fig. 25** SPARQL-based event inference framework.

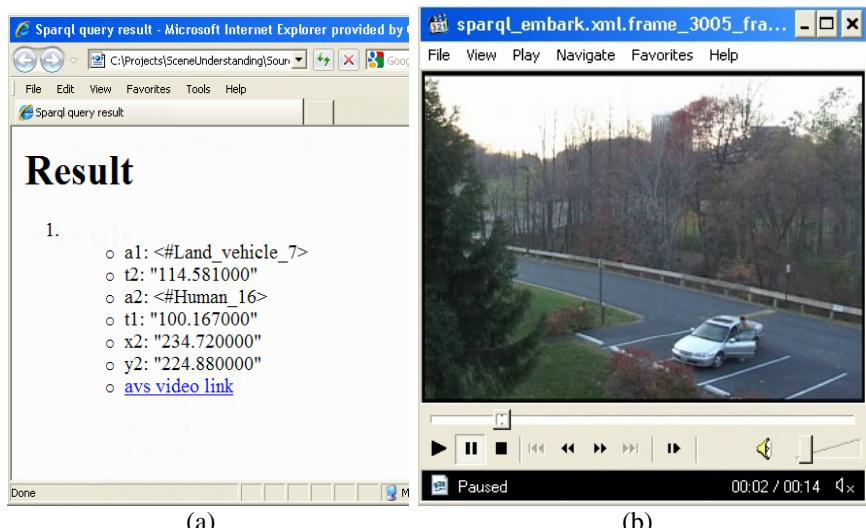
### 5.3 Video Event Query and Retrieval

Video event query and retrieval was implemented in the SPARQL framework. Figure 25 gives an overview of the SPARQL-based event inference framework. We use an open-source Java-based *Jena* RDF Framework as the underlying SPARQL computing engine. This *Jena* framework allows us to write additional customized functions such as computing geographical distance between two points. From the detected atomic events, SPARQL is used to define complex events using a high-level description. We detect “embark” and “disembark” events from atomic events using SPARQL-query. Figure 26 shows a graphical representation for an “embark” event which is a composition of two sub-events: (1) a person disappears, (2) a vehicle starts moving. The two sub-events should be close to each other, in time and space.



**Fig. 26** Graphical illustration of the definition of “embark” event.

Based on the query text, the Jena RDF framework automatically detect “embark” event from the set of atomic events. The output is a text file which is parsed to generate a html page and video snippets to show the detected events, and an example is shown in Figure 27.



**Fig. 27** SPARQL result of an embarking event. (a) Html output after parsing the SPARQL result text file. (b) automatically-generated video snippet of event based on “t1” and “t2” fields.

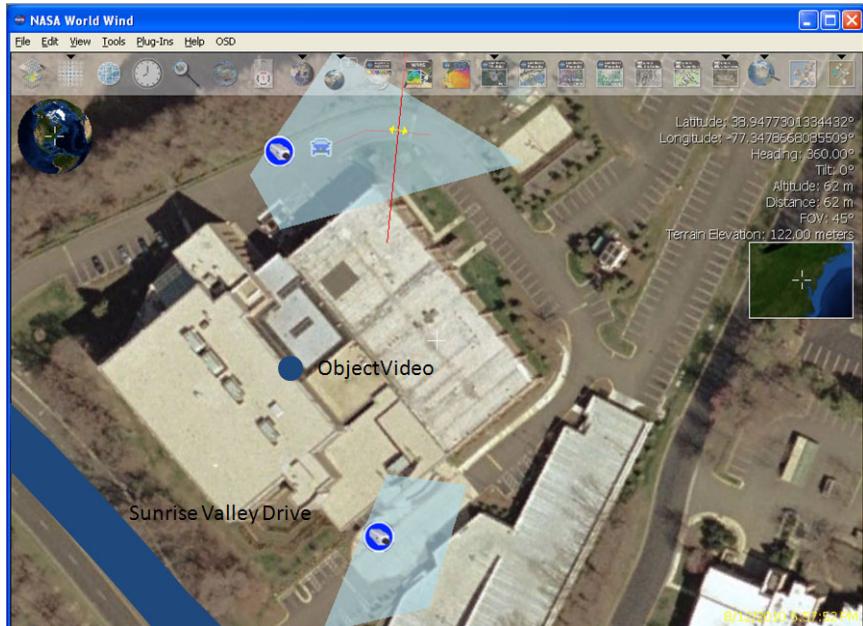
## 6 User Interfaces and Customer Experience

The final components of video analytics for business intelligence are tools and user interfaces that provide situational awareness and provide metrics to improve the quality of service provided by a business. These are described in the following sections.

## 6.1 Site Wide Awareness Using Geo-browser UI

Regardless of the accuracy of underlying intelligent systems, the interface used to provide the automatically processed information to the operator is the key component that determines the effectiveness and usefulness of the information for the task of situational awareness. A map-based interface for visualization and browsing provides a strong geographic and spatial context of site wide video content. We have leveraged our geo-browser interface, in the form of NASA's WorldWind UI plug-in [35] (as shown in Figure 28), that provides customizable visualization of data including integrated meta-data, inferred mission state representation, and views from different sensors with the ability to seamlessly transition between disparate views, enabling timely decision-making and rapid threat identification and response. This plug-in enables interfacing with WorldWind to enable the user to:

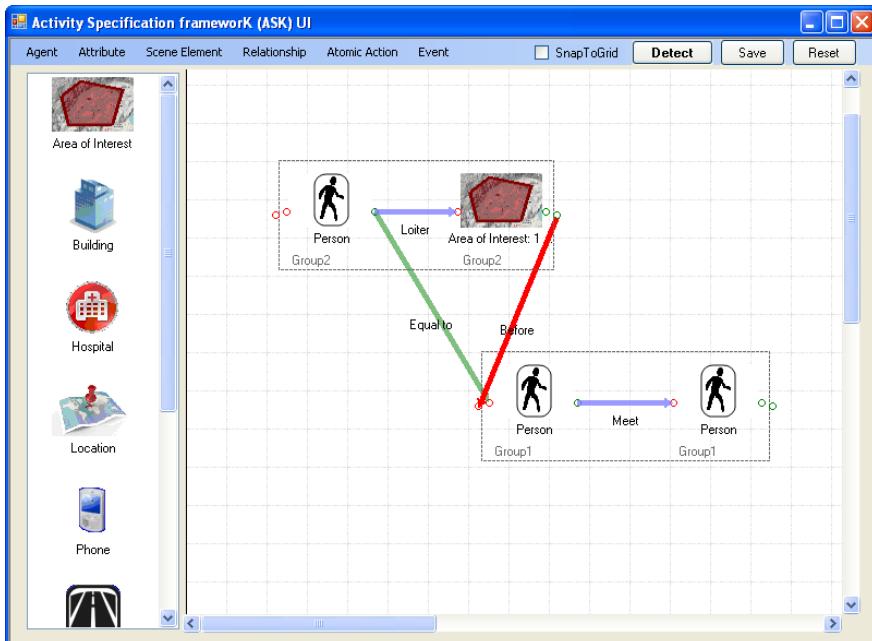
1. View each sensor's location on the map
2. View each sensor's field of view (FOV) on the map
3. Monitor target icons (indicating their latitude/longitude, speed, heading, and size) on a map
4. Track targets across sensors with overlapping and non-overlapping fields of view
5. Define metric rules based on geo-location, dimension and speed of targets
6. Receive alerts when the active rules trigger
7. Find different views for targets of interest monitored by multiple sensors



**Fig. 28** NASA's WorldWind geo-browser. The GUI displays the following information on the map: sensor locations, camera FOVs, rules, target icons, target trajectory, target attributes like location, heading etc. and GIS information.

## 6.2 Activity Specification framework (ASK)

The current state-of-the-art activity recognition systems typically provide a cumbersome text interface that is perhaps useful to researchers, but not intuitive for operators at a place of business. Furthermore, the systems support only a fixed set of pre-defined complex activities from which the user has to choose. These systems only provide a limited benefit, as the complex activity an operator is interested in based on the dynamic conditions in the place of business, may not be present in the pre-defined activity set. Under this effort, we have developed a GUI based on Microsoft's .NET framework, for an interactive Activity Specification framework (ASK) that enables the user to define human activities in terms of building blocks of atomic actions of people, and people-to-people and people-to-scene element interactions. Having an interactive activity specification and automated recognition system significantly improves the operator battlefield awareness and performance in executing mission-critical decisions. Figure 29 shows a graphical representation of the *loiter before meet* complex event specified by using the drag-and-drop interface of the ASK UI. The different components of ASK are described below:



**Fig. 29** Graphical representation of *loiter before meet* complex event in ASK UI.

**Agents:** The different object types such as humans, vehicles, trucks, etc. that are involved in the complex activity.

**Object Attributes:** The various object attributes such as size, speed etc. which act as an object filter.

**Scene Elements:** The various scene elements such as building, road etc. which are based on the contextual information, and narrow down the location where the activity is performed.

**Spatial Relationships:** This defines the spatial relationship of agents with respect to scene elements such as person behind a building, car inside area of interest, etc.

**Temporal Relationships:** This defines the temporal order of atomic actions and includes the temporal constructs such as before, during, after etc.

**Atomic Actions:** This includes simple actions and interactions such as enter, leave, meet, etc. that an agent or multiple agents are involved in during a complex activity.

The complex activity is composed of a sequence of atomic actions performed by agents such as vehicle stopping and person exiting. The ASK framework is hierarchical in nature and simple activities can be combined to describe progressively more complex activities. The ASK framework also enables the operator to quickly modify pre-defined complex events and label them as well as create new ones according to the situation in the battlefield, as well as, store their specified activities of interest so that they are available for future usage.

### 6.3 Interactive Search

Visual analysis systems are increasingly being deployed at a large scale for monitoring of vehicles, pedestrians and other objects of interest. One of the major goals for such systems is to give forensic analysts the capability to search surveillance videos for persons or vehicles of interest. The analysts are usually interested in establishing the time-line of the target object as it moves across areas being observed by various surveillance cameras. The target search involves matching of detected objects in multiple video feeds using low level image features, such as color, texture, and shape. However, significant variations in low level image features due to changes in scene illumination and viewing geometries of cameras makes it extremely hard to accurately match low level image features. The search problem can be constrained to some extent by using spatio-temporal constraints for inter-camera tracking. However, hard temporal constraints cannot be employed to localize targets when cameras are widely spaced or in areas where targets can disappear from view for long durations.

Target retrieval is a hard problem, and though progress has been made in recent years, retrieval accuracy is not high enough to warrant use in realistic surveillance scenarios. One reason for the low accuracy is the gap between semantic concepts and low-level image features. For different target search queries, various types of features have different significance. For example, in one particular query, the color of a pedestrian's jeans might be important for matching, but for another query, the texture on the pedestrian's jersey might be important for valid retrieval. Due to this changing notion of the similarity in feature space, it is hard to pre-define a feature weighting scheme or a distance metric that will work for all scenarios all the time. Here, the user's feedback can play a crucial role in learning a suitable similarity/distance metric for a particular query.

Our aim is to employ the user's judgment on relevance and irrelevance of targets to learn a distance metric; such that the retrieval system ranks the target of interest higher than other objects using that metric. In addition, since the user needs to interact with the retrieval system in real time, the metric learning algorithm should be sufficiently fast to keep the user engaged. Moreover, a practical retrieval system should require minimal user interaction, i.e., the algorithm should be able to learn from a very few examples with minimal user feedback iterations.



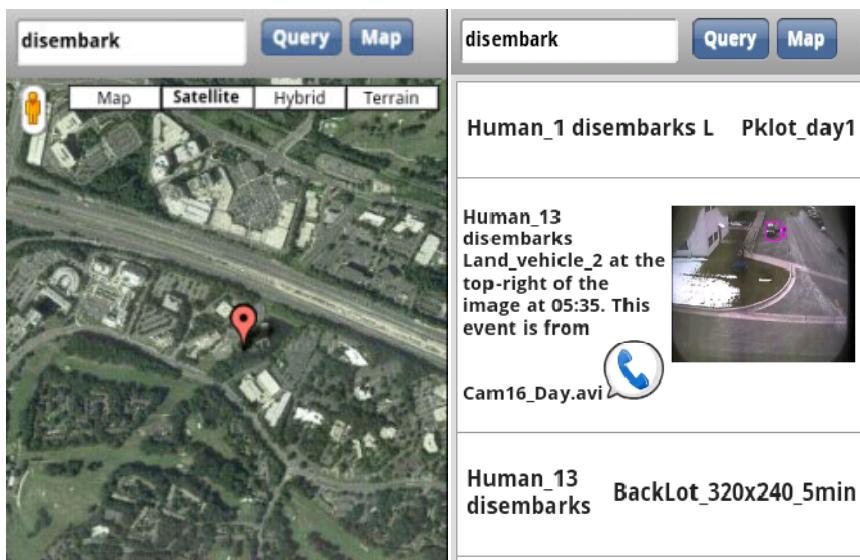
**Fig. 30** Example of pedestrian retrieval by the proposed system. The data set (VIPER [39]) contains pedestrian images from multiple viewpoints and in a variety of illumination conditions. The left-most column in each row shows the query target. The rest of rows show top target retrieval results. Each target is represented by a 100 dimensional feature vector based on color and texture features. First row shows ranking based on Euclidean distances. Note that the clothing of top retrieved pedestrians has a variety of textures and colors. The second row shows the results for the distance metric learned after 3 iterations. Note that, in this iteration the retrieved pedestrians all have white shirts, similar to the target. The third row shows retrieval results after 4 iterations of metric learning. The valid match is ranked at the top by the new metric.

With the aim to achieve all these objectives, we propose an interactive framework that allows efficient and quick learning of a Mahalanobis distance metric for target retrieval. The learning problem is cast as a convex optimization problem with rank based constraints. The proposed algorithm iteratively learns the distance metric. During each iteration the distance metric is transformed to decrease the rank of irrelevant examples and increase the ranks of relevant examples. The proposed approach avoids heavy computations over the whole data-set. The objective of minimal user interaction is achieved by using the generalized risk zone

principle [38] to select examples for user feedback that will result in significant information gain for metric learning. Moreover, unlabeled data is also incorporated in the optimization process to get better generalization performance. An example result of interactive search is shown in Figure 30.

## 6.4 Query UI on the Smart Phone

The emergence of commercial handheld devices now enables users to access and visualize rich geospatial data, even in remote locations. However, to make a smart phone application usable, we need to overcome challenges like limited bandwidth and small display screen size. We describe a smart phone application has been developed to submit user queries and display event detection results in both text and map modes (See Figure 31).

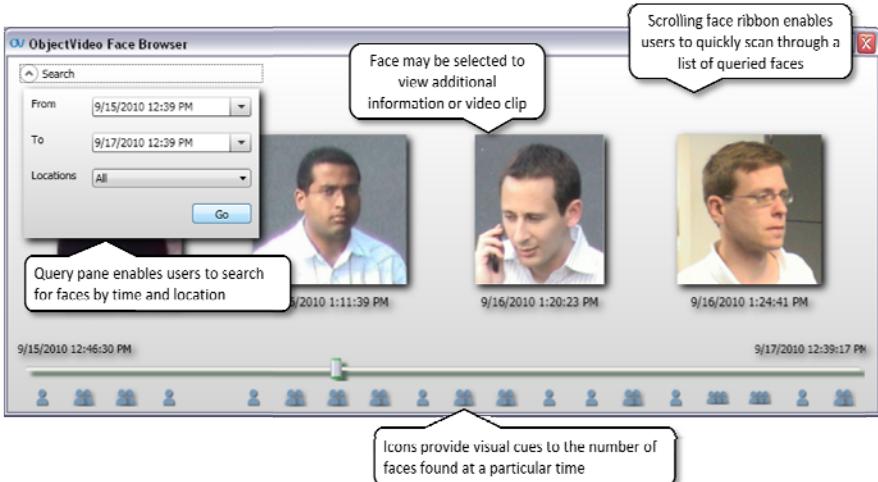


**Fig. 31** Query UI on smart phone. (Left) Initial map view (Right) Query result.

Users can either type in or speak keywords to search for and click “Query” button. Android built-in *speech-to-text* (STT) feature is used to translate user's speech into key words. The user query is processed and the matched events along with their text description are listed (see Figure 31). Tapping on an event description will bring up details such as a snapshot of the event with target markup and detailed textual description. A *voice synthesis* option is provided and the details of the event description can be “read” to the user. The query UI is developed using jQuery [36] and a Google map plug-in is used to provide the map data. On the backend, ASP.NET is hosting a RESTful (Representational State Transfer) web service which runs the query commands against a SQL server database. Query results are returned to web UI in the form of JavaScript Object Notation [37].

## 6.5 Face Browser

Face Browser is a software application that stores and displays face thumbnails in an intuitive interface, as shown in Figure 32. The key features include: 1) Fast and accurate best-shot face cataloging using state of the art computer vision technology. 2) Time-based, inertial ribbon for rapid search across multiple video sources. 3) Click-through to video for added scene context. 4) People count indicators.

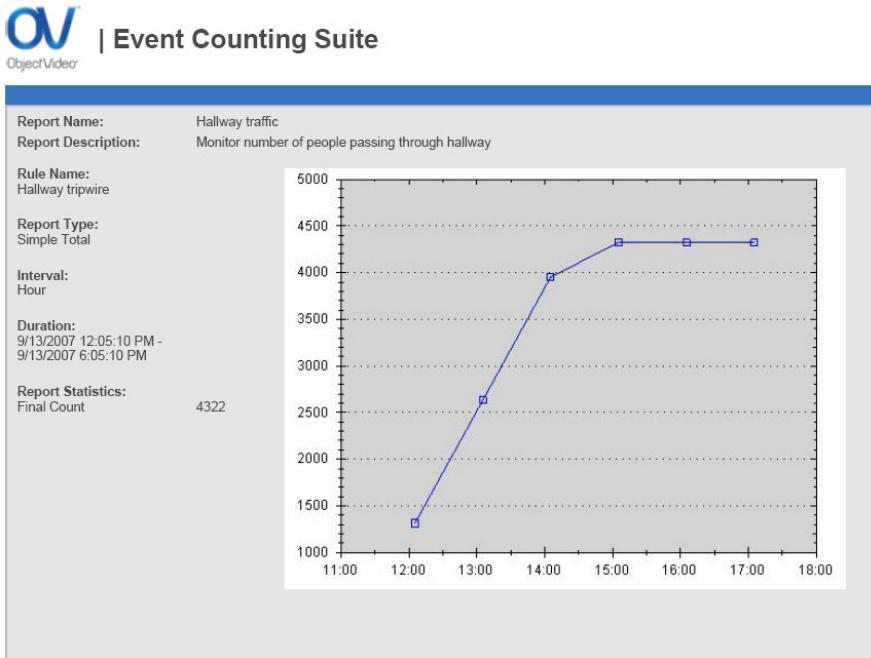


**Fig. 32** Face browser is an intuitive interface to browse and display faces.

## 6.6 Event/Person Counting Suite

The event counting suite (see Figure 33) offers a comprehensive set of functionality and flexible camera placements to address vital people and vehicle counting applications, including occupancy and dwell time detection features. Multi-camera and multi-rule functions allow users to aggregate data from across the enterprise while web-based monitoring and reporting interfaces enable them to access their desired data output anywhere, anytime. It provides real-time intelligence gathering through:

1. Event counting through tripwire technology and other key event types for the purpose of counting people or vehicles
2. Occupancy sensing continuously outputs the number of people present in any user-defined area and alerts when thresholds are exceeded
3. Dwell time monitoring outputs the length of time people remain in a user-defined area.



**Fig. 33** Event counting suite provides real-time business intelligence data.

## 7 Applications

In this section, we discuss some of the applications of the technologies and interfaces described earlier in the chapter. From a business intelligence standpoint, some of the technologies described can be of great help in improving the efficiency of the business.

**Person Counting:** The number of customers visiting a retail store is an extremely important metric for a number of reasons. Using person counting technology described earlier in this chapter, store supervisors can improve the efficiency of various aspects of their business by accurately counting the number of people entering and leaving the premises. The retailer can also use the Event Counting Suite (ECS) for web-based monitoring and intuitive reporting interfaces. Such statistics provide a much deeper insight than simple sales statistics and can lead to much more efficient business operation. An estimate of the customer traffic offers store managers the following:

- Visitor traffic by hour, day, week, month and year
- Schedule staffing level more accurately
- Understand retail store conversion rate, when sales information is integrated. Compute various statistics like spend per head etc.

- d) Compare store occupancy to store opening hours
- e) Know dwell time (average stay) of visitors
- f) Queue length monitoring and average wait/transaction times
- g) Measure impact of latest marketing campaign

**People Movement Patterns:** In a retail environment, one of the key pieces of information required for section planning, aisle designs and product placements is the movement patterns of customers. Using cross camera tracking technology, we can track people across the whole store and gather information like where people go first/most, where they spend most time, how they move from section to section, or how they move generally within the store. Once the movement patterns have been identified, the retailers can make better and more informed decisions like modifying the layout of the store to increase sales, placing marketing materials at appropriate locations for greater visibility etc.

**Shoplifting:** Shoplifting is one of the most common property crimes in a retail environment. Retailers report that shoplifting has a significant impact on their bottom line. One of the main security goals in this context is to identify and prosecute people who are involved in shoplifting. In such a scenario, the technology described earlier for detection of concealed objects based on automated 3D human model and pose estimation is extremely relevant. Such technology can be very useful for detection of people who conceal medium/big size objects under their clothes and try to leave the store with unpaid merchandise.

**Sweethearing:** Another major source of loss in a retail environment is sweethearing. Sweethearing is the act of a store employee giving a friend or family member free merchandise by pretending to scan items at the register. In such a scenario, the store security manager can create complex event rules involving data from multi-modal sensors. During a genuine transaction at a checkout register, there should be a store employee present on one side of the register and a customer present on the other side of the register. Video cameras can be used to validate this configuration at the time of checkout. Using human model estimation and fitting technology described earlier, using video cameras, we can detect and recognize human actions like a typical scan motion by the register attendant. Using the Activity Specification Framework (ASK) UI, the security supervisor can define complex rules such as:

- a) Person present in register attendant area
- b) Person present in customer area
- c) Multiple human "item scan" actions
- d) Each "item scan" action is validated by the electronic cash register data

If any of these rules are violated during an ongoing checkout transaction, the system can automatically notify the store supervisor for appropriate action.

**Customer Demographics:** A study of customer demographics can be a key component of setting up and operating a successful business. Using our technology for

gender and age classification, a business owner can collect data about the demographics of the customer base. At the time of setup, such data can greatly increase the chances of selecting the correct area and location for the business. For example, demographics data suggesting that a large fraction of people visiting a mall are "young women" provides a strong case for opening a cosmetics store. Within existing business establishments, customer demographics data can prove invaluable for day-to-day business operations like inventory control, product placement and pricing.

**Person Best Shot:** As mentioned earlier, shoplifting is a major source of revenue loss for retailers. It is important to identify and prosecute these individuals to deter future crimes. Using best shot technology, a face catalog of all people entering a store can be created. In scenarios where the image resolution is not sufficient to identify an individual, PTZ cameras can be automatically pointed at the target of interest to capture high-resolution images. The face browser UI can be used to query the face database and visualize the results. These best face shots can be displayed on "Wanted Person" bulletin boards, or matched against a face database of known offenders using face recognition technology.

**Target Retrieval:** In a retail store, from a surveillance stand point, it is very important that the security supervisor have the capability to search archived videos for person and vehicles of interest for forensic purposes. Typically in such a situation, the user would be interested in establishing the time-line of the target of interest (e.g shoplifter) as it moves across the site observed by multiple surveillance cameras. The target search involves matching of detected objects in multiple video feeds using low level image features, such as color, texture, and shape. However, significant variations in low level image features due to changes in scene illumination and viewing geometries of cameras makes it extremely hard to accurately match low level image features. The interactive search technology described earlier in this chapter seeks to overcome these limitations by employing the user's judgment on relevance and irrelevance of targets to learn a distance metric; such that the retrieval system ranks the target of interest higher than other objects.

**Event/Activity Specification:** In a retail environment with video surveillance, it is important for the security supervisor to have the capability to define the events of interest in a simple and intuitive manner. Most existing interfaces are too cumbersome and complex for a typical security personnel and therefore have very limited utility from a practical standpoint. The Activity Specification Framework (ASK) UI enables the user to define human activities in terms of building blocks of atomic actions of people, people-to-people, and people-to-scene element interactions. The UI provides a hierarchical framework in which simple activities can be combined to describe progressively more complex activities. As an example, the complex event of interest described earlier in the sweethearts subsection can be defined using the ASK UI in a matter of few clicks.

**Unusual Event Detection:** As described earlier, from a security standpoint, the security supervisor of a retail store might be interested in detecting a set of pre-defined events of interest. But, from a practical standpoint, it is impossible to

define every event that might be of interest. Sometimes, it is even difficult to know what exactly constitutes an "event of interest", and in such cases all one probably cares about is detecting anything that is unusual. What is unusual? Anything that has not been seen frequently enough in the past or doesn't fit the normal pattern of behavior. Using technology described earlier, we can build site-wide normalcy models and target property maps to learn normal target behavior and movement patterns and flag anything doesn't fit these models. An example of such an event might be detection of a speeding car (25mph) in the store parking lot where the average vehicle speed is 5 mph. The speeding car would be flagged as an unusual event, and could potentially be a shoplifter trying to escape as fast as possible.

**Visualization Interface:** One of the key features of a practically useful video surveillance system is an interface which enables to user to visualize the activities around the site in a centralized manner. Such an interface allows the user to define rules on a map/site layout or overhead imagery, monitor target movement patterns and visualize events of interest (with location, time, type etc.). A map-based interface for visualization and browsing provides a strong geographic and spatial context of video content. In a retail environment, the security supervisor needs to have the ability to define store wide rules such as: count number of people entering the new "Asian Food" section, or be able to visualize on a store map which check-out is overly crowded so that more registers can be opened. NASA's WorldWind UI plug-in described earlier meets these requirements, and provides customizable visualization of data including integrated metadata, inferred state representation, and views from different sensors.

**Daily Summary Report Generation:** In a retail environment with video surveillance, it is important for the security supervisor to have the capability to review a daily event summary text report at the end of the day. Such a report can effectively update the supervisor on the events of interest during the day, without taking too much time or requiring constant supervision throughout the day. Such a text report can be easily documented, stored, retrieved, transmitted, and read by other analysts. The pixel-to-text generation technology described earlier in this chapter can be used for generation and archiving of such text reports. Additionally, a textual representation of video events enables well established text searches using generic search engines such as Google.

## References

- [1] Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall (1999)
- [2] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., Wixson, A.: A System for Video Surveillance and Monitoring. CMU Technical Report CMU-RI-TR-00-12 (2000)
- [3] Isard, M., Blake, A.: CONDENSATION – conditional density propagation for visual tracking. IJCV 29(1), 5–28 (1998)

- [4] Rosales, R., Sclaroff, S.: Improved Tracking of Multiple Humans with Trajectory Prediction and Occlusion Modeling. In: IEEE Conf. On Computer Vision and Pattern Recognition, Santa Barbara, CA (1998)
- [5] Wren, C., Azarbayejani, A., Darrel, T., Pentland, A.: Pfnder: Real-time tracking of the human body. In: Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp. 51–56 (1996)
- [6] Isard, M., MacCormick, J.: BraMBLe: A Bayesian multiple-blob tracker. In: Proc. ICCV (2001)
- [7] Isard, M., Blake, A.: ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 893–908. Springer, Heidelberg (1998)
- [8] Viola, Jones: Robust Real-time Object Detection. In: IJCV (2001)
- [9] Kanaujia, A., Sminchisescu, C., Metaxas, D.: Semi-supervised Hierarchical Models for 3D Human Pose Reconstruction. In: Computer Vision and Pattern Recognition (2007)
- [10] Sminchisescu, C., Kanaujia, A., Metaxas, D.: BM3E: Discriminative Density Propagation for Visual Tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(11), 2030–2044 (2007)
- [11] Gilks, W., Richardson, S., Spiegelhalter, D.: Markov Chain Monte Carlo in Practice. Chapman and Hall (1996)
- [12] Lee, M., Cohen, I.: Proposal Maps driven MCMC for Estimating Human Body Pose in Static Images. In: Proc. Computer Vision and Pattern Recognition, pp. 334–341 (2004)
- [13] Lee, M., Nevatia, R.: Dynamic Human Pose Estimation using Markov chain Monte Carlo Approach. In: MOTION, pp. 168–175 (2005)
- [14] CAESAR, Civilian American and European Surface Anthropometry Resource Project (2002)
- [15] Sun, H., Goswami, A., Metaxas, D., Bruckner, J.: Cyclogram planarity is preserved in upward slope walking. In: Proc. International Society of Biomechanics XVII Congress, Calgary, Canada (1999)
- [16] Grasso, R., Bianchi, L., Lacquaniti, F.: Motor patterns for human gait: Backward versus forward locomotion. J. Neurophysiology 80, 1868–1885 (1998)
- [17] Borghese, A., Bianchi, L., Lacquaniti, F.: Kinematic Determinants of Human Locomotion. J. Physiology (494), 863–879 (1996)
- [18] Guo, Y., et al.: Matching Vehicles under Large Pose Transformations using approximate 3D. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (2008)
- [19] Leotta, M.J., Mundy, J.L.: Predicting high resolution image edges with a generic, adaptive, 3-D vehicle model. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (2009)
- [20] Lou, J., Tan, T.: 3-D Model-Based Vehicle Tracking. IEEE Transactions on PAMI (2005)
- [21] Zhu, S.C., Mumford, D.B.: Quest for a stochastic grammar of images. In: Foundations and Trends of Computer Graphics and Vision (2006)
- [22] Tu, Z.W., Zhu, S.C.: Parsing images into regions, curves and curve groups. IJCV 69(2), 223–249 (2006)
- [23] Babenko, B., Yang, M.H., Belongie, S.: Visual Tracking with Online Multiple Instance Learning. In: CVPR (2009)

- [24] Chen, K.W., Lai, C.C., Hung, Y.P., Chen, C.S.: An adaptive learning method for target tracking across multiple cameras. In: CVPR (2008)
- [25] Dietterich, T.G., Lathrop, R.H., Lozano-Perez, T.: Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence* 89, 31–71 (1997)
- [26] Javed, O., Shafique, K., Shah, M.: Appearance modeling for tracking in multiple non-overlapping cameras. In: CVPR 2005 (2005)
- [27] Porikli, F.: Inter-camera color calibration by correlation model function. In: ICIP 2003 (2003)
- [28] Viola, P., Platt, J.C., Zhang, C.: Multiple instance boosting for object detection. In: NIPS 2005 (2005)
- [29] Moore, D., Essa, I.: Recognizing Multitasked Activities using Stochastic Context-Free Grammar. In: CVPR 2001 (2001)
- [30] Earley, J.C.: An Efficient Context-Free Parsing Algorithm. PhD thesis, Carnegie-Mellon University (1968)
- [31] Jhuang, H., et al.: A Biologically Inspired System for Action Recognition. In: ICCV 2007 (2007)
- [32] Langkilde-Geary, I., Knight, K.: HALogen Input Representation, <http://www.isi.edu/publications/licensed-sw/halogen/interlingua.html>
- [33] Knight, K.: Unification: A Multidisciplinary Survey. *ACM Computing Surveys* 21(1) (1989)
- [34] Pollard, C., Sag, I.A.: Head-Driven Phrase Structure Grammar. University of Chicago Press, Chicago (1994)
- [35] Rasheed, Z., Cao, X., Shafique, K., Liu, H., Yu, L., Lee, M., Ramnath, K., Choe, T., Javed, O., Haering, N.: Automated Visual Analysis in Large Scale Sensor Networks. In: ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC) (September 2008)
- [36] jQuery. JavaScript library, <http://www.jquery.com>
- [37] JSON, <http://www.json.org>
- [38] Peres, R., Pedreira, C.: Generalized risk zone: Selecting observations for classification. *IEEE PAMI* 31(7) (2009)
- [39] Gray, D., Tao, H.: Viewpoint Invariant Pedestrian Recognition with an Ensemble of Localized Features. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part I*. LNCS, vol. 5302, pp. 262–275. Springer, Heidelberg (2008)
- [40] Kuhn, H.W.: The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly* 2, 83–97 (1955)
- [41] Comaniciu, D., Meer, P.: Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (IEEE) 24(5), 603–619 (2002)
- [42] Dukette, D., Cornish, D.: The Essential 20: Twenty Components of an Excellent Health Care Team, pp. 72–73. RoseDog Books (2009)

# Design and Validation of a System for People Queue Statistics Estimation

Vasu Parameswaran, Vinay Shet, and Visvanathan Ramesh

**Abstract.** Estimating statistics of people queues is an important problem for many businesses. Monitoring statistics like average wait time, average service time and queue length help businesses enhance service efficiency, improve customer satisfaction and increase revenue. There is thus a need to design systems that can automatically monitor these statistics. Systems that use video content analytics on imagery acquired by surveillance cameras are ideally suited for such a monitoring task. This chapter presents the systematic design of a general solution for automated visual queue statistics estimation and its validation from surveillance video. Such a design involves the careful consideration of multiple variables such as queue geometry, service-counter type, illumination dynamics, camera viewpoints, people appearances etc. We address these variabilities via a suite of algorithms designed to work across a range of queuing scenarios. We discuss factors involved in the systematic validation of such a system such that realistic performance assessment over a wide range of operating conditions can be ensured. We address validation, evaluation parameters and deployment considerations for this system and demonstrate the performance of the proposed solution.

---

Vasu Parameswaran

Nokia Research Center, Palo Alto, CA,

Work performed when author was at Siemens Corporate Research, Princeton, NJ

e-mail: [vasu.parameswaran@nokia.com](mailto:vasu.parameswaran@nokia.com)

Vinay Shet

Image Analytics and Informatics, Siemens Corporate Research, Princeton, NJ

e-mail: [vinay.shet@siemens.com](mailto:vinay.shet@siemens.com)

Visvanathan Ramesh

Goethe University, Frankfurt/Main, Frankfurt, Germany,

Work performed when author was at Siemens Corporate Research, Princeton, NJ

e-mail: [Ramesh@fias.uni-frankfurt.de](mailto:Ramesh@fias.uni-frankfurt.de)

## 1 Introduction

Managing human queues is important for a variety of businesses. Amusement parks need to handle queues of people waiting their turn for rides, airports need to manage people waiting to be screened by security, retail stores need to serve patrons waiting to pay for their purchases. If such queues of people are not actively monitored by the business, it often results in lower customer satisfaction, inefficient resource allocation and loss of revenue. There is thus a need, to not only monitor queue statistics in real-time but also to better understand trends that emerge in such queues over time. Automated video content analytics provide one means to design queue monitoring systems that are capable of doing both.

This chapter presents the systematic design and validation of a general solution for a video-surveillance-based, automatic, visual queue statistics estimation. The design of such a system requires the careful consideration of several variables, which include the type of queue (e.g. snaking queue, linear queue etc), the type of service counters (check-in counters, baggage screening counters etc.), level and dynamics of illumination, camera viewpoint, people appearances (with and without bags) etc. These variabilities are addressed by decomposing the main task into two sub-tasks, and combining their outputs. The subtasks are: (a) Queue size estimation, and (b) Service time estimation. We show that these modules can be combined together to analyze a range of queuing scenarios. The two tasks are solved using algorithmic components for crowd count estimation, and person detection & localization. In order to obtain a realistic assessment of the performance of a solution, it is necessary that a wide range of operating conditions be covered while keeping validation costs to a minimum. In this chapter we discuss the validation process we followed for a particular deployment of the proposed queue solution, the evaluation metrics used, and the results of the process for our solution.

A complete queue statistics monitoring system needs to perform several tasks, (i) estimate the number of people waiting in the queue, (ii) estimate the average service time at all the service counters, (iii) in case of multiple service counters, detect whether they are operational, and (iv) estimate the expected waiting time. A critical component in the system is one that estimates the number of people waiting in the queue. Prior research has focused on two related problems - crowd density estimation and crowd count estimation. Crowd density estimation involves devising measures related to the overall crowdedness of the scene while crowd count estimation, as the term implies, involves estimating the number of people in the scene. Both tasks involve using image features such as as the weighted area of foreground blobs in the image and their mapping to either crowdedness or crowd count estimation. Approaches differ based on the measure that is used to characterize the crowd, features selected on image and the type of learning used to map features to the measure.

Paragios and Ramesh [14] use a discontinuity preserving Markov Random Field based approach to perform smooth foreground segmentation that combines spatial constraints with intensity modeling. They then combine the change detection map with a geometry module that performs soft-auto calibration to estimate a

crowdedness measure in a given region. In [1] Chan et al describe a system to estimate the size of an inhomogeneous crowd by first segmenting the crowd into components of homogeneous motion using mixtures of dynamic texture-motion models. They extract a set of simple holistic features from these segmented regions and learn the correspondence between number of people in each segment and the extracted features using Gaussian process regression. In [9], Kong et al describe a learning based approach for view point invariant crowd counting. They first extract edge orientations and blob size histograms as features. They compute a density map measuring relative size of individuals and global scale measuring camera orientation and use it to normalize the extracted features. They learn the relationship between extracted features and crowd density using linear fitting and neural networks. In [8] Kilambi et al describe a system to estimate the counts of people in groups and track them. They learn the mapping between scene and camera calibration parameters to the counts of people visible in the image. Zhao and Nevatia [23] use a generative model for the formation of image silhouettes and infer the number and positions of people using the well-known Markov Chain Monte Carlo method.

Another class of approaches directly detects humans in the scene. Approaches here tend to fall primarily in two categories: those that detect the human as a whole and those that detect humans based on part detectors. Among approaches that detect humans as a whole, Leibe et.al [10] employs an iterative method combining local and global cues via a probabilistic segmentation, Papageorgiou et. al. [13] uses SVM detectors, Felzenszwalb [4] uses shape models, and Gavrilla [6, 5] uses edge templates to recognize full body patterns. A popular detector used in such systems is a cascade of detectors trained using AdaBoost as proposed by Viola and Jones [19]. Such an approach uses haar wavelets features and has been successfully applied for face detection in [19]. In [20] Viola and Jones applied this detector to detect pedestrians by augmenting haar wavelets with simple motion cues for enhanced performance. Another popular feature is the histogram of oriented gradients, introduced by Dalal and Triggs [2] who used it with a SVM based classifier. This was further extended by Zhu et. al [24] to detect whole humans using a cascade of histograms of oriented gradients. Tuzel et al [18] use covariance matrices to represent humans and learn a classifier on a Riemannian manifold.

Part based representations have also been used to detect humans. Mikolajczyk et al. In [11] Mikolajczyk et al divide the human body into seven parts and for each part, a Viola-Jones approach is applied to orientation features. [12] divides the human into four different parts and learns SVM detectors using Haar wavelet features. Wu and Nevatia [21] use edgelet features and learn nested cascade detectors [7] for each of several body parts and detect the whole human using an iterative probabilistic formulation. Mohan et.al. [22, 21, 10] follow up low level detections with high level reasoning that allows them to enforce global constraints, weed out false positives, and increase accuracy. Shet et al [16] use part based detectors with logical reasoning to fuse the results into scene-consistent human hypotheses.

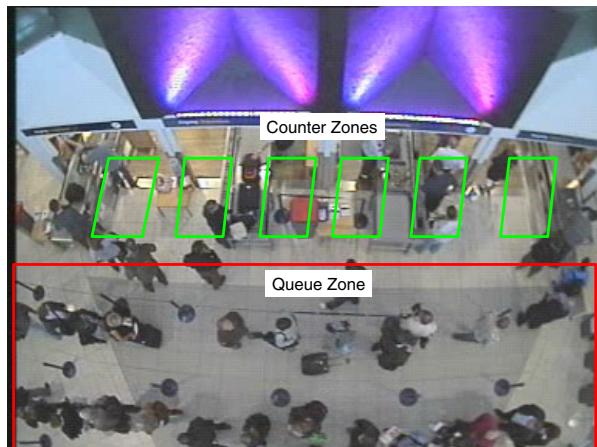
Our queue monitoring system is expected to have high accuracy, easy deployment, and easy configuration. It is expected to work well under a wide variety of operating conditions including different queue structures, viewpoints and person

appearances. Furthermore, the system is expected to be deployed in a fixed-camera setting allowing the possibility of performing background maintenance and change detection. Due to these factors, we chose not to use methods that are based primarily on image-features and machine learning. We instead chose to use and build upon foundational algorithms designed for a fixed-camera setting.

The rest of the article is organized as follows: In section 2 we decompose the queue statistics monitoring problem into its constituent components and make explicit our assumptions. In section 3, we identify several sources of variation that can adversely affect such a queue monitoring solution. In the following sections we describe techniques that help us handle these variability. In section 4 and section 5 we describe our approach for crowd count estimation and people detection & localization respectively. In section 6 we describe the queue monitoring system that combines all these technologies. In section 7 we list issues that need to be considered during deployment. We then describe how the solution is validated in section 8. Finally, in section 9 we conclude with a review of the solution including rationales for various choices made, a discussion of the results and insights obtained from the evaluations, current limitations of the solution and items for future work.

## 2 Problem Decomposition

One can model a queue as consisting of one or more queue-zones where people wait to be served, and one or more counter zones where people are provided service (see figure 1). We assume that the amount of time each person requires to get service at a counter is a random variable  $S$  with probability density:  $p_s(t)$ . The total time that



**Fig. 1** Queue and Counter Zones

a given person needs to wait in the queue before he gets served is a function of two variables - the number of people ahead of him in the queue, and the time taken for each person to be served. If there are  $N$  people in the queue, the expected waiting time for a person just entering the queue at the end,  $T_{avg}$ , is given by:

$$\begin{aligned} T_{avg} &= E(S_1 + S_2 + \dots + S_N) \\ &= N\bar{S}_N \\ &\approx N \int_0^{\infty} s p_s(s) ds \end{aligned} \quad (1)$$

where  $S_i$  are independent samples drawn from the service time density  $p_s$  and  $\bar{S}_N$  is the sample mean. As we discuss later, the number of people in queue may not be known precisely and instead also come from a probability density function  $p_N(x)$ . In this case the waiting time is given by:

$$T_{avg} \approx \left( \int_0^{\infty} x p_N(x) dx \right) \left( \int_0^{\infty} s p_s(s) ds \right) \quad (2)$$

Thus, a queue solution can be based upon two modules - a queue module (QM) that analyzes images from one or more cameras observing the people waiting in queue, and a counter module (CM) that analyzes images from one or more cameras observing the counter areas. QM estimates the number of people in the queue while CM estimates how long each person waits in a counter zone to complete service.

Modeling the problem in this manner offers several advantages in the design of a solution. Firstly, there is no implication that the solution necessarily depends upon detecting and tracking multiple individuals throughout the queue. Secondly, there is no explicit assumption made on the queue structure (e.g. straight-line queue, snaking queue, or combinations) - one just needs to know the number of people waiting in the queue. If the structure of the queue is known apriori, we could exploit it and obtain better accuracy. However, in our analysis we assume that the queue structure is unknown. Thirdly, there are no explicit assumptions made on the imaging conditions, e.g. camera position, person appearances, illumination level, etc. We incorporate robustness to these variabilities into the design of QM and CM which are based on low level analytics. Lastly, decomposing the queue problem as described here enables the re-use and adaptation of available algorithms for crowd density estimation, e.g. [14] and person detection e.g. [3] for QM and CM. In the following sections, we describe use and adaptation of such algorithms in greater detail.

## 2.1 Assumptions

The design of algorithms for QM and CM makes the following assumptions that are reasonable for a typical surveillance scenario:

1. The camera is static (although the scene may be dynamic)
2. Vertical lines in the world project to vertical lines in the image. Extension to the non-vertical projection case is in principle straightforward.
3. The projected width and height of a person of average size as a function of image position is provided. Since this is partial calibration information and not a full 3D geometric transformation, we refer to our scenes as ‘quasi-calibrated’, and the functions as ‘quasi-calibration’.

A static camera allows the possibility of maintaining a statistical model of the ‘empty’ scene, i.e. the scene devoid of foreground objects, and of computing a change image, which is a binary mask  $B(x,y)$  that corresponds to foreground objects in the scene. Background maintenance and change detection are rapidly maturing sub-fields within video surveillance. We assume the availability of modules for generating  $B$ .

### 3 Robustness to Variability

Several sources of variability influence the images coming into the queue processing system which QM and CM need to be robust to. Mainly, the following are the classes of variable:

1. Illumination (slow and fast changing, local and global)
2. Camera viewpoint (top-down and general)
3. Variation in person appearances (pose, articulation, etc)

For fast and slowly changing local and global illumination, we use our prior work on illumination-robust change detection [15], [17]. Taken together, the methods effectively deal with most of the illumination changes found in a typical queue monitoring environment. Therefore the binary mask  $B(x,y)$  can be expected to be largely free of illumination effects. Camera viewpoint and person appearance variability are handled by the respective modules for QM and CM. The scene background is maintained over time and updated conservatively. In other words, rather than updating the background at a specific exponential adaptation rate at all times which causes foreground objects to become blended into the background, the background is updated only when it is determined with high probability that the change is due to illumination.

In the next two sections we describe two key components of the overall solution, namely crowd count estimation and person detection & localization. Subsequently, we describe the visual queue analysis system.

## 4 Crowd Count Estimation

The task of the crowd count estimation component is to estimate the number of people in a given region of interest in the scene. We adapt and improve the crowd density estimation work of [14] to carry out the task.

Denote the width by  $w(y)$  and height by  $h(y)$ . We work with the  $y$  axis pointing downwards as in an image. Let the width and height of the image be  $W$  and  $H$  respectively. We propose that the crowd size  $C$  can be expressed as a weighted area of  $B$ :

$$C = \sum_{i=1}^H \sum_{j=1}^W \theta(i) B(i, j) \quad (3)$$

Paragios and Ramesh [14] choose  $\theta(i) = 1/(w(i)h(i))$ . Although this is approximately position invariant and a reasonable weight function, in this work, we derive a weight function that incorporates position invariance explicitly. Assume that there is one person in the scene and that the person is modeled by a rectangle with its top left corner at  $(x, y)$ . In this case we seek a function  $\theta(\cdot)$  such that:

$$\sum_{i=y}^{f(y)} \theta(i) \sum_{j=x}^{x+w(f(y))} B(i, j) = 1 \quad (4)$$

Here  $f(y)$  is the  $y$  coordinate of the person's foot  $f(y) - y = h(f(y))$ . Let the  $y$  coordinate of the horizon be  $y_v$  which can be obtained by solving  $h(y) = 0$ . The smallest  $y$  coordinate for a person's head we consider is given by :

$$y_0 = \max(0, y_v + \varepsilon) \quad (5)$$

Let  $y_{max}$  be the maximal head positon above which the feet are below the image. Equation 4 applies for positions  $y_0 \leq y \leq y_{max}$ . For  $y > y_{max}$  the weighted sum is adjusted to the fraction of the visible height of the person. We thus have  $(H - y_0 + 1)$  equations in as many unknowns and the linear system of equations can be solved to yield  $\theta(\cdot)$ . Although this is in principle correct, the equations do not enforce smoothness of  $\theta(y)$  and hence the resulting weight function is not typically smooth (see figure 2). We could remedy this problem using regularization (e.g. Tikhonov) but we found the following method quite effective in our case: We first define the cumulative sum function

$$F(y) = \sum_{t=y_0}^y \theta(t) \quad (6)$$

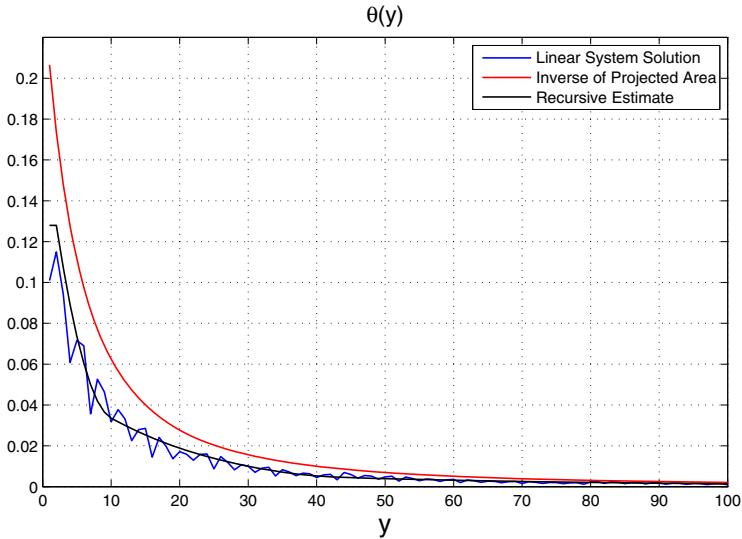
The inner sum in equation 4 is  $w(f(y))$  and hence equation 4 can be written as

$$F(f(y)) = F(y) + \frac{1}{w(f(y))} \quad (7)$$

This is a recurrence relation in  $F$ . We arbitrarily set  $F(H) = 1$  and obtain  $F$  at sparse locations:  $y = \{H, H - h(H), \dots, y_0\}$ . Next we interpolate  $F$  using a cubic spline and finally obtain  $\theta$  as follows:

$$\theta(y) = F(y) - F(y-1) \quad (8)$$

Figure 2 shows the estimate obtained by this recurrence method and the original system. Also shown is the projected area inverse function used in [14].  $\theta(\cdot)$  will be dependent upon scene-geometry but it needs to be calculated only once at system setup time. The quasi-calibration prior coupled with position invariance helps us deal with a variety of viewpoints, thereby satisfying the viewpoint robustness requirement.



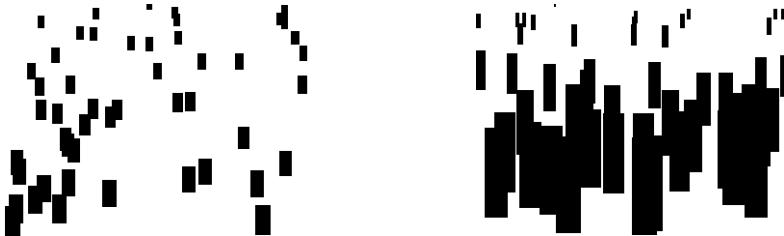
**Fig. 2** Weight function (calculated versus projected area based)

For  $N > 1$ ,  $C$ , the crowdedness measure obtained from equation 4 above will approximately equal  $N$  if the people do not occlude each other. However, if there are occlusions,  $C$  will not be unique, but can be described by a probability distribution function  $P(C|N)$ . The entropy of  $P(C|N)$  will depend upon the camera angle and be lowest for a top-down view. We estimate this pdf by simulating  $N$  humans in various configurations and degrees of overlap and calculating  $C$  using the resulting binary image and the scene-specific weight function we calculated in section 2. This process allows the inclusion of more detailed human body models and specific perturbation processes to the binary image. For example, if the sensor noise characteristics are available we could incorporate them into the simulation process. Similarly, structured perturbations such as shadows, reflections, and carried luggage can also

be introduced into the simulation process, allowing the scaling up to more complex scene conditions. While we do not attempt to mathematically aim for invariance to such perturbations, including them into the simulation and mapping process allows us to be robust to them. The essential output of the simulation process is an estimate  $P(C|N)$ . Let the maximum number of people in the scene be  $N_{max}$ . The simulation process produces  $P(C|i), 1 \leq i \leq N_{max}$ . At runtime, Bayes rule is used to find the posterior:

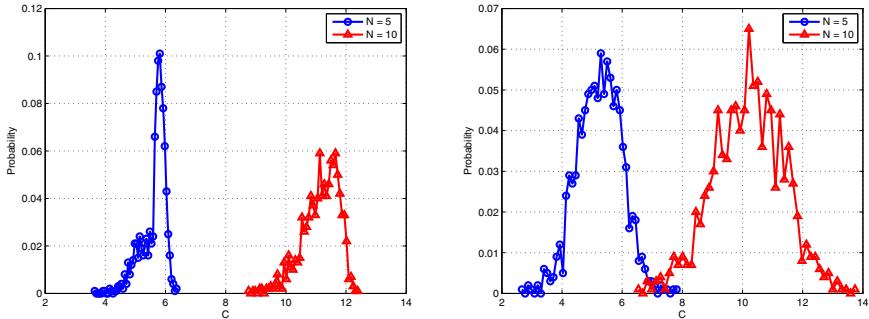
$$P(N|C) = \frac{P(C|N)P(N)}{\sum_{i=1}^{N_{max}} P(C|i)P(i)} \quad (9)$$

We further reduce the computational burden at run time by storing the posterior (rather than the likelihood) in a look up table. Hence, all that needs to be done at run-time is the calculation of  $C$  and a lookup into the table to obtain  $P(N|C)$ . We also approximate  $P(N|C)$  as a normal distribution and simply store the mean and standard deviation in the table. This has been found to work quite well in practice.



**Fig. 3** Example Images showing high-ceiling (left) and low-ceiling (right) camera positions

Figure 3 shows simulated images from a camera position higher up in the ceiling and a general view (top row). Figure 4 shows  $P(C|N)$  for  $N = 5$  and  $N = 10$  for the respective views using simulations. As expected, the separation between the two pdfs is better for the approximately top-down view, which also has a lower variance. It can be noted that there is a slight overestimate introduced due to the weight-function calculation procedure we used, resulting in  $C$  not exactly equaling  $N$ . The overestimate does not have a practical impact because we use equation 9 to estimate  $N$  from  $C$ . It can be recalled that most view-based training is done automatically at system setup-time by simulating different numbers of people-silhouettes in different configurations and establishing the distribution  $P(C|N)$ . While simulations can take account for a lot of variability in people appearances, we still found it necessary to fine-tune the crowd count output to handle unmodeled effects. This is done by comparing the output to manually annotated ground truth and estimating a scale and offset that can best match the two. In practice, while the amount of data needed for the fine-tuning is small, it is essential that different levels of crowding be present in the data. The validation numbers reported in section 8 are from fine-tuning the



**Fig. 4** Example results for  $P(C|5)$  and  $P(C|10)$  using simulations for high-ceiling (left) and low-ceiling (right) camera positions

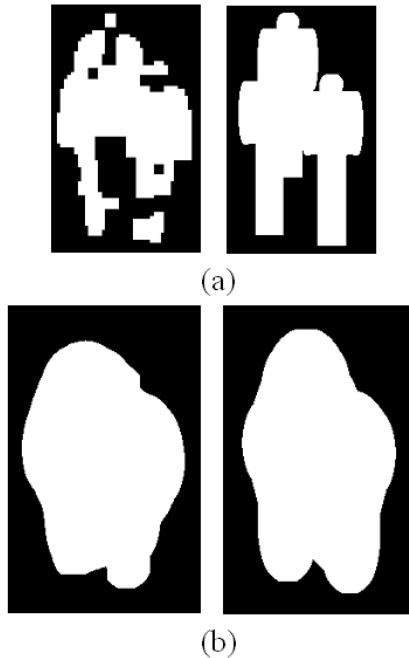
mapping based on a few hours of ground truth data covering different levels of crowding.

## 5 Person Detection and Localization

The goal of this component is to detect and localize people in the scene. Similar to the crowd count estimation component we start with the change image  $B(x,y)$ . We use our previously published method [3] to carry out the task of detecting and localizing people. For completeness, we briefly review the essentials of the algorithm leaving out mathematical details which can be found in [3]. The input to the algorithm is the binary change mask image  $B$  and the quasi-calibration. The output is the number and location of people. For an example, please see figure 5. A key goal in the work was fast operation so that real time operation is possible. To this end, the solution is calculated in two stages: First, a quick guess is calculated which



**Fig. 5** Crowd Segmentation Input and Output



**Fig. 6** Comparison of two input images of same parameter. (a) input image: left is from real video (b) right is from simulation. (b) filtered output.

maps the size and shape of the foreground silhouette (e.g. as seen in the left image in figure 5) to the number and positions of people in the silhouette. We refer to this stage as an *indexing* stage which produces a solution that is quite close to the true solution. The approximate solution is refined, as necessary, using a Markov Chain Monte Carlo based search method (specifically the method of [23]).

Key sources of variability are the relative poses of people in the silhouette, their articulation, and errors in change detection due to camouflage (i.e. where the foreground and background appearances are similar) and sensor noise. These variabilities are handled by filtering the change mask using a morphological filter of rectangular shape. The filtered shape is represented by Fourier descriptors of the contour. Figure 6 shows the filtering process on a real image and an idealized image. The resulting filtered shapes are quite similar in appearance and result in similar values of the Fourier descriptors. Prior to deployment, given the quasi-calibration, the system goes through a training phase where it generates silhouettes containing different numbers of people, articulations and positions and builds a look up table that maps Fourier descriptors of the morphologically filtered silhouettes to the number and positions of people in the silhouette. In order to limit the size of the look up table, the system has an upper bound of six people in a given blob (note that the scene may have several such blobs).

## 6 The Queue Analysis System

We are now ready to describe the overall queue analysis system. As discussed in section 2 the queue analysis system can be composed by combining two modules - a queue module (QM) that estimates the number of people waiting in the queue and a counter module (CM) that estimates the service time, i.e. the time required to obtain service at a counter. QM is built using the crowd count estimation component to estimate a probability distribution over the total number of people waiting. As crowd counts vary slowly over time we also found it useful to low-pass filter the crowd count estimate such that sudden changes in value were disallowed. In the scenario where a single queue is covered by multiple cameras, the probability distributions obtained from each camera are combined together.

CM is built using the person detection & localization component. Firstly zones are configured on the scene that correspond to regions where the person waits for service (e.g. as in figure 1). The person detection & localization component is used to localize people within the defined service counter zones. Typically only one person stands in the zone at a time. The system estimates the amount of time that a person waits in the service counter zone, i.e. the “service time”, and maintains statistics of this quantity over time. Errors in this process can arise from intermittent missed detections as well as false alarms. Such errors are sporadic and random in nature. These errors are handled quite effectively by a temporal smoothing process that makes use of two parameters: One is the minimum inter-arrival time between two people. Another is the minimum service time. Provided that the probability of successive miss detections over the minimum inter-arrival time, and the successive false alarm rate over the minimum service time are negligible, the system can recover from both types of error. These parameters were set based on training data from the installation.

Finally, equation 2 is used to estimate the expected waiting time of people in the queue.

## 7 Deployment Considerations

The algorithmic components that make up the queue monitoring system are designed for general views. In principle they can thus be deployed at sites where cameras have already been installed. However, where customer projects allow, it is useful to choose camera positions for best overall effectiveness. Generally, the more the number of cameras deployed, the better the coverage of the queue becomes. However, computational as well as economic constraints often limit the number of cameras that can be deployed in a real world setting. An optimal setup is one that minimizes the number of cameras required, maximizes the coverage of these cameras, and maximizes the accuracy of the system. A top-down view optimizes these different considerations (also recall from figure 3 that top-down views result

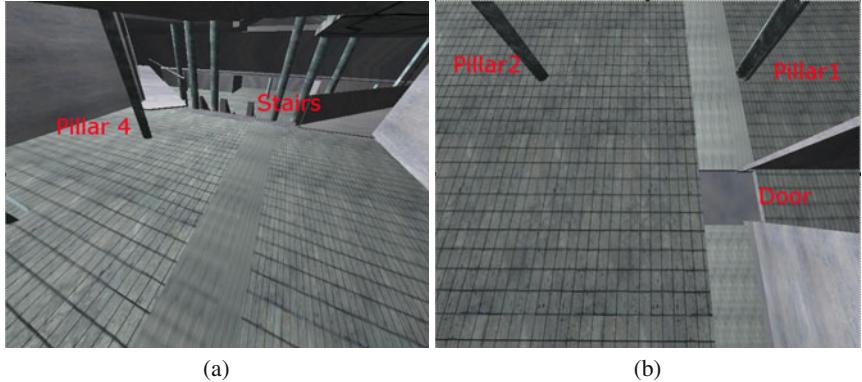
in lower uncertainty for crowd count estimation). Such views are easiest to achieve in installations sites with high ceilings. Figure 7(a) provides a good example of such a camera set up. In installation sites where ceiling height is low, cameras need to be positioned in an oblique set up. Figure 7(b) provides an example of such an installation. Such oblique views provide smaller coverage and thus such situations typically need a higher number of cameras.



**Fig. 7** Camera view of (a) high ceiling mounted camera and (b) low ceiling mounted oblique camera

For economic reasons, it is impractical to install a set of cameras, observe the goodness of the camera set up, and iteratively refine camera placement and numbers. Camera installations in major businesses like airports typically involve structural modifications to the building that make it impractical to perform anything but minor adjustments post installation. This implies that the initial recommendation provided to the camera installer needs to be as optimal as possible. One possible approach to accomplish this is to perform 3D modeling of the installation site and use this 3D model to plan for the optimal number and placement of cameras.

To construct 3D models, it is possible to use any existing CAD drawings of the building or even detailed measurements of different structures such as distances between walls, height of ceiling etc. Such 3D models can easily be constructed using 3D modeling software such as AutoCAD and even modern 3D game engines. Figures 8 shows example screen shots of the point of view of the proposed camera installation when viewed within such a 3D model. Use of such 3D models to plan the number and placement of cameras greatly minimizes costly errors in camera setup. Care must however be taken to account for structures in the building not typically modeled in CAD drawings. Examples of such structures include overhead signage in an airport (as can be seen in Figure 7(b)), ceiling mounted display monitors etc.



**Fig. 8** Simulated camera views from 3D model of target installation site.

## 8 Validation Process

The queue analysis system was validated on data from a baggage screening checkpoint at an airport. The queue in this scenario spanned multiple large areas and required five cameras whose views were mostly non-overlapping. On the other hand a single camera was sufficient to view the baggage screening zones (shown in figure 1). The system was required to flag situations when the average waiting time exceeded a maximum value or was lower than a minimal threshold. In general, validation requires balancing two competing needs: On the one hand, in order to obtain a realistic assessment of the performance, validation data needs to be representative of the range of situations that will occur in practice. It needs to cover different levels of illumination, different sizes of crowd, different queue formations, and different wait times. On the other hand, data collection and ground truthing costs need to be kept as low as possible. To this end, we selected eight cumulative hours per camera that spanned a range of operating conditions. Figure 9 shows an example from the camera covering the largest number of people. The variability shows low and high crowding, strong illumination effects, and evolving queue structures.

As crowd counts are expected to vary slowly over time, we reduced ground truthing costs by annotating at a low frequency (typically every 1000 frames, which corresponds to about 40 seconds) and interpolating the data for intervening times. Annotations were done for the following four quantities:

1. Number of people.
2. Average Service Time.
3. Waiting Time.
4. The number of counters open and closed.

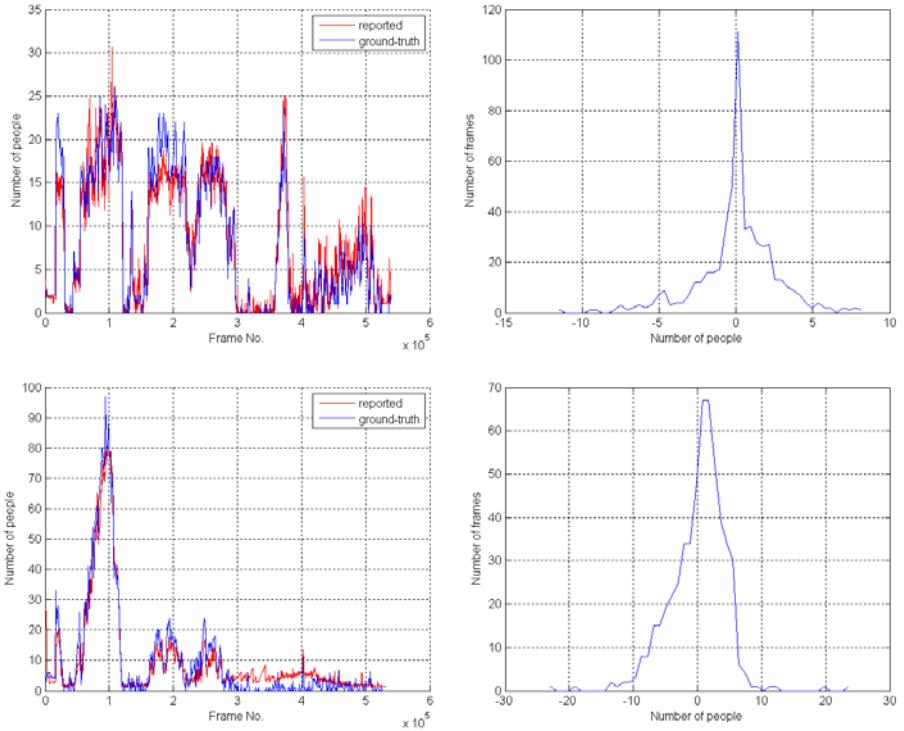
Although there were a total of five cameras viewing the queue, most of the time, people were concentrated in two cameras, the results for which are shown in figure 10. The left column shows the trend for the number of people seen in the camera. The blue line shows the actual number of people while the red line shows the number



**Fig. 9** Different Scene Conditions Included for Validation. Clockwise starting from top-left: Low Crowding, High Crowding, Strong Illumination Effects, Different Queue Structures.

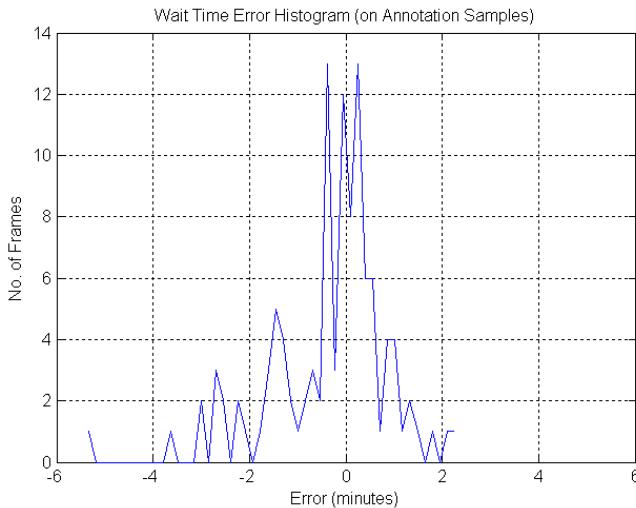
of people estimated by the system. The right column shows the error histograms for the first two cameras. Overall, it can be seen that the ground truth and the estimates agree fairly well. For the first camera (top row) which holds up to 30 people, the absolute error was less than 6 people for 96% of the time, and the average absolute error was 1.72 people. For the second camera (bottom row) which holds up to about 100 people, the absolute error was less than 8 people for 96% of the time, and the average absolute error was 3.25 people.

Although the counts agree reasonably well in practice, upon closer examination, it can be seen that there are errors at certain times. We traced these errors to two main sources. Firstly some errors occurred due to insufficient calibration. Recall that we only ‘quasi-calibrated’ the scene. Radial distortion contributed to errors when people were near the boundary of the scene (the heavy-crowd case). Secondly, sudden illumination changes due to sunlight streaming in from the ceiling were sometimes not fully compensated by the system, which resulted in residual error (especially for the second camera). Nevertheless the amount of error was small enough to have negligible impact on the overall waiting time.



**Fig. 10** Crowd Count Comparision and Error Histograms (best viewed in color)

There are two parts to service time estimation. Firstly, the time required for a person to pass through a given counter zone is needed (i.e. counter specific service time). Secondly, it is also required to determine the open/closed state of each service counter. For the specific deployment scenario under consideration, namely baggage screening at an airport, it was determined externally that the per counter-screening time held fairly constant at about 15 to 17 seconds on average. This is to be expected because the type of ‘service’ is the same. Consequently, it was decided to use the externally determined service time at this particular deployment of our queue solution. Determining the open/closed state of each counter was still necessary, and for this task we used the person detection and localization algorithm of section 5 to determine if a person occupied a given counter zone. The accuracy of counter open/close detections was found to be 96%. Finally, figure 11 shows the error histogram for the average waiting time estimation. The average absolute error was found to be 0.8736 minutes. Recall that the waiting time is a statistic derived from the queue and counter modules, which in turn rest upon the illumination robust change detection module. It is therefore to be expected that errors in any of the foundational modules will propagate up to the waiting time calculation. Besides the two sources of error described above that adversely impacted the crowd count estimation



**Fig. 11** Wait Time Error Histogram

component, for this particular deployment, we also traced some error down to the fluctuations in the service time. Nevertheless, as the impact on the overall waiting time was minimal, this source of error was ignored for this particular deployment.

## 9 Conclusions

This chapter presented the design and validation of a system for automatic, video based, people queue statistics estimation. In particular, we designed a system that estimates statistics over the total number of people waiting, the service time per counter, the number of operational service counters, and the waiting time. Such statistics are very useful not only for optimized resource allocation and improved customer satisfaction, but also to understand trends over longer periods of time, thus providing valuable input for business intelligence.

In practice, a queue monitoring system is expected to handle successfully a wide variety of variation in scene conditions arising due to several classes of influencing variable. Mainly these include illumination, queue structures, person appearances, viewpoints, etc. We followed a systematic approach for the design of the system. We first modeled the queue at an abstract level that allowed us to identify the need for two key modules - a queue monitoring module that estimated the total number of people waiting in the queue, and a service-counter monitoring module that estimated the service time per counter as well as determined which service counters are open. The modules are in turn built upon the algorithmic components of crowd count estimation, and person detection & localization. Robustness to illumination variation is largely achieved by a foundational change detection module we have previously reported in [15] and [17], allowing us to focus on the remaining variables

including queue structure, person appearance, and viewpoint. By design the algorithmic components do not need knowledge of the queue structures and are hence automatically robust to different queue formations. We imposed position invariance into the formulation, which, along with quasi-calibration information allowed us to achieve robustness to different viewpoints. The training process for both algorithmic components included simulations of different person appearances and allowed us to achieve robustness to variability in person appearances. We also discussed important factors to consider for the deployment of the queue monitoring system. We validated the queue monitoring system on representative data from a real airport. The data spanned a wide variety of conditions expected under normal operation of the system at the airport. The results demonstrated the overall effectiveness of the queue monitoring system.

We believe that several factors contributed to the successful design of the queue monitoring solution: (1) The systematic approach for the analysis of a queue and the design of the solution based on that analysis that also helped avoid the need to solve harder problems such as long range tracking of people in crowded situations, (2) The systematic enumeration of key sources of variability in the scene and ensuring that they are all addressed either mathematically or via training, and (3) A modular approach which allows us to ‘divide and conquer’ the space of variability as well as more easily pinpoint weaknesses and areas for further work on the system.

**Acknowledgements.** The authors would like to thank Imad Zoghla and Thomas Baum for helpful discussions and valuable feedback as well as Ningping Fan for his contribution to the project. The authors would also like to thank the airport authorities for providing permissions to use the images included in this article.

## References

1. Chan, A.B., Liang, Z.S.J., Vasconcelos, N.: Privacy preserving crowd monitoring: Counting people without people models or tracking. In: Proc. IEEE IEEE Conference in Computer Vision and Pattern Recognition (CVPR), June 2008, pp. 1–7 (2008)
2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR 2005, pp. I:886–I:893 (2005)
3. Dong, L., Parameswaran, V., Ramesh, V., Zoghla, I.: Fast crowd segmentation using shape matching. In: Proc. ICCV (2007)
4. Felzenszwalb, P.: Learning models for object recognition. In: CVPR 2001, pp. I:1056–I:1062 (2001)
5. Gavrila, D.M.: Pedestrian Detection from a Moving Vehicle. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1843, pp. 37–49. Springer, Heidelberg (2000)
6. Gavrila, D., Philomin, V.: Real-time object detection for smart vehicles. In: ICCV 1999, pp. 87–93 (1999)
7. Huang, C., Al, H., Wu, B., Lao, S.: Boosting nested cascade detector for multi-view face detection. In: ICPR 2004, pp. II:415–II:418 (2004)
8. Kilambi, P., Ribnick, E., Joshi, A.J., Masoud, O., Papanikolopoulos, N.: Estimating pedestrian counts in groups. In: Computer Vision and Image Understanding, pp. 43–59 (2008)

9. Kong, D., Gray, D., Tao, H.: Counting pedestrians in crowds using viewpoint invariant training. In: Proc. British Machine Vision Conference, BMVC (2005)
10. Leibe, B., Seemann, E., Schiele, B.: Pedestrian detection in crowded scenes. In: IEEE CVPR 2005, San Diego, CA, May 2005, pp. 878–885 (2005)
11. Dalal, N., Triggs, B., Schmid, C.: Human Detection using Oriented Histograms of Flow and Appearance. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 428–441. Springer, Heidelberg (2006)
12. Mohan, A., Papageorgiou, C., Poggio, T.: Example-based object detection in images by components. PAMI 23(4), 349–361 (2001)
13. Papageorgiou, C., Evgeniou, T., Poggio, T.: A trainable pedestrian detection system. In: Intelligent Vehicles, October 1998, pp. 241–246 (1998)
14. Paragios, N., Ramesh, V.: A mrf-based approach for real-time subway monitoring. In: Proc. IEEE IEEE Conference in Computer Vision and Pattern Recognition (CVPR), pp. I:1034–I:1040 (2001)
15. Parameswaran, V., Singh, M., Ramesh, V.: Illumination compensation based change detection using order consistency. In: Proc. IEEE CVPR (2010)
16. Shet, V., Neumann, J., Ramesh, V., Davis, L.: Bilattice-based logical reasoning for human detection. In: CVPR (2007)
17. Singh, M., Parameswaran, V., Ramesh, V.: Order consistent change detection via fast statistical significance testing. In: Proc. IEEE CVPR (2008)
18. Tuzel, O., Porikli, F., Meer, P.: Pedestrian detection via classification on riemannian manifolds. IEEE Transactions on Pattern Analysis and Machine Intelligence 30, 1713–1727 (2008)
19. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2001 (2001)
20. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: ICCV 2003, pp. 734–741 (2003)
21. Wu, B., Nevatia, R.: Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In: ICCV, Beijing (October 2005)
22. Zhao, T., Nevatia, R.: Bayesian human segmentation in crowded situations. In: CVPR, vol. 2, pp. 459–466 (2003)
23. Zhao, T., Nevatia, R.: Segmentation and tracking of multiple humans in crowded environments. In: IEEE PAMI, vol. 30(7), pp. 1198–1211 (2008)
24. Zhu, Q., Yeh, M., Cheng, K., Avidan, S.: Fast human detection using a cascade of histograms of oriented gradients. In: CVPR 2006, pp. II:1491–II:1498 (2006)

# Author Index

- Bazzani, Loris 271  
Bialkowski, Alina 199  
Chaudhry, Rizwan 69  
Choe, Tae Eun 309  
Collins, Roderic 241  
Cristani, Marco 271  
Denman, Simon 161, 199  
Duncan, Kester 133  
Fookes, Clinton 161, 199  
Gunda, Kiran 309  
Guo, Guodong 101  
Gupta, Himaanshu 309  
Haering, Niels 309  
Hakeem, Asaad 309  
Hoogs, Anthony 241  
Ivanov, Yuri 69  
Kanaujia, Atul 309  
Khemlani, Amit 133  
Menegaz, Gloria 271  
Micusik, Branislav 43  
Murino, Vittorio 271  
Paggetti, Giulia 271  
Parameswaran, Vasu 355  
Perera, A.G. Amitha 241  
Pflugfelder, Roman 43  
Picus, Cristina 43  
Porikli, Fatih 3  
Ramesh, Visvanathan 355  
Rasheed, Zeeshan 309  
Ryan, David 161  
Sarkar, Sudeep 133  
Scanlon, Andrew 309  
Shet, Vinay 355  
Sridharan, Sridha 161, 199  
Swears, Eran 241  
Tosato, Diego 271  
Turek, Matthew 241  
Venetianer, Peter 309  
Yilmaz, Alper 3  
Yu, Li 309  
Zhang, Zhong 309