Tracking Moving Objects in Surveillance Video

Peter John Dunne

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy at the University of Central Lancashire, UK.

April 2012

Declaration

Concurrent registration for two or more academic awards

I declare that, while registered as a candidate for the research degree, I have not been a registered candidate or enrolled student for another award of the University or other academic or professional institution.

Material submitted for another award

I declare that no material contained in this thesis has been used in any other submission for an academic award and is solely my own work.

Signature:	
Type of award:	Doctor of Philosophy
School:	Computing, Engineering and Physical Sciences

Abstract

The thesis looks at approaches to the detection and tracking of potential objects of interest in surveillance video. The aim was to investigate and develop methods that might be suitable for eventual application through embedded software, running on a fixed-point processor, in analytics capable cameras.

The work considers common approaches to object detection and representation, seeking out those that offer the necessary computational economy and the potential to be able to cope with constraints such as low frame rate due to possible limited processor time, or weak chromatic content that can occur in some typical surveillance contexts.

The aim is for probabilistic tracking of objects rather than simple concatenation of frame by frame detections. This involves using recursive Bayesian estimation. The particle filter is a technique for implementing such a recursion and so it is examined in the context of both single target and combined multi-target tracking.

A detailed examination of the operation of the single target tracking particle filter shows that objects can be tracked successfully using a relatively simple structured grey-scale histogram representation. It is shown that basic components of the particle filter can be simplified without loss in tracking quality. An analysis brings out the relationships between commonly used target representation distance measures and shows that in the context of the particle filter there is little to choose between them. With the correct choice of parameters, the simplest and computationally economic distance measure performs well. The work shows how to make that correct choice. Similarly, it is shown that a simple measurement likelihood function can be used in place of the more ubiquitous Gaussian.

The important step of target state estimation is examined. The standard weighted mean approach is rejected, a recently proposed maximum a posteriori approach is shown to be not suitable in the context of the work, and a practical alternative is developed.

Two methods are presented for tracker initialization. One of them is a simplification of an existing published method, the other is a novel approach. The aim is to detect trackable objects as they enter the scene, extract trackable features, then actively follow those features through subsequent frames. The multi-target tracking problem is then posed as one of management of multiple independent trackers.

Contents

List of Figures ix

List of Algorithms x

List of Symbols xii

List of Acronyms xiv

1 Introduction 1

1.1 Intelligent video analysis 1

1.2 Aims and constraints 2

1.3 Approaches to object detection for tracking 5

1.4 Probabilistic vs concatenative tracking 6

1.5 Thesis structure 7

1.6 Contributions 8

1.7 Mathematical notation 9

2 Object detection and representation 11

2.1 Introduction 11

2.2 Detection approaches based upon background subtraction 12

2.2.1 Identification of temporally consistent pixel values 13

2.2.2 Temporally consistent pixel regions and colour edges 14

2.2.3 Temporal median of pixel values 15

2.2.4 Gaussian representation of background pixel values 16

2.2.5 Mixture of Gaussians 17

2.2.6 Multimodal mean 17

2.2.7 Local region histograms 18

2.2.8 Local region classifiers 18

2.3 Approaches using temporal frame differencing 19

2.3.1 Using statistics of differences 19

2.3.2 Looking for object boundary evidence 20

2.3.3 Regions of consistent optical flow 20

2.4 Object detection using target model fitting and tracking 21

2.4.1 Generalized cylinders 21

2.4.2 Overlapping ellipsoids 22

2.4.3 Coaxial cylinders 22

2.4.4 Simple human model and mean-shift 23

2.5 Approaches based upon salient object characteristics 23

2.5.1 Haar wavelets used with a Support Vector Machine 23
2.5.2 Haar wavelets used with Adaboost 25
2.5.3 Edge detection 27
2.5.4 Intensity gradients 28
2.6 Target representation 29
2.6.1 Colour histogram representation 29
2.6.2 Spatiograms 31
2.6.3 Edge orientation histograms 32
2.6.4 Multi-part histograms 32
2.7 Discussion 33

3 Sequential Bayesian Estimation 35
3.1 Introduction 35
3.2 Bayesian recursion 35
3.3 The Kalman filter 38
3.4 The particle filter 40
3.4.1 Filter degeneration 44
3.4.2 Re-sampling 46
3.5 Particle filter variants 46
3.6 Discussion 49

4 Multi-target tracking 50
4.1 Introduction 50
4.2 Multi-target Bayesian recursion 51
4.3 Classical approaches 52
4.4 Extensions to Condensation 55
4.4.1 Single multimodal pdf 55
4.4.2 Probabilistic exclusion principle 56
4.4.3 BraMBLe 57
4.5 Mixture particle filters 58
4.5.1 The mixture tracker 58
4.5.2 The boosted particle filter 60
4.5.3 MCMC particle filter 61
4.6 Joint Multi-target Probability Density 62
4.7 The Probability Hypothesis Density filter 64
4.8 Discussion 69

5 Particle filter implementation 71
5.1 Introduction 71
5.2 Basic tools for tracking 72
5.2.1 Simple histogram based representation 72
5.2.2 Bhattacharyya distance and the likelihood function 73
5.2.3 Integral histogram 75
5.3 Particle filter stages 78
5.3.1 Initialization 78
5.3.2 Process step 79
5.3.3 Measurement 81
5.3.4 State estimation 83

5.3.5 Re-sampling 83
5.4 General behaviour of the simple implementation 84
5.4.1 Dealing with depth views 85
5.4.2 Occlusion behaviour 85
5.4.3 Inadequacy of weighted mean estimate 86
5.4.4 Process noise feedback 87
5.5 Improving the representation 88
5.5.1 Using a simple structured histogram 91
5.6 Discussion 91

6 Distance measures and particle likelihood 93
6.1 Introduction 93
6.2 Choice of Gaussian likelihood standard deviation in published work . 93
6.3 Choice of the likelihood standard deviation for particle weighting and re-sampling 94
6.4 Distance measures other than Bhattacharyya 98
6.4.1 Histogram Intersection distance 99
6.4.2 The χ^2 distance 102
6.4.3 Choosing an appropriate histogram distance 104
6.5 Likelihood functions other than Gaussian 105
6.6 Investigating the effects of using the simplified distance/likelihood combination 107
6.7 Discussion 111

7 State estimation and particle re-sampling 113
7.1 Introduction 113
7.2 State estimation 114
7.2.1 Using post-measurement processing 114
7.2.2 Approximate particle based MAP estimator 115
7.2.3 Testing the Driessen and Boers MAP estimator 116
7.2.4 Mode selection in the presence of other trackers 118
7.3 Re-sampling 123
7.4 Discussion 128

8 Tracker initialization 129
8.1 Introduction 129
8.2 Background image for initial detection 131
8.3 Mean-shift based object detection 134
8.4 Histogram based detection 139
8.4.1 Constructing an object indicator histogram 140
8.4.2 Finding a histogram to track 143
8.4.3 Tracker initialization and subsequent management 144
8.5 Discussion 146

9 Conclusions and further work 148
9.1 Review of aims and findings 148
9.1.1 Approaches to object detection 149
9.1.2 Consideration of theoretical foundations 150

9.1.3 Multi-target tracking 150
9.1.4 Implementation of the particle filter 151
9.1.5 Choice of similarity distance and particle likelihood 151
9.1.6 State estimation 152
9.1.7 Particle re-sampling 152
9.1.8 Object detection for tracker initialization 152
9.2 Further work 153

A Video sequences linked to the figures **155**

B Publications **157**

References **158**

Acknowledgements

I would like to acknowledge the crucial support given by my supervisor Bogdan Matuszewski. He always managed to find time, within a very busy schedule, to discuss theoretical and practical aspects. His insightful guidance, questioning, and insistence on attention to detail, has been invaluable in the development of the work.

I would also like to thank Mike Newton, of AD-Group, for having the vision to provide time and resources to allow me to pursue the study.

My colleagues at AD Group have been a valuable source of inspiration and support. Both Mike Newton and Richard Patel had the foresight to see how the company's research and development work would be enhanced by the academic links. Wei Wang often gave time to look at papers related to the work, to correct my misinterpretations, offer suggestions for improvement to techniques, and to act as a moderating influence at times when my proposals were a little too speculative. He experimented with aspects of the particle filter, integral histogram implementation and likelihood calculation timings, using the Chipwrights simulator, and gave valuable feedback.

I am grateful for the contributions of my colleagues at UCLAN, especially ChunXiao Zhang , Wei Quan, and Xin Chen. They gave time to make me feel at home in the department. I benefited from their contributions and feedback in presentations and reading group sessions.

I am indebted to my wife Joan, and children Anna and Michael, for their tolerance and support. They allowed me the space and the time to concentrate on the task when I should really have been paying attention to them.

List of Figures

1.1 Example frames from sequences used 4

2.1 Background subtraction: (a) intensity image, (b) blob image 13
2.2 Edge detecting Haar wavelets: (a) vertical, (b) horizontal and (c) diagonal 24
2.3 Haar wavelet convolution responses: (a) training image, (b) vertical edge wavelet, (c) grey scale representation of wavelet response 24
2.4 Chamfer transform: (a) pedestrian image, (b) edge image (Sobel) (c) transform distances 28
2.5 Illustrating target variability. (a - c) showing different object shapes, (d - e) same objects some frames later 30
2.6 The Epanechnikov kernel 30

3.1 Illustrating Kalman filter behaviour 40
3.2 Illustrating multinomial re-sampling using 10 weights: (a) weight values, (b) weight re-selection using the cumulative weights, the yellow points indicate sampling numbers $u^{(k)}$. The horizontal dotted lines from the yellow points indicate weight index selection. 45

4.1 PHD particle weights using simulated measurements, Overhead sequence 68

5.1 Variation of histogram similarity in the vicinity of the target 72
5.2 Histogram Bhattacharyya distance scan across target: (a) frame 6, (b) frame 8, (c) distance scan across target, same frame, (d) original histogram scan across target 2 frames later. 75
5.3 Illustrating the tracking rectangle with dimensions height $h = |C - A|$, width $w = |D - A|$; and the integral histogram bin extraction for $u = 2$ 76
5.4 Matlab comparison of multiple histogram extractions: direct calculation(red points), integral histogram calculation(blue points) 77
5.5 Illustrating the particle sequence 80
5.6 Gaussian weightings for the particle set 81
5.7 Temporal difference motion feature 82
5.8 Examples of simple 8 bin histogram tracking with manual initialization 84
5.9 Occlusion behaviour with simple 8 bin histogram tracking 86
5.10 Failure of weighted mean state estimate with large process noise . . . 87
5.11 Improved target representation using a 32-bin four-quadrant histogram. 89
5.12 Improved representation response using the four-quadrant (4q) histogram. 90

6.1 Gaussian weighting of the Bhattacharyya distance measure for a typical set of 512 particles from a tracking sequence. The standard deviation of the distribution is set to $\sigma = 0.2$. The curves to the left and right of the central one have standard deviations of $\sigma = 0.1$ and $\sigma = 0.3$ respectively. The lower figure shows the frequency distribution of the distances. 95
6.2 Illustrating the relationship between Histogram Intersection and Bhattacharyya distances 100
6.3 Plot of Histogram Intersection distance components against the corresponding Bhattacharyya distance components for 512 x 8 $\{p, q\}$ bin pairs. 101
6.4 Looking at the relationship between the χ^2 and the Bhattacharyya distances 105
6.5 Scattered distribution of χ^2 distance weightings with a Gaussian likelihood, using $\sigma = 0.35$ 106
6.6 Using a simple triangular likelihood function 107
6.7 Composite images and ground truth to tracker bounding box overlap data for the sequences. Blue ($\cdot$) points - Gaussian likelihood\Bhattacharyya distance, red (+) points - triangular likelihood\Histogram Intersection distance 110
6.8 Illustrating the track differences for Gaussian/Bhattacharyya (left) and triangular/Histogram Intersection (right) 112

7.1 Tracking similar targets using the Driessen and Boers MAP 117
7.2 Sort and count tracking for crossing similar targets 121
7.3 Tracker mode identification by sort and count 122
7.4 Comparison of re-sampling processes: examples using 10 weights . . . 125
7.5 Comparison of weight distribution before (a) and after re-sampling (b) 127

8.1 Comparison of Approximating Median and Mosaic of Means background development. It can be seen that by the 100th frame the Mosaic of Means has cleared the foreground content of frame 1. By the 260th frame the Mosaic of Means remains clear, whereas the Approximating Median shows signs of traffic contamination. 133
8.2 Illustrative Beleznai mean-shift start points and convergence 135
8.3 Pad based mean-shift detection 136
8.4 Mean-shift detection with target shadow 138
8.5 Mean-shift detection as a source for a PHD filter 139
8.6 Trigger frames for pad histogram based detection 140
8.7 Histograms used in pad proposal histogram construction 141
8.8 Particle filter type search around the pad and initial four quadrant tracking region 143
8.9 Initialized trackers in the i-LIDs sequence 144
8.10 Tracker initialization with multiple triggers 145
8.11 Tracker initialization, Overhead sequence 145

List of Algorithms

1 Multinomial resample 47
2 SIR particle filter - with multinomial re-sampling 48
3 Particle PHD filter 67
4 Extract dominant and subsidiary mode 120
5 Produce ordered random numbers 124
6 RSR re-sampling 126
7 Mosaic of short term means 132

List of Symbols

Operations

$\lvert a-b \rvert$	Absolute difference between a and b
$\lVert \mathbf{a}-\mathbf{b} \rVert$	Euclidean distance between $\mathbf{a}$ and $\mathbf{b}$
$\lfloor a \rfloor$	Rounding down of a
Δ	Difference
θ	Threshold
$\ast$	Convolution
$\cdot *$	Pixel-wise multiplication
$\bigwedge$	Multi-variable logical conjunction

Intervals and sets

$[a,b]$	Closed interval $a \leqslant x \leqslant b$
(a,b)	Open interval $a < x < b$
$[a,b)$	Half open interval $a \leqslant x < b$
$\{\ \}$	Set

Functions

$f(\cdot)$	Motion model
$g(\cdot)$	Measurement model
$\delta(\cdot)$	Dirac delta function

Vectors

$\mathbf{x}_t$	Target state at time t
$\mathbf{z}_t$	Measurement at time t
$\mathbf{X}_t$	Multi-target state at time t
$\mathbf{Z}_t$	Multi-target measurement at time t
$\boldsymbol{\nu}$	Independent identically distributed process noise
$\boldsymbol{\eta}$	Independent identically distributed measurement noise
$\mathbf{p}$	Candidate histogram
$\mathbf{q}$	Reference histogram
$\lVert \cdot \rVert$	Vector norm

Scalar variables

n_s	Number of states
n_{eff}	Effective number of states
n_m	Number of measurements
n_τ	Number of targets
ρ	Histogram dissimilarity distance
w_t^i	Un-normalized weight associated with particle i at time t
$\tilde{w}_t^i$	Normalized weight associated with particle i at time t
$\hat{w}_t^i$	Proposal weight associated with particle i at time t

Images and Matrices

I_t	Intensity image at time t, or current image
$I\left(x, y\right)$	Image pixel value at coordinates x, y
$I_{B,t}$	Background image at time t
$I_{\Delta B}$	Difference image formed by background subtraction
$I_{\Delta B\theta}$	Binary image formed by background subtraction and thresholding
$I_{\Delta t}$	Difference image formed by subtraction of temporally successive frames
$I_{\Delta t\theta}$	Binary image formed by subtraction of temporally successive frames and thresholding
F	State transition matrix
M	Measurement matrix

Statistical quantities

μ_t	Mean value at time t
σ_t	Standard deviation at time t
$\mathcal{N}(\mu, \sigma)$	Normal distributed random variable
$\mathcal{U}\left[0, 1\right]$	Uniform distributed random variable over the closed interval
$\mathrm{E}\left[\,\cdot\,\right]$	Expectation

Probability density functions

$p\left(\mathbf{x}_t \vert \mathbf{x}_{t-1}\right)$	State transition
$p\left(\mathbf{z}_t \vert \mathbf{x}_t\right)$	State measurement likelihood
$p\left(\mathbf{x}_t \vert \mathbf{z}_{1:t}\right)$	State posterior at time t

List of Acronyms

APF	Auxiliary Particle Filter
AnPF	Annealed Particle Filter
ANPR	automatic number plate recognition
CCTV	closed circuit television
DBT	detect-before-track
DSP	digital signal processor
EKF	Extended Kalman filter
EM	Expectation Maximization
EMD	Earth Mover's Distance
fps	frames per second
KDE	Kernel Density Estimate
FISST	Finite Set Statistics
KF	Kalman filter
GPF	Gaussian Particle Filter
HOG	Histogram of Oriented Gradients
HSV	Hue-Saturation-Value
JMPD	Joint Multi-target Probability Density
JPDAF	Joint Probabilistic Data Association Filter
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo
MHT	Multiple Hypothesis Tracking
MOG	Mixture of Gaussians
MTT	multiple-target tracking

MRF	Markov random field
NN	nearest neighbour
pdf	probability density function
PF	particle filter
PHD	Probability Hypothesis Density
RFS	random finite set
RGB	Red, Green, Blue
RPF	Regularized Particle Filter
RSR	residual systematic re-sampling
SAT	summed area table
SIFT	Scale Invariant Feature Transform
SIMD	single instruction multiple data
SIR	Sampling Importance Re-sampling
SMC	Sequential Monte Carlo
SoC	System on Chip
SVM	Support Vector Machine
TBD	track-before-detect
VMD	video motion detection
VSP	video signal processor
YUV	luminance chromacity

Chapter 1

Introduction

1.1 Intelligent video analysis

The development of automatic intelligent video content analysis [1], or video analytics, is an important goal for both the commercial and public sector surveillance industry [2, 3]. The field of computer vision has made some progress towards that goal in the last twenty years. The detection and tracking of well defined objects such as faces is now a standard feature in many low cost digital cameras; video systems capable of automatic number plate recognition (ANPR) and analysis are commonplace on motorways, controlled car parks and petrol station forecourts. The ubiquity of such systems might suggest that machines are approaching animal vision capabilities. The reality is that the success in the fields of face detection and ANPR has been reliant on the particular distinctiveness and repeatability of the features being detected. Computational systems in the field of general object detection and tracking are still very far from those developed by nature.

The aspirations and expectations of the closed circuit television (CCTV) and surveillance community for video analytics are high. The mainstay of the CCTV industry remains that of recording footage for post-event forensics. If it were possible for systems to recognize image sequences with potential content of interest, within hours of relatively event free video, then they might be able to intelligently adjust recording rates and increase the efficiency of image storage. Alternatively, analytics systems might simply produce a record of frame numbers and times at which events of interest may have occurred, so that scrutiny of the recorded footage might be made a little more efficient. A current common approach is to use triggers based upon video motion detection (VMD). This works by monitoring predefined regions within the video frame and simply looking for increased pixel variability. But video analytics aims for much more than simply archiving footage showing increased pixel intensity variation. Potential content of interest might be unusual behaviour of objects in the field of view, such as running, loitering, fighting, increased crowd activity, area perimeter violation, illegal parking or vehicle driver behaviour etc.

In addition to event logging, automatic recognition of this content might be used to alert CCTV control room operators with the responsibility of monitoring multiple feeds in real-time leading to greater observational efficiency. Detection and tracking of objects of interest can also lead to statistical analysis of video content to determine commercially useful information such as navigational strategies of customers in stores, consumer pause behaviour linked to either product placement or promotional signage etc.

Analytics applications such as those described above require the detection and tracking of objects that are more variable and less well defined than faces and vehicle licence plates. Pedestrians and vehicles, for example, can present a variety of appearances and colours dependent upon angle of view, perspective and lighting. The intense academic activity in the analytics related fields during the last twenty years has led to a wide range of avenues being explored with some impressive theoretical advances [4, 5, 6]. But despite those theoretical advances there has been relatively little successful transfer to commercially useful analytics products.

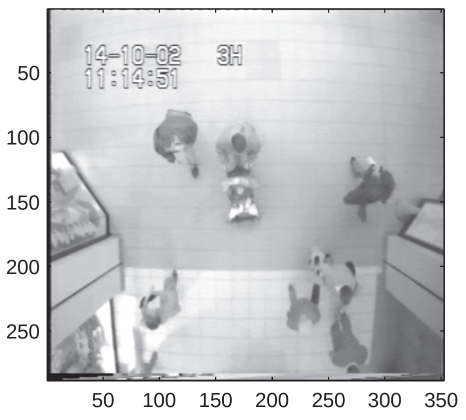
1.2 Aims and constraints

The overall aim of the work was to examine approaches to object detection and tracking with a focus on the development of computationally economic analytics-related algorithms for possible product development in specific surveillance CCTV systems [7]. The eventual analytics real-time applications would be expected to run as embedded software in a particular digital signal processor (DSP). It was not an aim of the work to develop the embedded software itself; it was, instead, to look for pathways from the recent theoretical advances to practical interpretations, approximations and simplifications in order to increase the chances of commercially useful implementation.

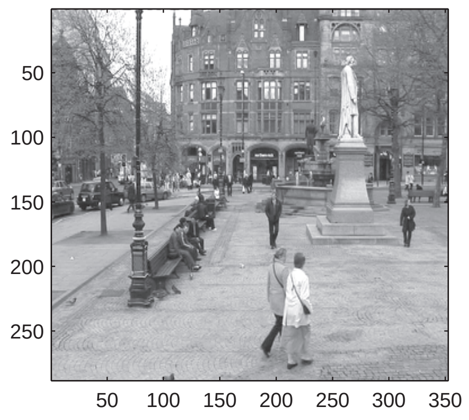
The constraints were defined both by the architecture of the processor and the typical circumstances in which the analytics software would be expected to operate. The processor of interest was from the CW5XXX family of video signal processors (VSPs) [8]. The CW5631 System on Chip (SoC) processor was designed specifically for imaging applications. It is a single instruction multiple data (SIMD) device providing sixteen 32-bit parallel processor data paths that can work off a single instruction. The SoC contains a fixed-point DSP and an ARM RISC processor for control operations and system code. The main role of the processor is to deal with video capture from the imaging sensor, carry out basic image processing and camera related tasks such as contrast adjustment, MPEG compression etc. Any analytics software would have to share processor time with those tasks. The maximum processor speed is 360MHz, so processing time would be limited in comparison to PC based systems. An initial design aim was for a working rate of 5 frames per second (fps) for analytics applications. The low frame rate would put constraints on the techniques available: calculations of optical flow, for example, are not going to be reliable at those frame rates.

The fixed-point nature of the DSP meant that operations like division, extraction of square roots, calculations of logarithms and trigonometric functions etc. are going to be time consuming, so an aim was to find approaches that do not rely heavily on such calculations. The absence of floating point representation meant that it will be prudent to avoid approaches that might be sensitive to arithmetic rounding errors. The SIMD nature of the processor meant that algorithms that are parallelizable would be attractive; algorithmic stages that are not readily parallelizable would lead to computational bottlenecks. Analytics software for CCTV security and surveillance applications can expect to meet situations in which chromatic information is weak [1]. Street sodium lighting can return a near monochromatic image. Similarly, fluorescent lighting in public areas such as subways can return little colour information. This, coupled with the tendency for people to adopt low-key colours for their clothing, and for some locations to have a predominance of overcast weather, suggests that it would be prudent to develop a tracker capable of operating with intensity information only. Systems developed to operate effectively with intensity only input have the added commercial attractiveness of the potential to be used with legacy monochromatic systems and with the developing infra-red camera technology [9].

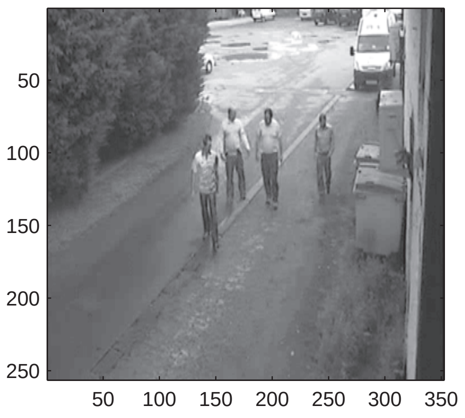
The video sequences used for the exploration of the techniques were chosen to reflect the constraints and present particular challenges. Example frames from some of the sequences are shown in Figure 1.1. The 'Overhead' sequence, Figure 1.1(a), shows targets that change shape significantly as they move through the view. There is a step change in the background illumination (around row 200) such that the targets show an additional appearance change as they move across the step. In addition there are illumination changes as the camera auto-iris responds to sunlight on the floor in the upper half of the frame. The 'Square' sequence, Figure 1.1(b), offers a perspective view with pedestrian targets having varying heights in the frame. Two of the targets have similar appearance and their paths cross. The overlapping pedestrians in the foreground of the frame have clothes items with colors close to that of the background. The 'Yard' sequence, Figure 1.1(c), involves a group of pedestrians with similar appearance. They move closely and overlap in some frames.



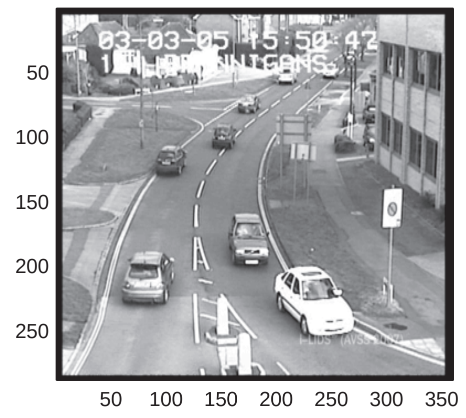
(a) Overhead



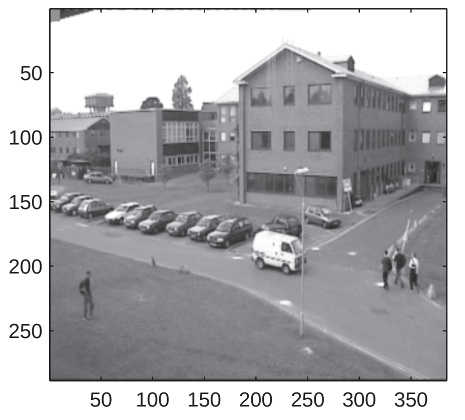
(b) Square



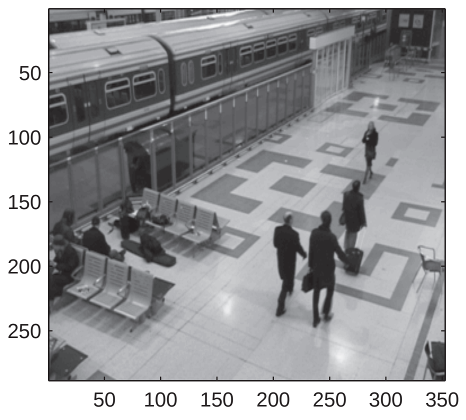
(c) Yard



(d) iLIDs



(e) PETS2001



(f) PETS2006

Figure 1.1: Example frames from sequences used

The ‘iLIDs’ sequence [10], Figure 1.1(d), focuses on vehicles for which the size varies significantly as they move through the frame. The sequence involves some slight camera movement and sunlight illumination changes. The ‘PETS2001’ sequence [11], Figure 1.1(d), offers a mixture of pedestrians and vehicles. The pedestrians are relatively small in the image and hence involve the need for the system to track small groups of pixels. The vehicle moves across the frame, stops, and then reverses into the road junction. It presents a significant change of appearance, over a short sequence of frames, as it moves out of the junction. The ‘PETS2006’ sequence [12], Figure 1.1(d), presents a group of pedestrians, with similar and relatively uniform appearance, that overlap as they move past each other.

The aim is to produce a single tracking system capable of dealing with the variety of appearances found in such sequences.

1.3 Approaches to object detection for tracking

Animal vision systems have evolved to respond to changes against a background scene, focus attention on that change and track it [13]. They are capable of quickly evaluating the change and determining if it is associated with a known object class or otherwise. There is no fundamental requirement that artificial vision systems should have algorithmic architectures directly drawn from animal ones but the broad principles do apply: objects of interest show up in frame-to-frame differences and image regions that differ from background. One approach to object tracking is to detect those differences, find some features of the object linked to those differences and track the signature.

An alternative approach is to start from the characteristics of a known object class and carry out frame-to-frame searches for evidence of that class. Once an object of interest is detected, keeping track of it can reduce the search area and help to resolve difficulties when it might become temporarily occluded. Detection and tracking complement each other, high-quality detection helps to maintain reliable tracking, reliable tracking can help to produce high-quality detection. The difficult step is that of extracting high-quality trackable characteristics at the point where a target of interest is first detected. Cannons [4] points out it is very common in the research field to see tracker initialization carried out either manually or via a separate detection module. Commercial systems cannot have the convenience of manual tracker initialization, they must automatically respond, discriminate and track.

Typical commercial systems rely on static camera views, background scene modeling and ‘blob’ production by thresholded subtraction of a background reference image from the current video frame. While background subtraction blob-based systems are common, they can suffer significantly at the blob interpretation step. The blobs do not necessarily correspond to complete targets, multiple targets can merge into single blobs. Simple tracking of blob centroids is not sufficient for analytics application. The property of being ‘not background’, however, is one of a number of features that can be drawn upon for potential target identification.

With this in mind the thesis starts by considering the common background modeling approaches found in the literature. It recognizes that animal vision supplements the ‘not background’ and ‘consistent motion’ features by interpreting the information in terms of known object models. It is common in tracking to try to fit ‘top down’ information, such as templates or contours, onto the blob patterns [4]; examples are considered where appropriately sized simple shapes are used in this way. In attempts to be free of difficulties linked to background modeling some tracker initialization and update approaches have been based upon salient characteristics of target objects; examples of such approaches are considered in order to illustrate their computational overheads.

1.4 Probabilistic vs concatenative tracking

In cases where targets are expected to be well separated and the image preprocessing returns distinct blobs corresponding to the targets then tracks can be constructed by connecting blob centroids with their nearest neighbour in the preceding frame. A similar process can be used if object positions are extracted in the salient feature approaches. But the aim of tracking is much more than simply constructing paths by concatenation of target detections, its role is to produce a predictive system in which a model of the motion can be developed.

Comaniciu’s popular mean-shift approach to tracking [14] has concatenative characteristics; it builds on recent observations using the last frame object position as the starting point of a gradient-descent style optimization movement towards the position in the current frame. It does not have an in-built mechanism for speculative search. The previous-frame object tracking window has to overlap the target in the current frame for the process to succeed. Fast moving or occluded objects are likely to be lost.

Probabilistic tracking of objects in video sequences allows the system to use information from recent frames to predict the location in the current frame. This limits the object detection search space so that computational resources can be efficiently used. It also helps in the resolution of occlusion issues when similar objects pass each other, or when tracked objects disappear behind obstacles.

The sequential nature of video frames means that Bayesian probability approaches will be appropriate. In a Bayesian approach the probability density of a proposed target position can be constructed using a transition probability, derived from the estimated target position in the preceding frame, complemented by a likelihood factor derived from measurements in the current frame.

1.5 Thesis structure

The thesis aims to explore all aspects of the target tracking task. This involves making decisions about the most appropriate target representation, efficient tracker maintenance, whether to track targets individually or to see individual target trackers as components in a composite tracking system. An overview of object detection techniques is presented in Chapter 2. It is a context-setting treatment. It focuses on approaches that have become standard and form the basic toolkit for workers in the field. It allows for the development of broad concepts and terminology that can be drawn on in later chapters.

Chapter 3 looks at the general basics of sequential Bayesian estimation. The mathematics of Bayesian recursion is reviewed and details of approaches that implement the recursion are given. In the restricted case in which prediction and measurement error are assumed to be Gaussian, and the prediction and measurement functions are linear, the recursion can be implemented using the Kalman filter (KF). In cases where the linearity constraints are not fully met the Extended Kalman filter (EKF) can be used. Both the KF and the EKF require that a separate detection, or measurement, step be used to deliver the proposed target position. The task of the filters is then to return the best estimate of state and uncertainty given the detections. In the case of noisy measurements the best estimates of state can return a relatively smooth trajectory through the scatter. In the event of occlusion or other track loss events the trajectory can be maintained until reliable detection is recovered.

A more general method that can incorporate the measurement step directly in its implementation, and return estimates of state and trajectory, is the particle filter (PF). This is a Sequential Monte Carlo (SMC) method that does not rely upon function linearity or Gaussian uncertainty. The particle filter is introduced, a broad picture of its implementation is developed, and an indication is given of some variants found in the literature.

Chapter 4 presents a selection of approaches to the multi-target tracking problem. The aim is to give an overview of the range of approaches developed in attempts to deal with multi-target tracking problems that arise in different contexts, and to consider if the approaches, or elements of them, should be adopted in the context of this work.

Given the scene setting aspects of chapters 2 to 4, Chapter 5 takes the first steps in looking at practical tracker implementation issues. It looks at each stage of the filtering process, and illustrates the stages with simple single tracker examples.

Chapter 6 makes a deeper consideration of implementation practicalities with a critical examination of the choice of particle filter parameters, likelihood functions and distance measures etc. The aim is twofold: to see if any choice of parameters or functions is more likely than others to give more reliable tracking, and to consider if computationally more attractive simplifications of the functions could be applied.

Chapter 7 focuses on the state estimation and particle re-sampling steps. The consideration of the re-sampling step is an examination of approaches in terms of their quality and computational demand. The consideration of the state estimation step is motivated, on one hand, by a recent claim of a theoretically optimal approach, on the other hand by its importance in the issue of maintaining tracker identity in multi-target situations with occlusion and possible track coalescence.

Chapter 8 considers practicalities of tracker initialization. The overview of object detection and representation techniques in Chapter 2 led to the conclusion that only limited aspects of them would be usable given the constraints of the problem being considered. Two practical alternative detection approaches are suggested for the identification of trackable objects. One of the approaches calls for background image construction; a simple and effective method is described.

Overall conclusions, followed by suggestions for further development, are given in Chapter 9.

1.6 Contributions

The work suggests that object tracking can be carried out effectively in low-resolution grey scale video using simple particle filters. It examines the choice of feature distance measures, likelihood functions and parameters in the basic particle filter and concludes that, if the parameters are chosen appropriately, the choices of distance measure and likelihood function are not critical. The findings were published in the journal *Image and Vision Computing* with the title '**Choice of similarity measure, likelihood function and parameters for histogram based particle filter tracking in CCTV grey scale video**' [15].

It is common in the application of the particle filter to choose the weighted mean of the states as the filtered state. It is also common to see that choice being criticized in the field. In Chapter 7 a recent (2008) suggestion [16] of a theoretically sound particle-based *maximum a posteriori* (MAP) estimator is examined and found to be not adequate in practice. A practical alternative is developed, based upon the recognition that target probability density functions (pdfs) often are characterized by only a couple of dominant modes. The alternative simply identifies the dominant and the strongest sub-dominant modes and chooses the one of them most likely to belong to the tracked object. It is shown that this mode selection process works well with crossing and closely moving similar targets.

In Chapter 8 a practical simplification of an established pedestrian detection technique is described. In addition, an alternative and novel histogram based detection method is developed. The method was presented at the 'International Conference on Emerging Security Technologies 2011' and published in *International Journal of Grid and Distributed Computing* with the title '**Histogram-based detection of moving objects for tracker initialization in surveillance video**' [17]

1.7 Mathematical notation

It is a feature of the field of study that a consistent symbolic convention for the description and labeling of images and object tracking processes has yet to emerge. There are almost as many conventions as there are academic groups working in the field. Different groups often use the same notation to describe different things, and many differ in the notation they choose to describe things that are common across the groups. An attempt has been made in this thesis to describe the work of others using a consistent mathematical notation.

Scalar variables are described using European or Greek letters in lower case normal font e.g. n_s for number of states, ρ for histogram dissimilarity distance etc. A lower case subscript t is used to indicate both continuous or frame stepped time. Lower case superscripts are used to label members of a set e.g. w_t^i indicates the i^{th} weight at time t. Where there might be confusion between a label index and a mathematical power the index is enclosed in brackets e.g. $w_t^{(i)}$.

Vectors are described using lower case bold font e.g. $\mathbf{x}_t$ for target state at time t, $\boldsymbol{\nu}$ for multi-dimensional process noise etc. Matrices are described using upper case normal font. All images are described using a normal font upper case I and their type is indicated using subscripts. For example, the Greek Δ indicates differences so a difference image produced by background subtraction would be $I_{\Delta B}$, a binary image produced by background subtraction and then thresholding with a threshold θ would be $I_{\Delta B\theta}$. A temporal difference image will be $I_{\Delta t}$, but to indicate an image at time t the time subscript is separated from the description subscripts by a comma e.g. $I_{\Delta B,t}$ to represent the image produced by background subtraction at time t.

In order to be consistent with the notation used in the majority of papers in the field of multi-target tracking, the multi-target state and measurement are described using upper case bold letters e.g. $\mathbf{X}$ and $\mathbf{Z}$ respectively. In some works the multi-target state represents a set of single target states, in others it is a vector formed by concatenation of single object states, and in others it is a matrix. The notation chosen embraces all of the alternatives and it is assumed that the interpretation can be inferred from the context.

There are a few instances where it is clearer to use a research group's original notation; in such situations the alternative symbolic convention is described in the text.

Chapter 2

Object detection and representation

2.1 Introduction

The aim of this chapter is to provide an overview of the object detection 'toolkit' that has underpinned approaches to the development of algorithms for object tracking in video. Most of the approaches generally have been associated with desktop PC based implementation with access to GHz processing and no restrictions on floating point operations. Their transferability to a more restricted processing environment is limited, but a consideration of their principles of operation is of value. It provides the opportunity for informed rejection of some approaches whilst selecting useful elements of some of them for practical application. Object detection can have a role both in the initialization of trackers and the subsequent update of the trackers as the object moves through the scene, although it is not essential that the detection method is the same in both cases.

A good proportion of commercial analytics products developed to date, such as virtual tripwires and object trackers, appear to use static cameras and background subtraction, so a significant section of the chapter, Section 2.2, is spent considering basic background modeling methods. Alternatives based upon temporal frame differencing are considered in Section 2.3. Section 2.4 looks at approaches that fit shapes to blob patterns resulting from background subtraction

In recognizing the limitations of background subtraction based methods other research paths have led to systems designed to look for specific characteristics of objects of interest; Section 2.5 looks at methods that involve image search for salient object characteristics. Some particle filter based trackers have incorporated such approaches in the initial detection stage so it is of interest to outline the principles of the methods.

It is common, in published work, to see studies of tracker development and behaviour after manual initialization of the track. A representation, or signature, of the target is extracted at the initial step and then subsequent frames are searched for evidence of that representation. The outcome of the search then constitutes the frame detection. The representation is updated to accommodate slight variations evidenced by that outcome. Vision related particle filters, being based upon a likelihood calculation, tend to be representation based. Section 2.6 looks at a range of methods that have been used to extract a trackable target representation.

2.2 Detection approaches based upon background subtraction

In background modeling based target identification a representation of the image background is developed and subtracted from the current image in order to produce foreground. It is generally assumed that both the background scenery and the camera remain fixed in position. A background reference image I_B is built up over a sequence of frames. A binary 'blob' image $I_{\Delta B}$ is made by pixel-wise subtraction between the background and the current image frame I_t and then thresholding the absolute differences:

$$I_{\Delta B}(x, y) = (|I_B(x, y) - I_t(x, y)| > \theta) \tag{2.1}$$

where θ is a scalar threshold.

Drawing on the psychophysical definition of visibility of an object, Fuentes and Velastin [18] chose to use luminance contrast instead of intensity difference. Working with the luminance component of the luminance chromacity (YUV) color system, they defined luminance contrast as the relative difference between object luminance Y_t and the surrounding background luminance Y_B i.e.

$$C(x, y) = \frac{Y_t(x, y) - Y_B(x, y)}{Y_B(x, y)} \tag{2.2}$$

As the expected component values were in the range $[0, 255]$ they redefined null (zero) values for the background to be 1 to avoid infinities in the contrast. They found that a manually selected absolute contrast threshold value of between 0.15 and 0.2 produced satisfactory segmentation.

A problem with background subtraction, especially in the case of monochrome images, is that regions within the object of interest having pixel values close to background will not be identified. It is generally necessary to apply to the blob

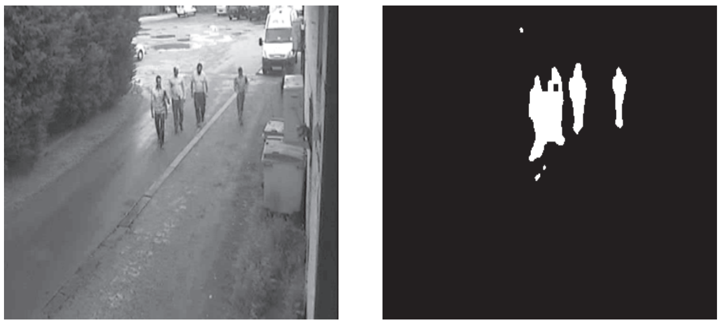


Figure 2.1: Background subtraction: (a) intensity image, (b) blob image

image morphological operations of dilation, erosion, hole filling etc. to remove noise and produce well defined regions for further analysis. A typical resulting binary image is shown in Figure 2.1(b).

Background modeling approaches that are commonly quoted in CCTV surveillance analytics systems are identification of pixel temporal consistency, approximating median, Gaussian representation, mixture of Gaussians, local region histograms and local region classifiers. The underlying principles associated with those approaches are described in the following sections.

2.2.1 Identification of temporally consistent pixel values

An early method for background reference image construction involved a simple classification of pixel values into those associated with probable moving objects and those that are probably part of the fixed scene. Collins et al. [19] determined that a pixel could be classified as belonging to a moving object if the ‘three frame’ differences were greater than a threshold i.e.

$$|I_t(x,y) - I_{t-1}(x,y)| > \theta_t(x,y) \quad \wedge \quad |I_t(x,y) - I_{t-2}(x,y)| > \theta_t(x,y) \tag{2.3}$$

If the absolute differences were less than the threshold, resulting in a ‘non-moving’ classification, then the pixel value could be classed as background. The initial background was taken to be the first image, i.e. $I_{B,0}(x,y) = I_0(x,y)$ and the initial threshold $\theta_0(x,y)$ was manually set to a non-zero value. With the assumption that the variation of background pixel intensity over time might be described by a normal distribution, the threshold was developed to be an approximation to 5 times the standard deviation of the pixel value. When the temporal differences indicated that the pixel should be classed as ‘non-moving’ the background value was fractionally updated with the current intensity value, otherwise the background value was unchanged i.e.

$$I_{B,t}(x,y) = \begin{cases} \alpha I_{B,t-1}(x,y) + (1-\alpha)I_t(x,y) & x \text{ is non-moving} \\ I_{B,t-1}(x,y) & x \text{ is moving} \end{cases} \tag{2.4}$$

The threshold was updated in a similar way using the absolute deviation of the intensity value from the current background:

$$\theta_t(x,y) = \begin{cases} \alpha\theta_{t-1}(x,y) + (1-\alpha)(5 \times |I_t(x,y) - I_{B,t}(x)|) & \text{x is non-moving} \\ \theta_{t-1}(x,y) & \text{x is moving} \end{cases} \tag{2.5}$$

where α is a constant determining the update rate.

In a given frame the pixels classed by the temporal differences as ‘moving’ were clustered into connected regions. Holes in the resulting ‘blobs’ were filled using background subtraction to identify the additional pixels associated with the moving objects.

2.2.2 Temporally consistent pixel regions and colour edges

A related approach was used by Piscaglia et al. [20], but in this case the foreground-background weighting was a little more complicated. Pixels in the incoming frame $I_t(x,y)$ were subtracted from the current background $I_{B,t}(x,y)$ to produce a background subtraction image. The incoming frame was also subtracted from the immediately preceding frame to produce the temporal difference image. A moving object presents foreground intensity values at its leading edges and uncovered background values at its trailing edges. By comparing pixel values in the temporal difference image and the background subtraction image, and then using a region growing process to find ‘flat’ intensity zones in the background and current images, i.e. groups of pixels that are locally connected by intensity uniformity, they were able to assign a background probability weight $W_{I_t}(x,y)$ to current frame pixels. The new background pixel value was a weighted combination of current frame and recent background values:

$$I_{B,t}(x,y) = \frac{W_{I_{B,t-1}}(x,y)I_{B,t-1}(x,y) + W_{I_t}(x,y)I_t(x,y)}{W_{I_{B,t-1}}(x,y) + W_{I_t}(x,y)} \tag{2.6}$$

$$W_{I_{B,t}} = W_{I_{B,t-1}}(x,y) + W_{I_t}(x,y) \tag{2.7}$$

The approach was developed further by Cavallaro et al. [21, 22, 23]. They looked at local 5×5 pixel windows. It was assumed that background pixel temporal variation could be described by a Gaussian distribution. The squared deviation of each pixel intensity from the mean of the background values within the window was calculated. The sum of the 25 pixel squared deviations from the mean in the window could then be expected to have a χ^2 distribution. For foreground object detection the segmentation threshold could then be set as a standard statistical significance value based upon a χ^2 test.

Object detection was also reinforced by focusing on the consistency of colour edges in the separate Y, Cr, and Cb channels of the video. The absolute differences between each channel in the current frame and those in the background reference frame were extracted. A Sobel edge detector was used on the differences. The edge maps from each channel were then fused using a logical *or* operator. This allowed the identification of edges that would be lost in a straight intensity subtraction where different hues might have similar brightness. The resulting fused edge map was then successively dilated and eroded to close the contours and fill in the region.

2.2.3 Temporal median of pixel values

Using a simplifying assumption that in a sequence of video frames a pixel value might represent the background more frequently than it represents an object moving through the scene, one could take either the mode or the median of the sequence of values to represent the background. It would, however, demand memory space in order to store a representative batch of frames from which to extract the statistics. To avoid the need for accumulating a batch of pixel values, McFarlane and Schofield [24] maintained a running estimate of the median, referred to as the 'Approximating Median', by incrementing the background pixel value by 1 if the corresponding pixel values in the current image is greater than background, and decrementing by 1 if it is less than it. This estimate eventually converges to a value for which half of the input pixel values are larger than and half are smaller than it i.e. the median.

The principle was extended in the $\Sigma - \Delta$ (i.e. sum-difference) background estimation method described by Manzanera and Richefeu [25] by applying the same principle to the absolute deviation of the current pixel value from the background in order to maintain an approximating standard deviation. This was then used for segmentation purposes: pixel values that were more than one standard deviation from the median were taken to be foreground.

A drawback to the Approximating Median approach is its slow response to change and progressive contamination by slow moving foreground objects. If an initial background representation is being built up from a starting frame, with objects moving to reveal background, then an intensity difference having a numerical value Δ between background pixel and an object pixel, with unit pixel value increments, would require a sequence of frames Δ long to adapt.

The slow update means that it is sensitive, like other background schemes, to rapid global illumination changes. Vijverberg et al. [26], using the Approximating Median, compensated for fast illumination changes by first computing the current background differences $I_{\Delta B,t}(x, y) = I_t(x, y) - I_{B,t}(x, y)$, then building a histogram of differences $\mathbf{h}_\Delta[u]$ with $u \in [-255, 255]$ for all pixel locations. They applied a best fit Gaussian, a mixture of Gaussians or a Laplacian to the histogram. They then segmented the foreground $I_{F,t}(x, y)$ if the measured differences $I_{\Delta B,t}(x, y)$ differed significantly from the mean μ_t of the fitted distribution i.e:

$$I_{F,t}(x, y) = \begin{cases} 255 & \text{if } |I_{\Delta,t}(x, y) - \mu_t| > \theta \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

2.2.4 Gaussian representation of background pixel values

An alternative to recursive median or mean update is to explicitly represent the background pixel value by a Gaussian i.e. $I_{B,t}(x, y) \sim \mathcal{N}(\mu_t(x, y), \Sigma_t(x, y))$. The model mean μ_t and covariance Σ_t are learned from observations of the pixel value over a sequence of frames. Rather than extracting the parameters from a batch of image frames they can be estimated recursively from a sequence. Wren et al. [27], in their human body motion modeling system, modeled each background pixel as a multi-variate Gaussian in YUV colour space. The authors' main focus was that of tracking a single person in a relatively static office type environment. They identified the person by the colour and 'blob' shape in the background subtracted image. Non-person regions could then be classed as background. For each background pixel the Gaussian mean and variance were updated using simple adaptive filters:

$$\mu_t(x, y) = \alpha I_t(x, y) + (1 - \alpha)\mu_{t-1}(x, y) \tag{2.9}$$

$$\sigma_t^2(x, y) = \alpha(I_{t-1}(x, y) - \mu_{t-1}(x, y))^2 + (1 - \alpha)\sigma_{t-1}^2(x, y) \tag{2.10}$$

The absolute difference threshold in Eq.(2.1) would then be set as a simple multiple of the standard deviation i.e. $\theta(x, y) = k\sigma(x, y)$.

Bramberger et al. [28] use a similar Gaussian and simple filter approach with grey scale video in their Smart Camera Stationary Vehicle Detection system. In this case, rather than using a simple filter update, a frame buffer holds a small batch of frames from which the statistics update is drawn.

2.2.5 Mixture of Gaussians

The approaches described above assume that each background pixel can be represented by a single set of statistical parameters. In practice the distribution of values at a single pixel can often be multimodal due to periodic motion of background objects e.g. swaying tree branches, or regular passage of shadows etc. Stauffer and Grimson [29, 30] developed an adaptive method that uses a Mixture of Gaussians (MOG) to model a pixel value. It is assumed that each mode in the background values will have a relatively narrow variance and that the variances of the different components are independent. Temporal adaptation of the means and variances of the (typically 2 or 3) Gaussians are determined using an incremental version of the Expectation Maximization (EM) algorithm. Pixels in the current frame are checked against the background model with each Gaussian having a probability associated with it. If a match is found then the parameters of the matched Gaussian are updated. If no match is found then a new Gaussian is introduced into the mixture with the mean equal to the current pixel value and some initial variance assigned.

Isard and MacCormick [31] took the Gaussian principle further in their BraMBLe multiple-blob tracking system. They used a 4-component Gaussian mixture for background pixels and a 16-component mixture for foreground. They used a training image sequence with foreground objects moving against a static background in order to determine the most likely component parameter sets.

2.2.6 Multimodal mean

Apewokin et al. [32] developed a system that had similar segmentation performance to that of the MOG but operated about six times faster, required less storage per pixel and used only integer operations. Each pixel is associated with a set of up to $k = 4$ mean background representations called 'cells', i.e. 4 different background states are allowed per pixel. Each cell maintains three mean colour component values. A pixel is associated with a cell if each of the incoming frame pixel colour components i are within an error margin θ_{E_i} of the mean μ_i for that cell i.e.

$$\bigwedge_i |I_{t,i}(x,y) - \mu_{t-1,i}(x,y)| < \theta_{E_i} \tag{2.11}$$

Where $\bigwedge$ represents multi-variable logical conjunction. When a pixel is found to match a cell, the component values are used to update a running sum $S_{i=1,2,3}(x,y)$ and increment a counter $C_i(x,y)$ associated with each of the three RGB colours for that cell. If a match is found for a sequence greater than a count threshold θ_{FG} i.e.

$$\left(\bigwedge_i |I_{t,i}(x,y) - \mu_{t-1,i}(x,y)| < \theta_{E_i} \right) \wedge (C_{t-1}(x,y) > \theta_{FG}) \tag{2.12}$$

where θ_{FG} is a count below which the pixel is classed as foreground, then the sums and the counts are updated. At any given time t the component mean values for a colour component at a given pixel location are computed as $\mu_{t,i}(x,y) = S_{t,i}(x,y)/C_{t,i}(x,y)$. In the authors' experiments they used $\theta_{FG} = 3$ and $\theta_{E_i} = 30$. In order to stop the background becoming unrepresentative of gradual change they decimated the cells every 400 frames by halving both the sum and the count. The approach has similar characteristics to the MOG, having a variable number of means, but with preset measures of spread and avoiding the analytical maximization elements. Its simple counting nature makes it attractive for embedded computing applications.

2.2.7 Local region histograms

Rather than modeling the behaviour of single pixels, local region histograms look at the colour or intensity profile within a neighborhood region of pixels. This can accommodate background pixel intensity variation due to slight movement. Ko et al. [33] allowed for the region intensity histogram associated with each pixel location to be temporally updated using a simple recursive filter of the form $\mathbf{p}_t = (1-\alpha)\mathbf{p}_{t-1} + \alpha\mathbf{q}_t$, where $\mathbf{p}_t$ was the developing background histogram associated with the region, and $\mathbf{q}_t$ was the corresponding current frame pixel region histogram. Foreground segmentation was carried out by thresholding the Bhattacharyya distance [34] between the current and background histograms at each pixel.

Noriega et al. [35] describe a more sophisticated approach in which the background reference image was partitioned into 12×12 pixels overlapping squares. A colour histogram was computed for each square. The pixel contributions to the histogram were modified by a spatial Gaussian kernel such that the central region of the square carried a higher weight than those at the edges. Within the histogram the effects of bin quantization were mitigated by sharing each pixel value between bins using a Gaussian weight.

2.2.8 Local region classifiers

Li, Huang et al. [36] described a Bayesian approach using probabilities of pixel features based on colour intensity, local colour gradients and temporal colour consistency to characterize the background. The probabilities were updated frame by frame using recursive temporal updates of the form described above. They developed histograms of the principal features to allow them to select the dominant ones at any time.

Grabner and Roth [37] developed a background modeling approach using a grid of small overlapping image patches. For each of the patches a classifier was trained using Adaboost [38] with Haar features, gradient orientation histograms and local binary patterns. The grid of background classifiers was developed first in a learning phase and then updated frame by frame using on-line boosting. A similar classifier was developed for expected foreground objects and the background was updated when no foreground classes were identified at a grid position. This meant that all non-object background elements could be incorporated into the background classifier making it robust to dynamically changing scenery, illumination variations etc.

2.3 Approaches using temporal frame differencing

Temporal frame differencing offers the advantage of computational simplicity but its disadvantages make it a poor candidate for object detection. Stationary objects cannot be segmented. The change mask shows object boundary pixels at the motion leading edge and background uncovered regions (‘ghosts’) at the trailing edge. It suffers from a type of ‘aperture problem’ in which uniformly coloured regions within a moving object boundary do not show in the difference image. However, the approach has been combined with background approaches described above to reinforce segmentation decisions.

2.3.1 Using statistics of differences

Cheung and Kamath [39] combined information from both background subtraction and temporal frame differencing in order to identify foreground objects. They first found the mean μ_d and the standard deviation σ_d calculated from all the pixel absolute intensity differences between the current and preceding frame. A binary temporal difference mask was then constructed by identifying pixels for which the absolute intensity difference was a chosen number of standard deviations away from the mean i.e:

$$I_{\Delta t}(x,y) = \left(\frac{||I_t(x,y) - I_{t-1}(x,y)| - \mu_d|}{\sigma_d} > \theta_t \right) \tag{2.13}$$

The background model was maintained by tracking the pixel intensity values with a KF. The state of the KF was described by the background intensity value $I_{B,t}(x,y)$

and its temporal derivative $I'_{B,t}(x, y)$. The intensities of those pixels for which the temporal differences were less than the threshold were used as measurements for the KFs associated with the pixels.

They partitioned the image into a regular array of rectangular regions. For each region they constructed a colour histogram for those pixels identified as background. They fitted bounding ellipses to the combined leading and trailing edges in associated object blobs in the temporal difference mask. They found the nearest corresponding object in the previous frame and used the intersection between the two objects to identify likely object intensities and build a histogram of them. They then used the background and object histograms to determine the pixels within the current bounding ellipse that had the greatest probability of being not background and set them to represent the object.

2.3.2 Looking for object boundary evidence

Yoo and Park [40] used temporal differencing directly to identify foreground objects. They suggested that when an object moves in front of a uniform background, covered and uncovered regions appear in pairs around the boundaries and that they have opposite signs in the difference image. They matched components of the pairs using the Earth Mover's Distance (EMD). In this approach local regions of positively signed difference values were taken to be able to 'fill' the negatively signed regions. The cost of the filling process was proportional to the distance between the regions. Pixels from positive and negative regions were used as nodes in a bipartite graph. The capacity of each node was set to the difference associated with it and the cost of transporting values from one node to the next was set to be the Euclidean distance between corresponding positions in the difference image. The bipartite graph was solved using linear programming. To reduce the computational cost they split the image into 16×16 pixel rectangular blocks and calculated the cost of transporting the information between blocks. Successful pairing allowed them to identify blocks with significant motion. Appropriate combinations of blocks could then be classified as target objects. However, the assumption of a uniform background is a weakness in the approach.

2.3.3 Regions of consistent optical flow

Bugeau and Peréz [41] developed a change detection approach capable of segmenting and tracking objects that can be applied to both static and moving cameras. Each point in a sub-grid of image pixels was described by a 3D YUV colour vector and a 2D optical flow vector. Optical flow was calculated using the Lucas-Kanade method [42]. Points in the feature space were clustered using the mean-shift algorithm [43]. Primary object segmentation was carried out by identifying outliers against the dominant optical flow. Moving objects were finally segmented by representing the clusters as a graph in a bi-partitioning graph-cut framework [44]. The energy function that was minimized had terms describing probabilities for between-pixel colour, colour gradient, and optical flow.

2.4 Object detection using target model fitting and tracking

A difficulty with background subtraction blob-based detection is that the detected foreground regions do not always correspond to objects. The output is dependent upon the thresholding strategy used. The process can result in single objects segmenting as multiple disconnected blobs, single objects producing larger blobs due to local soft shadowing, and multiple close or overlapping objects producing single blobs. The problem can be addressed by either fitting object shape models to the blob patterns or directly computing the likelihood of a non-background region corresponding to target objects. In contrast to the methods described above, the examples described in this section also include tracking through video sequences as the segmentation is generally based upon information update from frame to frame.

2.4.1 Generalized cylinders

In the BraMBLe multi-blob likelihood approach [31] humans were modeled using a set of four horizontal discs to produce a ‘generalized cylinder’. The discs, specified by their radius and height above the ground, defined the areas in the world coordinate system corresponding to the feet position, the waist position, the shoulders and the top of the head. The transformation of the cylinder disc extrema from world coordinates to image coordinates resulted in a stretched octagonal shape in the image plane. The process required camera calibration. Using a 20-component Gaussian mixture (4 background classes, 16 foreground classes) they could assign class labels to points in a grid of 5×5 pixel rectangular regions. Object shapes were then fitted to the likelihood image and the overall coverage maximized. The objects were tracked through a multi-frame sequence using a particle filter in which the state vectors had dimensionality encompassing position, velocity and object shape, and state weights linked to the model coverage likelihood. The system was computationally intensive but the authors report that they could achieve real-time tracking, for a small number of people, using a 447MHz Pentium II workstation.

2.4.2 Overlapping ellipsoids

Zhao and Nevatia [45] modeled the human shape using overlapping ellipsoids to describe the head, torso and legs. They allowed for limited configurations of the leg ellipsoids: both legs together, left leg forward and right leg forward. Similarly, the allowed angles between the components in the various configurations were quantized into a limited set. The world plane 3D shape model was projected into a 2D mask using a camera calibration model and an assumption that the objects move in the ground plane. The identity of individuals was determined by extracting a colour histogram of the model region. The pixels contributing to the histogram were weighted using a convex kernel centred on the model. The effect was to give more weight to pixels in the centre of the object than to those on the boundary so that contributions from the background were minimized.

A Gaussian background was modeled with components in each of the three colour channels. The segmentation task was formulated as a Bayesian problem in which the most probable outcome was expressed in terms of a likelihood given a prior state. The process sought to maximize the likelihood. The prior had two parts. The first part was constructed with object area based penalties for overlapping human models and small size. It also set the expectation of the height, fatness and inclination of the models with their values being drawn from Gaussian distributions. The second part was the prior dependent on the previous frame. The temporal development of the shape components was tracked using Kalman filters. Probabilities associated with object addition and subtraction from the set were linked to distances from entrances and exits. Single object likelihoods were calculated in terms of the Bhattacharyya distances between the proposal histograms and the object and background histograms. The object transition from the last state to the current one was carried out using colour-based mean-shift. The likelihood of the whole image was maximized rather than separate maximization of individual object likelihoods. The maximum a posteriori (MAP) state space was explored using a Markov chain Monte Carlo (MCMC) approach with Metropolis-Hastings acceptance.

2.4.3 Coaxial cylinders

A simpler approach is described by Zhang, Venetianer and Lipton [46]. They used a standard background modeling of the type described in Section 2.2.1 and produced blob images by thresholded background subtraction. Their 3D human target model consisted of three coaxial cylinders representing head, torso and legs. Using a pinhole camera calibration they mapped the 3D model into the 2D image plane so that they could determine the expected object appearance for all image positions. They matched models to blobs by adjusting positions and maximizing a matching score based upon the percentage of the blob covered by the human target model, the percentage of the human model covered by the observed target and penalties based upon model overlaps. The score maximization also involved some refinement of blob appearance by filtering out peripheral pixels that might be associated with shadows and reflections. The 3D target footprint locations were tracked using Kalman filters. The system was implemented on a Texas Instruments TMS320DM642/600 DSP, in a commercial application, and ran at 10fps.

2.4.4 Simple human model and mean-shift

Beleznai et al. [47] worked on the unthresholded intensity difference image produced by current frame and background subtraction. The background reference image was constructed using the Collins background modeling method described in Section 2.2.1. They identified positions of intensity difference maxima and used those as starting points for mean-shift iteration in the difference image. They used human sized rectangular kernels initially centred on the maxima. This particular version of the mean-shift procedure involved finding the difference intensity centre of gravity of the rectangle and relocating the centre to that point. The process was repeated until the stepwise shift became insignificant. The integral image (Section 2.5.2) was used for rapid calculation of image areas. They identified proposed person positions by locating basins of convergence in the mean-shift paths. Once identified, people could be tracked by shifting from the last known position. Groups of people were segmented by looking for maximum-likelihood combinations of rectangles. The authors reported that the overall process was capable running in real-time on a 2.5Ghz PC, and worked well for separated objects, but it could be prone to track association errors when targets crossed.

2.5 Approaches based upon salient object characteristics

2.5.1 Haar wavelets used with a Support Vector Machine

Papageorgiou and Poggio [48] describe a car and pedestrian detector based upon local regional differences in image intensity typical of the target object. Their approach to pedestrian detection is considered in order to illustrate the general processes involved.

They used two dimensional Haar wavelets of the kind shown in Figure 2.2. Ap-

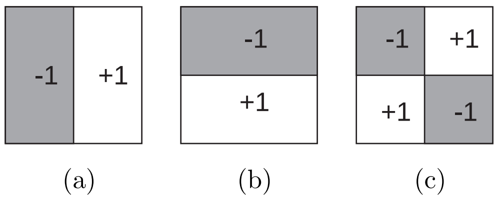


Figure 2.2: Edge detecting Haar wavelets: (a) vertical, (b) horizontal and (c) diagonal

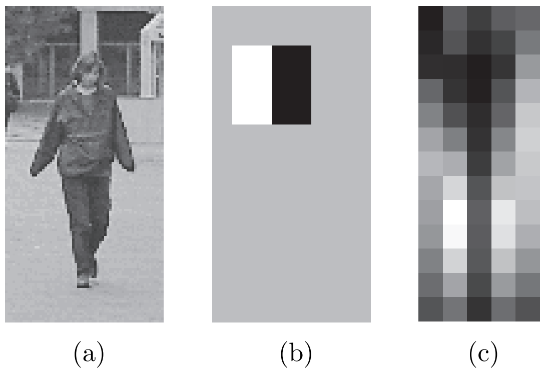


Figure 2.3: Haar wavelet convolution responses: (a) training image, (b) vertical edge wavelet, (c) grey scale representation of wavelet response

plication of a wavelet to a region within an image involves subtraction of adjacent image areas and the absolute value of the response is returned. A high wavelet response indicates the presence of an intensity edge, a weak response indicates a uniformly textured region. 1800 images of people were used, together with the images horizontally reflected, for the training phase. The images were scaled and clipped to 128 by 64 pixels. Wavelets with dimensions 16×16 and 32×32 pixels were convolved with the images using a shift of 4 and 8 pixels respectively. Figure 2.3(a) shows a typical training image, 2.3(b) shows a 32×32 Haar wavelet in one of the 65 positions that it takes as it is scanned across and down the image. Figure 2.3(c) shows a grey scale representation of the strength of the convolution response associated with that wavelet. It can be seen that the response emphasizes the symmetry of the target.

The wavelet scans resulted in 1326 dimensional feature vectors. Moving the 32×32 wavelets in steps of 8 pixels in both the vertical and horizontal directions produces 65 of the vector components, moving the 16×16 wavelets in steps of 4 pixels in both the vertical and horizontal directions produces a further 377 of the vector components. The use of three different wavelets results in a total of $3 \times (377 + 65) = 1326$.

The feature vectors, developed from both positive and negative examples, were used to train a Support Vector Machine (SVM). The SVM classifier clusters data into two classes by finding the maximum marginal hyperplane separating them. The maximized margin of the hyperplane is defined as the distance between the hyperplane and the closest data points, referred to as the 'support vectors'. The classes correspond to object and non-object.

With the characteristics of the separating hyperplane known, an image region feature vector can be assigned to one of the two classes. In order to classify regions of an image wavelet, sets are scanned across the image at different scales and feature vectors are built at each position.

The images were scaled between 0.2 and 1.5 times the original image size in increments of 0.1 to deal with varying object size in the image plane. To reduce computational cost they concentrated their search into regions likely to contain the objects of interest e.g. doorways, roadways etc.

The use of the high dimensional vector produced a slow overall response. Performance was improved by selecting a subset of 29 the strongest components: 6 vertical and 1 horizontal at the coarse wavelet scale, 14 vertical and 8 horizontal at the finer scale.

This approach to pedestrian detection has limits. Pedestrians come in a variety of shapes, garment colours and types etc. They can be occluded and have parts that can be difficult to distinguish from background. Recognizing that it might be more efficient to detect individual components, such as head, legs and arms, and then combine the detections with geometrical constraints on their positions, the team developed a wavelet and SVM based pedestrian component detector [49] and used a further SVM to classify the combinations.

2.5.2 Haar wavelets used with Adaboost

Rather than scan the wavelet across the sub-image it is more common for Haar wavelets to be used in a different way. The feature classifier is defined by fixing the position of the wavelet within the sub-image window and then determining the best positions, wavelet style and size that result in the feature response able to distinguish between class and non-class. A manually labelled training set, similar to the one described above, is used. Features selected are those that produce at least some bi-modality in the frequency distribution of responses. In the version of the classifier described by [50, 51, 52], for each wavelet labeled υ the weak classifier $h_k(\upsilon)$ consists of a feature response $f_k(\upsilon)$ and a parity p_k:

$$h_k(\upsilon) = \begin{cases} 1 & \text{if } p_k f_k(\upsilon) < p_k \theta_k \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

where θ_k is a threshold situated between the modes and p_k indicates the direction of the inequality sign i.e. whether the target mode is above or below the chosen threshold.

The object classifier is built by combining the feature classifiers. The process of selecting the most discriminatory features and combining them is known as 'adaptive boosting' or Adaboost [38]. Each selected weak classifier is given a weight $\alpha = \frac{1}{2}\log\frac{1-\varepsilon}{\varepsilon}$, indicating its discriminatory strength, where ε is a weighted measure of the error rate for that classifier. The error weight is increased for misclassified examples forcing the next feature chosen to focus more on those difficult to classify.

The final classification is carried out using a weighted sum of n_k feature classifications:

$$C(v) = \begin{cases} 1 & \text{if } \sum_{k=1}^{n_k} \alpha_k h_k(v) \geqslant \frac{1}{2}\sum_{k=1}^{n_k} \alpha_k \\ 0 & \text{otherwise} \end{cases} \tag{2.15}$$

The ubiquity of the rectangular Haar-based classifier is due to the fact that it can be computed quickly and in constant time independent of area using the 'integral image' introduced by Viola and Jones [53]. The integral image is a summed area table (SAT) computed once over the entire image such that the value at each pixel location is the sum of values from the origin to that point i.e.

$$SAT(x,y) = \sum_{i<x,j<y} I(i,j) \tag{2.16}$$

The pixel sum $S(x,y)$ over a rectangular area of size $w \times h$ based at the location (x_0,y_0) can then be extracted quickly using:

$$S(x_0,y_0) = SAT(x_0,y_0) + SAT(x_0+w,y_0+h) - SAT(x_0,y_0+h) - SAT(x_0+w,y_0) \tag{2.17}$$

Viola, Jones and Snow [51] went beyond the use of Haar wavelets with the basic pedestrian shape and incorporated motion as an additional distinguishing characteristic. They used pairs of images of size 20×15 pixels taken from successive frames in video sequences. From those they generated five further images: the temporal intensity difference image and four intensity difference images formed by subtracting the second image displaced in the horizontal and vertical directions. They applied the Haar features to the first image of the pair and to each of the difference images. They also extracted a measure of the motion in the images by a simple rectangular sum of the difference image pixel values. This would give a minimum value in the displacement image in the direction of pedestrian motion. They trained their Adaboost based classifier using this rich feature set and reported a much higher ratio of detection to false positives than using a simpler detector trained on static images. They allowed for varying pedestrian size in the image plane by running the detection over an image pyramid using a scale factor of 0.8 for each layer. The detection rate was reported as being of the order 90%.

2.5.3 Edge detection

The detectors described above operate on the basis that regions of the object of interest have average intensity values differing from the immediately surrounding regions and show characteristic classifiable patterns. Wu and Nevatia developed a part based detection approach based upon characteristic edges [54]. Their 'Edgelet' features are short linear or curved groups of pixels linked to edge shapes from head and shoulder, torso, and leg regions of the body. With the positions and normal vectors of the k points in an edgelet denoted by $\{\mathbf{u}_i\}_{i=1}^{k}$ and $\{\mathbf{n}_i^E\}_{i=1}^{k}$ respectively, and the edge intensity and normal at position $\mathbf{p}$ in the image $\mathbf{I}$ represented by $M^I(\mathbf{p})$ and $\mathbf{n}^I(\mathbf{p})$, the affinity between the Edgelet and the image at position $\mathbf{w}$ was calculated by:

$$S(\mathbf{w}) = ({}^1\!/\!_k) \sum_{i=1}^{k} M^I(\mathbf{u}_i + \mathbf{w}) \langle \mathbf{n}^I(\mathbf{u}_i + \mathbf{w}), \mathbf{n}_i^E \rangle \tag{2.18}$$

This is a sum of products of edge strengths and cosines of the angle between the Edgelet and image edge derived from the scalar product between the normals. They simplified the calculation by quantizing the angle range $[0°, 180°)$ into six bins and approximating the cosines for those bins as $\{1, {}^4\!/\!_5, {}^1\!/\!_2, 0, {}^1\!/\!_2, {}^4\!/\!_5, 1\}$.

They used a variant of the Adaboost algorithm [38] to train their feature classifiers for individual Edgelets on 1700 sample images of size 58×24 pixels. For detection they used an image pyramid with scale factor 1.2 and used a probabilistic combination of feature responses to determine the presence or otherwise of a body part. They reported a processing rate of 1 frame per second with 288×384 pixel images on a 2.8GHz processor.

Gavrila [55] describes an edge based method in which the extracted candidates are matched against a database of edge-based shape exemplars. The matching is carried out by computing the Chamfer Transform [56] of the edge image. This transform converts a binary image into a non-binary one in which each pixel value denotes a distance to the nearest edge pixel in the edge image. The transform is illustrated in Figure 2.4. Edge templates are matched to the edge image by placement of the template on the chamfer transform image and summing the chamfer values under the template edge pixels. The template is translated, rotated and scaled to find the minimum match sum. The exemplars are selected by a Bayesian

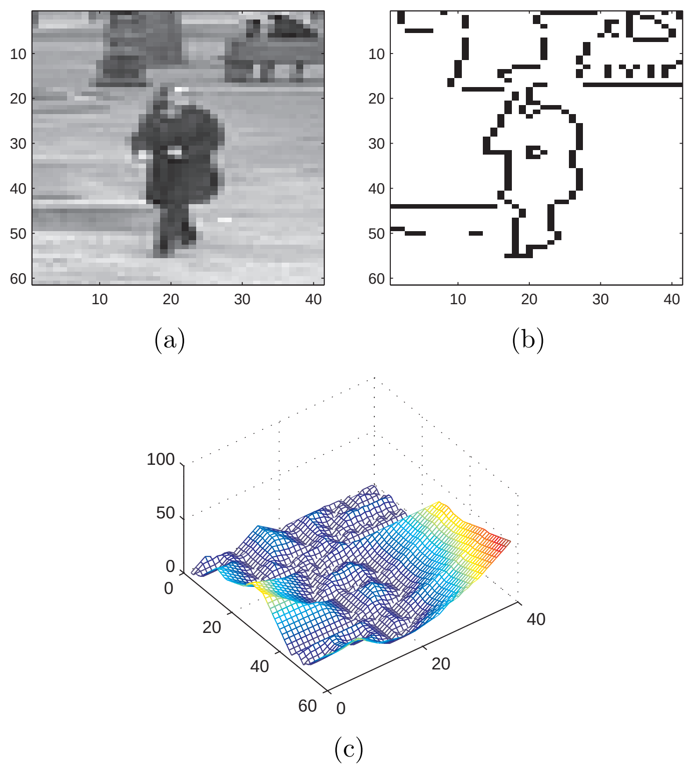


Figure 2.4: Chamfer transform: (a) pedestrian image, (b) edge image (Sobel) (c) transform distances

probabilistic stepwise path through a database tree structure until an adequate match is found. The detector ran at 7-15 Hz on a 2.4GHz processor.

2.5.4 Intensity gradients

Shapelets, developed by Sabzmeydani and Mori [57], follow a similar path to Edgelets but focus on local intensity gradient patterns rather then edge segments. Low-level shape features S_d, linked to a given direction d, are formed by convolving the intensity image I with the basic gradient kernel $G_d = [-1, 0, 1]$ and its transpose to extract gradient information in the directions $d \in \{0°, 45°, 90°, 135°\}$. The gradient magnitudes in each direction are smoothed using a 5×5 box filter B i.e.

$$S_d = |I \ast G_d| \ast B \tag{2.19}$$

where $\ast$ denotes convolution.

The detection window, similar to that in Figure 2.3(a), is subdivided into subwindows of sizes 5×5, 10×10 and 15×15 pixels. Each of the low level features within the window are used in an Adaboost training cycle to produce a weighted combination of them. The combination is the Shapelet feature. The object classifier is then constructed by using Adaboost again to produce a weighted combination of

Shapelet features within the detection sub-window. The overall effect is to produce a classifier based upon the gradient information around the periphery of the object of interest.

Dalal and Triggs also developed a human detection system based upon intensity gradient patterns [58]. The detection windows had typical size 128×64 pixels with the target centred such that there was a 16 pixel margin on all sides. They reported that the best gradient operator was the simple Sobel $[-1\ 0\ 1]$ used with no smoothing. The detection window was subdivided into 16×16 pixel blocks consisting of four 8×8 cells. For each cell an oriented gradient 9-bin histogram was extracted. The orientations were quantized within the range $[0,\ \pi)$. Each pixel voted for its gradient orientation weighted by its gradient magnitude with the vote interpolated linearly between neighbouring bins. Neighbouring cells were grouped into the blocks and the bin values normalized. Blocks were stepped through the window with an 8-pixel stride and the histograms concatenated to form a high dimensional Histogram of Oriented Gradients (HOG) feature vector. The vectors were then used to train a SVM classifier.

2.6 Target representation

The problem with all of the methods described up to this point is that they assume that the objects of interest can each be described by a relatively small range of characteristic shapes. But, in practice, trackable objects expected to be of interest in a surveillance setting can be subject to significant unpredictable deformation and rotation. Figure 2.5 shows some typical examples. From a commercial point of view it is better to find systems that are applicable in a range of situations and can be adapted easily to new ones. It would be of value to develop a single approach that could be applied to objects as dissimilar as vehicles and pedestrians. It is true that object classifiers of the type described in Section 2.5 can be adapted to any object through appropriate training or template development. But commercially it is inconvenient to develop products that need tailoring to specific situations, so the aim is to find a target representation that is generalizable and can cope with variability such as object deformation and rotation. Colour and edge histograms have been studied as potential representations capable of responding to target variation.

2.6.1 Colour histogram representation

Updateable colour histogram based representations are capable of distinguishing a trackable object from background. They can be independent of object type and can

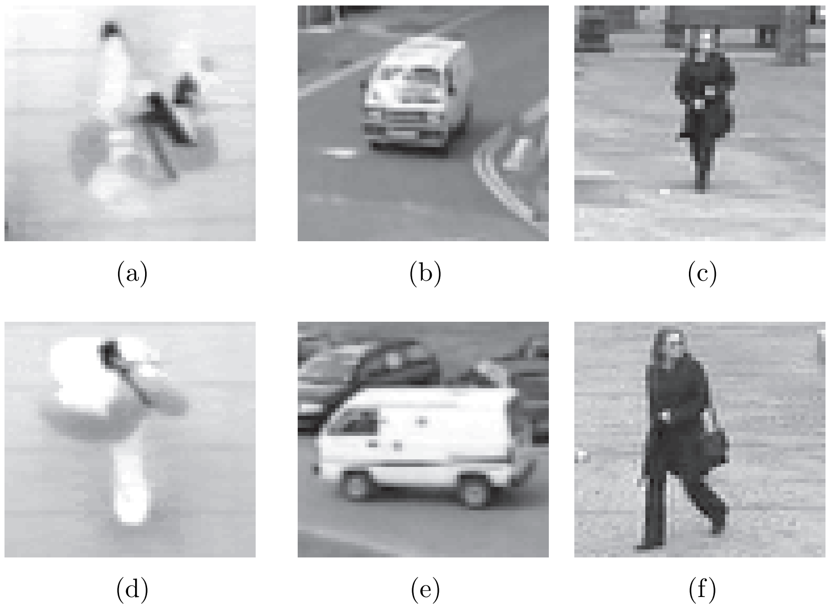


Figure 2.5: Illustrating target variability. (a - c) showing different object shapes, (d - e) same objects some frames later

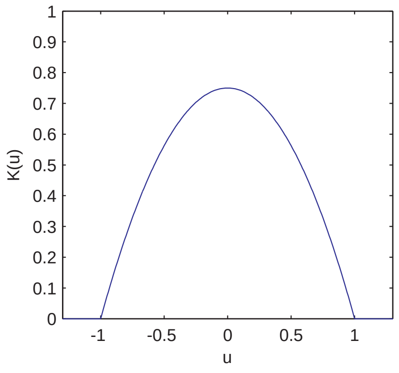


Figure 2.6: The Epanechnikov kernel

cope with partial occlusion and object deformation. A variety of colour spaces have been investigated for their suitability. Comaniciu et al. [14] opted for the standard Red, Green, Blue (RGB) colour space with their mean-shift based tracker. They used histograms with $16 \times 16 \times 16$ bins extracted from elliptical regions. The pixels were weighted with an Epanechnikov kernel that diminished the background pixel contribution from the edges of the ellipses. The kernel for an object with width $2u$, shown in Figure 2.6, has the form:

$$K(u) = \begin{cases} \frac{3}{4}\left(1-u^2\right) & \text{if } |u| \leqslant 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.20}$$

A similar approach was taken by Nummario et al. [59] but they used a more economical $8 \times 8 \times 8$ bins. Czyz et al. [60] also used $8 \times 8 \times 8$ bin RGB but with rectangular regions and lower weighting applied to periphery pixels. Maggio and Cavallaro [61] carried out a study of the effectiveness of seven different colour spaces, with 10 bins for each dimension, and concluded that the RGB space gave the best performance. But others have opted for the Hue-Saturation-Value (HSV) space in order to decouple chromatic information from shading effects. Lu et al. [62] followed Pérez et al. [63] in using a 2 dimensional 10×10 bin histogram for the HS components and a 10 bin one for the V component, to give a dimensionality of 110 bins.

2.6.2 Spatiograms

Whilst a colour histogram representation allows targets to deform and rotate, it also has the potential of making it difficult to distinguish individual targets that might have similar overall colour signatures but with different colour layouts. Similarly regions of background might match closely to the simple histogram. Compromising on the versatility of the histogram, some have looked to make the representation a little more descriptive by incorporating either general shape or spatial colour distribution information. Birchfield and Rangarajan [64] introduced the 'Spatiogram' as a way of doing this. Their representation augments the histogram bins with spatial moments of the pixels contributing to the bin i.e:

$$h(b) = \{n_b, \mu_b, \Sigma_b\} \tag{2.21}$$

where n_b is the number of pixels falling in the bin b, μ_b and Σ_b are respectively the mean vector and covariance matrix of the pixel coordinates. The distances between the target and candidate histograms $\rho(h, h^*)$ were computed with an additional bin-wise weighting based upon the spatial information:

$$\rho(h, h^*) = \sum_{b=1}^{B} \psi_b \rho(n_b, n_b^*) \tag{2.22}$$

where

$$\psi_b = \eta \mathcal{N}\left(\mu_b; \mu_b^*, 2(\Sigma_b + \Sigma_b^*)\right) \tag{2.23}$$

where $\mathcal{N}\left(a; \mu, \Sigma)\right)$ is a (μ, Σ) Gaussian evaluated at a. The multiplier η is a normalizing factor. The authors reported that they had applied the approach to both mean-shift and particle filter trackers, and demonstrated its superiority over a range of alternative features on the sequences used. They used 3D colour histograms with good resolution.

2.6.3 Edge orientation histograms

Yang et al. [65] supplemented object colour information with an additional edge representation. Instead of the standard colour histogram they split the target into rectangular regions and calculated the mean RGB values within that region. They then converted the colour image into greyscale and calculated the vertical and horizontal Sobel edges. At each pixel they calculated the edge magnitude and direction. The histogram was constructed with the bins representing the orientations and the bin value representing the sum of the strengths for each angle range. By constructing integral images for each bin they could build up an integral histogram for fast feature extraction. Colour and edge histograms similarities were calculated separately to produce likelihoods of the form:

$$p\left(\mathbf{z}_t|\mathbf{x}_t\right) \propto \exp\left(-\rho(\mathbf{p}(\mathbf{x}_t), \mathbf{q})^2/\sigma^2\right) \tag{2.24}$$

where $\rho(\mathbf{p}(\mathbf{x}_t), \mathbf{q})$ is the Euclidean distance between the candidate vector $\mathbf{p}(\mathbf{x_t})$ and the reference vector $\mathbf{q}$.

2.6.4 Multi-part histograms

Others encoded spatial information by simply concatenating histograms extracted from different regions across the target to be tracked. For example, Okuma et al. [66] presented a colour histogram particle filter tracker in which the target regions were split into upper and lower rectangular sub-regions. But whilst the combination of the regions was simple, they still felt a need to use a 110 bin HSV representation. Iwahori et al. [67] went further by splitting a target rectangular region into a number of vertically stacked rectangular sub-regions. For each of the sub-regions they extracted a HSV histogram and calculated the similarity against the reference region histogram. They then returned an overall similarity using the mean value of them.

Maggio and Cavallaro [61] complemented their colour space study with an investigation into multi-part histogram representations. They devised a multi-part elliptical representation of the target. A histogram was constructed for the full ellipse, four histograms were extracted from quadrants bounded by the major and minor axes, one was extracted from a half sized version of the full ellipse, and another from the ‘annulus’ formed by subtracting the half sized from the full sized one. The seven histograms were then concatenated to produce the feature vector. The dissimilarity between the model and candidate histograms was calculated using the Bhattacharyya distance. The four quadrants introduced spatial information to the measure in order to recognize rotations. They compared the responses of a vertical two part rectangular histogram representation, the concentric ellipses, the four ellipse quadrant representation, and the full seven part combined representation. They found that the seven part one outperformed the others.

2.7 Discussion

The methods described in this chapter paint a general picture of the way that the problem of target object detection has been tackled in the last two decades. In many of the cases the focus has been on object identification and localization within the images without necessarily using the detection as a starting point for tracking. If the methods were efficient then it might be argued that frame by frame detection could be adequate. But one aim of tracking is to maintain some model of the motion so that problems thrown up by occlusion and loss of track might be resolved. As computational resources are limited, and the implementation of a tracker might draw substantially on those resources, it is essential to keep the demands of the detection element under control. In addition, in the context of commercial CCTV it would be useful to develop a tracker that is not finely tuned to following a single class of object. The tool developed has to work equally for vehicles as for pedestrians or small groups of people; so detection methods, such as the SVM and Adaboost approaches, that rely on training specific object classifiers, are unattractive.

It is true that a real-time particle filter tracker incorporating Adaboost person detection has been reported [66], and continues to be developed [62], but it relies on desktop PC level GHz processing, deals with a rather special case and would be difficult to extend into a general purpose approach. The edge based methods similarly draw on either classifier training or tree based template matching. Enzweiler and Gavrila's edge based pedestrian detection approaches [55, 68] were explored specifically for in-vehicle embedded processing. Their work indicated a preference for HOG/SVM approaches at high resolution and a Haar-type Adaboost approach at lower resolution. But they still demand GHz processing, and that is before a tracker has been incorporated.

Even with static cameras the background modeling approaches with current frame subtraction do not offer a full solution. No matter how much computational effort is put into the development of the reference image, there is still a need to develop intelligent thresholding procedures to produce good quality blob images. Also, the blob image has to be further interpreted to identify objects of interest. The Gaussian based background approaches offer a degree of mathematical sophistication but there is no fundamental reason to believe that temporal sequences of image pixel intensity values follow a Normal distribution. The histogram and classifier based approaches consume memory, call for distance measures that rely on floating point calculations, and return blob images that are not significantly superior to the others. Out of the background methods considered the multi-modal mean is the closest candidate for consideration. It assumes no underlying statistical model and is computationally simpler than the others.

But the approaches considered offer some tools that can be employed. In the spirit of animal vision the general features that will guide detection are that the targets are not background, that they have movement, they will have a size characteristic of the targets of interest, and that they have a trackable chromatic signature. So a foundation element, at least for the initial detection of objects for tracking, will be background subtraction. A mean-shift approach similar to that of Beleznai et al. (Section 2.4.4) will allow the placement of a simple template over blobs entering the field of view, so it can be expected that the integral image will be used.

Without definitely classifying the detected object it will be necessary to identify which of the blob pixels are foreground and which are background so it is likely that there will have to be inclusion of some element of background region histogram, at least at the tracker initialization stage. A histogram based signature, that can be used to follow a deforming and rotating object in subsequent frames, will have to be built from the proposed foreground pixels

The suitability of the spatiogram approach was considered, but it was found that with 8-bin grey scale histograms the contribution of the spatial component did not justify the extra computation involved. Object deformation and rotation, coupled with unpredictable illumination and histogram quantization effects, meant that the spatial component could vary significantly within a grey scale slice from frame to frame. This was especially noticeable at low frame rates. But combinations of 8-bin grey scale histograms with a rectangular spatial layout, as described in Section 2.6.4, showed promise in terms of tracking consistency and computational economy, so that approach suggests it will be the first choice for target representation for tracking.

Chapter 3

Sequential Bayesian Estimation

3.1 Introduction

This chapter considers basic theory associated with the particle filter. It starts from first principles and traces out the steps from basic Bayesian recursion to the Sampling Importance Re-sampling (SIR) particle filter. It draws upon material from standard sources but supplements it in order to clarify steps in the arguments. For notational convenience the concepts are developed in terms of a single target state $\mathbf{x}$ and measurement $\mathbf{z}$. Extension to the multi-target state occurs in the following chapter.

The two main tools for tracking are the Kalman filter (KF) and the particle filter (PF) so both are described with a degree of detail. Although the PF is the tool of choice for this work, the discussion of the KF is included to clarify the reasons for that choice.

An important practical step in the operation of the PF is *re-sampling*, so the basics of that process are described in order to set the context for a more detailed discussion later in the thesis dealing with simplifications and improvements. The chapter concludes with a brief overview of variants of the basic filter.

3.2 Bayesian recursion

Sequential estimation relationships can be derived using Bayes' probability rule and the Chapman-Kolmogorov equation. Given the equivalent conditional probabilities:

$$p(A, B) = p(B|A)p(A) \tag{3.1}$$

and

$$p(A, B) = p(A|B)p(B) \tag{3.2}$$

Bayes' rule follows as:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)} \tag{3.3}$$

The Chapman-Kolmogorov equation is:

$$p(A|B) = \int p(A|C)p(C|B)dC \tag{3.4}$$

It is essentially a marginalization with respect to the variable C and represents the sum of all pathways through which state A can be returned given B.

The Bayesian tracker [69, 70] is based upon the use of a possibly non-linear state transition model subject to additive uncertainty $\boldsymbol{\nu}_{t-1}$, a Markov assumption, in which a current state $\mathbf{x}_t$ depends on the immediately preceding state $\mathbf{x}_{t-1}$,

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\nu}_{t-1}), \tag{3.5}$$

and the use of a measurement model,

$$\mathbf{z}_t = h(\mathbf{x}_t, \boldsymbol{\eta}_t), \tag{3.6}$$

in which a current measurement is taken to be a possibly non-linear function of the current state with measurement uncertainty $\boldsymbol{\eta}_t$.

The objective is to recursively estimate $\mathbf{x}_t$ at a time t from the set of measurements up to time t represented by $\mathbf{z}_{1:t}$. With uncertainty, or 'noise', in both prediction and measurement, the tracking process is necessarily probabilistic. Given a belief represented by the pdf $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ at time $t-1$, we want to recursively update the degree of belief $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ at time t.

Bayes' rule says that for time t we can write:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_{1:t})} \tag{3.7}$$

from which we can derive a $t-1$ to t recursion relationship. With an extension of Eq.(3.1) to three variables we can write:

$$\begin{aligned} p(\mathbf{z}_t, \mathbf{z}_{1:t-1}, \mathbf{x}_t) &= p(\mathbf{z}_t|\mathbf{x}_{t,}\mathbf{z}_{1:t-1})p(\mathbf{x}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1}) && (3.8)\\ &= p(\mathbf{x}_t|\mathbf{z}_{t,}\mathbf{z}_{1:t-1})p(\mathbf{z}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1}) && (3.9)\\ &= p(\mathbf{x}_t|\mathbf{z}_{1:t})p(\mathbf{z}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})) && (3.10) \end{aligned}$$

It follows, using Eq.(3.8) and Eq.(3.10), that Eq.(3.7) can be written as:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{z}_{1:t-1})p(\mathbf{x}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})} \tag{3.11}$$

If we take it that the state $\mathbf{x}_t$ is complete, meaning that no past measurements such as $\mathbf{z}_{1:t-1}$ can provide additional information for the prediction of the measurement $\mathbf{z}_t$, then we can say:

$$p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{z}_{1:t-1}) = p(\mathbf{z}_t|\mathbf{x}_t) \tag{3.12}$$

Eq.(3.11) can then be simplified to:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \tag{3.13}$$

The link to $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ from $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ can be introduced through the replacement of the numerator term $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ with the Chapman-Kolmorogov marginalization over $\mathbf{x}_{t-1}$:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \tag{3.14}$$

The denominator in Eq.(3.13) can be linked to the numerator through a similar Chapman-Kolmogorov marginalization over $\mathbf{x}_t$:

$$p(\mathbf{z}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t \tag{3.15}$$

The full Bayesian recursion relationship can then be stated as:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}}{\int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t} \tag{3.16}$$

Eq.(3.14) defines a Bayesian *prior*, or a prediction of the state at time step t based upon the knowledge of the state at $t-1$. The numerator term $p(\mathbf{z}_t|\mathbf{x}_t)$ in Eq.(3.13) defines a *likelihood* based upon the measurement model being used, and the denominator term, Eq.(3.15), is a normalizing constant linked to the likelihood function. It is generally referred to as the *evidence.*

The Bayes relationship can be seen to have the general form:

$$\text{posterior} = \frac{\text{likelihood}}{\text{evidence}} \cdot \text{prior} \tag{3.17}$$

i.e. the posterior probability is the product of the likelihood given the evidence, and the probability of the state based upon the prior knowledge.

3.3 The Kalman filter

The optimal Bayesian recursion (3.16) can be implemented analytically using a Kalman filter [71]. It can be shown that if the density $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ is Gaussian, then the KF recursion can return a Gaussian $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ if the process and measurement functions Eq.(3.5) and Eq.(3.6) are linear, and the noise terms are also Gaussian. Thrun [72] provides an example of a full and analytical derivation of the KF update equations. In that derivation equations Eq.(3.12) and Eq.(3.14) are rewritten with the probability terms replaced by their Gaussian equivalents. Equations Eq.(3.5) and Eq.(3.6) are realized in the forms:

$$\mathbf{x}_t = F_t\mathbf{x}_{t-1} + \boldsymbol{\nu}_{t-1} \tag{3.18}$$

and

$$\mathbf{z}_t = H_t\mathbf{x}_t + \boldsymbol{\eta}_t \tag{3.19}$$

where F_t is a matrix defining the linear transition function and H_t is a matrix defining the linear relationship between the measurement and the proposed mean. The covariances of the noise terms $\boldsymbol{\nu}_{t-1}$ and $\boldsymbol{\eta}_t$, indicating the uncertainties in prediction and measurement, are represented by Q_{t-1} and R_t respectively.

If the multidimensional state at time $t-1$ is represented by a Gaussian having mean $\boldsymbol{\mu}_{t-1}$ and covariance P_{t-1} then the filter starts by making a proposal for the mean at time t using:

$$\hat{\boldsymbol{\mu}}_t = F\boldsymbol{\mu}_{t-1} \tag{3.20}$$

The proposed state $\hat{\boldsymbol{\mu}}_t$ has prediction uncertainty $\hat{P}_t$ derived from a combination of the state uncertainty P_{t-1} at time $t-1$ and the process noise covariance Q_{t-1}:

$$\hat{P}_t = Q_{t-1} + F_tP_{t-1}F_t^T \tag{3.21}$$

Given a measurement $\mathbf{z}_t$, with uncertainty R_t, the filter makes a correction to the proposed mean state:

$$\boldsymbol{\mu}_t = \hat{\boldsymbol{\mu}}_t + K_t(\mathbf{z}_t - H_t\hat{\boldsymbol{\mu}}_t) \tag{3.22}$$

where K_t, called the Kalman gain, is a combination of the prediction uncertainty $\hat{P}_t$ and the measurement uncertainty R_t:

$$K_t = \frac{\hat{P}_tH_t^T}{H_t\hat{P}_tH_t^T + R_t} \tag{3.23}$$

The difference between the measurement and the predicted measurement, based upon the proposal for the mean, i.e. $(\mathbf{z}_t - H_t\hat{\boldsymbol{\mu}}_t)$, is referred to as the 'innovation'.

The state uncertainty at time t, P_t, is then returned as a correction to the estimated uncertainty $\hat{P}_t$:

$$P_t = \hat{P}_t - K_t H_t \hat{P}_t \tag{3.24}$$

It is useful to illustrate the logic behind the filter, and the nature of the Kalman gain, by considering a one-dimensional situation with $F = 1$ and $H = 1$. The estimated uncertainty in prediction at time t, $\hat{P}_t$, is given by a linear combination of the process noise uncertainty Q_{t-1} and the state uncertainty at time $t-1$, P_{t-1}, i.e

$$\hat{P}_t = Q_{t-1} + P_{t-1} \tag{3.25}$$

The filter takes the mean μ_t to be a weighted combination of the prediction $\hat{\mu}$ and the measurement z:

$$\mu_t = \frac{R_t}{(\hat{P}_t + R_t)}\hat{\mu}_t + \frac{\hat{P}_t}{(\hat{P}_t + R_t)}z_t \tag{3.26}$$

It can be seen from Eq.(3.26) that if the prediction uncertainty $\hat{P}_t$ is large in comparison to the measurement uncertainty R_t then the resultant state is biased towards the measurement. If the converse is true then the resultant state is biased towards the prediction. The equation can be rearranged to give:

$$\mu_t = \hat{\mu}_t + \frac{\hat{P}_t}{\hat{P}_t + R_t}(z_t - \hat{\mu}_t) \tag{3.27}$$

The Kalman gain can now be recognized as the fraction of the estimated uncertainty to the combined estimated and measurement uncertainties:

$$K = \frac{\hat{P}_t}{\hat{P}_t + R_t} \tag{3.28}$$

The state uncertainty at time t is derived from a reciprocal combination of the uncertainties $\hat{P}_t$ and R_t, analogous to the combination of electrical resistances in parallel:

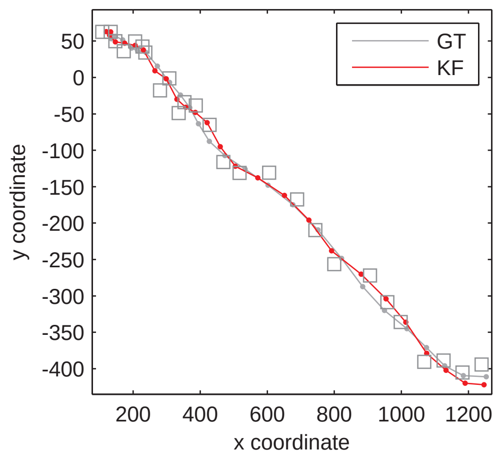
$$\frac{1}{P_t} = \frac{1}{\hat{P}_t} + \frac{1}{R_t} \tag{3.29}$$

With this combination it can be seen that if the uncertainty $\hat{P}_t$ is large in comparison to R_t then the value of P_t approaches R_t. Conversely, if the value of R_t is large in comparison to $\hat{P}_t$ then the uncertainty approaches $\hat{P}_t$. Eq.(3.29) can be rearranged to give:

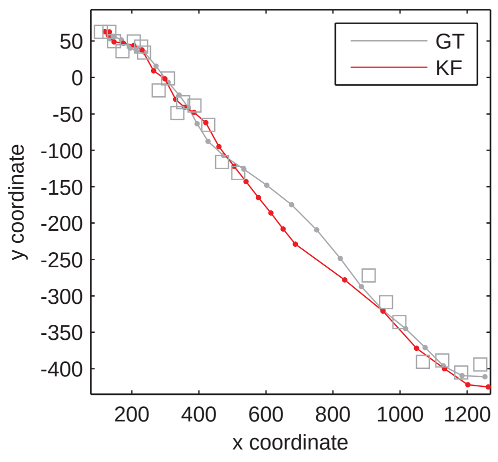
$$P_t = \hat{P}_t - K\hat{P}_t \tag{3.30}$$

i.e. the-one dimensional form of Eq.(3.24).

An illustration of the Kalman filter in action is given in Figure 3.1. In this



(a) Tracking through noisy measurements



(b) Tracking through simulated occlusion

Figure 3.1: Illustrating Kalman filter behaviour

simulation the grey line indicates the ground truth, the gray squares indicate the noisy measurements and the red line shows the KF response. The track starts at the top left of the figure. The simulated track was constructed by sequentially adding Gaussian random increments in the x and y directions. The simulated noisy measurements were then produced by adding further x and y Gaussian increments to the track points. Figure 3.1(a) shows the ability of the KF to produce a good estimate of the track through the noisy data points. In Figure 3.1(b), data points 16 to 20 have been artificially removed to simulate an occlusion. In the absence of a measurement the KF the system places track points at predictions based upon the last known state and the state transition equation. When measurements return, the filter picks up the track again.

But in general the tracking process will have non-linear measurement steps and non-Gaussian noise factors hence the Kalman filter has limited applicability in those circumstances. An alternative is to use the Extended Kalman filter (EKF). The EKF works by linearizing equations Eq.(3.5) and Eq.(3.6) using a Taylor expansion. But it still approximates the posterior to be Gaussian and this is a condition that cannot always be satisfied when tracking in clutter. For non-linear, non-Gaussian situations we turn to the particle filter

3.4 The particle filter

The particle filter is a sub-optimal technique for implementing the Bayesian recursion using Monte Carlo approximations. The posterior pdf is represented by a set of weighted, randomly placed, points in the state space. The points are referred to as particles. The particle set $\{\mathbf{x}^i_{0:t}, w^i_t\}^{n_s}_{i=1}$, with the states up to time t represented by

$\{\mathbf{x}_{0:t}^i\}_{i=1}^{n_s}$ and weights at time t represented by $\{w_t^i\}_{i=1}^{n_s}$, characterizes the posterior pdf $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ where $\mathbf{z}_{1:t}$ is the set of all measurements up to that time [69]. If the weights are normalized such that $\sum_i w_t^i = 1$ then a discrete weighted approximation to the true posterior $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ at time t can be formed as:

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{n_s} \tilde{w}_t^i \delta(\mathbf{x}_{0:t} - \mathbf{x}_{0:t}^i) \tag{3.31}$$

where $\delta(\cdot)$ is the Dirac delta function and $\tilde{w}$ is the normalized weight. The weights are chosen using *importance sampling*.

Importance sampling [73] is a method used to calculate the expectation of a probability distribution in situations where it is difficult to directly generate samples from that distribution. The expectation of a function $f(x)$ over a probability distribution $p(x)$ is given by:

$$\mathrm{E}_p\left[f(x)\right] = \int p(x)f(x)dx \tag{3.32}$$

If we could generate samples $\{x^i\}_{i=1}^{n_s}$, independently drawn from $p(x)$, then the sampled approximation to the expectation would be:

$$\hat{f} = \frac{1}{n_s}\sum_{i=1}^{n_s} f(x^i) \tag{3.33}$$

with

$$\lim_{n_s\to\infty} \hat{f} = \mathrm{E}_p\left[f(x)\right] \tag{3.34}$$

If we are working with an un-normalized distribution $p'(x) = c_p p(x)$, from which it is difficult to draw samples, and for which the normalization constant c_p is unknown, then the importance sampling approach allows us to use samples from another un-normalized distribution $q'(x) = c_q q(x)$, with unknown normalization constant c_q, but from which it is easy to draw samples, i.e.:

$$\begin{aligned}\mathrm{E}_p\left[f(x)\right] &= \int p(x)f(x)dx \\ &= \frac{1}{c_p}\int p'(x)f(x)dx \end{aligned} \tag{3.35}$$

Introducing the new sampling distribution we get:

$$\mathrm{E}_p\left[f(x)\right] = \frac{1}{c_p}\int \frac{p'(x)}{q'(x)} q'(x) f(x) dx \tag{3.36}$$

$$= \frac{c_q}{c_p}\int \frac{p'(x)}{q'(x)} q(x) f(x) dx \tag{3.37}$$

and for discrete samples we can write:

$$\mathrm{E}_p\left[f(x)\right] \approx \frac{c_q}{c_p n_s}\sum_{i=1}^{n_s} \frac{p'(x^i)}{q'(x^i)} f(x^i) \tag{3.38}$$

$$= \frac{c_q}{c_p n_s}\sum_{i=1}^{n_s} w^i f(x^i) \tag{3.39}$$

where $w^i = \frac{p'(x^i)}{q'(x^i)}$ is defined as an un-normalized importance weight.

If we consider now the normalization constant c_p:

$$c_p = \int p'(x) dx \tag{3.40}$$

$$= \int \frac{p'(x)}{q'(x)} q'(x) dx \tag{3.41}$$

$$= c_q \int \frac{p'(x)}{q'(x)} q(x) dx \tag{3.42}$$

$$\approx \frac{c_q}{n_s}\sum_{i=1}^{n_s} w^i \tag{3.43}$$

i.e.

$$\frac{c_p n_s}{c_q} \approx \sum_{i=1}^{n_s} w^i \tag{3.44}$$

It follows that Eq.(3.39) can be re-written as:

$$\mathrm{E}_p\left[f(x)\right] \approx \frac{1}{\sum\limits_{i=1}^{n_s} w^i}\sum_{i=1}^{n_s} w^i f(x) \tag{3.45}$$

$$\approx \sum_{i=1}^{n_s} \tilde{w}^i f(x) \tag{3.46}$$

where $\tilde{w}_t^i$ is the normalized weight:

$$\tilde{w}_t^i = \frac{w_t^i}{\sum\limits_{i=1}^{n_s} w_t^i} \tag{3.47}$$

The importance sampling principle is now extended to the particle filter representation. If the required samples $\{\mathbf{x}_{0:t}^i\}_{i=1}^{n_s}$ are drawn from an importance density $q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ then the normalized weights in Eq.(3.31) can be derived from the unnormalized weights:

$$w_t^i = \frac{p(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t}^i)}{q(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t}^i)} \tag{3.48}$$

The recursive element of the particle filter then involves a Bayesian update of the weights as each measurement is received at each time step, i.e., relating w_t^i to w_{t-1}^i as shown below.

Arulampalam et al. [69] argue that if the importance density is chosen to factorize such that:

$$q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})q(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \tag{3.49}$$

then we can get the new state by taking samples from the state $q(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1})$ and augmenting them with samples from the state $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})$.

Following the approach used in equations Eq.(3.8), Eq.(3.9) and Eq.(3.10), Bayes' theorem can be used to express the numerator in Eq.(3.48) as:

$$\begin{aligned} p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) &= \frac{p(\mathbf{z}_t|\mathbf{x}_{0:t}, \mathbf{z}_{1:t-1})p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \\ &= \frac{p(\mathbf{z}_t|\mathbf{x}_{0:t}, \mathbf{z}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t-1})p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \end{aligned}$$

If the process is Markovian, i.e. the current state is dependent only on the immediately preceding state, then:

$$p(\mathbf{z}_t|\mathbf{x}_{0:t}, \mathbf{z}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t-1}) \rightarrow p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{3.50}$$

leading to:

$$\begin{aligned} p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) &= \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \\ &\propto p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \end{aligned} \tag{3.51}$$

By substituting Eq.(3.49) and Eq.(3.51) in Eq.(3.48), the weight update equation becomes

$$\begin{aligned} w_t^i &\propto \frac{p(\mathbf{z}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)p(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})}{q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i, \mathbf{z}_{1:t})q(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})} \\ &= w_{t-1}^i \frac{p(\mathbf{z}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i, \mathbf{z}_{1:t})} \end{aligned} \tag{3.52}$$

in which:

$$w_{t-1}^i = \frac{p(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})}{q(\mathbf{x}_{0:t-1}^i|\mathbf{z}_{1:t-1})} \tag{3.53}$$

Applying a Markov restriction to the denominator of Eq.(3.52) we have:

$$q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i, \mathbf{z}_{1:t}) \rightarrow q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t) \tag{3.54}$$

and the weight update equation becomes:

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\mathbf{x}_t^i) p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)} \tag{3.55}$$

If the importance density is taken to be the prior, i.e.

$$q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t) = p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i) \tag{3.56}$$

then the weight equation Eq.(3.55) reduces to:

$$w_t^i \propto w_{t-1}^i p(\mathbf{z}_t|\mathbf{x}_t^i) \tag{3.57}$$

The posterior density $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is then approximated as:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{n_s} \tilde{w}_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i) \tag{3.58}$$

where $\tilde{w}_t^i$ is the normalized weight as described by Eq.(3.47).

The basic filter algorithm consists of a recursive update of the weights as each measurement is received at each time step, resulting in a sequentially updated particle representation of the pdf. The object state is usually extracted as either a weighted mean:

$$\mathrm{E}\left[\mathbf{x}_t|\mathbf{z}_{1:t}\right] \approx \sum_{i=1}^{n_s} \tilde{\omega}_t^i \mathbf{x}_t^i \tag{3.59}$$

or as a maximum a posteriori (MAP) estimate:

$$\mathbf{x}_t^{MAP} = \arg\max_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \arg\max_{\mathbf{x}_t} w_t \tag{3.60}$$

3.4.1 Filter degeneration

A difficulty with the recursive propagation of the weights is that it can quickly result in a degeneration of the sample set. The weight becomes concentrated in a small fraction of the particle population and the lower weight particles diffuse in the state space. This weakens the Monte Carlo representation of the pdf. The estimation

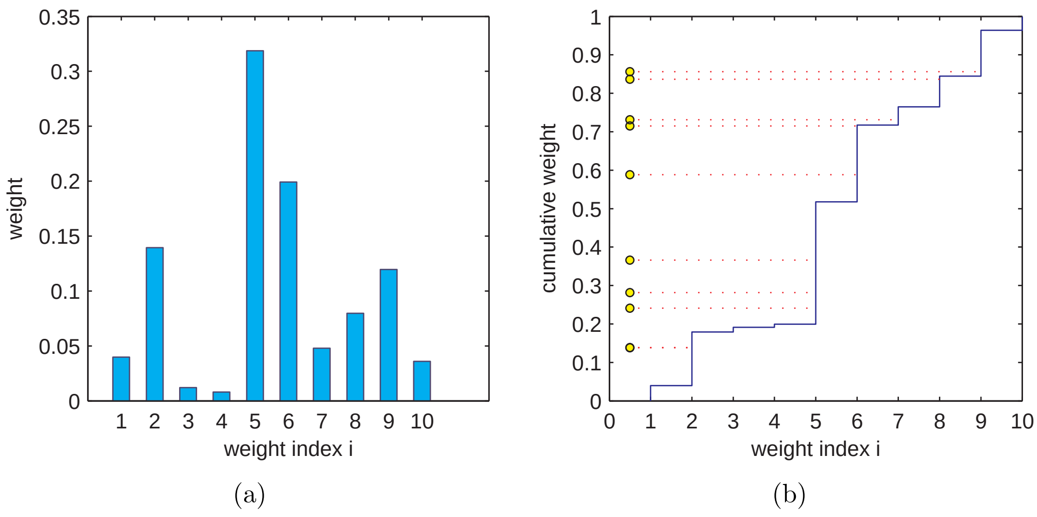


Figure 3.2: Illustrating multinomial re-sampling using 10 weights: (a) weight values, (b) weight re-selection using the cumulative weights, the yellow points indicate sampling numbers $u^{(k)}$. The horizontal dotted lines from the yellow points indicate weight index selection.

of the state suffers from the reduced particle density in the vicinity of the target mode. In addition, computational resources are wasted dealing with particles that have lost their link to the target.

The number of effective particles n_{eff} is linked to the total number n_s and the variance of the weights through [69]:

$$n_{eff} = \frac{n_s}{1 + \mathrm{Var}(w_t^i)} \tag{3.61}$$

where $w_t^i = p(\mathbf{x}_t^i|\mathbf{z}_{1:t})/q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$ is the un-normalized weight, also referred to as the 'true weight'.

The degeneration could be countered by using a large number of particles n_s, but this increases the computational load. Alternative importance densities $q(.)$, from which the samples are chosen, could be explored with a view to ensuring that the variance of the weights is minimized. Doucet et al. [74] show that the optimal importance density is $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)$, i.e. the pdf of the state given both the prediction and the current measurement. The drawback to this density is that it is not always easy to sample from it.

An alternative is to regenerate the particle set, through re-sampling, when n_{eff} falls below a pre-set threshold. It is common for re-sampling to be applied at each time step irrespective of the value of n_{eff}.

3.4.2 Re-sampling

Resampling is an evolutionary type process in which higher-weight particles are reproduced at the expense of lower-weight ones with the aim of reducing the variance of the set. Particles are sampled with replacement n_s times from the original particle set $\{\mathbf{x}^i, w^i\}_{i=1}^{n_s}$ to produce a new set $\{\mathbf{x}^j, \tilde{w}^j\}_{j=1}^{n_s}$ such that $P(j) \propto w^i$, i.e. the probability of selecting a particle is proportional to the weight associated with it. A basic approach [75] is to form a cumulative sum of the normalized weights and a set of n_s fractional sampling numbers $u^{(k)} \in [0, 1]$. The indices j of the re-sampled particle set are generated using the inverse F^{-1} of the cumulative distribution of the particle weights with:

$$i^j = F^{-1}(u^{(k)}) \tag{3.62}$$

$$= i | u^{(k)} \in \left[\sum_{s=1}^{i-1} \tilde{w}_s, \sum_{s=1}^{i} \tilde{w}_s \right) \tag{3.63}$$

where i^j is the index from the original particle set assigned to the j^{th} re-sampled particle. The re-sampling process is summarized in Algorithm 1.

The re-sampling step is illustrated, using an example with 10 particles, in Figure 3.2. The example is one of *multinomial* re-sampling in which the sampling numbers $u^{(k)}$ are drawn from a uniform random distribution across the range of the weights. Of the original particle set, those labelled {1,3,4,10} are discarded and the more strongly weighted {2,5,6,7,8,9} are reproduced with multiplicities {1,3,2,1,1,1} respectively. This approach to re-sampling featured in early versions of particle filters but was quickly recognized to be inefficient on both computational and statistical grounds. A range of alternative re-sampling approaches have been proposed, they will be discussed in Chapter 7.

3.5 Particle filter variants

The importance sampling particle filter with the addition of re-sampling is referred to as the *Sampling Importance Re-sampling* (SIR) filter. It is summarized in Algorithm 2. Research in the last decade has explored a range of variations on the basic particle filter theme. It has looked at issues such as reducing the number of particles required, enhancing the state estimation, and streamlining computational throughput.

The *Auxiliary Particle Filter* (APF) [76] was proposed to produce an improved importance density closer to the optimal $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{z}_t)$. It introduces an additional re-sampling step. The particles are first re-sampled as in Algorithm 2 to produce

Algorithm 1 Multinomial resample

```
**Function definition:** $\left[\left\{\mathbf{x}_t^j, \tilde{w}_t^j\right\}_{j=1}^{n_s}\right] = resample\left(\{\mathbf{x}_t^i, w_t^i\}_{i=1}^{n_s}\right)$

**Stage 1**: normalize the weights

- calculate the total weight: $w_\Sigma \leftarrow \sum_{i=1}^{n_s} w_t^i$

**for** $i = 1$ to $n_s$ **do**
  - normalize $\tilde{w}_t^i \leftarrow w_\Sigma^{-1} w_t^i$
**end for**

**Stage 2**: construct cumulative density function (CDF) $c$

- initialize: $c^1 \leftarrow 0$

**for** $i = 1$ to $n_s$ **do**
  $c^i \leftarrow c^{i-1} + \tilde{w}_t^i$
**end for**

**Stage 3**: select samples

**for** $j = 1$ to $n_s$ **do**

  - draw a sampling number $u^j$
  - start at the bottom of the CDF: $i \leftarrow 0$
  **while** $u^j > c^i$ **do**
    $i \leftarrow i + 1$
  **end while**

  - assign sample: $\mathbf{x}_t^j \leftarrow \mathbf{x}_t^i$
  - assign weight: $\tilde{w}_t^j \leftarrow n_s^{-1}$

**end for**
```

Algorithm 2 SIR particle filter - with multinomial re-sampling

```
**Function definition:** $\left[\{\mathbf{x}_t^j, \tilde{w}_t^j\}_{j=1}^{n_s}\right] = SIR\left(\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^{n_s}\right)$

**Stage 1**: apply process step and weight particles:

**for** $i = 1$ to $n_s$ **do**
  - draw $\mathbf{x}_t^i \sim p(\mathbf{x}_t|\mathbf{x}_{t-1})$
  - calculate $w_t^i \leftarrow p(\mathbf{z}_t|\mathbf{x}_t^i)$
**end for**

**Stage 2**: call resample function:

$\left[\{\mathbf{x}_t^j, \tilde{w}_t^j\}_{j=1}^{n_s}\right] = resample\left(\{\mathbf{x}_t^i, w_t^i\}_{i=1}^{n_s}\right)$
```

an improved representation $\{\mathbf{x}_t'^j\}_{j=1}^{n_s}$ around the dominant modes in the pdf, with a reduced representation in the tails and with a likelihood $p(\mathbf{z}_t|\mathbf{x}_t'^j)$. Gaussian noise is added to these auxiliary states to produce $\{\mathbf{x}_t^j\}_{j=1}^{n_s}$. A new likelihood $p(\mathbf{z}_t|\mathbf{x}_t^j)$ is calculated and the particle is weighted according to:

$$w_t^j = \frac{p(\mathbf{z}_t|\mathbf{x}_t^j)}{p(\mathbf{z}_t|\mathbf{x}_t'^j)} \tag{3.64}$$

The particles are then re-sampled using this weight. The result is that preference will be given to those states that improve the likelihood after the auxiliary step.

The approach increases the amount of computation per particle but it is argued that it is offset by a better representation of the state resulting in the need for fewer particles. A possible drawback to the approach is that it can unnecessarily enhance modes associated with clutter if the target mode is small.

The *Regularized Particle Filter* (RPF) [77] was introduced to counteract re-sampling problems that emerge from the discrete rather than continuous nature of the probability distributions. The discontinuous representation can lead to some particles being over-representative. The filter is identical to the one described in Algorithm 2 except at the final sample assignment step. Instead of assigning the re-sampled particle directly, the new particle is drawn randomly from a continuous kernel centred on the re-sampled state. This 'kernelized' representation provides an approximation to a continuous distribution.

The *Gaussian Particle Filter* (GPF) [78] aims to reduce computational complexity by using a particle based Gaussian approximation to the probability densities. Particles are generated from a Gaussian distribution with an initialized mean and covariance. They undergo a process step to diffuse them from the last known state to the proposed one. The likelihood weights are calculated and normalized as in the basic particle filter. There is no re-sampling step. The updated Gaussian parameters, the mean and covariance, are then calculated using the weighted states.

The *Annealed Particle Filter* (AnPF) [79] has the same general structure as the filter described in Algorithm 2 but aims to improve the final estimation of state by repeating the re-sampling step a number of times using conditioned particle weights. The conditioned weights are the original weights raised to a fractional power, i.e.

$$\tilde{w}_m = \tilde{w}_t^{\beta_m} \tag{3.65}$$

where $\beta_0 = 1$ and $\beta_0 > \beta_1 > \beta_2 ... > \beta_m$. As both the weights and the powers are fractional the effect of large m (smaller values of β) is to smooth out the likelihood distribution. Particles are re-sampled from the smoothed distribution to produce a set that captures the broad structure of the search space. By gradually increasing β the fine structure of the distribution becomes more pronounced but the particle representation accumulates in the region of the dominant mode. This results in a better estimate of the mean or MAP state and a better conditioned posterior distribution for progression to the next state. It does so, however, at a cost of increasing the computation per particle.

3.6 Discussion

Sequential Bayesian Estimation provides a basic framework for the development of active tracking systems, and the particle filter offers an approach that is not constrained by linearity and Gaussianity. But it does not provide a single universal solution to the tracking problem; it simply provides a paradigm from which tracking systems can be engineered. The existence of the variants illustrate the need to tailor such a filter to specific applications. There is no theoretical framework that dictates aspects like the optimum number of particles to use, the choice of proposal density, which re-sampling method is appropriate etc. These are engineering aspects and different solutions will emerge in different contexts. In the absence of compelling evidence found to suggest otherwise, the SIR variant will form the basis of the work in the tracking context of this thesis.

Chapter 4

Multi-target tracking

4.1 Introduction

The Bayesian recursion of the last chapter was described in terms of predictions and measurements involving a single target state. The arguments apply equally to a multi-target state. The aim of this chapter is to give an indication of the variety of approaches to the multi-target tracking problem. It focuses on frequently cited work that can be taken to form, to some degree, a sampled approximation to the academic background to the field. It is of interest to consider issues such as the chosen target representation, the nature of the state vector, the construction of the multi-target proposal and likelihood densities, approaches to dealing with target occlusion, tracker birth and death etc.

As the selected cited work had to deal, to varying extents, with those issues, the published accounts tended to range from those that included detailed theoretical development to ones that relied upon highlighting the main points and referring back to earlier sets of publications containing varying degrees of clarification. The very brief summaries presented in this chapter necessarily exclude the detail but attempt to draw out the essential elements of the work.

As with the work cited in earlier chapters, there is no accepted notational convention in the field of multi-target tracking. The need to accommodate notation for multiple targets, multiple particle representations of those targets, multiple measurements etc. often leads to complex symbolic representations that can change between publications from the same group, and can lead to situations in which chosen symbols can mean different things between groups. An attempt has been made, in this chapter, to describe the work of others using a single consistent notational convention.

The main motivation for the development of multi-target trackers in recent decades has been in the context of radar or sonar detections, where a stream of signals from one or more sensors have to be linked to a possibly varying number of targets. A comprehensive taxonomy of the expanding range of multiple-target tracking (MTT) approaches in that field has been compiled by Pulford [80]. Developments in the related field of tracking multiple extended objects in image sequences have tended to draw on a few of the principles categorized by Pulford, but they have generally had to go beyond them to accommodate differences in aspects such as object detection methods and approaches to occlusion handling.

The chapter starts by considering the 'classical' Kalman filter type treatments that underlie the methods considered by Pulford. The direct extension of those treatments to the particle filter context is then considered. Non-classical approaches based upon multi-target extensions to the basic SIR filter (Algorithm 2) are looked at. The development goes on to consider treatments managing the multi-target state using mixture particle filters in preference to having a single particle filter for the joint state.

Work in the original field of radar and sonar multi-target tracking appears to have developed along two fronts. One pathway argues that the theoretical foundations for multi-target state vector treatments are weak and that the problem needs to be tackled using random finite sets. This leads to the Probability Hypothesis Density (PHD) filter. The other pathway disputes the challenges to the theoretical foundations and considers a vectorial approach in which the trackers are seen as independent partitions of a Joint Multi-target Probability Density (JMPD). Out of the two pathways only the particle PHD filter has claimed success in real-time tracking of multiple extended objects in the typical surveillance context. It is of interest to consider its mode of operation in a bit of detail to see if there are strong foundations in that claim to success.

4.2 Multi-target Bayesian recursion

The Bayesian recursion equations and the particle filter representations for multi-target states are similar to those for the single target state as described in Chapter 3. The multi-target state $\mathbf{X}_t$ can be represented by the set of individual states:

$$\mathbf{X}_t = \{\mathbf{x}_t^\tau\}_{\tau=1}^{n_\tau} \tag{4.1}$$

where n_τ represents the number of targets τ visible at time t. The history of states up to time t is represented as $\mathbf{X}_{0:t}$. Similarly the multi-target measurement $\mathbf{Z}_t$, with a history $\mathbf{Z}_{1:t}$, can be represented as the set

$$\mathbf{Z}_t = \{\mathbf{z}_t^m\}_{m=1}^{n_m} \tag{4.2}$$

where n_m represents the number of measurements $\mathbf{z}$ to be interpreted at time t. Equation Eq.(3.13) then becomes:

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) = \frac{p(\mathbf{Z}_t|\mathbf{X}_t)p(\mathbf{X}_t|\mathbf{Z}_{1:t-1})}{p(\mathbf{Z}_t|\mathbf{Z}_{1:t-1})} \tag{4.3}$$

For the multi-target particle filter each particle state can be represented by a weighted particle set $\{\mathbf{X}_t^i, w_t^i\}$ although other particle filter approaches to multi-target tracking will be described.

With the weighted particle set, as described above, the Bayesian recursion of the multi-target particle filter leads to a weight update equation of the form:

$$w_t^i \propto w_{t-1}^i \frac{p\left(\mathbf{Z}_t|\mathbf{X}_t^i\right) p\left(\mathbf{X}_t^i|\mathbf{X}_{t-1}^i\right)}{q\left(\mathbf{X}_t^i|\mathbf{X}_{t-1}^i, \mathbf{Z}_t\right)} \tag{4.4}$$

In general the differences between single and multi-target target tracking are to be found in the way the likelihood component $p(\mathbf{Z}_t|\mathbf{X}_t)$ is constructed from individual measurement likelihoods and target states, and the way the process is made computationally efficient by using measurement directed proposals in the approximation of the optimal importance density $p\left(\mathbf{X}_t|\mathbf{X}_{t-1}, \mathbf{Z}_t\right)$.

4.3 Classical approaches

The recognized 'classical' approaches to the MTT problem are the Multiple Hypothesis Tracking (MHT) filter and the Joint Probabilistic Data Association Filter (JPDAF) [81]. In the radar and sonar fields, or similar, it is assumed that point sources are being tracked. Difficulties to be accommodated are possible multiple sensor signals from a single target, observations from multiple targets fusing to produce a single detection, noisy signals, signals from clutter etc. The track hypotheses are generally maintained via Kalman filters.

The simplest MHT solution uses a nearest neighbour (NN) track update. This updates a track using only the measurement closest to the predicted track position and ignores other measurements in the data set. It can be adequate for widely spaced targets but can run into difficulties with close or crossing ones. Techniques designed to deal with such difficulties have been developed [81] but have limited applicability. The more general MHT solution [82] is a batch process, combining information from several frames to produce a track update with greater reliability. It looks at all possible association hypotheses between measurements and tracks but produces an exponentially growing tree structure that demands measurement gating and hypothesis pruning in order to maintain manageability. The general MHT approach has been applied in the image processing context of corner tracking [83], demonstrating the method's capabilities of track initiation, termination, occlusion handling etc. Its computational complexity was kept under control by dealing with a subset of 'best' hypotheses rather than explicitly enumerating all possible ones.

The JPDAF was developed as a sub-optimal solution to the MTT problem. It was designed to counter limitations associated with the need for heuristic pruning strategies in the MHT approaches. It has its own limitation, though, in that it assumes that the number of targets is known and constant. In its original implementation it used Kalman filters to track the targets. Instead of attempting to solve the target to measurement data association problem directly it used the approach of associating all measurements with every target to produce a combined innovation term to be used by the KFs. The KF innovation term, $\mathbf{n}$, is described in Section 3.3 as the difference between the target predicted state and the measurement. For measurement $\mathbf{z}^m$ and the predicted position $\hat{\mathbf{x}}^\tau$ associated with track τ it can be written as:

$$\mathbf{n}^{m,\tau} = \mathbf{z}^m - H\hat{\mathbf{x}}^\tau \tag{4.5}$$

In Eq.(3.22) it was seen that the innovation had a role in the correction to the estimated mean state for a single target; in the multi-target JPDAF context the KF for track τ is updated with the combined innovation [4]:

$$\mathbf{n}^\tau = \sum_{m=1}^{Z} \beta^{m,\tau}\mathbf{n}^{m,\tau} \tag{4.6}$$

where $\beta^{m,\tau}$ is the probability that measurement $\mathbf{z}^m$ originated from track τ.

In early considerations of the choice between the use of MHT and JPDAF approaches, Blackman [81] concluded that whilst the JPDAF offered computational convenience, the main disadvantage of the JPDAF was the absence of explicit mechanisms for track initiation and deletion. Those mechanisms had to be provided by external routines that could reduce the computational advantages. The MHT approaches had no difficulties associated with track initiation and deletion. However, at least one recent work has demonstrated an efficient incorporation of target initiation, continued existence and deletion in the KF based JPDAF [84]. In the context of radar tracking the most important factor in the choice between the approaches turned out to be the false alarm density. For high false alarm densities MHT became computationally heavy and JPDAF was favoured, for low false alarm densities the MHT approach was the one of choice.

The JPDAF approach has also been used with particle filters in place of Kalman filters [85, 86, 87, 88]. In this setting the JPDAF combines tracks and measurements through a modification of the particle weights. It is useful to describe a typical implementation [86] in order to illustrate its computational demands in comparison to the single target particle filter described in earlier chapters.

Assuming n_τ targets, the multi-target state at time t is represented as in Eq.(4.1). The n_s particles are then defined as:

$$\left\{\mathbf{X}_t^i\right\}_{i=1}^{n_s} = \left\{\mathbf{x}_t^{i,1}, \ldots.., \mathbf{x}_t^{i,n_\tau}\right\}_{i=1}^{n_s} \tag{4.7}$$

where each initial target state is given by $\mathbf{x}_0^\tau$ for targets $\tau = 1, \ldots., n_\tau$.

The n_m position measurements at each time step t are denoted by $\{\mathbf{z}_t^m\}_{m=1}^{n_m}$. The innovation between measurement m and the prediction associated with track τ is given by Eq.(4.5). The normalized innovation, with innovation covariance matrix $S_t^{m,\tau}$, is given by:

$$d_t^{m,\tau} = (\mathbf{n}_t^{m,\tau})^T \, (S_t^{m,\tau})^{-1} \, \mathbf{n}_t^{m,\tau} \tag{4.8}$$

and the n_m-dimensional association likelihood is:

$$p(\mathbf{n}_t^{m,\tau}) = \frac{1}{(2\pi)^{n_m/2} \, |S_t^{m,\tau}|^{1/2}} \exp\left\{\frac{-\,(d_t^{m,\tau})^2}{2}\right\} \tag{4.9}$$

The measurement-to-track association hypotheses are built by supplementing the product of the association likelihoods with factor representing the probability of detection P_D, the probability of detection failure $(1 - P_D)$ and the probability of a false alarm P_{FA} in a given hypothesis. A general expression for the probability in the n^{th} association hypothesis h_n is then:

$$P(h_n) = \delta_n P_D^{n_\tau - \phi_n} (1 - P_D)^{\phi_n} P_{FA}^{n_m - (n_\tau - \phi_n)} \prod_{m,\tau} p(\mathbf{n}_t^{m,\tau}) \tag{4.10}$$

where ϕ_n is the number of false alarms in hypothesis h_n and $\delta_n \in \{0, 1\}$ is a switch that can be used in a gating mechanism allowing only measurements close to the predictions to be involved.

Each particle is then associated with a weight:

$$w_t^i = \sum_{n=1}^{n_\lambda} P(h_n^i) \tag{4.11}$$

where n_λ is the number of possible association hypotheses.

The steps are the same as in the standard particle filter except in the weight calculation. Each target has n_s particles, each particle is then given a weight determined by all the measurement to track associations. The weight set for each state is then normalized and re-sampled. Individual target state estimation is carried out by calculating the weighted mean state of the target particle set.

4.4 Extensions to Condensation

4.4.1 Single multimodal pdf

The basic particle filter, as described in Section 3.4, was labeled as the Condensation (Conditional Density propagation) algorithm by Isard and Blake [89] in 1998. In that application it was used to track the motion of a contour of a single target in visual clutter. Koller-Meier and Ade [90] used a single Condensation-type tracker to track multiple objects with the process step, measurement step and estimation step implemented similarly to those of the basic filter. In this approach, the particles did not represent multi-target states as defined in Eq.(4.1); as in the basic Condensation algorithm they represented hypotheses on individual target states with the state vector carrying information about target position, velocity and shape. The complete particle set, however, represented samples from the whole multi-target state pdf. Individual targets were associated with modes in that pdf.

The measurements, carried out in the context of robot estimation of its surroundings, came from either a range sensor or an image-like matrix sensor. The robot was not interested in extracting the full tracks of the various objects, it was only interested in avoiding obstacles by keeping a frame-to-frame track of the modes. The measurement set consisted of either local environment positions or blob centroids in a simple image. Each particle likelihood was calculated using a truncated Gaussian having an argument based upon the distance between the particle position and the incoming measurement, i.e.

$$w_t^i = \begin{cases} e^{-\frac{u^2}{2\sigma^2}} & u < \delta \\ \rho & \text{otherwise} \end{cases} \tag{4.12}$$

where u is given by $u_t^i = \min_k \rho(\mathbf{z}_t^k, H\hat{\mathbf{x}}_t^i)$ and $\rho = \sqrt{\left[(\mathbf{p} - \hat{\mathbf{p}})^2 + (e_1 - \hat{e}_1)^2 + (e_2 - \hat{e}_2)^2\right]}$. The arguments of ρ represent measured ($\mathbf{p}$) and predicted ($\hat{\mathbf{p}}$) positions together with measurement and prediction for each of two dimensions e_1 and e_2. So it is a nearest neighbour type likelihood.

The approach dealt with newly appearing objects by re-sampling only a subset $n_{t-1} - n'_{t-1}$ of the particles at each step. The remaining n'_{t-1} particles were placed at the measurement positions $\mathbf{Z}_{t-1}$ to form an initialization density $p\left(\mathbf{X}_{t-1}|\mathbf{Z}_{t-1}\right)$ associated with that time step. The subset was then combined with the *a posteriori* density from the previous step:

$$p'\left(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1}\right) = \gamma p\left(\mathbf{X}_{t-1}|\mathbf{Z}_{t-1}\right) + (1-\gamma)p\left(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1}\right) \tag{4.13}$$

The weighting factor for combining the densities was chosen to be $\gamma = \frac{n'_{t-1}}{n_{t-1}}$.

For true newly appearing objects the initialization density particles would be in the vicinity of the associated measurement at time step t and would multiply through re-sampling. The pdf would then grow to accommodate newly appearing objects; particles linked to clutter would be less likely to reproduce.

The Koller-Meier approach was developed further for person tracking by Marrón et al. [91]. They increased the likelihood of true detections by incorporating measurements from stereovision cameras [92]. Their treatment did involve mode finding: they used a k-means clustering approach to link particle groups to tracked objects and to create new tracked objects based upon incoming measurements. They ensured that the n'_{t-1} new target samples were assembled by taking a proportion of particles from each group to counter the possibility that any one became over or under representative of a target.

4.4.2 Probabilistic exclusion principle

MacCormick and Blake [93] aimed to deal with problems that develop when several independent Condensation-type trackers use the same target model to describe the tracked objects. In their case the target representation was a head and shoulders contour. With all tracked objects sharing the same representation, closely moving or occluding objects can suffer either tracker 'hijacking' or tracker coalescence, where tracks can become attached to the incorrect object or merge into one. The authors sought to tackle the situation by developing a probabilistic exclusion principle. The aim was to allow multiple targets to occupy the same image position yet retain their independent tracks. Their Condensation modification involved extending the state dimensionality to accommodate two targets and developing a filter likelihood function that compounded the measurement probabilities associated with the possible observation states.

The exclusion principle was specific to the target representation used. Each head and shoulders contour was described by lines perpendicular to it and measurements consisted of the positions of points on the lines that intersected with target outlines in the edge image. In order to describe adequately the outlines the representation model had up to 50 dimensions.

In addition to the model components describing the shape configuration there was a label indicating the depth priority in the scene. Shape representations were allowed to overlap so that a single measurement line might intersect with a number of possible target outlines. The authors restricted the calculations to the assumption of possibly 0, 1 or 2 objects. Each particle had the capacity to describe a potentially occluding foreground object and a potentially occluded background object. Each object was described by a shape space vector $\mathbf{x}$. The state vector had the form $\mathbf{X} = (\mathbf{x}, y)$, with $\mathbf{x} = (\mathbf{x}^{\tau_1}, \mathbf{x}^{\tau_2})$ and y being a label indicating depth priority. The label values were $y = 1$ if τ_1 was nearer to the camera than τ_2 and $y = 2$ for the alternative. The particle filter process step incorporated a transition matrix T based upon a small fixed probability that y will change i.e

$$p\left(\mathbf{X}_t|\mathbf{X}_{t-1}\right) = T_{\tau_1\tau_2}\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right) p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) \tag{4.14}$$

The filter likelihood function involved the product of the probabilities of possible configurations:

$$p\left(\mathbf{Z}|\mathbf{x}\right) = \prod_{i=1}^{m} p_{c(i,y)}\left(\mathbf{z}^{(i)}|\mathbf{v}^{(i,y)}\right) \tag{4.15}$$

where $\mathbf{v}$ is a vector of proposed coordinates of the visible boundary intersections with the measurement lines, and $\mathbf{z}$ is a vector of the measured intersections. The probability densities p_c described the cases for $c = 0, 1, 2$ edge-boundary interactions at each of the m measurement lines in an outline. The target boundary positions $\mathbf{v}$ depended on the discrete occlusion state y.

The basic elements of the SIR filter were retained i.e. a process step, a measurement step, state estimation carried out by a weighted mean of the particles, and a re-sampling step to avoid particle degeneration. The re-sampling step, however, had the additional complication that the composite particles were decomposed into foreground and background elements which were then re-sampled separately.

4.4.3 BraMBLe

In the same spirit as the probabilistic exclusion principle, Isard and MacCormick [31] developed a Condensation extension, the Bayesian Multiple Blob tracker(BraMBLE), in the context of background subtraction blob-based human tracking measurements. The aim was to tackle the same problem of tracker confusion in occlusion situations. Their generalized cylinder model for the human appearance has been described earlier in Section 2.4.1.

The multi-target state was represented as $\mathbf{X} = \{n_m, \mathbf{x}^1, ..., \mathbf{x}^{n_m}\}$ for n_m objects, or blobs, and $\mathbf{x}^\tau$ is a vector representing the state of the τ^{th} object. The state vector for an individual object included a 2D floor position in world coordinates, velocity components, and parameters associated with the object shape as a vertically aligned generalized cylinder. The particle component associated with an object carried a label specifying to which object the component belonged. The particle set was the standard representation of weighted states: $\{\mathbf{X}_t^i, w_t^i\}$.

Central to the approach was the multi-blob likelihood function $p(\mathbf{Z}|\mathbf{X})$ derived using a fixed grid of G points across the image. At each grid location a measurement $\mathbf{z}_g$ was derived by considering the responses returned after applying radially symmetric Gaussian and Mexican hat filters to image patches centred on the grid point in each of the colour channels. The response vector was used to assign a predetermined likelihood $p(\mathbf{z}_g|l_g)$ that the measurement corresponded to background, with label $l_g = 0$, or to one of n_l pre-defined foreground models, i.e. $l_g \in \{1...n_l\}$. The likelihood was then:

$$p\left(\mathbf{Z}|\mathbf{X}\right) = \prod_{g=1}^{G} p\left(\mathbf{z}_g|l_g\right) \tag{4.16}$$

The process step was applied individually to each target component of $\mathbf{X}$. It used a first order velocity model, with Gaussian noise, for the spatial components, and Gaussian random variations for the velocity and shape components. Target labels were preserved in the step. Possible new targets were initialized using a Poisson probability assigning a zero velocity and set of initialization shape components at a randomly chosen floor position.

The multi-target particle likelihoods, $p\left(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{X}_t^i\right)$, were calculated by accumulating pre-calculated grid likelihoods in the image region proposed to be occupied by the object states associated with the particle. The objects were sorted into depth order of their real world position and their likelihoods were then calculated first for the nearest and then in depth order. The algorithm enforced an exclusion principle by assigning a zero likelihood to hypotheses in which distinct objects would occupy the same physical space in the world coordinates.

Individual target estimation was carried out by extracting the particle states and the weights for each labeled object, then calculating a weighted mean.

4.5 Mixture particle filters

4.5.1 The mixture tracker

Vermaak et al. [94] aimed to deal with the problem of tracker hijacking by using a separate particle filter for each object being tracked but developing the multi-target state as a weighted mixture of the filters, i.e.

$$p\left(\mathbf{X}_t|\mathbf{Z}_{1:t}\right) = \sum_{\tau=1}^{n_\tau} \pi_t^\tau p\left(\mathbf{x}_t^\tau|\mathbf{z}_{1:t}^\tau\right) \tag{4.17}$$

where the number of components in the mixture is n_τ and $\sum_{\tau=1}^{n_\tau} \pi_t^\tau = 1$. Each individual component behaved as a standard SIR filter with a normalized set of weights. In the particle representation Eq.(4.17) is written as:

$$p\left(\mathbf{X}_t|\mathbf{Z}_{1:t}\right) = \sum_{\tau=1}^{n_\tau} \pi_t^\tau \sum_{i\in\mathcal{I}_\tau} w_t^i \delta\left(\mathbf{x}_t - \mathbf{x}_t^i\right) \tag{4.18}$$

The authors showed that the Bayesian recursion of the mixture pdfs led to a sequential update of the mixture weights π. The component particle un-normalized weights w_t^τ were extracted as the sum of the individual particle un-normalized weights associated with that component:

$$w_t^\tau = \sum_{i\in\mathcal{I}_\tau} w_t^i \tag{4.19}$$

where $\mathcal{I}$ was a component indicator variable. The filter mixture weight was then updated as:

$$\pi_t^\tau = \frac{\pi_{t-1}^\tau w_t^\tau}{\sum_{\tau=1}^{n_\tau} \pi_{t-1}^\tau w_t^\tau} \tag{4.20}$$

After the mixture weight had been calculated the individual particle weights were normalized. A k-means style spatial re-clustering of the particles was applied and new mixture weights were calculated as:

$$\pi_t^{\prime\tau} = \sum_{i\in\mathcal{I}^{\prime\tau}} \pi_t^{c_t^i} \tilde{w}_t^i \tag{4.21}$$

where c_t^i is the re-allocated cluster for the i^{th} particle at time t. The individual particle weights were re-calculated as:

$$\tilde{w}_t^{\prime i} = \frac{\pi_t^{c_t^i} \tilde{w}_t^i}{\pi_t^{\prime c_t^i}} \tag{4.22}$$

to produce the updated mixture:

$$p\left(\mathbf{X}_t|\mathbf{Z}_{1:t}\right) = \sum_{\tau=1}^{n_\tau} \pi_t^{\prime\tau} \sum_{i\in\mathcal{I}_\tau} w_t^{\prime i} \delta\left(\mathbf{x}_t - \mathbf{x}_t^i\right) \tag{4.23}$$

The particles were then re-sampled.

The claim was that the re-clustering and re-weighting maintained the distinctiveness of the target modes. The particle filters within the mixture interacted only in the computation of the weights. Experiments were carried out on a synthetic example consisting position measurements of two targets. It showed that a standard pair of particle filters coalesced as the targets approached but the mixture particle filter retained the independent tracks. Experiments were also carried out using a video sequence of a football scenario using target colour and shape as a representation. The mixture particle filter managed to track four targets successfully where separate filters failed.

4.5.2 The boosted particle filter

Okuma et al. [66] worked in the context of tracking multiple interacting hockey players. They used a Vermaak-style mixture particle filter approach with a measurement directed proposal function. Targets were detected using a 23-layer cascaded Adaboost classifier. The classifier was trained using 6000 player images automatically extracted using software that looked for regions with low intensity surrounded high intensities indicative of the skating rink surface. Gaussian distributed particles placed in the regions associated with the Adaboost detections constituted a detected object proposal pdf $q_{ada}(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{z}_t)$. The detection pdfs were combined with the transition density of existing tracks to produce the 'boosted' proposal:

$$q_B\left(\mathbf{x}_t|\mathbf{x}_{0:t-1},\mathbf{z}_{1:t}\right) = \alpha q_{ada}\left(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{z}_t\right) + (1-\alpha)\,p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) \tag{4.24}$$

The parameter α was adjustable so that the relative importance between the detections and the transition density could vary depending upon tracking situations such as cross-overs and occlusions. If the Adaboost detections were not close to existing tracks then a setting of $\alpha = 1$ would suggest a new track, a setting of $\alpha = 0$ would produce a proposal distribution associated only with a known track.

They worked with a predefined multi-part object model based upon 110 component HSV colour histograms. The players were defined by bounding box rectangular regions with separate histograms for the upper and lower halves of the rectangle. The histogram extractions were weighted using an Epanechnikov kernel, as described in Section 2.6.1. For each individual target the combined Adaboost and transition particle sets were weighted using the Bhattacharyya distance and a Gaussian likelihood. The position estimation for each component of the mixture was carried out using the standard weighted mean.

The approach was developed further [95] by detecting linear and circular markings on the skating rink surface, constructing a homographic transformation from image coordinates to a top-view system and then tracking the players from that perspective

Cai et al. [96] built upon the development by adding a bipartite matching type data association step to produce a better linking of the Adaboost detections with existing tracks. They also applied a mean-shift procedure at the histogram measurement step to move particles closer to the likelihood peaks. Lu et al. [62] went even further and combined a HOG descriptor likelihood with the colour based one, although the focus of the increased sophistication was towards recognizing actions of the detected players rather than simply tracking them.

4.5.3 MCMC particle filter

Khan et al. [97, 98] describe a multi-target tracker designed explicitly to track a large number of interacting targets. The subjects were ants moving against a plain featureless background. The target representations were ant-size rectangular regions described by their length, width and rotation angle. The authors used a single Gaussian image background, built by averaging the images over the image sequence, and a single foreground indicator derived from the average appearance of the targets. This meant that all the targets used the same appearance model.

The joint state of all the targets at time t was represented by $\mathbf{X}_t$, as in Eq.(4.1). The particle set consisted of n_s samples $\{\mathbf{X}_t^i, w_t^i\}_{i=1}^{n_s}$ with each joint state given a weight w_t.

The appearance log likelihood, associated with an individual target state, was defined in terms of the summed pixel intensity deviations from the foreground and background mean values:

$$\log p\left(\mathbf{z}_t|\mathbf{x}_t\right) = c + \frac{1}{2}\sum_{p \in F}\left[\left(\frac{I_p - \mu_{bp}}{\sigma_{bp}}\right)^2 - \left(\frac{I_p - \mu_{fp}}{\sigma_{fp}}\right)^2\right] \tag{4.25}$$

where subscript p represents a pixel with intensity I_p taken from region F associated with the target, μ and σ represent the means and standard deviations of the background and foreground models, c is a constant.

Interactions between targets were modeled by a pairwise Markov random field (MRF) with interaction potential between individual target states τ_1 and τ_2 given by $\psi\left(\mathbf{x}_t^{\tau_1}, \mathbf{x}_t^{\tau_2}\right) \propto \exp\left(-g\left(\mathbf{x}_t^{\tau_1}, \mathbf{x}_t^{\tau_2}\right)\right)$, where $g\left(\mathbf{x}_t^{\tau_1}, \mathbf{x}_t^{\tau_2}\right)$ was a penalty term derived from the degree of pixel overlap between two states. The penalty reduced the weights of particles that had overlapping areas. The authors showed that the interaction term becomes a simple additional factor in the importance weight for the single target state so that:

$$w_t^{\tau_1} = p(\mathbf{z}_t|\mathbf{x}_t^{\tau_1}) \prod_{\tau_2 \in E_{\tau_1}} \psi(\mathbf{x}_t^{\tau_1}, \hat{\mathbf{x}}_t^{\tau_2}) \tag{4.26}$$

where $\hat{\mathbf{x}}_t^{\tau_2}$ is the estimated state for target τ_2 at time t given its state at $t-1$, and E_{τ_1} is the MRF edge subset with connections to the target τ_1.

The authors replaced the traditional particle filter importance sampling step with a MCMC Metropolis-Hastings procedure. Instead of applying the state transition step to the whole particle set and then calculating the likelihood, they made a change to the state of one target at a time, and accepted the change using a Metropolis-Hastings based decision. At time $t-1$ the state of all targets would be represented by a set of samples $\left\{\mathbf{X}_{t-1}^{i}\right\}_{i=1}^{n_s}$ each containing the joint state for the n_τ targets:

$$\mathbf{X}_{t-1}^{i} = \left\{\mathbf{x}_{t-1}^{i,1}, ..., \mathbf{x}_{t-1}^{i,n_\tau}\right\} \tag{4.27}$$

The update procedure was then to randomly select a joint sample $\mathbf{X}_{t-1}^{r}$ from the set of unweighted samples from the previous time step, and randomly select a target τ_1 from the n_τ targets. Using the previous state of this target, $\mathbf{x}_{t-1}^{r,\tau_1}$, the motion model $p(\mathbf{x}_{t}^{r,\tau_1}|\mathbf{x}_{t-1}^{r,\tau_1})$ would be applied to obtain a proposed individual target state $\mathbf{x}_{t}^{r,\tau_1}$, and a stochastic element added to produce a state $\mathbf{x}_{t}^{r*,\tau_1}$. The Metropolis-Hastings acceptance ratio was then calculated as:

$$a_s = \min\left(1, \frac{p(\mathbf{z}_t^{\tau_1}|\mathbf{x}_t^{r*,\tau_1})\prod_{\tau_2\in E_{\tau_1}}\psi(\mathbf{x}_t^{r*,\tau_1},\hat{\mathbf{x}}_t^{r,\tau_2})}{p(\mathbf{z}_t^{\tau_1}|\mathbf{x}_t^{r,\tau_1})\prod_{\tau_2\in E_{\tau_1}}\psi(\mathbf{x}_t^{r,\tau_1},\hat{\mathbf{x}}_t^{r,\tau_2})}\right) \tag{4.28}$$

If it turned out that the ratio in Eq.(4.28) was $\geqslant 1$ then the proposal would be accepted and the state of the target corresponding to τ_1 in $\mathbf{X}_t^r$ would be set to $\mathbf{x}_t^{r*,\tau_1}$. Otherwise it would be accepted with a probability a_s. If the proposal was rejected the selected target in $\mathbf{X}_t^r$ would be left unchanged and a copy of $\mathbf{X}_t^r$ would be added to the sample set. The sample set $\{\mathbf{X}_t^i\}_{i=1}^{n_s}$ would then represent the new multi-target state. The effect was to progressively increase the likelihood of each multi-target state through the incremental adjustments to the individual components. The estimated state for each individual target was then calculated using the weighted mean of the particle partitions associated with that target.

The approach was developed further by Smith et al. [99] who applied it to track varying number of human targets using colour histogram models with a Bhattacharyya based likelihood.

4.6 Joint Multi-target Probability Density

Kreucher, Kastella, Morelande et al. [100, 101, 102] chose to work with a Bayesian recursion of the full JMPD using the Bayesian recursion outlined in Section 4.2. The approach treats the density as a single probabilistic entity capable of tracking both the multiplicity and individual states of the targets.

The authors worked in the context of military vehicle tracking, using measurements of target position based upon devices such as ground movement sensors, radar or GPS. They emphasized that their treatment could be used with either thresholded or unthresholded measurements and that the use of unthresholded data would lead to greater accuracy in tracking.

It is generally recognized that dealing directly with the JMPD leads to computational intractibility due to the increased and varying dimensionality of the state space associated with multiple targets. Particle filter approximations to the JMPD demand large numbers of particles in order to sample the state space efficiently. Mahler [103] argued that beyond the computational difficulties were fundamental problems with the generalization of single target Bayesian filtering to multi-target situations, arguing that the classical techniques for optimally determining parameters of interest, such as the MAP and the expectation were not definable in multi-target situations. Kreucher, Kastella, Morelande et al. [100, 101, 102] aimed to address the problems by careful construction of the particle filter used to approximate the multi-target state. They allowed for factorization of the multi-target state when the targets were well separated. In effect this meant that the multi-target filter behaved as multiple independent ones. The SIR particle filter uses the sub-optimal kinematic prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as the sampling importance density in place of the optimal density $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)$; this allows the particle weights to be represented by the measurement likelihood as described in Section 3.4. Kreucher et al. aimed to construct a sampling density closer to the optimal by using measurement directed particle proposals. They also reduced the particle dimensionality by applying Rao-Blackwellization, i.e. using Kalman filter recursion on variables that had linear and Gaussian behaviour, to the target velocities.

Their measurement model divided the observation region into cells so that clusters of measurements within cells were returned as a single cell measurement. This reduced the dimensionality of the measurement vector. The particle filter probabilities were based upon the position vectors of the targets only, so the filter dealt with targets moving from cell to cell. The actual positions of the targets within the cell could be inferred at the state estimation step. The process step for each target, necessarily involving both position and velocity, were the same as described in Section 5.3.2.

An important element of their approach was target clustering within the multi-target particle representation. Targets with position Euclidean separations less than a given threshold were grouped into a cluster, and the process and measurement steps were then applied to clusters as though they were a single targets. Well separated targets were treated as single element clusters. A particle consisted not as a collection of independent target states, but as a collection of independent states and clusters. The particle components were referred to as either independent or coupled (clustered) partitions. At the re-sampling stage the independent partitions of the particle set were separated out and re-sampled according to their weights in the standard way. The coupled partitions went through an Auxiliary Particle Filter type stage with alternative realizations of the coupling being generated, re-weighted and re-sampled. The resulting particles, with new component proposals, were then re-sampled according to the particle weights. Individual target states were estimated using the weighted mean of the relevant partition. Components were re-grouped in a k-means clustering step.

Target birth and death were dealt with by the construction of an 'existence' grid which linked a measurement within a cell to a prior probability of a false alarm or target arrival. Similarly the absence of a measurement could be linked with a target death. The existence variable then enabled a measurement directed proposal for arriving or departing particle components.

This particle filter representation of the JMPD was applied to simulated measurement sets having a maximum of 10 targets. The authors worked with samples sizes ranging from 25 to 500 particles, but with each particle carrying position and velocity information for a number of targets. The observation region covered an area of $25km^2$ with each observation cell having dimensions $100m \times 100m$. Starting the simulation with 0 targets they reported successful acquisition and tracking of 10 targets across a range of signal-to-noise ratios using 200 particles.

4.7 The Probability Hypothesis Density filter

Mahler's criticisms [104, 103] of the JMPD approach included the recognition of difficulties associated with a representation of the multi-target state in which the individual states are concatenated into a single vector. If the multi-target posterior probability density accommodates the probability of varying numbers of targets it becomes difficult to define expectation values with particles of differing dimensionality. This was the basis of his proposal that multi-target vector based recursions were without adequate theoretical foundations. Mahler recognized that random finite sets (RFSs) offered a more appropriate representation and developed the field of Finite Set Statistics (FISST). A FISST Bayesian multi-object recursion was found to be computationally intractable, but it was shown that recursive propagation of the first-order moment of the multiple target state, the Probability Hypothesis Density (PHD), was possible. The PHD is a function defined in the single target state space. The integral of the PHD over a region of the state space gives the expected number of targets in the region. The modes of the PHD indicate the highest local concentration of contributions to the expected number of targets. The positions of the modes in the state space can be used to generate estimations of the target states.

SMC implementations of the PHD filter were developed [105, 106] and showed successful multi-target tracking in a diverse range of scenarios; for example: simulated tracks in heavy clutter [105], simulated vehicle movements in terrain [106], feature points in image sequences [107], radar [108], sonar [109], human groups [110], metallic objects in security screening image sequences [111], faces [112], vehicles on roads [113] etc.

The probability hypothesis density associated with the multi-target posterior $p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1})$, at time $t-1$, is denoted as $D\left(\mathbf{x}_{t-1}|\mathbf{Z}_{1:t-1}\right)$. In this case the variable $\mathbf{x}$ is interpreted not as a single target, but simply as a state in the single target state space. The PHD filter is a standard two-stage procedure: prediction and update. The PHD prediction is:

$$D(\mathbf{x}_t|\mathbf{Z}_{1:t-1}) = b(\mathbf{x}_t) + \int \varphi(\mathbf{x}_t, \mathbf{x}_{t-1}) D\left(\mathbf{x}_{t-1}|\mathbf{Z}_{1:t-1}\right) d\mathbf{x}_{t-1} \tag{4.29}$$

with

$$\varphi\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right) = p_s\left(\mathbf{x}_{t-1}\right) p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) + b\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) \tag{4.30}$$

where $b(\mathbf{x}_t)$ represents the intensity function of contributions to the PHD associated with targets newly appearing at time t, $b\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right)$ is the intensity function of the RFS of new targets emerging from the previous state $\mathbf{x}_{t-1}$, $p_s\left(\mathbf{x}_{t-1}\right)$ is the probability that the PHD contribution still exists at time t given that it had a previous state $\mathbf{x}_{t-1}$, and $p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right)$ is the single target transition probability density. The PHD update equation is:

$$D\left(\mathbf{x}_t|\mathbf{Z}_{1:t}\right) \simeq p\left(\mathbf{Z}_t|\mathbf{x}_t\right) D\left(\mathbf{x}_t|\mathbf{Z}_{1:t-1}\right) \tag{4.31}$$

with

$$p\left(\mathbf{Z}_t|\mathbf{x}_t\right) = 1 - p_D(\mathbf{x}_t) + \sum_{\mathbf{z}\in\mathbf{Z}_t} \frac{p_D(\mathbf{x}_t) p\left(\mathbf{z}|\mathbf{x}_t\right)}{\lambda c\left(\mathbf{z}\right) + \int p_D\left(\mathbf{x}_t\right) p\left(\mathbf{z}|\mathbf{x}_t\right) d\mathbf{x}_t} \tag{4.32}$$

where $p_D(\mathbf{x}_t)$ is the probability of detection, $p\left(\mathbf{z}|\mathbf{x}_t\right)$ is the likelihood of an individual contribution, λ is the average number of contributions due to clutter points per scan, and $c\left(\mathbf{z}\right)$ is the probability distribution of the clutter points.

The SMC implementation of the filter approximates the PHD with a large set of weighted particles. The filter works with an input of the measurement set $\mathbf{Z_t}$ consisting of possible target positions and clutter. The particle set is split into two groups n_L and n_J. The group $n_{L,t-1}$ represents the number of particles linked to the states $\mathbf{x}_{t-1}$ at time $t-1$, and $n_{J,t}$ represent particles linked to newly appearing targets at time t. At the prediction step the $n_{L,t-1}$ particles are proposed to new states using the standard process equation i.e. $\mathbf{x}_t = F\mathbf{x}_{t-1} + G\boldsymbol{\nu}_{t-1}$. The $n_{J,t}$ particles are placed within the state space according to a target arrival probability model. The $n_{L,t-1}$ state transition particles are given proposal weights:

$$\hat{w}_t^i = \frac{\varphi\left(\mathbf{x}_t^i, \mathbf{x}_{t-1}^i\right)}{q_t\left(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{Z}_t\right)} w_{t-1}^i \tag{4.33}$$

where $q_t\left(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{Z}_t\right)$ is the importance sampling function. The $n_{J,t}$ particles associated with target arrival are given weights:

$$\hat{w}_t^j = \frac{b_t\left(\mathbf{x}_t^j\right)}{n_{J,t} p\left(\mathbf{x}_t^j|\mathbf{Z}_t\right)} \tag{4.34}$$

where $p\left(\mathbf{x}_t^j|\mathbf{Z}_t\right)$ is the importance sampling function for new targets. At the update step the particle weights, representing the amplitude of the PHD, are then recalculated using the measurement likelihood and equations Eq.(4.31) and Eq.(4.32).

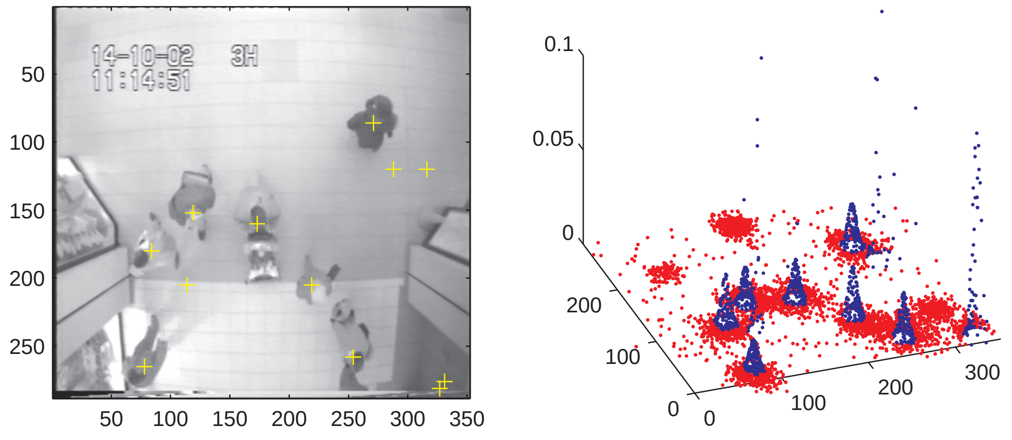
An example of a typical particle PHD recursion is shown in Algorithm 3. As it is a particle based Bayesian recursion it has some recognizable general features such as: generate particle proposals, assign weights to the particles, make state estimates, re-sample etc. But as the PHD filter is doing a job different to that of the SIR filter it is of interest to examine the differences and similarities in a bit more detail.

The input to the filter consists of: the particle states and weights from the previous iteration, the complete measurement set including false alarms and clutter, and a variable indicating the number of particles to be allocated to newly born targets. At Stage 1 the particles $\left\{\mathbf{x}_{t-1}^i, w_{t-1}^i\right\}_{i=1}^{n_{L,t-1}}$ from the preceding time step undergo a standard process diffusion. The weights for each particle at this stage represent the particle's fractional contribution to the previous estimate of the target number. In Algorithm 3 the sampling density is taken to be the proposal density, as in the simple SIR particle filter, so the weights are simply carried forward.

At Stage 2, in order to accommodate the possible appearance of $\Delta\tau$ new targets, a set of n_J additional particles are created with states drawn from a birth proposal distribution. For this distribution Wang et al. [110] used a uniform distribution across the image region, reflecting the possibility that a new target could appear anywhere within the field of view. In contrast, Maggio et al. [112] used a mixture of Gaussians centred on the observations $\mathbf{Z}_t$. Their choice was based on the recognition that drawing samples from a uniform distribution could require too many particles in their higher dimensional state space (involving target shape as well as position and velocity). Stage 3 is the weight update step. In this example the particle likelihood $p\left(\mathbf{z}|\mathbf{x}_t^i\right)$ is taken to be Gaussian centred on the measurement $\mathbf{z}$. The coefficient C sums the likelihoods of all particles for each measurement. If many particles are close to a measurement then the corresponding value of C will be high. The value

Algorithm 3 Particle PHD filter

```
Function def:
$\left[\left\{\mathbf{x}_t^j, w_t^j\right\}_{j=1}^{n_{L,t}}, \tau_t\right] = PHDfilter\left(\left\{\mathbf{x}_{t-1}^i, w_{t-1}^i\right\}_{i=1}^{n_{L,t-1}}, \mathbf{Z}_t, n_{J,t}, n_{L,t-1}, F, G, p_t, p_D, \Delta\tau\right)$

Stage 1
for $i = 1$ to $n_{L,t-1}$ do
   - generate proposed states:
   $\mathbf{x}_t^i \leftarrow F\mathbf{x}_{t-1}^i + G\boldsymbol{\nu}_{t-1}$
   - calculate the proposal weights using Eq.(4.33):
   $\hat{w}_t^i \leftarrow w_{t-1}^i$
end for

Stage 2
for $j = n_{L,t-1} + 1$ to $n_{L,t-1} + n_{J,t}$ do
   - generate new samples from a measurement based birth proposal function:
   $\mathbf{x}_t^j \sim p_t(\cdot|\mathbf{Z}_t)$
   - allocate weights to the new samples using Eq.(4.34):
   $\hat{w}_t^j \leftarrow \Delta\tau / n_{J,t}$
end for

Stage 3
- compute the weight sum coefficient
for $m = 1$ to size($\mathbf{Z}_t$) do
   $C_t(\mathbf{z}^m) \leftarrow \sum_{i=1}^{n_{L,t-1}+n_{J,t}} p_D(\mathbf{x}_t^i)\, p(\mathbf{z}^m|\mathbf{x}_t^i)\, \hat{w}_t^i$
end for

- update the weights
for $i = 1$ to $n_{L,t-1} + n_{J,t}$ do
   $w_t^i \leftarrow \left[1 - p_D(\mathbf{x}_t^i) + \sum_{\mathbf{z}\in\mathbf{Z}_t} \frac{p_D(\mathbf{x}_t^i)p(\mathbf{z}|\mathbf{x}_t^i)}{\lambda c(\mathbf{z}) + C_t(\mathbf{z})}\right] \hat{w}_t^i$
end for

Stage 4
- estimate the number of targets
$\hat{\tau}_t \leftarrow \sum_{i=1}^{n_{L,t-1}+n_{J,t}} w_t^i$

- call re-sample function to get $n_{L,t}$ new particles

$\left[\left\{\mathbf{x}_t^j\right\}_{j=1}^{n_{L,t}}\right] = resample\left(\left\{\mathbf{x}_t^i, w_t^i\right\}_{i=1}^{n_{L,t-1}+n_{J,t}}\right)$

- re-scale the weights so that they sum to the target number
for $i = 1$ to $n_{L,t}$ do
   $w_t^i \leftarrow \frac{\hat{\tau}_t}{n_{L,t}}$
end for
```



(a) Overhead frame, simulated detections (b) PHD particle weight distribution

Figure 4.1: PHD particle weights using simulated measurements, Overhead sequence

of C has implications in the following weight update. The weight update makes a likelihood style correction to the carried forward and new birth weights. If we consider an ideal case with $p_D \approx 1$ and $\lambda \approx 0$ then the weight update reduces to $w_t^i \approx \sum_{\mathbf{z}\in\mathbf{Z}_t} \frac{p\left(\mathbf{z}|\mathbf{x}_t^i\right)}{C_t(\mathbf{z})} \hat{w}_t^i$. The update factor is then a sum of likelihoods of all the measurements for a given particle, and we see $C_t(\mathbf{z})$ as a normalizing factor. If the particle is far from the measurements then the update factor is small. Similarly, if clutter is then included and the detection probability is reduced then the overall weight update factor is reduced. If, on the other hand, the particle density is low near a measurement, but there are a few new particles near that measurement, then the weight update factor can be relatively high. The effects are seen in Figure 4.1. The frame is from the Overhead sequence with simulated measurements added at each frame. In Figure 4.1(a) seven real targets can be seen together with 5 added clutter points. The corresponding updated particle weights are shown in Figure 4.1(b). The red colored particles have weights below an arbitrarily chosen threshold, and the blue particles have weights above. Successive weight updates have led to stable groupings of particles centred on the real targets. Low weight groupings of particles can be seen centred on clutter positions from the preceding frame. Low density, but high weight, particles can be seen centred on the current frame clutter points. The high weights in those regions mean that the weight based re-sampling step at Stage 4 will lead to reproduction of particles in those regions. If it had been that the measurements at those points had been associated with real targets then the new local high density of re-sampled particles would be placed close to the target in the next frame and the related particle weights would grow to form a ‘blue’ cluster at the update step. If, on the other hand, the measurements were randomly placed clutter then the new group of particles would not have favorable weight update at the next step and would die away through the re-sampling process.

The overall process is a feedback system in which particles close to consistent tracks reproduce positively and ones not associated with consistency die away. A significant drawback to the PHD filter is speed of collapse of a particle group if a detection is missed. In such cases the particles will have low weights and would not survive resampling.

Given consistent detections and consistent particle groupings it remains to have a cluster finding procedure in order to extract the target states. Maggio et al. [113] used a k-means approach coupled with a graph-based track update for data association. The graph analysis dealt with tracker issues such as misdetections, absence of detections, occlusions, target addition and removal.

4.8 Discussion

The chapter has presented an overview of frequently cited approaches to multi-target tracking. The aims were twofold: one aim was to give some indication of the academic background to the field, the other was to consider if any of the approaches offered tools or solutions that might be useful in the context of the situations being addressed by this thesis.

The content of the chapter suggests that there is no clear convergence of opinion as to the best way to deal with the general task of tracking a number of targets. The absence of convergence may be due, in part, to the differing degree of importance attached to the different challenges presented to the tracker in the given tracking contexts.

In the multi-modal pdf example [90], the problem, whilst presenting itself as multi-target tracking, was one of keeping track of multiple obstacles whilst the detector moved. There was no real interest in maintaining the specific identities of the obstacles or tracking through occlusion. It extended the formalism of the single target tracker, produced a measurement set at each time step, used a nearest neighbour likelihood to weight the particles and required a particle clustering step to update the obstacle position. It did not maintain an independent tracker for each target. It had a built in mechanism to deal with newly appearing obstacles.

It had features in common with the mathematically more sophisticated PHD filter. It distributed particles across the entire surveillance region, or in places where new targets were more likely to appear, and then gave each particle a weight reflecting its closeness to a measurement position. Both approaches then needed a clustering procedure to find the resulting modes in the distribution. The procedures were not designed to deal with occlusion or tracker hijack, those aspects needed a further data association step. The PHD filter could be seen simply as a clutter filter, the Koller-Meier tracker would behave as a clutter filter but with less sophistication than the PHD filter.

The question of whether to track the likelihood associated with the full multi-target state, or whether to manage a mixture of individual trackers has not been resolved. Those that tracked the multi-target state had to incorporate algorithmic steps to deal with the changing dimensionality of the multi-target state vector, incorporation of new targets, and then marginalization of the representation in order to estimate the individual target states. It was necessary to restrict the total number of targets that could be tracked at any one time.

Procedures for dealing with target occlusion, overlap and track crossing were not entirely satisfactory. Targets will overlap in the image plane. Whilst Khan's target proximity penalty might be useful in the special context of tracking in an overhead view, it is likely to be less useful in an oblique view where we would want trackers to appear to pass through each other in the image.

Those opting for the mixture particle filter approach introduced computations that involved particle weights associated with a specific target having a calculation link with all the other targets. But there was no compelling evidence to suggest that this linkage improved the overall performance of the tracking. Widely separated targets should be treated as independent and some of the approaches accommodated this. This was coupled with a recognition that special attention is needed when similar targets move closely together. Closely moving targets will present problems in radar and sonar type applications, but such problems might be less evident when targets have their own identifiable representation.

The approaches have been underpinned by a faith that Bayesian procedures applied to various degrees of a multi-target state might lead to computational simplicity, mathematical coherence, and the resolution of tracking problems in their implementation. It is suggested that the illustrative sample of work in this chapter does not strengthen that faith.

The next step in the work is to return to a more in-depth analysis of the actual behaviour of a single target particle filter tracker. It will look at the factors that determine the choice of parameters, approaches to various implementation techniques and state estimation etc. The proposal is that for a single target tracker, tracking an object with a clear representation, others in the vicinity should be classified as clutter. The work becomes a re-examination of the approach, implicitly rejected by standard multi-target tracking approaches, of the application of multiple instances of tracking a single target through clutter.

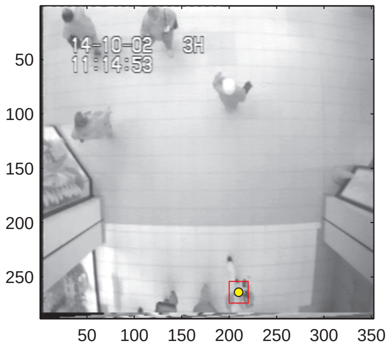
Chapter 5

Particle filter implementation

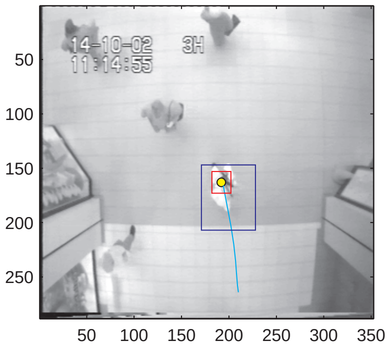
5.1 Introduction

This chapter deals with the basic implementation of the SIR particle filter as applied in the context of histogram-based object tracking in typical surveillance video. It considers the practical operation of the filter in some detail, rather than the theoretical aspects, in order to clarify the way that it works. It lays the foundations upon which refinements will be built in the later chapters. The behaviour of the filter is governed by the interplay of a number of parameters and it is only through the understanding and control of those parameters that generalizable aspects can be teased out. The parameters are introduced in the following sections, they are analyzed in more depth in Chapter 6.

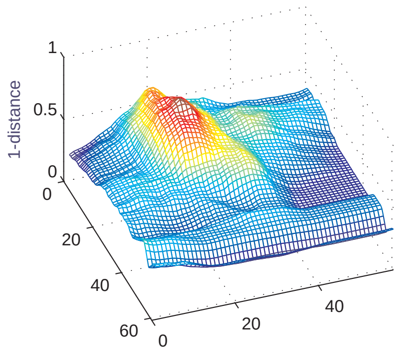
The treatment starts with a discussion of the basic tools used with the proposed particle filter. The approach is illustrated using a simple 8-bin histogram as the trackable representation. The standard feature distance measure, the Bhattacharyya distance, is described. The efficient extraction of multiple histograms is achieved through the use of the integral histogram, so the method used to implement it is described. The filter process step is considered in relation to the basic dynamics of the expected targets, the choice of process noise and the need for feedback. The appropriateness of the common approach to state estimation, using the weighted mean of the states, is considered. The operation and effectiveness of this implementation of the filter is illustrated using manual initialization of trackers in a range of settings. The drawbacks of the approach are considered in order to produce pointers to improvement.



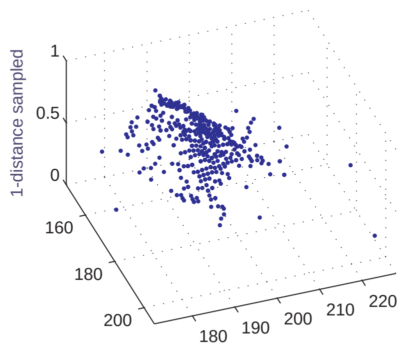
(a) initialization frame



(b) frame 10, with scan region highlighted



(c) frame 10, full scan similarity response



(d) frame 10, particle representation

Figure 5.1: Variation of histogram similarity in the vicinity of the target

5.2 Basic tools for tracking

5.2.1 Simple histogram based representation

It was concluded, in Chapter 2, that a colour intensity histogram representation would be appropriate for general purpose tracking of deforming and rotating objects. The approaches discussed in that chapter considered polychromatic representations with some indication of the spatial layout of the colour components. But the kind of feature vector dimensionality associated with the histogram representations described in that chapter can present a severe computational bottleneck in particle filter tracking where each object can demand a large number of particles. In an embedded system where the tracker algorithms are competing with other routines for shares of processor time it is necessary to consider lower dimensionality and reconsider the appropriateness of the chosen colour space. It was stated, in Section 1.2, that it would be prudent to assume that the chromatic content of CCTV footage would be weak. Initial experimentation in this work showed that grey scale histograms with $u = 8$ bins could be used to track objects and so this was taken as a starting point. Where the video was available as a standard YCrCb signal the Y channel was used as the monochromatic source. In cases where the video was RGB it was converted to monochrome (V of HSV) using the standard conversion [114]:

$$V = 0.299R + 0.587G + 0.114B \tag{5.1}$$

Figure 5.1 introduces the particle representation. Figure 5.1(a) shows an example tracker initialization. The reference eight-bin grey scale histogram to be tracked is constructed from the pixel values within the small rectangle shown in that figure. The dimensions of the rectangle are taken to be half those of the typical average object bounding box i.e. the average area of image that would just cover a target object. The expected bounding box size for pedestrians in this overhead view was taken to be about 40×40 pixels so the tracking rectangle is 20×20 pixels. This is, in effect, a use of a uniform kernel that gives equal weight to pixels within the tracking rectangle and zero weight to ones beyond the edges. The centre of the tracking box was placed approximately close to the centre of the object region to be tracked. Figure 5.1(b) shows the state of the tracker after 10 frames. The tracker rectangle has managed to maintain its position close to the central region of the object being tracked even though the object has changed shape and appearance. Figure 5.1(c) shows a similarity surface produced using a comparison between the reference histogram, initialized in the first frame, and ones extracted at pixel positions in a uniform grid placed over the target in frame 10. The similarity surface illustrates the trackable mode. The borders of the similarity grid are shown as a blue rectangle. Figure 5.1(d) shows a typical particle approximation to the distribution, the task of the particle filter is to extract the most appropriate mode from the particle representation.

5.2.2 Bhattacharyya distance and the likelihood function

Central to the operation of the particle filter is the likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$. This is used to update a weight given to a proposed state, and from those weights we can estimate the most likely state of a tracked object. We focus on the weight update equation as described in equation Eq.(3.57), i.e.

$$w_t^i \propto w_{t-1}^i p(\mathbf{z}_t|\mathbf{x}_t^i)$$

The likelihood is commonly realized through a Gaussian function of the form:

$$p(\mathbf{z}_t|\mathbf{x}_t) \propto \exp\left(-\frac{\rho^2}{2\sigma^2}\right) \tag{5.2}$$

where ρ is a comparison distance between a reference representation and one extracted at a position (x, y) in the image. The parameter σ is the standard deviation of the distribution of the distances. The function characterizes the distribution of the measurement noise $\boldsymbol{\eta}_t$ in Eq.(3.6).

In histogram-based tracking, the Bhattacharyya distance measure ρ_B [115], derived from the Bhattacharyya coefficient, is a popular choice for the distance measure ρ in Eq.(5.2). If two n-bin histograms are represented as $\mathbf{p} = \{p_u\}_{u=1...n}$ and $\mathbf{q} = \{q_u\}_{u=1...n}$, and normalized such that $\sum\limits_{u=1}^{n} p_u = 1$ and $\sum\limits_{u=1}^{n} q_u = 1$, then the Bhattacharyya *coefficient* is defined as:

$$\rho[\mathbf{p}, \mathbf{q}] = \sum_{u=1}^{n} \sqrt{p_u}\sqrt{q_u} \tag{5.3}$$

It has a geometrical interpretation [34] as the cosine of the angle between the n-dimensional unit vectors $\left[\sqrt{p_1}, ..., \sqrt{p_n}\right]^{\mathrm{T}}$ and $\left[\sqrt{q_1}, ..., \sqrt{q_n}\right]^{\mathrm{T}}$.

The Bhattacharyya *distance* is a dissimilarity measure. It is defined [115] to be:

$$\rho_B = \sqrt{1 - \rho[\mathbf{p}, \mathbf{q}]} \tag{5.4}$$

This formulation ensures that the distance between two identical histograms has $\rho_B = 0$ and dissimilar histograms have values $0 < \rho_B \leqslant 1$. The upper limit of 1 (rather than $\sqrt{2}$) occurs because the histogram bin values cannot be negative. This means that the unit vectors will have no negative components, the maximum angle between the two unit vectors will be 90° and hence $\rho_{B,max} = \sqrt{1 - 0}$.

Figure 5.2 shows the effect of a Bhattacharyya distance measure as a monochrome 8-bin histogram, extracted at the centre of the target, is compared with ones extracted at points along the scan line indicated. The vertical dotted lines to each side of the central ones shown in Figures 5.2(c) and 5.2(d) indicate the scan positions corresponding to the sides of the rectangular region. In the same frame comparison, Figure 5.2(c), the distance goes to zero at the extraction point. Figure 5.2(d) shows the scan response, taken two frames later, in which the extracted histograms are compared with the reference histogram taken from the earlier frame. It is seen that the overall shape of the response is similar with a well defined object localization. The distance does not go to zero in the second case because the histogram extracted at the centre of the target differs slightly from the reference histogram due to target change of shape and illumination variation. The characteristic of the response is important for particle filter tracking. If the response is too sharply peaked then it can be difficult to locate the target if it moves into a region where the particle density is low, as might be the case when the target undergoes sudden acceleration.

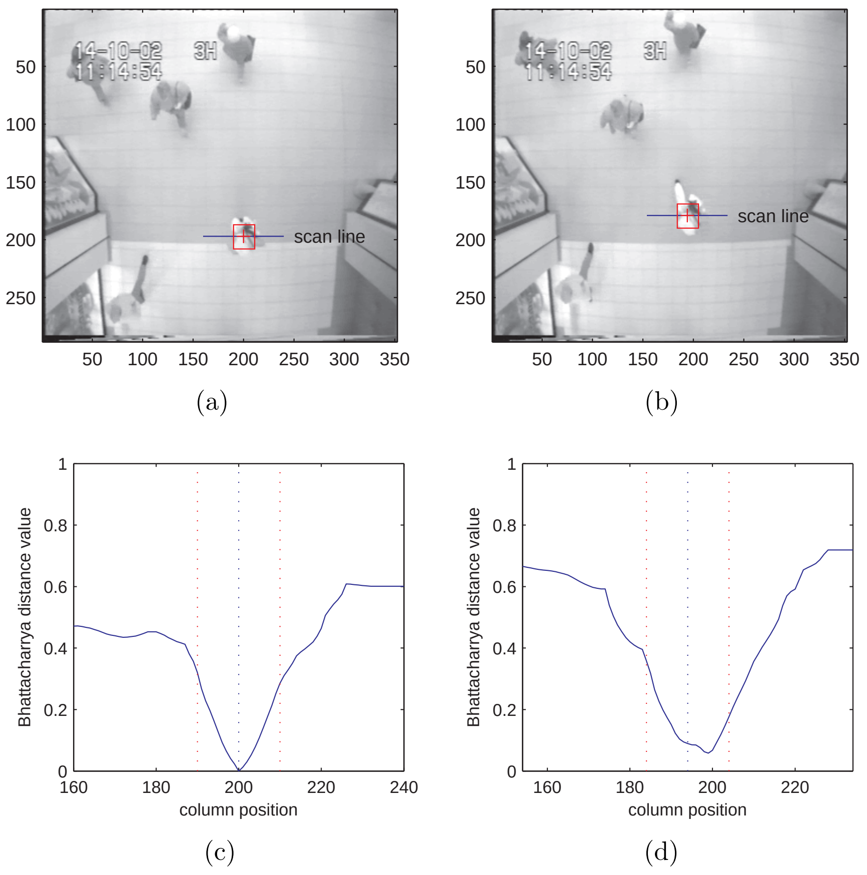


Figure 5.2: Histogram Bhattacharyya distance scan across target: (a) frame 6, (b) frame 8, (c) distance scan across target, same frame, (d) original histogram scan across target 2 frames later.

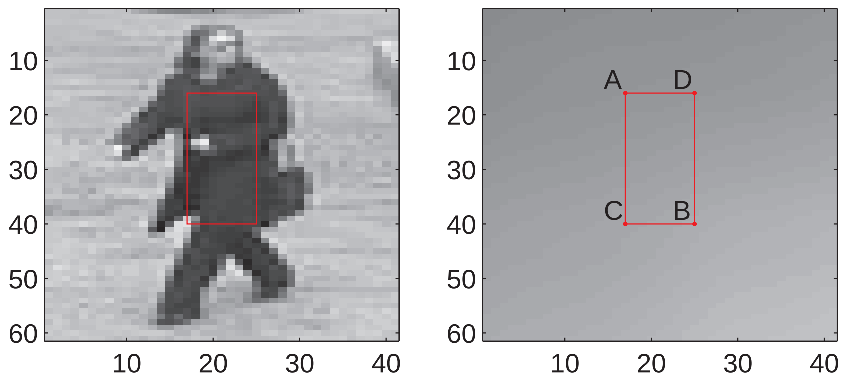
In practice, during tracking, the reference histogram is not allowed to 'age' with frame count. At each target detection in subsequent frames the reference histogram is updated using a fraction of the histogram $\bar{\mathbf{p}}_t$ extracted at the determined target location:

$$\mathbf{q}_t = (1 - \alpha)\mathbf{q}_{t-1} + \alpha\bar{\mathbf{p}}_t \tag{5.5}$$

The fraction α typically has the value 0.2 but it can be greater or smaller depending upon the confidence in the update histogram. An alternative is to use values that are multiples of 1/8, more suitable for fixed-point processing.

5.2.3 Integral histogram

Even with low-dimensional monochromatic histograms, the particle filter need for multiple histogram calculations from image regions with varying areas can have an undesirable computational cost. The 'integral histogram' [116] provides a tool to offset this cost. The integral histogram associated with a grey scale image I is constructed by producing a binary image slice S_u associated with each histogram



(a) tracking rectangle

(b) example histogram bin extraction for $u = 2$

Figure 5.3: Illustrating the tracking rectangle with dimensions height $h = |C - A|$, width $w = |D - A|$; and the integral histogram bin extraction for $u = 2$

bin u and a bin width of b:

$$S_u(x, y) = \begin{cases} 1 & \text{if } b(u-1) \leqslant I(x, y) < bu \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

Each binary image slice is then summed into an integral image representing bin u of the integral histogram P, i.e.

$$P(x, y, u) = \sum_{i<x, j<y} S_u(i, j) \tag{5.7}$$

An n-bin histogram, represented as $\mathbf{p} = \{p_u\}_{u=1\ldots n}$, can then be extracted in a constant number of operations independent of the target region size. As the value in each pixel in the image slice corresponding to a bin u is the sum of the binary values from the origin to that pixel, the bin value $p_u(x, y)$ for a region with upper left hand corner (x, y) and width and height (w, h) is obtained by:

$$p_u(x, y) = P(x, y, u) + P(x + w, y + h, u) - P(x, y + h, u) - P(x + w, y, u) \tag{5.8}$$

A typical tracking region and an example integral histogram bin extraction area for one bin is shown in Figure 5.3. The histogram bin extraction described by Eq.(5.8) translates to $p_u(x, y) = A + B - C - D$ in Figure 5.3(b). Porikli [116] showed that for an area search in a grey level image, using 8 bin histograms, the integral histogram method can run significantly faster than a conventional search. Whilst the integral histogram reduces the cost of multiple histogram extractions, it incurs an overhead cost in terms of the time required for its initial construction, and of memory allocation consumed by the need for an image for each bin. This restricts

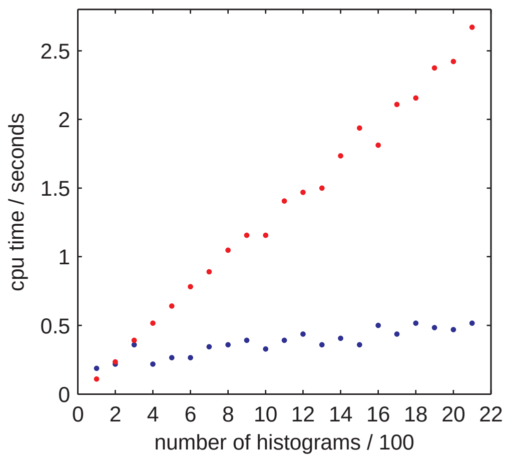


Figure 5.4: Matlab comparison of multiple histogram extractions: direct calculation(red points), integral histogram calculation(blue points)

the design dimensionality of the representation: using more grey scale levels would be more descriptive, but the marginal increase in quality of the description might not justify the increased cost in terms of resource consumption.

It has been stated that adequate tracking could be achieved using 8-bin grey scale histograms. Experimentation using an un-optimized Chipwrights hardware simulator showed that an 8-bin grey scale integral histogram could be constructed in less than 5ms for an image size of 320×256 pixels, an acceptable time for the targeted frame rate of 5fps (Source: Wang, W. AD-Group, personal communication). For small particle sets the integral histogram, with its initial overhead, does not give a time advantage; but if tracking multiple objects, with multi-part histograms, and each object demanding a few hundred particles, the advantages of the integral histogram approach become evident. Figure 5.4 illustrates the point. A simple Matlab routine was written to measure the processor time taken to extract groups of simple structured histograms, as described later in Section 5.5.1, for grey scale image regions having dimensions 10×25 pixels. It can be seen that the time for direct calculation of the histograms rises linearly as the number of histograms extracted increases. The integral histogram based extraction shows a much smaller increase in time with the number of histograms, and an intercept indicating the initial overhead. For numbers of histograms greater than 200 the integral histogram extraction shows a clear advantage. The advantage would be greater for histogram calculation involving larger image regions as the integral histogram based extraction time is independent of region size.

5.3 Particle filter stages

This section describes the implementation of the basic particle filter for single target tracking in the monochrome sequences described in Section 1.2. It outlines the steps of target initialization, motion prediction based upon basic dynamics, likelihood calculation via the measurement step, state estimation, and particle re-sampling. The purpose is not to focus, at this stage, on analysis to determine the optimal choice of parameters, it is to illustrate the operation of the filter and show that single target tracking can be achieved with parameter values manually chosen both for computational convenience and to reflect the expected dynamics of the objects in the given scenarios. For example, the number of particles was chosen to be a power of 2 because the number of data-paths in a SIMD processor is likely to be such a value. The number of particles had to be large enough to give a minimum density of one per target bounding box area in the event of particle set spread due to target occlusion etc., but not too large such that the computation load was too great. The process noise components were chosen such that the particle diffusion accommodated the anticipated velocities of the targets in the sequences etc.

It is recognized [4] that tracker initialization is a difficult and critical step. In order to illustrate the operation of the particle filter the initial tracking point was selected manually. This is an approach that is common in published work. Methods of automated tracker initialization will be discussed later in Chapter 8.

5.3.1 Initialization

The implementation is described with reference to the Overhead sequence, although the behaviour of the tracker is illustrated using the other sequences as shown later in Figure 5.8.

A tracker initialization point is shown in Figure 5.5(a). The approximate centre of the object anticipated bounding box was chosen as the point to track and an 8-bin histogram representation $\mathbf{q} = \{q_u\}_{u=1\ldots8}$ was extracted from a rectangular area with dimensions half those of the bounding box, as described in Section 5.2.1. The object state was described by the vector $\mathbf{x}_t = [x_t, v_{x,t}, y_t, v_{y,t}]^{\mathrm{T}}$ incorporating Cartesian position and velocity components.

An initial set of $n_s = 512$ particles was produced using a four-dimensional Gaussian distribution centred on the manually chosen starting coordinates $[x_0, y_0]^{\mathrm{T}}$ and initial velocities $[v_{x,0} = 0, v_{y,0} = 0]^{\mathrm{T}}$, i.e.

$$\mathbf{x}_0^i = \mathbf{x}_0 + \mathcal{N}(0, \Sigma) \tag{5.9}$$

The particles represent hypothetical states.

Within the diagonal noise covariance matrix Σ the standard deviations of the spatial components (σ_x, σ_y) were set to be one quarter of the tracking box width in pixels, and those of the velocity components $(\sigma_{v_x}, \sigma_{v_y})$ were set to be half of those of the spatial components. The values were chosen to give a distribution of the spatial components roughly within the bounding box of the object to be tracked. The spread of the particles represents measurement uncertainty in both the initial position and velocity of the object. A typical initial spread of particles is shown in Figure 5.5(b).

5.3.2 Process step

The motion of the target is described by the kinematic equations:

$$\begin{aligned} x_t &= x_{t-\delta t} + v_{x,t-\delta t}\delta t + \tfrac{1}{2}a_{x,t-\delta t}\delta t^2 \\ v_{x,t} &= v_{x,t-\delta t} + a_{x,t-\delta t}\delta t \\ y_t &= y_{t-\delta t} + v_{y,t-\delta t}\delta t + \tfrac{1}{2}a_{y,t-\delta t}\delta t^2 \\ v_{y,t} &= v_{y,t-\delta t} + a_{y,t-\delta t}\delta t \end{aligned}$$

The equations can be written in the form:

$$\mathbf{x}_t = F\mathbf{x}_{t-\delta t} + G\mathbf{a}_{t-\delta t} \tag{5.10}$$

with state vector $\mathbf{x}_t = [x_t, v_{x,t}, y_t, v_{y,t}]^{\mathrm{T}}$, time-step matrices F and G,

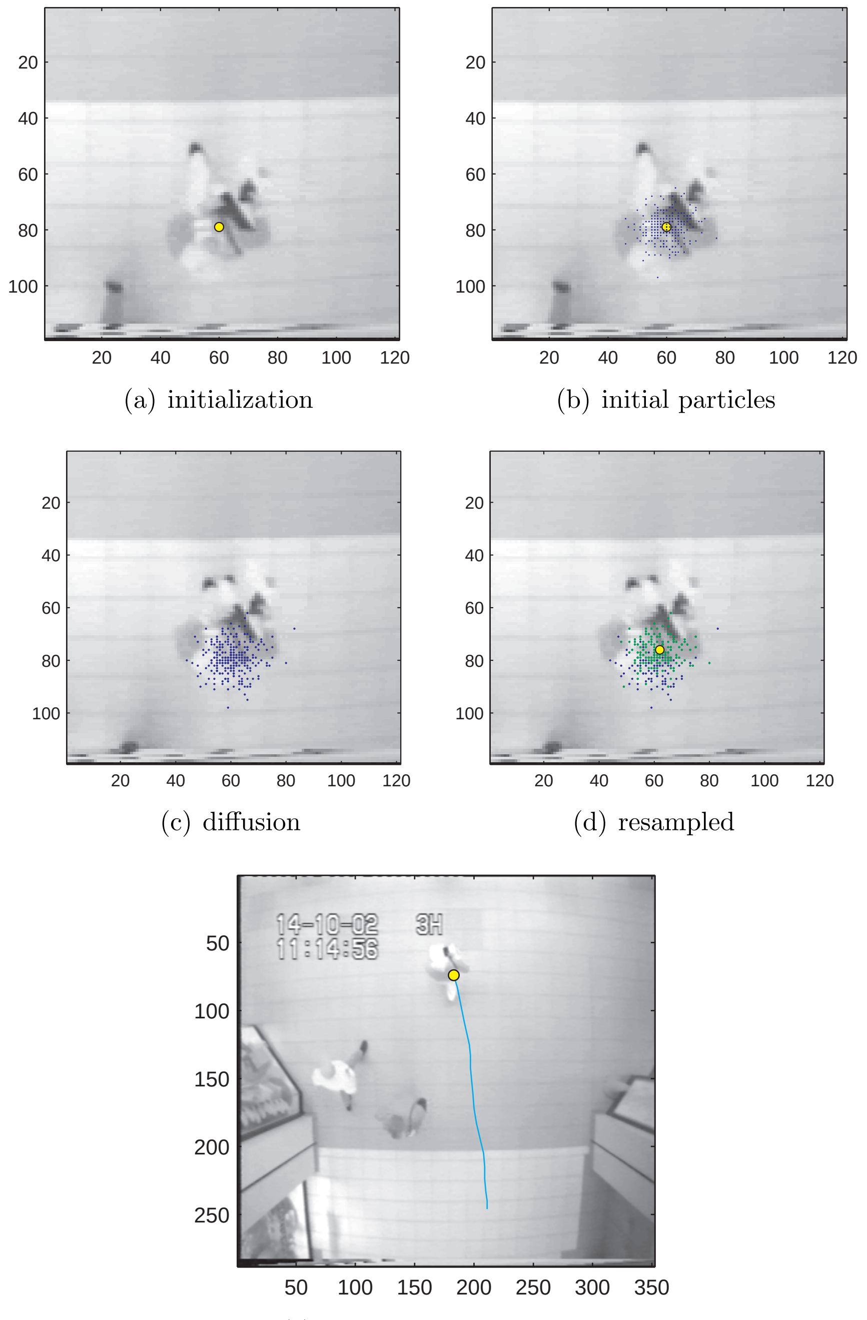
$$F = \begin{bmatrix} 1 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad G = \begin{bmatrix} \frac{1}{2}\delta t^2 & 0 & 0 & 0 \\ 0 & \delta t & 0 & 0 \\ 0 & 0 & \frac{1}{2}\delta t^2 & 0 \\ 0 & 0 & 0 & \delta t \end{bmatrix} \tag{5.11}$$

and acceleration

$$\mathbf{a}_{t-\delta t} = \left[a_{x,t-\delta t}\ a_{x,t-\delta t}\ a_{y,t-\delta t}\ a_{y,t-\delta t}\right]^{\mathrm{T}}. \tag{5.12}$$

For computational convenience we set $\delta t = 1$ to represent a frame step. The distances are measured in pixels rather than metres. With these settings the velocity units end up being pixels per frame rather than metres per second.

With the particle representation, the motion model $\mathbf{x}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\nu}_{t-1})$ is realized by splitting Eq.(5.10) into a deterministic component and a stochastic one, and using $\delta t = 1$. The stochastic component is introduced by replacing the acceleration



(a) initialization

(b) initial particles

(c) diffusion

(d) resampled

(e) track through 20 frames

Figure 5.5: Illustrating the particle sequence

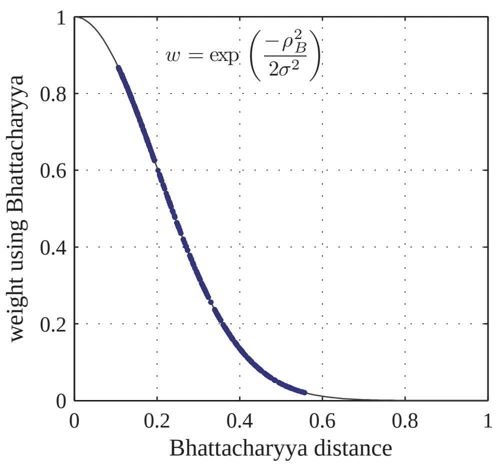


Figure 5.6: Gaussian weightings for the particle set

terms with Gaussian random values to get, for the i^{th} particle,

$$\mathbf{x}_t^i = F\mathbf{x}_{t-1}^i + G\boldsymbol{\nu}_{t-1}^i \tag{5.13}$$

with the process noise $\boldsymbol{\nu}_{t-1} \sim \mathcal{N}(0, \Sigma)$, having zero mean and covariance diagonal $(\sigma_x^2, \sigma_{v_x}^2, \sigma_y^2, \sigma_{v_y}^2)$. The transition probability density $p(\mathbf{x}_t^i | \mathbf{x}_{t-1}^i)$ is characterized by Eq.(5.13).

The particle aspect of the process step can be thought of as a search for the next position through the diffusion of the particles under the motion model. Each particle represents a hypothetical object path. The process noise variances have to be chosen such that the particle placement is neither too wide or too narrow. If the placement is too wide then there is a chance that the overall density of particles is too small and few end up in the region of the target mode at time t. If it is too narrow then a rapidly accelerating target can be lost as the particles fail to keep up. Figure 5.5(c) shows the frame following that of Figure 5.5(b) with the particles diffused symmetrically around the last position. The symmetrical diffusion arises because the target velocity is unknown at that step and so is assumed to be zero.

5.3.3 Measurement

The measurement step involves using the integral histogram to extract an 8-bin histogram at each new particle position. The Bhattacharyya distance between the reference histogram and the particle histogram is calculated, and a weighting obtained using Eq.(5.2) with a measurement noise standard deviation $\sigma = 0.2$. The weightings for the particle set in Figure 5.5(d) are shown in Figure 5.6.

The high-weight particles correspond to the ones in the upper region of the particle set as seen in Figure 5.5(d). Those are the ones with random velocities

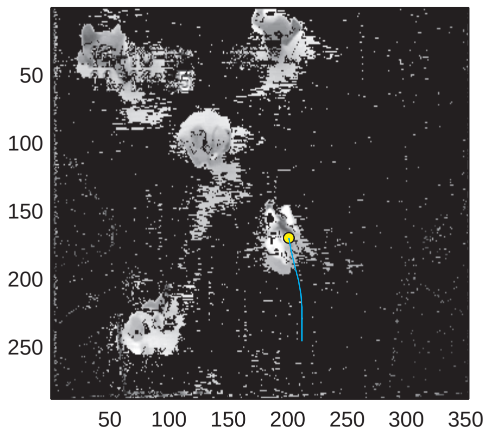


Figure 5.7: Temporal difference motion feature

similar to the true velocity of the target. Particles with Gaussian weights less than 0.01 were reset to $w = 0.01$. This introduces a ‘tail’ to the distribution. It ensures that the particle population survives re-sampling in the event of object occlusion. It also ensures a surviving contribution of alternative hypotheses in the particle population.

Whilst the histogram is the dominant trackable feature in this version of the particle filter it is not sufficient for effective tracking. Early experimentation showed that as the histogram aged static elements in the scene could present distracting histograms. It is necessary to supplement the main feature. A simple feature to add is object motion, i.e. the object has a trackable histogram feature and it also has motion. The feature was generated by simple temporal frame differencing with a low threshold to produce a binary motion image. The motion image, superimposed on the grey scale image is shown in Figure 5.7. At a frame rate of 5 fps the textured objects presented sufficient difference to survive the threshold; static background elements were below threshold. At each particle position a tracking box area was extracted from the binary temporal difference image and the motion pixel fraction of the area, corresponding to the fraction of white pixels within the area in the binary mask, was calculated. The fraction formed a motion weight w_m. The particle weight was then calculated by multiplying the Gaussian weighting w_G by the motion weight if it was determined that the object velocity was above a minimum value, chosen to be a fraction of the smallest dimension of the tracking box:

$$w_t^i = \begin{cases} w_G^i w_m^i & \text{if } \|\mathbf{v}\| > \|\mathbf{v}\|_{\min} \\ w_G^i & \text{otherwise} \end{cases} \tag{5.14}$$

5.3.4 State estimation

In this simple version of the particle filter the target state is extracted using the weighted mean of the states as described by Eq.(3.59) in Section 3.4, i.e:

$$\mathrm{E}\left[\mathbf{x}_t|\mathbf{z}_{1:t}\right] \approx \sum_{i=1}^{n_s} \tilde{\omega}_t^i \mathbf{x}_t^i$$

where E[·] is the expectation, and $\tilde{w}_t^i$ is the normalized weight.

This approach is adequate if the process noise is small such that the particles do not spread much beyond the bounding box of the target. However, it turns out to be not adequate in general; alternative methods, discussed in Chapter 7, have to be used.

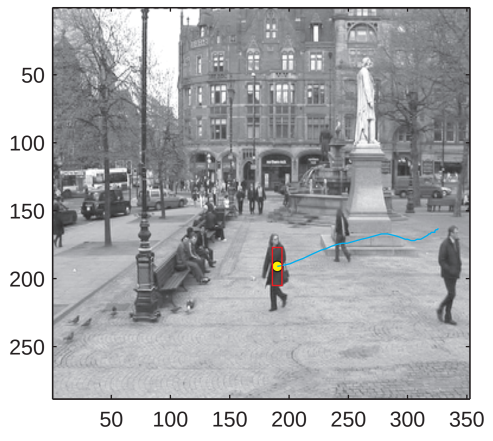
Once the object state has been determined it remains to update the histogram representation. A new histogram is extracted at the object position and is used to incrementally update the target representation as described by Eq.(5.5) in Section 5.2.2, i.e:

$$\mathbf{q}_t = (1-\alpha)\mathbf{q}_{t-1} + \alpha\bar{\mathbf{p}}_t$$

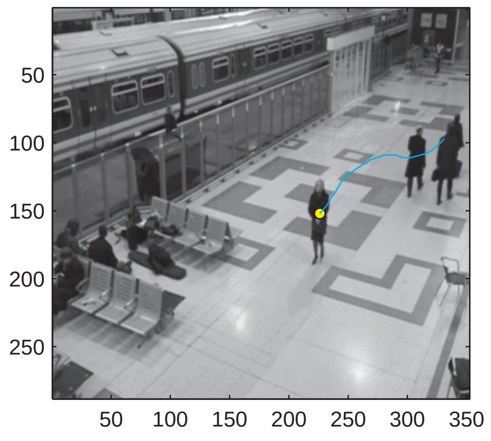
with $\alpha = 2/8$. This slow update ensures that the tracker retains a 'memory' of the feature being tracked and does not easily switch to track a distraction. The update can be suppressed when the overall weights are low, indicating that the track has been temporarily lost.

5.3.5 Re-sampling

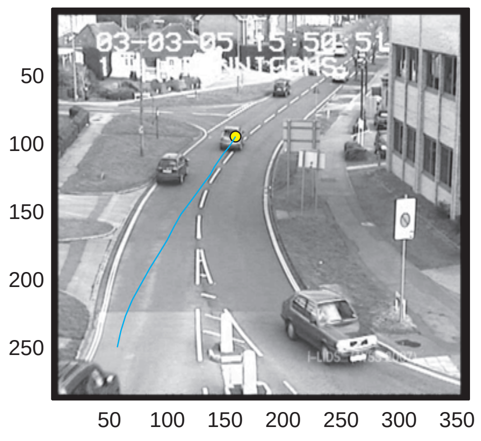
Once the object state has been extracted the particle set needs to be regenerated through re-sampling. The method used in this illustration is multinomial re-sampling as described in Section 3.4.2. In the tracking example shown in Figure 5.5(d) the original prior particle set is shown in blue and the re-sampled particles are indicated as green. The r-sampled set represents an estimate of the uncertainty of the measurement at that step. The surviving particles turn out to be those that had the randomized velocity at the last step appropriate to enable them to end up at the positions shown. Through the re-sampling process the tracker quickly learns the velocity of the target.



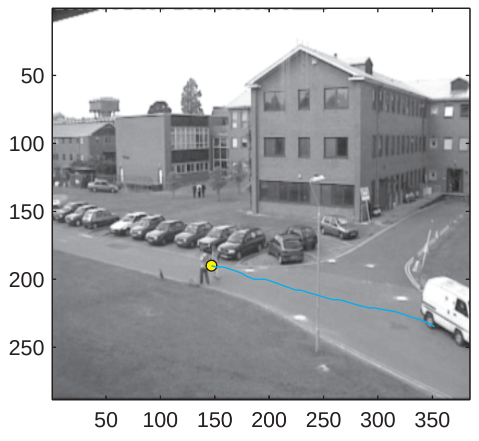
(a)



(b)



(c)



(d)

Figure 5.8: Examples of simple 8 bin histogram tracking with manual initialization

5.4 General behaviour of the simple implementation

Figure 5.8 shows typical examples of simple 8-bin grey scale tracking using the particle filter described above. The conditions are relatively controlled and have manual initialization. Under the controlled conditions the trackers respond well to brief and partial occlusion and deal with the apparent change of size as they move through the oblique views. In Figures 5.8(a) and 5.8(b) the tracked targets pass behind other objects having similar shape and histogram signatures. The tracking survives mainly because of the choice of the process noise values and a degree of feedback aimed at adjusting appropriately the size of the tracking box.

5.4.1 Dealing with depth views

Most typical surveillance applications of object detect and track systems are likely to have views similar to those in Figure 5.8. Objects will change size, shape and appearance as they move through the scene. Venegas et al. [117] sought to deal with such difficulties by transforming the oblique view image into a plan view and carrying out particle filter tracking in that plane. A common alternative approach is to carry out a perspective camera calibration [118, 119, 120] and apply motion models in the world coordinates. In the interest of keeping computation to the minimum the approach adopted in this work was to apply a simple linear scaling. The view was calibrated by extracting object width and height, in pixels, at a row position corresponding to a far view, then repeating the procedure at a near view row position. A linear relationship between object size and row position was derived and used to determine the expected bounding box dimensions given a row coordinate. The process noise parameter was taken to be proportional to the bounding box smaller dimension. This approach ignored true perspective but gave sufficient information to allow the tracker to adjust its search area appropriately as the object moved through the scene. The acceptable response to this first order approach can be seen in the examples shown in Figure 5.8.

5.4.2 Occlusion behaviour

A valuable aspect of the particle filter is its ability to deal with a degree of occlusion. The sequence of images in Figure 5.9 illustrate the behaviour of the particle filter in such circumstances. An artificial occlusion has been added to the view in the Overhead sequence. Figure 5.9(a) shows the object approaching the occlusion. In Figure 5.9(b) the object has been partially covered by the occlusion, the re-sampled particles cluster on the fraction of the object still uncovered but some of the low weight particles in the occlusion region survive the re-sampling process. As those particles are at the front of particle cloud they are likely to be carrying velocities sufficient to see them through the occlusion. In Figure 5.9(c) the object has emerged from the occlusion, the forward moving particles land in the region of the object and are strongly re-sampled. The particle cloud has found the object and by Figure 5.9(d) the particle set is beginning to group more closely on the target. The track through the occlusion is uninterrupted as though the obstacle was not there. With a broader occlusion there is always the chance that the track might not recover if the dispersion of the cloud is such that no particle lands on the object. But it only takes a few particles landing on the object for re-sampling to regenerate the tracker.

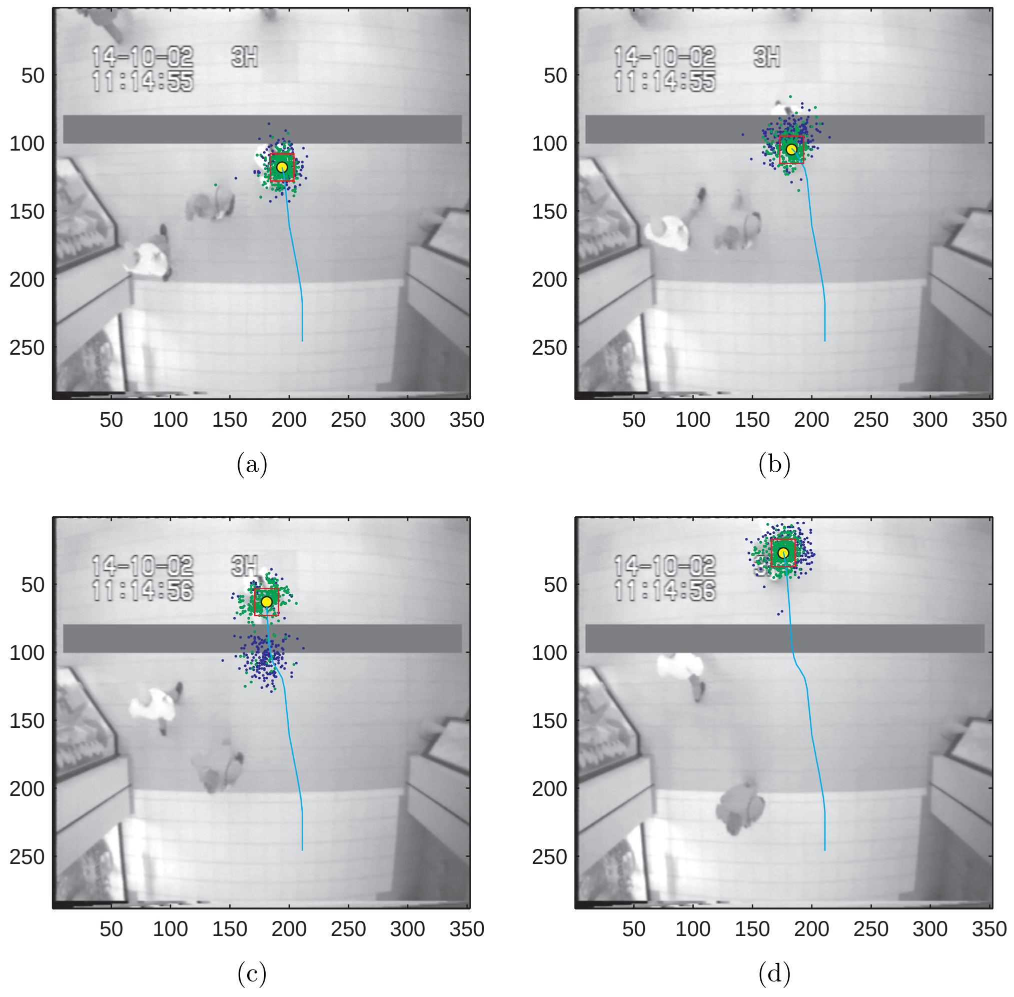
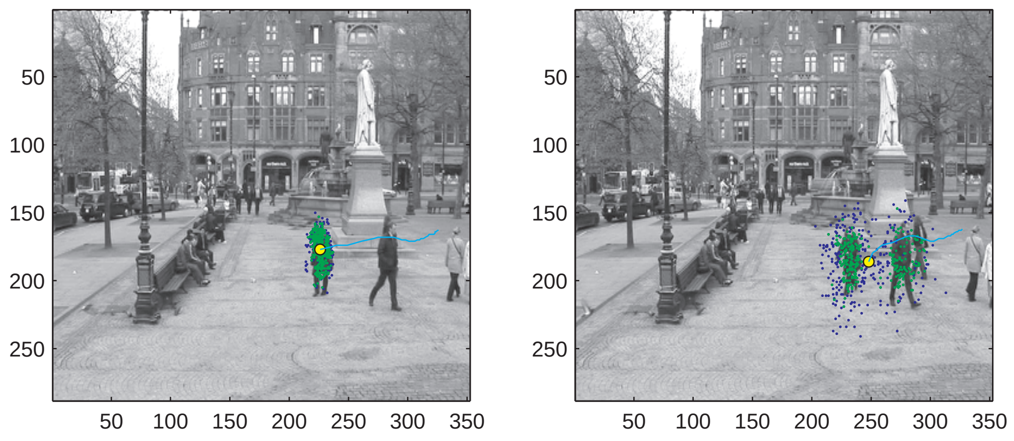


Figure 5.9: Occlusion behaviour with simple 8 bin histogram tracking

5.4.3 Inadequacy of weighted mean estimate

The minimum variance, weighted mean, estimate of target state delivers satisfactory tracking in the examples given above. If the process noise is chosen such that the particle cloud concentrates well on the object, suggesting low uncertainty in position, then the mean gives a good estimate. However, if we return to the view that the particle set is an approximation to a broad area representation of the pdf then we would expect to use a larger process noise and have a broader particle spread. Figure 5.10 illustrates the effect of using the weighted mean estimate with a tracker having large process noise. The example shows two objects with similar grey scale signature. The object being tracked passes behind the second object. In Figure 5.10(a) the process noise is small enough to keep the particle cloud compactly with the target being tracked. At the point of occlusion the momentum of the cloud is sufficient to carry most of the particles with the tracked object, the weighted mean remains within the bounding box of the object. In Figure 5.10(b) a larger process



(a) (b)

Figure 5.10: Failure of weighted mean state estimate with large process noise

noise has been used. At the point of crossing the larger spread of the particles means that a larger proportion migrate to the second object and re-sampling ensures the growth of the bimodal distribution. The weighted mean now falls between the two objects. The histogram update means that it loses the identity of the object being tracked and after a few frames the histogram converges to that of the local background with a consequent loss of object track. The example illustrates the need for control of the parameters.

5.4.4 Process noise feedback

A number of researchers have tried to incorporate process noise feedback into their systems in order to alleviate difficulties of the type illustrated in the last section. Bagdanov et al. [121] incorporated feedback moderated process noise in which the relative contributions of the static $[x, y]^{\mathrm{T}}$ and dynamic $[v_x, v_y]^{\mathrm{T}}$ parts of the process equation varied depending upon the likelihood of the proposal particle. Arguing that it was detrimental to introduce uncertainty into both position and velocity they introduced a sigmoid 'blindness' function which returned a small static process noise but large dynamic one if the measurement likelihood was high and the reverse if the likelihood was low. But it is difficult to see how this additional calculation has any overall effect. For example, considering a single dimension x_t, delivery of a particular uncertainty in state δx_t, which is the principal function of the process noise, can be achieved equally by a preceding uncertainty in position, i.e.

$$x_t + \delta x_t = (x_{t-\delta t} + \delta x_{t-\delta t}) + v_{t-\delta t}\delta t \tag{5.15}$$

or a preceding uncertainty in velocity i.e

$$x_t + \delta x_t = x_{t-\delta t} + (v_{t-\delta t} + \delta v_{t-\delta t})\delta t \tag{5.16}$$

It is irrelevant whether the required δx_t comes from the added $\delta x_{t-\delta t}$ or the alternative $\delta v_{t-\delta t}\delta t$.

Maggio and Cavallaro [122] described a simple adaptive transition model based upon the average changes of state in the previous κ frames:

$$\mathrm{E}_t[\Delta \mathbf{x}] = \frac{1}{\kappa}\sum_{n=t-\kappa}^{t} |\mathbf{x}_n - \mathbf{x}_{n-1}| \tag{5.17}$$

Their state transition model was:

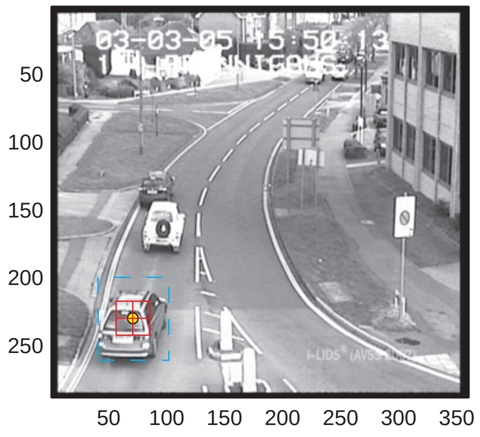
$$\mathbf{x}_t = \mathbf{x}_{t-1} + C_t \boldsymbol{\nu}_t \tag{5.18}$$

where $\mathbf{x} = [x, y]^{\mathrm{T}}$, $\boldsymbol{\nu} \sim \mathcal{N}(0, I)$ and $C_t \propto \mathrm{E}_\kappa [\Delta \mathbf{x}]$

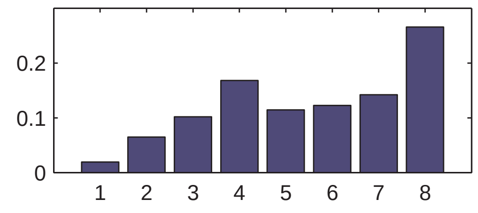
As it is a zero-th order motion model, equation Eq.(5.18) represents a search around the particle state at time $t-1$. In contrast, a first-order motion model, incorporating velocity as in Eq.(5.13), more usefully represents a search around the *predicted* state. A first-order model automatically takes account of recent velocities through the particle filter re-sampling step: those particles with velocities significantly different to that of the target will end up in low likelihood positions and reduce their chances of reselection. The particle filter quickly learns the current velocity of the object. The process noise added to this velocity has to be such that particles have a spread about the expected target position in the next frame sufficient to deal with large accelerations. In the approach described in this chapter the process noise is related to the size of the target bounding box which changes with depth in the field of view. It recognizes that, with typical frame rates, objects of interest rarely move beyond a bounding box dimension from frame to frame.

5.5 Improving the representation

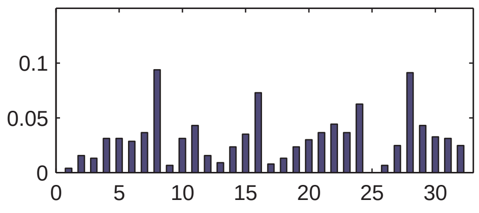
Given that 8-bin grey scale histogram tracking works at a basic level for single object, it is of value to consider the limitations of that representation and look at strategies for alleviating them. The strategies involve a compromise between developing a detailed representation that would clearly label and identify the individual targets to be tracked, but would make the similarity response around the target too narrow to be useful for particle filter analysis, and having a computationally economic but



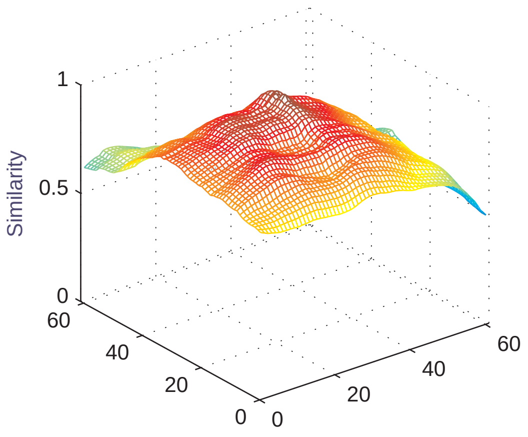
(a) histogram four quadrants and scan region



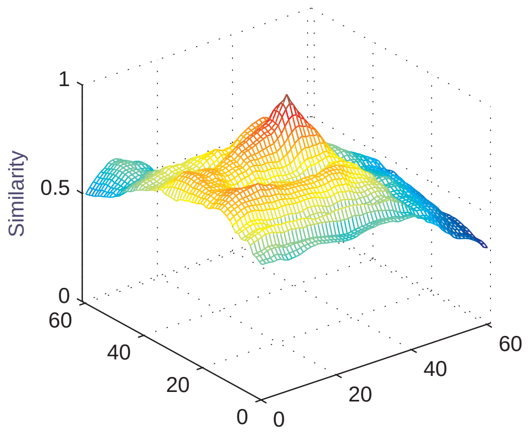
(b) 8-bin histogram



(c) 32-bin four quadrant histogram

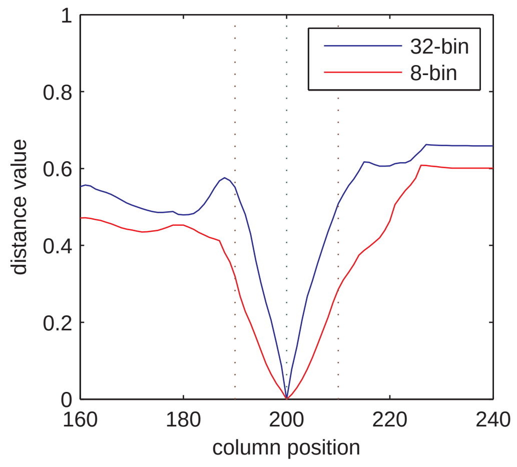


(d) 8 bin histogram similarity scan over the region

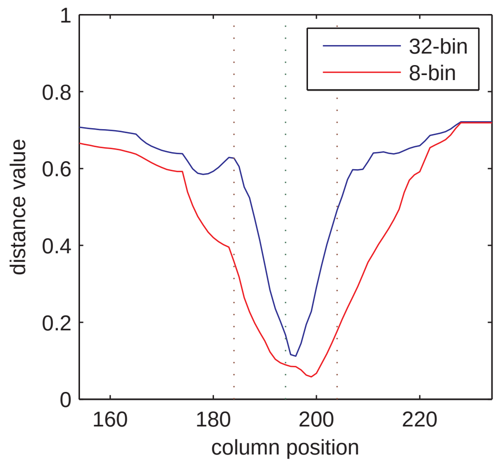


(e) 32 bin histogram similarity scan over the region

Figure 5.11: Improved target representation using a 32-bin four-quadrant histogram.



(a) scan at frame 6



(b) scan at frame 8

Figure 5.12: Improved representation response using the four-quadrant (4q) histogram.

simple representation with a broad similarity response that is particle filter friendly but might have an unacceptable chance of track loss.

Increasing the number of histogram bins would allow a clearer representation of the object to be tracked, but each histogram bin demands memory space equal to that required of a single image. On the other hand, representing a target with the broad grey scale quantization associated with 8-bins can lead to track loss if the grey scale range of pixels within the target is small and most of the grey scale values fall into a small number of the bins. In such cases, there is a chance that illumination changes or object deformation can transfer pixels from one histogram bin to a neighbouring one and the distance between the template histogram and that of the object can become artificially large. An alternative to the histogram representation is a Kernel Density Estimate (KDE), or Parzen Window [123], in which a kernel K is centred on each grey scale abscissa value, and the ordinate of the representation $\tilde{f}(x)$ at any abscissa value x is the sum of the contributions from the rest, i.e.

$$\tilde{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K\left(\frac{x - x(i)}{h}\right) \tag{5.19}$$

where h is the bandwidth of the kernel. But experimentation showed that the KDE did not, in general, deliver improvement in tracking sufficient to justify the extra computational cost.

In order to work with an 8-bin representation but increase reliability of target identity, a multi-part structured representation similar to, but simpler than, those described in Section 2.6.4 is considered.

5.5.1 Using a simple structured histogram

Subdivision of the tracking rectangle into regions provides a more detailed signature to track. Using an elliptical multi-part representation of the type described by Maggio et al. [61] would increase the demand for floating point calculation and possibly produce a representation too precise for effective particle filter tracking. The vertically stacked representations of Iwahori et al. [67] and Okuma et al. [66] restrict the signatures to vertical objects. Splitting the tracking rectangle into four quadrants, as shown in Figure 5.11(a), and concatenating 8-bin histograms from the quadrants in a clockwise fashion produced a 32-bin representation that was found to work equally well with all object types and views. Figure 5.11(d) shows the result of a Bhattacharyya similarity scan for the 8-bin histogram representation, with the scan over points within the outer rectangle shown in Figure 5.11(a). The reference histogram was taken at the centre of the outer rectangle in the frame shown. Figure 5.11(e) shows the equivalent scan using the four quadrant histogram. It can be seen that the response at the centre of the outer rectangle is more strongly pronounced yet still broad enough to produce a useful response surface in the particle filter representation. The strong central response survived in comparison to that for the 8-bin histogram in subsequent frames.

Figure 5.12 shows the improved response with the four quadrant(4q) histogram using the Overhead sequence frames as shown in Figure 5.2. Figure 5.12(a) shows that the 32-bin histogram produces a sharper response than the 8-bin one but retains a spread within the bounding box region of the object. In Figure 5.12(b) it can be seen that the 32 bin histogram response is retained and has a peak closer to the centre of the tracked object than does the 8 bin response. In general the 32-bin representation offers improved and more robust tracking than the simpler 8-bin representation.

5.6 Discussion

This chapter has described the basic tools used for particle filter histogram based tracking and has given an outline of a simple implementation of the SIR filter. It has illustrated that with the choice of a few constraints e.g. manual initialization, preset process noise etc. adequate tracking can be achieved with relatively simple representations. The filter described here is not robust enough for commercial applications, it is too dependent on those constraints. It needs to be developed beyond this first step. The treatment has hinted that, for robust tracking, issues like the use of the weighted mean estimate need to be re-examined and alternatives sought. Other aspects of the filter also need to be examined in the context of their intrinsic validity and their computational implications, for example: the appropriateness of the re-sampling method used, the choice of the Gaussian likelihood function, the choice of distance measure etc. These aspects will be the focus of the next two chapters.

Chapter 6

Distance measures and particle likelihood

6.1 Introduction

This chapter looks critically, and in detail, at implementation practicalities. It considers what simplifications can be made to produce a system that still has particle filter characteristics and is capable of acceptable tracking behaviour, but has more favorable computational characteristics. It questions the common choices of likelihood functions and representation distance measures. It looks to see if simpler likelihood functions can be used in order to reduce the number of floating point calculations required. It asks if the number of interacting adjustable parameters can be limited, or limits can be placed on the ranges of parameter values.

It starts by working with the common approach of a Gaussian likelihood and derives a rationale for the choice of its associated standard deviation. Popular choices of likelihood distance measures are reviewed. It is shown that they are interrelated, and that the simplest and most computationally attractive of them can be just as effective as the others. It then questions the choice of the Gaussian likelihood itself, and argues that a simple linear approximation is adequate.

6.2 Choice of Gaussian likelihood standard deviation in published work

There has been a good range of high-quality work relating to colour and grey-scale intensity or intensity gradient histogram based particle filter tracking during the last decade. The use of a Gaussian likelihood measure, as described in Section 5.2.2, has become standard and use of the Bhattacharyya distance [115] as the dissimilarity, or distance, measure ρ is common, although other measures are used. The Bhattacharyya distance returns a value in the range [0,1].

As the numerical value of the distribution standard deviation σ is generally taken to be a tuneable design parameter we can expect to see a range of reported values dependent on the choice and range of the distance measure. Out of a large body of work in the field some representative examples have been selected in order to make the point. A recent approach by Czyz et al. [60], using the Bhattacharyya distance measure reports using $\sigma = 0.8$ in one example and $\sigma = 0.6$ in others. Another, by Maggio et al. [124], using a multiple feature approach and the Bhattacharyya distance, reports using $\sigma = 0.09$ for the colour component. Early examples of Bhattacharyya distance colour based particle filter tracking [63, 95, 66] expressed the likelihood in the form $\exp(-\lambda\rho^2)$ with $\lambda = 20$. This corresponds to a value of $\sigma = 0.16$. A more recent example, by Snoek et al. [125], uses $\lambda = 50$ to give $\sigma = 0.1$. Lu et al. [62] use a HOG distance measure in a Gaussian likelihood with $\lambda = 10$ to give $\sigma = 0.22$. In a slightly different approach, Song et al. [126] uses $\exp(-(1-\alpha)\rho^2)$, with $\alpha = 0.6$ giving $\sigma = 1.12$. It is common, however, to see reported work in which numerical values for the likelihood parameters are not given [59, 96, 127, 128, 129].

Considering the reported variation of Gaussian likelihood parameters in the context of colour/grey-scale histogram based tracking involving the Bhattacharyya distance, and the apparently crucial choice of the parameters as design components, the role of the likelihood standard deviation and its relationship to the distance measure is examined in order to get better guidance on the choice of design values.

6.3 Choice of the likelihood standard deviation for particle weighting and re-sampling

The particle filter state transition equation, Eq.(5.13), and its associated noise $\boldsymbol{\nu}_{t-1}$ propagates the particle representation of the probability density function from the last frame into the current one. In normal operation a proportion of the particles will end up in the region of the target position with the remainder spread into places that the target might reach with changes of speed and direction. In the ideal case this would return a set of particle distance measures spread across the full range [0,1]. In practice, with occlusion or changes in scene illumination, the distribution of particle distances can cover less than the full range and show either a positive or negative skew. A typical particle distance frequency distribution from real data is shown in Figure 6.1. The Gaussian weighting for that set, using a likelihood standard deviation of $\sigma = 0.2$, together with curves for $\sigma = 0.1$ and 0.3, is also shown.

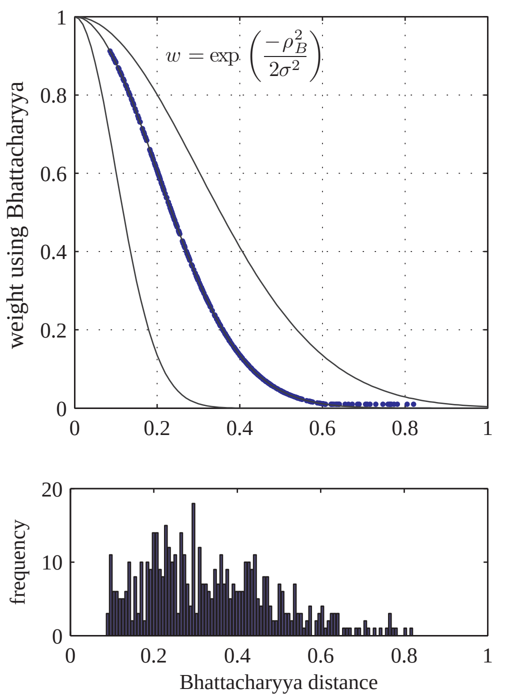


Figure 6.1: Gaussian weighting of the Bhattacharyya distance measure for a typical set of 512 particles from a tracking sequence. The standard deviation of the distribution is set to $\sigma = 0.2$. The curves to the left and right of the central one have standard deviations of $\sigma = 0.1$ and $\sigma = 0.3$ respectively. The lower figure shows the frequency distribution of the distances.

The particle weighting has two functions: it provides values for the estimate of the object state through the weighted mean, and it provides feedback for the evolution of the particle set. The process of re-sampling with replacement from the weighted particle set removes particles that are not very representative of the target state and produces multiple copies of those that are more representative. The weighting of the particle set changes the effective size of the set as unrepresentative particles end up making diminished contributions to the calculations at each stage. The effective particle number can be estimated [69, 74] using:

$$n_{eff} = \frac{1}{\sum_{i=1}^{n_s} (\tilde{w}_t^i)^2} \tag{6.1}$$

where $\tilde{w}_t^i$ is the normalized weight of the i^{th} particle at the t^{th} step, as defined in Eq.(3.47), and n_s is the number of particles.

In some particle filters the effective particle number is used to trigger re-sampling if n_{eff} falls below a given threshold. For example Doucet et al. [74] used a threshold of $n_s/3$ in their early simulations. It is common, however, for re-sampling to be implemented at each iteration irrespective of the value of n_{eff}. Even if it is not used to trigger re-sampling, n_{eff} can be used to give an indication of the degree of degeneracy of the filter.

It is possible to relate the effective particle number to the standard deviation of the Gaussian likelihood weighting function for the ideal case in which the dissimilarity measures are uniformly spread across the full [0,1] range. The particle weights are taken as $w^i = \exp\left(-\frac{l_i{}^2}{2\sigma^2}\right)$ where l_i has n_s uniformly incremented discrete values from 0 to 1. We consider $\sigma \leqslant 0.3$ so that we can assume that the area under the Gaussian in the range [0,1] is approximately equal to the area in the range [0,∞]. The area under a Gaussian curve is given by:

$$\int_{-\infty}^{+\infty} \exp\left(\frac{-l^2}{2\sigma^2}\right) dl = \sigma\sqrt{2\pi} \tag{6.2}$$

which is recognizable as the standard normalizing coefficient for a Gaussian distribution.

For the likelihood weighting we are only interested in the positive half of a Gaussian curve with area given by:

$$\int_{0}^{+\infty} \exp\left(\frac{-l^2}{2\sigma^2}\right) dl = \frac{\sigma\sqrt{2\pi}}{2} \tag{6.3}$$

In the case of the discrete distribution of Gaussian weights w_t^i, with $\delta l = \frac{1}{n_s}$, an element of area dA is given by:

$$dA^i = w_t^i \cdot \frac{1}{n_s} \tag{6.4}$$

In the limit of large n_s, the area under the weight curve is given by:

$$\sum_{i=1}^{n_s} dA^i = \sum_{i=1}^{n_s} w_t^i \cdot \frac{1}{n_s} = \frac{\sigma\sqrt{2\pi}}{2} \tag{6.5}$$

so that:

$$\sum_{i=1}^{n_s} w_t^i = \frac{n_s\sigma\sqrt{2\pi}}{2} \tag{6.6}$$

The normalized weights are then given by:

$$\tilde{w}_t^i = \frac{2}{n_s\sigma\sqrt{2\pi}} \exp\left(-\frac{l_i{}^2}{2\sigma^2}\right) \tag{6.7}$$

The effective particle number is given by:

$$n_{eff} = \frac{1}{\sum\limits_{i=1}^{n_s} \left(\tilde{w}_t^i\right)^2} \tag{6.8}$$

We can write the denominator of Eq.(6.8) as:

$$\sum_{i=1}^{n_s} \left(\tilde{w}_t^i\right)^2 = \frac{4}{{n_s}^2\sigma^2 2\pi} \sum_{i=1}^{n_s} \left(w_t^i\right)^2$$

If $w_t^i = \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right)$ then $\left(w_t^i\right)^2 = \exp\left(-\frac{{l_i}^2}{\sigma^2}\right)$. Equating the area under the squared weight curve with that under the half Gaussian we get:

$$\int_0^{+\infty} \exp\left(\frac{-l^2}{\sigma^2}\right) dl = \frac{\sigma\sqrt{\pi}}{2} = \sum_{i=1}^{n_s} \left(w_t^i\right)^2 \cdot \frac{1}{n_s}$$

so that:

$$\frac{n_s\sigma\sqrt{\pi}}{2} = \sum_{i=1}^{n_s} \left(w_t^i\right)^2 \tag{6.9}$$

and the denominator of Eq.(6.8) becomes:

$$\sum_{i=1}^{n_s} \left(\tilde{w}_t^i\right)^2 = \frac{4}{{n_s}^2\sigma^2 2\pi} \cdot \frac{n_s\sigma\sqrt{\pi}}{2}$$

leading to:

$$n_{eff} = \sqrt{\pi}\sigma n_s \tag{6.10}$$

For $\sigma = \{0.1, 0.2, 0.3\}$, we get $\sqrt{\pi}\sigma = \{0.18, 0.35, 0.53\}$, producing n_{eff} of approximately one fifth, one third and one half respectively of the particle set.

It is also useful to refer to the Bhattacharyya coefficient and consider the angle between the target vector and vectors from the particle set. For $\sigma = \{0.1, 0.2, 0.3\}$ the 3σ Bhattacharyya distances correspond approximately to angles between feature vectors of $\theta = \{25, 50, 79\}^\circ$ respectively. For a uniform distribution of distances, using a Gaussian with $\sigma = 0.1$ will result in an effective particle set of size $n_{eff} \sim n_s/5$ focusing on feature vectors within 25° of the target vector. This looks attractive at first sight but, in practice, distribution skew will mean that there will be many fewer than $n_s/5$ contributing. Re-sampling of such a diminished set is likely to result in loss of diversity amongst the particle population. It is suggested that such a value of σ is likely to make the filter have difficulty tracking the target.

If we look at the other extreme with $\sigma = 0.3$ we are likely to have an effective sample size greater than $n_s/2$ but with significant contribution from feature vectors that are at angles up to 80° from the target vector. If n_s is small then the large angle feature vectors could have too great an undesirable influence on the weighted mean and there is a significant chance of the particle cloud diverging.

The argument points to the suggestion that $\sigma = 0.2$ is a compromise value. This will maintain a particle set consisting of feature vectors within 50° of the target vector, it will produce an effective sample size of $n_s/3$, and drawing from 60% of the distance axis it will be reasonably tolerant to skew of the weight frequency distribution in the particle set. It is interesting to note that Doucet et al. [74] opted for an n_{eff} threshold of $n_s/3$ to trigger re-sampling in their particle filter simulations.

6.4 Distance measures other than Bhattacharyya

A general class of measures for distances between n-dimensional real valued vectors $\mathbf{a}, \mathbf{b}$, referred to as the L_r norm, has the form [130]:

$$L_r(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{n} |a_i - b_i|^r \right)^{1/r} \tag{6.11}$$

As we are using the normalized histograms $\mathbf{p}, \mathbf{q}$ as the feature vectors the vector components have to be the square roots of the bin values for the vectors to have unit length, hence we expect to see the equivalent L_r norm in the form:

$$L_r(\mathbf{p}, \mathbf{q}) = \left(\sum_{i=1}^{n} |\sqrt{p_i} - \sqrt{q_i}|^r \right)^{1/r} \tag{6.12}$$

The Matusita distance between two probability distributions $\mathbf{p} = [p_1, .., p_n]^{\mathrm{T}}$ and $\mathbf{q} = [q_1, .., q_n]^{\mathrm{T}}$ is defined as [131]

$$\rho_M(\mathbf{p}, \mathbf{q}) = \left(\sum_{i=1}^{n} (\sqrt{p_i} - \sqrt{q_i})^2 \right)^{1/2} \tag{6.13}$$

and hence can be seen as an L_2 norm.

The Matusita distance is directly related to the Bhattacharyya distance. The relationship is discussed in [34], although the authors state the Matusita distance as the square of the original definition in [131]. The original definition is used in this work. By expanding the square under the summation in Eq.(6.13) and gathering the terms it can be shown that $\rho_B = \rho_M/\sqrt{2}$. The link between the Bhattacharyya *coefficient* and the Matusita distance can be seen in terms of the triangle cosine rule: the Bhattacharyya *coefficient* is the cosine of the angle between the vectors, the Matusita distance is the magnitude of the difference between the vectors, the Bhattacharyya *distance* is then a simple fraction of that distance.

6.4.1 Histogram Intersection distance

The Matusita distance can be used to bring out a relationship between the Bhattacharyya distance, the L_1 norm, and the less often used Histogram Intersection distance. Histogram Intersection is defined as [132]:

$$\cap(\mathbf{p},\mathbf{q}) = \frac{\sum\limits_{u=1}^{n} \min(p_u, q_u)}{\sum\limits_{u=1}^{n} q_u} \tag{6.14}$$

The measure calculates the commonality (the intersection) between the two histograms $\mathbf{p}, \mathbf{q}$. The measure is normalized by the number of pixels in the target histogram in order to obtain a fractional match value between 0 and 1. In the case of normalized histograms the denominator in Eq.(6.14) is unity. For a perfect match between normalized histograms $\cap(\mathbf{p},\mathbf{q}) = 1$. To get a dissimilarity measure we take the complement of the intersection to get the Histogram Intersection distance ρ_H:

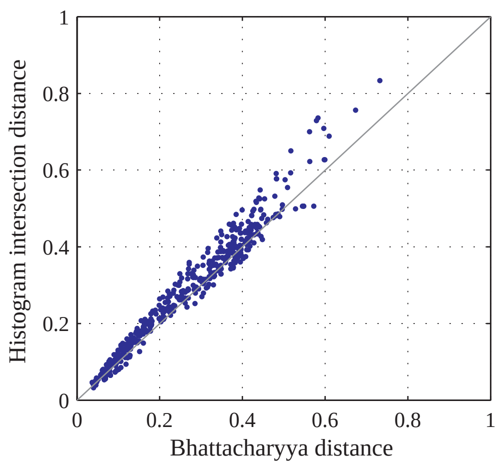
$$\rho_H = 1 - \cap(\mathbf{p},\mathbf{q}) \tag{6.15}$$

It can be shown [132] that for normalized histograms the Intersection distance ρ_H is linked to an L_1 type distance formed using the simple sum of absolute differences of the bin values:

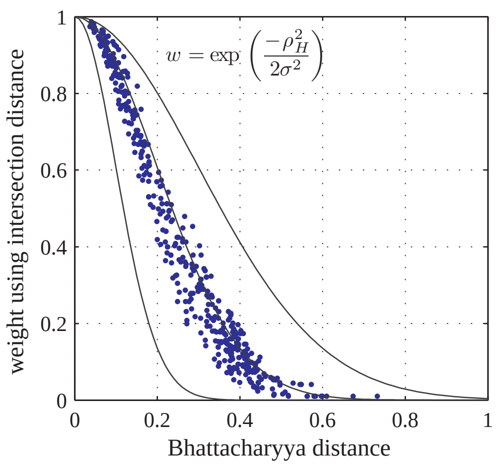
$$1 - \cap(\mathbf{p},\mathbf{q}) = \tfrac{1}{2}\sum_{u=1}^{n} |p_u - q_u| = \tfrac{1}{2}L_1 \tag{6.16}$$

Intersection has been used, for example, in the context of logo identification in video sequences [133], spatial histograms for region based tracking [64], via the L_1 form in 'Bag-of-Features' face matching [134] and in multi-segment histogram matching for particle filter tracking of crossing targets [67].

The Bhattacharyya and Histogram Intersection distances are linked by comparing the square of the Matusita distance and this binwise absolute difference form of



(a) Scatter of Histogram Intersection distance values plotted against the Bhattacharyya distance for a typical set of particles



(b) The effect of using the ρ_H distance with a Gaussian likelihood function, $\sigma = 0.2$.

Figure 6.2: Illustrating the relationship between Histogram Intersection and Bhattacharyya distances

the Intersection:

$$\begin{aligned} M^2 &= \sum_{u=1}^{n} \left(\sqrt{p_u} - \sqrt{q_u}\right)^2 \\ &= \sum_{u=1}^{n} \left(\sqrt{p_u} - \sqrt{q_u}\right)\left(\sqrt{p_u} - \sqrt{q_u}\right) \qquad (6.17) \\ L_1 &= \sum_{u=1}^{n} |p_u - q_u| = \sum_{u=1}^{n} |\sqrt{p_u}^2 - \sqrt{q_u}^2| \\ &= \sum_{u=1}^{n} |\left(\sqrt{p_u} - \sqrt{q_u}\right)|\left(\sqrt{p_u} + \sqrt{q_u}\right) \qquad (6.18) \end{aligned}$$

We compare equations Eq.(6.17) and Eq.(6.18). For a given set of histogram bin differences as indicated by a single Bhattacharyya distance associated with $\left(\sqrt{p} - \sqrt{q}\right)$, there will be a range of L_1 values determined by alternative combinations of component bin values ($0 \leqslant p \leqslant 1, 0 \leqslant q \leqslant 1$), that give rise to a set of given differences. The alternative combinations supply the factor $\left(\sqrt{p} + \sqrt{q}\right)$ in Eq.(6.18).

Figure 6.2(a) shows a comparison of the calculated Histogram Intersection distance measure (ρ_H) against the Bhattacharyya distance (ρ_B) for a set of 512 particles from a typical particle filter step. It can be seen that the scatter of the ρ_H measures about the line $\rho_H = \rho_B$ is reasonably compact.

It is not difficult to understand the nature of the scatter. Figure 6.3 shows the relationship between the Histogram Intersection type component plotted against

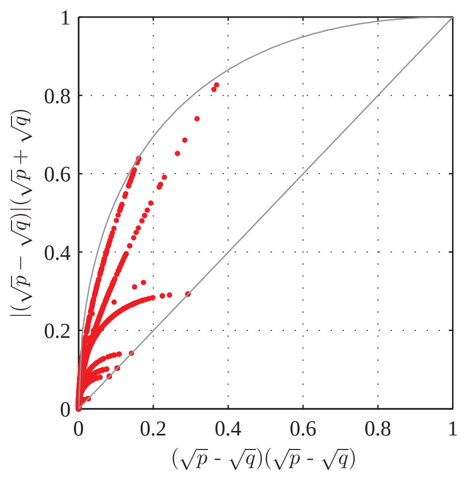


Figure 6.3: Plot of Histogram Intersection distance components against the corresponding Bhattacharyya distance components for 512 x 8 $\{p, q\}$ bin pairs.

the corresponding Bhattacharyya type component as described by equations (6.19) and (6.20) for the 512 x 8 $\{p, q\}$ pairs, where $\mathbf{p}$ represents normalized candidate 8-bin histograms and $\mathbf{q}$ represents the single normalized target 8-bin histogram, i.e.

$$2\rho_B^2 = \rho_M^2 = \sum_{u=1}^{8} (\sqrt{p_u} - \sqrt{q_u})(\sqrt{p_u} - \sqrt{q_u}) \tag{6.19}$$

$$\begin{aligned} 1 - \mathbf{p} \cap \mathbf{q} &= \frac{1}{2} \sum_{u=1}^{8} |p_u - q_u| \\ &= \frac{1}{2} \sum_{u=1}^{8} |(\sqrt{p_u} - \sqrt{q_u})|(\sqrt{p_u} + \sqrt{q_u}) \end{aligned} \tag{6.20}$$

The 45° straight line in Figure 6.3 corresponds to points in the plot where either $p = 0$ or $q = 0$. The upper bounding curve corresponds to points where either $p = 1$ or $q = 1$. In the case where either of the bin values is unity the functional relationship for the curve is given by $y = 2\sqrt{x} - x$. Points between the upper and lower boundaries correspond to $0 < p < 1$, $0 < q < 1$. The lower left hand corner of the graph corresponds to points where the bin values $\{p, q\}$ are similar. The upper right hand corner of the graph corresponds to points where the bin values $\{p, q\}$ are very different. There are eight curves in Figure 6.3 corresponding to the eight bin values q associated with the target histogram. The points along each curve represent the 512 p values linked to each q.

The Bhattacharyya distance calculation involves summing eight abscissa values from Figure 6.3. The Histogram Intersection distance involves summing from the figure eight ordinate values corresponding to those abscissa values. However, for each abscissa there can be a number of ordinate values where the vertical line associated with that abscissa intersects with the curves. This means that each Bhattacharyya distance can be associated with a number of Histogram Intersection distances hence we can expect to see a scatter if one distance is plotted against the other.

The relatively compact form of the scatter in Figure 6.2(a) can be understood by recognizing that its extent is limited by bounding 45° line and the upper curve in Figure 6.3. Also, each set of p and q values must sum to unity so most of the values tend to be concentrated in the lower left hand corner of that figure. These limitations mean that the resulting scatter of the points in Figure 6.2(a) is not large. If the sums of eight ordinate values and eight abscissa values taken from Figure 6.3 are plotted against each other in Figure 6.2(a) then the point will be placed above the 45° line. The multiplication by 1/2 in the Histogram Intersection distance displaces the sum of points vertically downwards in that figure, and the square root in the Bhattacharyya distance calculation shifts the sums parallel to the x-axis towards the right. The overall result, as seen in Figure 6.2(a), is the observed scattered distribution of the points around the 45° diagonal.

Figure 6.2(b) shows the effect of using the ρ_H measure with a Gaussian likelihood function with $\sigma = 0.2$. The Bhattacharyya distance is plotted on the horizontal axis and the weight associated with the corresponding ρ_H distance is plotted on the vertical axis.

It can be seen that the degree of scatter shown in Figure 6.2(a) translates to a scatter about the chosen Gaussian and is within the limits associated with $\sigma = 0.1$ and $\sigma = 0.3$.

This analysis suggests that if we take into account statistical averaging associated with using a weighted mean, or pdf mode finding based upon weighted particle densities, the use of the computationally simpler Histogram Intersection distance might produce outcomes that are not much different from those obtained using the Bhattacharyya distance.

6.4.2 The χ^2 distance

The χ^2 measure has also been used as a distance measure to compare histograms [135, 136]. The χ^2 probability density function describes the probability of occurrence of a given value of the sum:

$$Q = \sum_{u=1}^{n} {x_u}^2 \tag{6.21}$$

for n independent values of $x \sim \mathcal{N}(0, 1)$. It is used in testing the goodness-of-fit of experimental data to expectation. That kind of testing is based upon the recognition that for frequency binned data the distribution of

$$\sum_{u=1}^{n} \frac{(O_u - E_u)^2}{E_u} \tag{6.22}$$

where O_u and E_u represent observed and expected frequencies respectively is approximated by a χ^2 distribution.

For matched pairs of observed and expected histogram bin values p_u, q_u it can be shown [137] that the distribution of

$$X^2 = \sum_{u=1}^{n} \frac{(p_u - q_u)^2}{p_u + q_u} \tag{6.23}$$

is also approximated by a χ^2 distribution. This quantity X^2 is generally labelled as χ^2, and for consistency with the literature it will be useful to adopt that labeling.

Whilst the χ^2 statistic is used for significance/goodness-of-fit testing, its appropriateness as a distance measure is debatable. Aherne et al. [34] point out the non-linear nature of χ^2 type distances. They replace p and q in Eq.(6.23) by $\sqrt{p}$ and $\sqrt{q}$ respectively and apply a first-order Taylor approximation to the denominator to show that for small distances the χ^2 distance is twice their version of the Matusita distance, which is the actually the square of Matusita's original distance.

We can analyze the relationship between the χ^2 measure and the Bhattacharyya distance using an approach similar to that for the Histogram Intersection distance. The numerator in Eq.(6.23) expands in a way similar to that in Eq.(6.18) except that the factors are squared:

$$\chi^2(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} \frac{\left(\sqrt{p_i} - \sqrt{q_i}\right)^2 \left(\sqrt{p_i} + \sqrt{q_i}\right)^2}{p_i + q_i} \tag{6.24}$$

The first factor in the numerator is similar to the Matusita component of the sum. By combining values of $0 \leqslant p \leqslant 1$ and $0 \leqslant q \leqslant 1$ it can be shown that the remaining factor $\left(\sqrt{p} + \sqrt{q}\right)^2 / (p + q)$ varies between 1 and 2 with most of the values at the upper end of that range. Expansion of the numerator of the remaining factor gives:

$$\begin{aligned} \frac{\left(\sqrt{p} + \sqrt{q}\right)^2}{p + q} &= \frac{p + q + 2\sqrt{p}\sqrt{q}}{p + q} \\ &= 1 + \frac{2\sqrt{p}\sqrt{q}}{p + q} \end{aligned}$$

In a normalized histogram the values of p or q are limited to the range [0,1]. If

$p = q = 1$ or if $p \approx q \approx 0$ then the component $\frac{2\sqrt{p}\sqrt{q}}{p+q} = 1$ so that

$$\frac{\left(\sqrt{p}+\sqrt{q}\right)^2}{p+q} = 2$$

If one of p or $q = 0$ then $\frac{2\sqrt{p}\sqrt{q}}{p+q} = 0$ so that

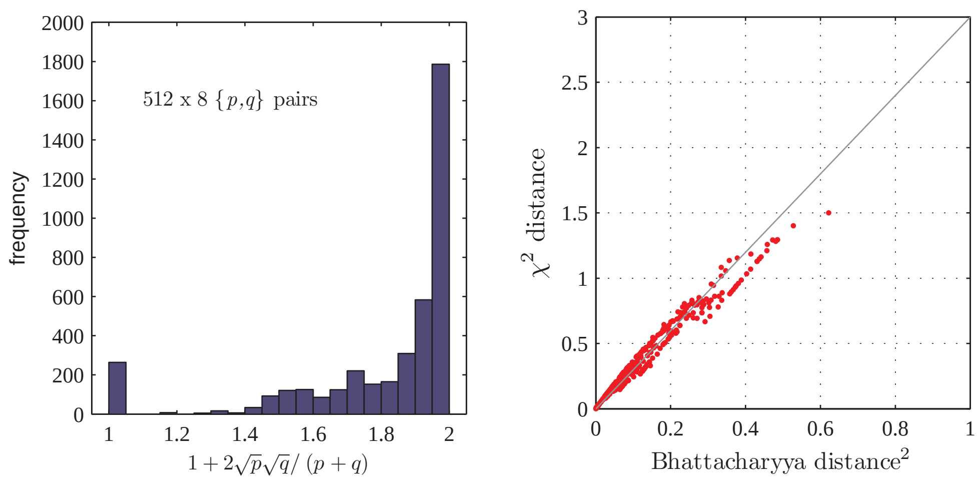
$$\frac{\left(\sqrt{p}+\sqrt{q}\right)^2}{p+q} = 1$$

Other combinations of p and q return values of the factor between 1 and 2. A typical frequency distribution of the factor for 512×8 $\{p, q\}$ pairs from a typical step of a particle filter track is shown in Figure 6.4(a). It can be seen that the factor varies between 1 and 2 with a tendency for most of the values to be at the upper end of the range. The relatively large number of entries in the lowest bin reflects the frequency of occurrence of one of p or $q = 0$. For a given set of $\{p, q\}$ pairs in a histogram Matusita/Bhattacharyya distance calculation a corresponding χ^2 distance is going to have different value due to the bin by bin contributions of the multiplicative factors. A given p, q difference can correspond to a range of p, q sums. We can expect to see a scatter of χ^2 distances for a given set of Matusita/Bhattacharyya distances. A typical scatter between the χ^2 distance and the square of the Bhattacharyya distance, using real particle filter data, is shown in Figure 6.4(b). The graph suggests that the relationship between between the χ^2 and the ρ_B^2 values can be approximated by $\chi^2 = 3\rho_B^2$. This relationship is seen in the experimental data in Figure 6.4(b).

If χ^2 is substituted for ρ_B^2 in the Gaussian likelihood then a standard deviation of $\sigma = 0.35 \approx \sqrt{3} \times 0.2$ should produce a distribution similar to one with a Bhattacharyya distance and standard deviation of $\sigma = 0.2$. A plot of such a χ^2 likelihood weighting is shown in Figure 6.5. The Bhattacharyya distances are plotted as abscissae and the weights produced using χ^2 with $\sigma = 0.35$ are plotted as ordinates. The scatter is less than in the Histogram Intersection distance case but the computational demand is a little greater.

6.4.3 Choosing an appropriate histogram distance

We can conclude from this analysis that for the purposes of estimation there is little to gain in choosing between the Bhattacharyya, Matusita and χ^2 distances. The first two differ only by a simple numerical factor, the first and the third differ approximately by such a factor. Their differences in the context of particle filter calculations can be compensated for by adjusting the value of the standard deviation



(a) Frequency distribution of the factor $1 + 2\sqrt{p}\sqrt{q}/(p+q)$ for $512 \times 8\{p,q\}$ pairs from a step in a typical particle filter track

(b) Plot of χ^2 vs the square of the Bhattacharyya distance for 512 typical particles, indicating the scatter associated with the multiplicative factor. Note: the Bhattacharyya distances range from 0 to ~ 0.8 hence the maximum of the squared distances is ~ 0.6.

Figure 6.4: Looking at the relationship between the χ^2 and the Bhattacharyya distances

in the likelihood function. The Histogram Intersection distance, however, offers potential computational economy through avoidance of the need to extract square roots. It does produce a scatter of likelihood weight values but they are generally within a range corresponding to the use of Bhattacharyya distances and a Gaussian likelihood with standard deviations between the useful limits of $\sigma = 0.1$ to $\sigma = 0.3$. As this standard deviation range would produce an acceptable re-weighting, dependent upon the prior spread of the distance values, it is reasonable to expect that given sufficient particles distributed reasonably symmetrically around the target location a position estimate would be returned not far from one calculated using the Bhattacharyya distance with a standard deviation of $\sigma = 0.2$. The behaviour of the Histogram Intersection distance suggests that it might produce particle filter tracking that is not much different from that produced with the Bhattacharyya distance.

6.5 Likelihood functions other than Gaussian

The basic role of the likelihood function is to give preferential weight to proposed states that are close to the target state and to suppress proposals that are far from the target state. The choice of a Gaussian likelihood function is due to the common assumption of normally distributed measurement error. It is of interest to note that

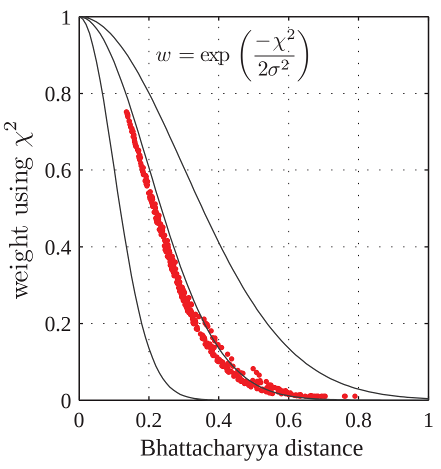


Figure 6.5: Scattered distribution of χ^2 distance weightings with a Gaussian likelihood, using $\sigma = 0.35$

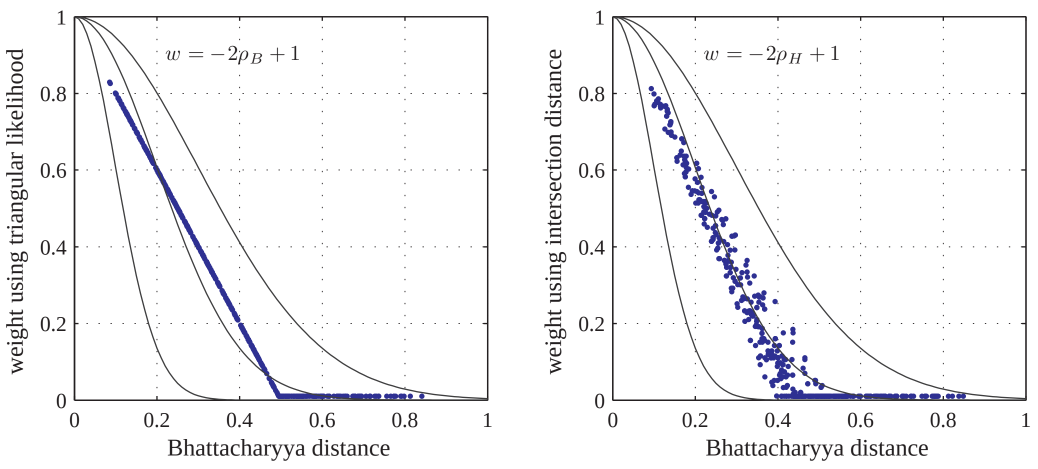
the appropriateness of the underlying Gaussian error model has been questioned by some authors [138, 139, 140] who point out that a heavier tailed Cauchy distribution might be more appropriate in some situations. An example of the use of a Cauchy distribution in the context of particle filter tracking is given in [141].

Some particle filter applications approximate the behaviour of such a distribution by adding a constant tail to the likelihood function [90]. The role of the 'tail' is to maintain the distribution through re-sampling in the event that the target object is not observed in an image. In such cases the distance measure would be large and the corresponding weights would be near zero. Allowing a small but non-zero weight for large distances ensures survival of the particle set at the re-sampling stage.

Whilst it might indeed be the case that measurement error is best described by either a Gaussian or a Cauchy distribution, it is proposed that there is nothing particularly special about the form of the particle filter likelihood function other than it must maintain the positive evolution of the particle set by preferentially selecting low distance vectors and suppressing vectors with a large distance.

In practice a Gaussian weighting is likely to be implemented via a look-up table. In order to illustrate the non-critical form of the likelihood function it is proposed that a simple triangular function of the form $w = -2\rho_B + 1$ might be used as a selection device. An example of such a triangular likelihood with a constant tail is shown in Figure 6.6(a). In this case the function drops to zero at $\rho_B = 0.5$ and the tail returns the weight $w = 0.01$ for $\rho_B \geqslant 0.5$.

When the Histogram Intersection distance is used in place of the Bhattacharyya distance the associated scatter re-distributes the weights in a similar way to that seen in Figure 6.2(b). The effect is shown in Figure 6.6(b). The combination of Histogram Intersection distance and the simple triangular likelihood function produces



(a) Weighting a Bhattacharyya distance function using a triangular likelihood given by $w = -2\rho_B + 1$ for $\rho_B < 0.5$, $w = 0.01$ for $\rho_B \geqslant 0.5$

(b) Weighting a Histogram Intersection distance function using a triangular likelihood given by $w = -2\rho_H + 1$ for $\rho_H < 0.5$, $w = 0.01$ for $\rho_H \geqslant 0.5$. Bhattacharyya distance is plotted on the horizontal axis but the weight is calculated using the ρ_H value corresponding to that Bhattacharyya distance.

Figure 6.6: Using a simple triangular likelihood function

a scatter of weightings that is not much different to that of the Gaussian/intersection combination.

6.6 Investigating the effects of using the simplified distance/likelihood combination

The proposal that the analytical form of the likelihood function is not critical in the context of grey scale histogram based particle filter tracking was explored by comparing tracks produced using the standard Bhattacharyya distance/Gaussian likelihood combination against ones produced using the simple Histogram Intersection distance/triangular likelihood combination. The Histogram Intersection distance was chosen rather than χ^2 because, whilst the scatter of distance measures appeared to have similar consequences for each of the approaches, the Intersection distance was computationally more attractive.

Targets were tracked using an elementary SIR particle filter [69], as described in Chapter 5. Static background elements were suppressed by the use of a temporal frame difference mask. Four image sequences were used, each of which presented its own challenges to the tracker.

The first image sequence consisted of an overhead view of shoppers walking through a mall. The appearance of the shoppers changes significantly as they move from the bottom of the scene to the top due to a mixture of object deformability, viewing angle and varying lighting conditions.

The second sequence is a daytime outdoor oblique view of pedestrians. The pedestrians often occlude each other. In the example shown the tracked pedestrian passes partly behind two foreground pedestrians having a different grey scale signature and then completely behind one with a similar signature.

The third was an outdoor view of traffic taken from the AVSS 2007 i-LIDS Vehicle Challenge data set [10]. In this sequence there is some slight camera movement and illumination changes.

The fourth sequence is taken from the PETS 2006 data set [12]. It is an indoor scene and the experiment tracks a pedestrian passing close to others of similar appearance.

In each of the sequences a selected object was tracked 100 times using the Gaussian likelihood/Bhattacharyya distance combination and then 100 times using the alternative combination. The frame rates were set to 5fps. The mean and variance of the 100 measurements of each track point position was determined for each combination. The mean track from the first likelihood/distance combination was then compared with the second. Investigations were carried out using smaller and larger statistical sample sizes (between 5 and 1000 track sequences) but the outcomes were only significantly different, as expected, for small sample sizes (e.g. < 10 measured tracks). The behaviour of most of the targets in each sequence was investigated and the outcomes for the ones reported here were typical of all.

The target starting position was manually initialized in order to ensure comparability of the resulting tracks. A single rectangular region with dimensions half those of the bounding box of the target object was tracked using a normalized 8-bin grey-scale histogram. For example, the tracking rectangle seen in Figure 6.7(a) had dimensions 17×17 pixels. The histograms were extracted using the integral histogram method [116]. A first order state transition step was used with the state vector $\mathbf{x} = [x, v_x, y, v_y]^T$. The number of particles used was $n_s = 512$.

In the cases where the view was at an oblique angle, the size of the rectangle was adjusted using a linear function of the target position in the image in order to retain its relative size in proportion to the size of the target.

The standard deviation of the additive Gaussian process noise ($\boldsymbol{\nu}_t$ in Eq.(3.5)) was set to be $0.2 \times$ the length of the side of the tracking rectangle for both the spatial and velocity components of the state vector. This choice produced a prior particle spread appropriate to the typical velocities of the targets in the sequence.

The template histogram $\mathbf{q}$ was updated at each frame using $\mathbf{q}_t = (1-\alpha)\mathbf{q}_{t-1} + \alpha\bar{\mathbf{p}}$ where $\bar{\mathbf{p}}$ was the histogram extracted at the current target position. The value of α

was set at 0.2 for all sequences.

The measurement noise, i.e. the standard deviation of the likelihood Gaussian function, was kept constant at $\sigma = 0.2$. The likelihood distributions were given a 3σ 'tail' value of 0.01 for all likelihood values that were initially below that value. The posterior particle set was updated at each step using standard stratified re-sampling.

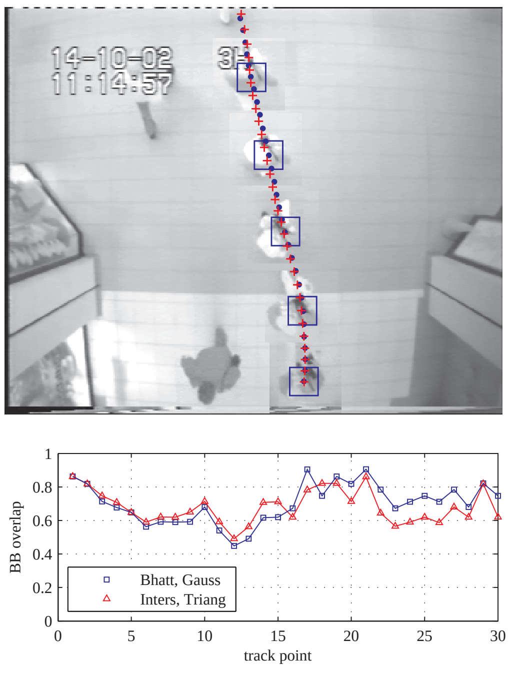
Track smoothing was carried out by adjusting the position components of $\mathbf{x}_{t-1}$ and $\mathbf{x}_{t-2}$ at each time step t by a recursive procedure that consisted of deriving and averaging velocities from measured positions at t-3 to t to produce an adjusted $\hat{\mathbf{x}}_{t-2}$ and $\hat{\mathbf{x}}_{t-1}$. The resulting comparison of the tracking approaches can be seen in Figure 6.7. Both the Bhattacharyya/Gaussian and Intersection distance/triangular tracks are shown for each sequence. Snapshots of the targets together with occluding objects at periodic track points have been overlaid so that the tracks can be seen in context.

The correspondence of the tracks with the geometrical ground truth of the tracked objects was investigated. For each sequence the ground truth was determined manually in each frame as the centroid of the bounding box of the objects. The tracks for each of the two approaches were compared to the ground truth using the bounding box spatial overlap [142], defined as the overlapping area $A(GT_t, ST_t)$ between the ground truth bounding box GT and the tracking system bounding box ST in frame t:

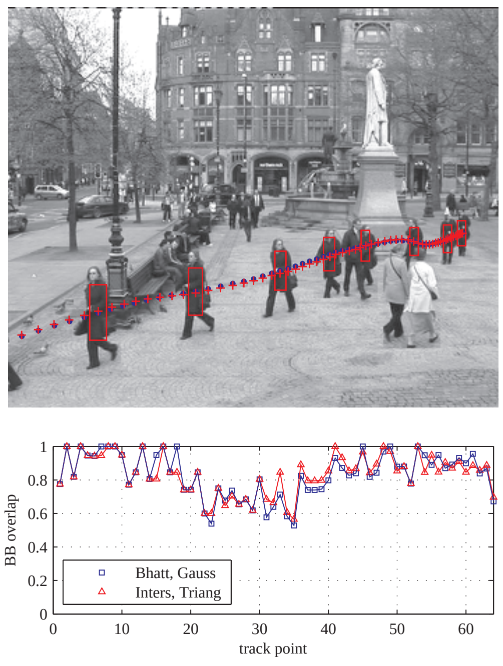
$$A(GT_t, ST_t) = \frac{Area(GT_t \cap ST_t)}{Area(GT_t \cup ST_t)} \tag{6.25}$$

This measure returns a value of 1 when the track corresponds exactly with the ground truth and 0 when bounding boxes do not overlap at all. The overlaps for each approach are plotted beneath the relevant track images in Figure 6.7. It can be seen that in general the overlap is between 0.6 and 1.0 giving sufficient correspondence for tracking in surveillance contexts. It can also be seen that the track points associated with each of the two approaches are closer to each other than to the geometrical ground truth. It is to be expected that the mean position of the tracked grey scale feature would not necessarily correspond to the geometrical centroid of the tracked object due to a combination of object deformation, perspective and shading effects. The closeness of the overlap for the two tracking approaches indicates that there is no significant difference between them in terms of the tracks returned.

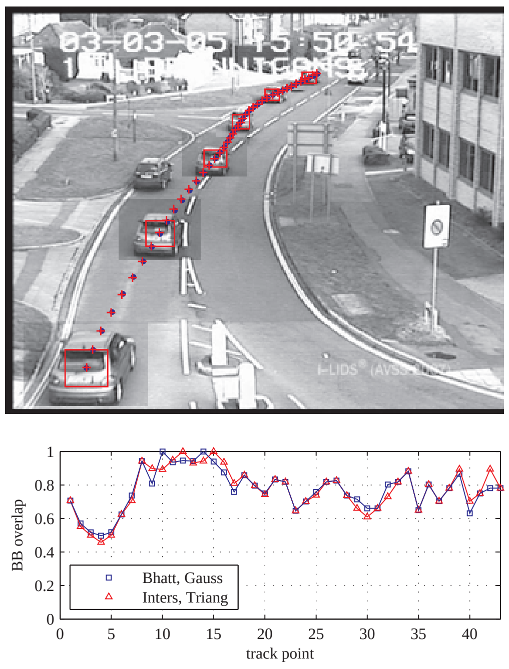
An investigation into the processor timings associated with the use of the Histogram Intersection distance/triangular likelihood, compared to the Bhattacharyya distance/Gaussian likelihood, was carried out using a Chipwrights CW5631 [8] simulator running at 300MHz (Source: Wang, W. AD-Group, personal communication). The results are shown in Table 6.1. It can be seen that for the floating-point calculations, the Intersection-based calculation ran twice as fast as the Bhattacharyya-



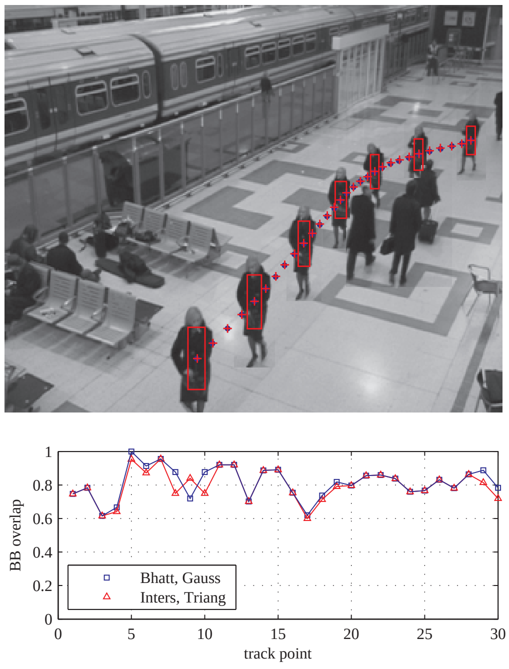
(a) Overhead



(b) Square



(c) i-LIDS



(d) PETS 2006

Figure 6.7: Composite images and ground truth to tracker bounding box overlap data for the sequences. Blue (·) points - Gaussian likelihood\Bhattacharyya distance, red (+) points - triangular likelihood\Histogram Intersection distance

	Histogram size		
	16 bins	128 bins	256 bins
Bhattacharyya/Gaussian(float)	0.0747	0.3447	0.6531
Intersection/triangular(float)	0.0197	0.1531	0.3049
Intersection/triangular(fixed)	0.0075	0.0575	0.1147

Table 6.1: Timings, in ms, for Bhattacharyya/Gaussian and Intersection/triangular likelihood calculations, using a Chipwrights CW5631 simulator running at 300MHz (Source: Wang, W., AD-Group, personal communication).

based calculation for the larger (128, 256 bin) histograms, and three times faster for the smaller (16 bin) histograms. But the attractiveness of the Intersection/triangular likelihood combination lies in its greater suitability for fixed-point implementation. The table shows that the use of fixed-point calculations for the Intersection-based combination produced a further factor of three speed increase. In addition to the basic calculation speed gain, fixed-point implementation opens up the advantages of the vector processing capabilities of the chip, producing further computational convenience and economy, especially in the case of large histograms.

6.7 Discussion

The review of distance measures and their inter-relationships led to the conclusion that the Bhattacharyya distance and the related Matusita distance had a clear interpretation in terms of the magnitude of the difference between histogram based vectors in a Euclidean feature space. In comparison to these measures others, like the Histogram Intersection distance and χ^2, had a less clear fundamental interpretation in this context and produced a scattering of distances in comparison to the Bhattacharyya distances. But it was suggested that the absence of the fundamental interpretation did not exclude the measures from consideration and that any scatter in comparison to the Bhattacharyya distance could simply contribute to the stochastic element of the filter.

The understanding of the effects of choosing a particular Gaussian standard deviation when using a normalized distance measure led to the proposal that the use of the potentially computationally advantageous Histogram Intersection distance, combined with a triangular likelihood function, might produce results comparable to those obtained using a Bhattacharyya measure/Gaussian likelihood combination. The application of the proposal was supported by the experimental findings. In general the mean track points for the two approaches stayed well within the bound-

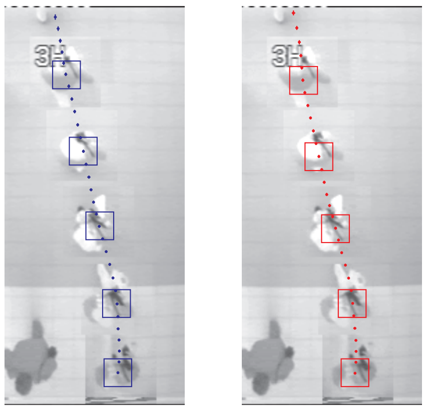


Figure 6.8: Illustrating the track differences for Gaussian/Bhattacharyya (left) and triangular/Histogram Intersection (right)

aries of the objects being tracked. It is true that they deviated from the geometrical centroids of the objects but the results were still within acceptable limits for the purpose. A comparison of the worst case differences, in the Overhead sequence tracks, is shown in Figure 6.8.

The broad range of likelihood standard deviations, discussed in Section 6.2, can be interpreted as indicating choices of process noise parameters that delivered ranges of similarity distance appropriate for particle re-weighting and re-sampling in those cases. It is suggested, however, that a Gaussian likelihood particle filter being operated with normalized distance measures and values of $\sigma < 0.1$ is likely to have a process stage consistently returning a high proportion of distances in the lower end of the [0,1] range in order to maintain a reasonable experimental value of n_{eff}. It is likely that this will not spread the prior particles sufficiently to accommodate extreme behaviour of the target. A particle filter being operated with normalized distances and values of $\sigma > 0.3$ might return distance measures across the full range, with $> 50\%$ of the particles contributing to n_{eff}, but will allow undue influence from histogram feature vectors that are very different from the target vector. It is likely that it would end up spreading the particles too widely.

The limitation of the range for the choice of the Gaussian standard deviation, and its link to the effective particle number, led to the recognition that the forms of the likelihood function and distance measure are not critical in the tracking situations examined. This opened the door to the use of the simpler forms capable of giving potential computational economy in systems where resources might be constrained.

Chapter 7

State estimation and particle re-sampling

7.1 Introduction

The particle filter provides a recursive update of a discrete weighted approximation to the true posterior state probability density. It was shown, in Section 3.4, that the weights could be derived using the principles of Importance Sampling. In the Importance Sampling treatment the relationship between the weights, the posterior density, and a sampling distribution, was derived in terms of the expectations of the distributions. But it does not necessarily mean that parameters of interest to be extracted from the approximation to the state have to be expectations. For tracking purposes, the parameter of interest is more likely to be one of the modes of the distribution. Even though that is the case it is still common to see, in the literature, the tracker state derived using the expectation. This implicitly assumes the existence of a strong Gaussian-like mode in the distribution. In other cases the state might be extracted by taking the sample with the maximum weight, or using some other mechanism to determine the MAP state. Those who opt for the maximum weight do so with the assumption that the particle density will be sufficient to guarantee that a particle will be positioned close to the true peak of underlying likelihood distribution. They also assume that the mode of interest is the dominant one, which is not always the case. There appears to be no clear consensus on the best way, if there is one, to extract the state estimate from the particle representation.

Similarly, there are conflicting views about the most appropriate re-sampling procedure to adopt. Some, e.g. [59, 113] follow Gordon et al. [75] and use the multinomial method. Many adopt the systematic approach described by Arulampalam et al. [69], whilst some early Sequential Monte-Carlo studies advocated procedures like deterministic [143], stratified [143], or residual [144] re-sampling. The re-sampling approaches are generally justified on grounds of statistical validity, leaving computational convenience do be dealt with as a later refinement. However, one approach [145] focuses closely on the computational aspects and is the one adopted in this work. The approach is analyzed in detail in order to justify the adoption.

This chapter looks critically at the state estimation and particle re-sampling steps. It does so in the same spirit as the last chapter: it questions the assumptions underlying the steps and looks to see what simplifications can be incorporated. It starts by considering some approaches to state estimation. It considers a recent claim to a provably correct approach to MAP estimation for particle filters, suggests that it might not be adequate in the context of multi-target tracking, and offers an alternative practical solution.

7.2 State estimation

7.2.1 Using post-measurement processing

Single target track determination calls for the extraction of a single point estimate from the particle approximation to the pdf. In Section 3.4 it was stated that the required state is commonly estimated as the expectation of the posterior density, i.e. Eq.(3.59)

$$\mathrm{E}\left[\mathbf{x}_t|\mathbf{z}_{1:t}\right] \approx \sum_{i=1}^{n_s} \tilde{\omega}_t^i \mathbf{x}_t^i$$

or as the MAP approximated by the maximum weighted state:

$$\mathbf{x}_t^{MAP} = \arg\max_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \arg\max_{\mathbf{x}_t} w_t^i$$

The use of the weighted mean as a state point estimator is inappropriate when the posterior distribution is multi-modal. The aim is to seek out the correct mode of the distribution.

One approach is to use the standard expectation, as in Eq.(3.59), but aim to ensure that the dominant mode is better represented by particles and reduce the contributions of the sub-dominant modes. In order to reduce the requirement of a large number of particles, some have adopted post-measurement processing strategies to redistribute the particles with a bias towards the suspected mode. The Auxiliary Particle Filter [76] did this by the addition of a second re-sampling step. Maggio and Cavallaro [122] introduced an intermediate mean-shift step in which the particles are moved closer to their local mode. Using a target histogram representation to determine the particle likelihood they mean-shifted the particles along the gradient of the histogram similarity surface to concentrate them at the mode. They claimed that they could track with as few as 30 particles using this approach. But it cannot guarantee that the local mode is the correct one, and the reduction in particle representation is offset by the requirement of a number of mean-shift iterations per particle. Naeem et al. [146] took the approach a step further by mean-shifting the particles on an annealed likelihood distribution. This reduced the local mode problem but introduced further processing per particle. It assumes that the dominant mode is the correct one. But it can suffer from the drawback that, in practice, there are occasions when the target mode is a sub-dominant one.

7.2.2 Approximate particle based MAP estimator

Driessen and Boers [16] argue that not only is the expectation based estimator inadequate but that the traditional MAP maximum weight estimator also gives an incorrect result. This is because in the particle representation it cannot be guaranteed that a particle ends up at the dominant mode of the underlying pdf. The authors point out that the traditional particle filter MAP estimation maximizes the likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$ rather than the posterior $p(\mathbf{x}_t|\mathbf{z}_{1:t})$. They suggest that their approach maximizes the posterior and gives the first provably correct algorithm for marginal MAP estimation that is applicable to particle filters.

Starting from the basic Bayesian recursion equation:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \tag{7.1}$$

they use the Monte Carlo equivalent of the Chapman-Kolmogorov equation:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \\ &\approx \sum_j p(\mathbf{x}_t|\mathbf{x}^j_{t-1})w^j_{t-1} \end{aligned} \tag{7.2}$$

so that the proposed MAP estimator becomes:

$$\mathbf{x}_t^{MAP} = \arg\max_{\mathbf{x}_t^i} p(\mathbf{z}_t|\mathbf{x}_t^i) \sum_j p(\mathbf{x}_t^i|\mathbf{x}^j_{t-1})w^j_{t-1} \tag{7.3}$$

They suggest [147] that this estimate converges to the true MAP in the limit for infinitely many particles. They illustrate the effectiveness of their theoretical approach using a basic particle filter with simulation examples. In one case [16] the point measurement based example involved two targets starting well separated, with one catching up and passing the other. The results of the simulation suggested that their MAP approach returned better estimations than the alternative weighted mean approach.

7.2.3 Testing the Driessen and Boers MAP estimator

The proposed MAP estimator was put to the test, in the context of this work, using a difficult sequence in which two targets with similar grey scale histogram signatures move parallel and close to each other. This sequence proved challenging using both the weighted mean and maximum weight estimators. In those cases the trackers would repeatedly jump from one target to another.

The initialization frame is shown in Figure 7.1(a). The two central targets were tracked. Figure 7.1(b) shows the result of an area similarity scan using the reference four quadrant histogram from the target on the right of the two in the figure. The main peak in the scan is that associated with the reference target. It can be seen that there is a strong subsidiary peak associated with the adjacent target on the left of the two. There is also evidence of similarity with the outer two targets. Figure 7.1(c) shows the distribution of weights associated with an initial set of particles distributed about the selected target. The grey footprints of the particle sets associated with the two trackers are also shown. It can be seen that set of weights shown in the figure is relatively compactly distributed on the footprint of the selected target.

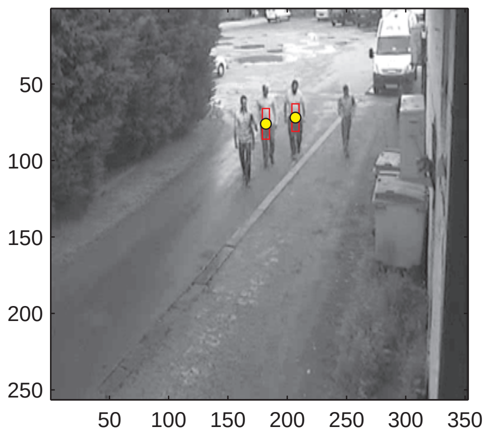
The transition density associated with the sum in Eq.(7.3) was realized using the process noise parameters as indicated in [16, 147], i.e.

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \frac{1}{\sqrt{2\pi Q_{t-1}}} \exp\left(-\frac{1}{2}(\mathbf{x}_t - F\mathbf{x}_{t-1})^{\mathrm{T}} Q_{t-1}^{-1}(\mathbf{x}_t - F\mathbf{x}_{t-1})\right) \tag{7.4}$$

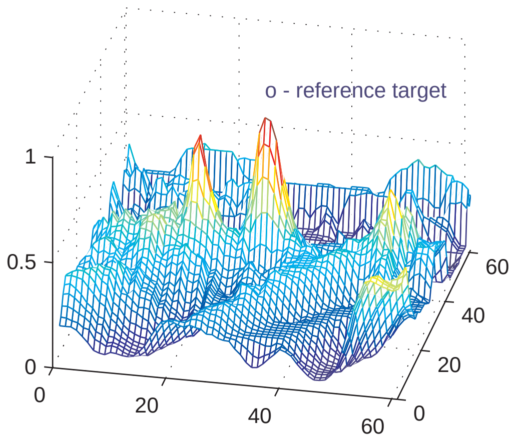
where Q_{t-1} is the covariance of the process noise associated with the step.

Figure 7.1(d) shows the particle weight distribution for $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ at the frame immediately after the initialization frame. Figure 7.1(e) shows the distribution for $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ at frame 18. It can be seen that at that stage the particle set has migrated into the region occupied by the left target and that the maximum value of $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is in that left region. Figure 7.1(f) shows the consequence with the track settling not on the correct target but on the neighbouring one. In some cases the correct track was restored in subsequent frames, in others the tracker settled on the new target.

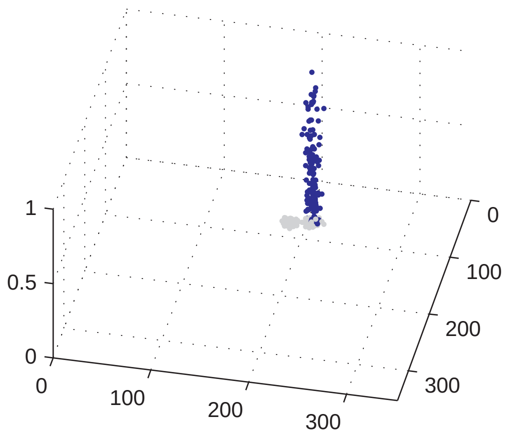
The rate of track loss was investigated by cycling the same sequence for a number of times (100 cycles, each having 40 track points) and signalling an error when the track point for either of the two targets being tracked ended up closer to the ground truth of the other than to its own. The ground truth for the sequence was extracted



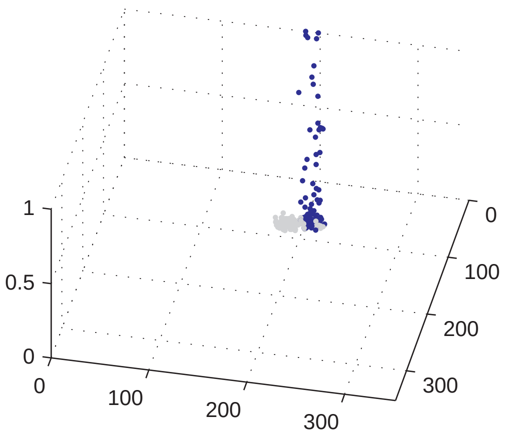
(a) initial frame



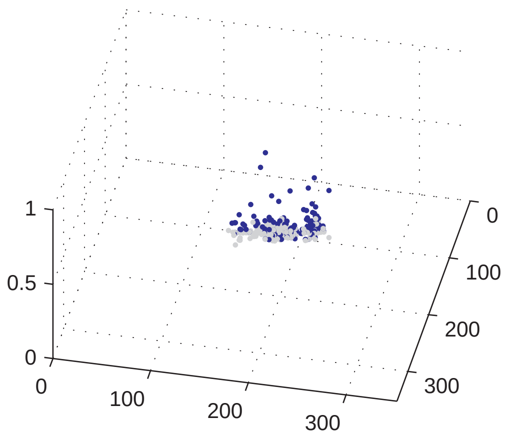
(b) similarity distribution for chosen target



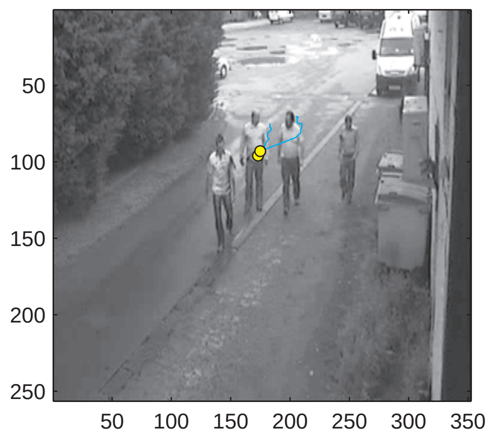
(c) target weights at the first frame



(d) Driessen weights at second frame



(e) Driessen weights in frame 18



(f) loss of track at frame 18

Figure 7.1: Tracking similar targets using the Driessen and Boers MAP

by manually determining the bounding box of each target, and hence its centre, in each frame. Repeated runs of the sequence showed that this track loss occurred for about 12% of the track points in this case.

It was possible to inhibit the tracker exchange by limiting the process noise such that the particles tended to pack closely to the target but that is contrary to the aim of the particle approach: the particles need to have a reasonable spread in order to carry out a realistic search of the area.

Whilst the Driessen and Boers MAP might return the true estimate from the particle distribution in the limit $n_s \to \infty$ it is not adequate in the example shown. Similar misbehaviour was demonstrated in sequences of similar targets with crossing tracks. The approach requires significant extra computation per particle which does not appear to be justified in terms of more reliable tracking. It fails because it has no information about the existence of other trackers in the vicinity and hence cannot take into account the possibility that a strong adjacent mode might belong to another target.

7.2.4 Mode selection in the presence of other trackers

A practical solution to single-target tracker state estimation with a multi-modal likelihood distribution was to recognize the multi-modality, locate the strongest modes and then choose between them. It was assumed, for computational economy, that there would be a dominant mode and only one significant subsidiary mode. It was recognized that in the case of interacting targets with similar representations it was not necessarily the case that the strongest mode was the appropriate one to choose.

The extraction of the dominant mode and the assumed single subsidiary one was carried out by first calculating the weights for all the particles associated with a given tracker. The weights were then sorted in descending order of their strength and a subset of the m top weights was formed. The position with maximum weight was taken to represent the dominant mode. The positions of particles associated with the rest of the weights were compared, one at a time, with that of the maximum weight. If the distance between a following weight particle and that of the maximum was less than a given multiple of the target half width then the particle was added to the set containing the maximum weight particle. If a weight was reached for which the distance to the maximum weight particle was greater than the allowed distance then it was assumed that the weight belonged to the subsidiary mode and a new set was started. The positions associated with the following weights were then compared to those of both of the modes and assigned to their sets accordingly. Once m comparisons had been made the mean positions associated with each set were determined to represent the dominant and the subsidiary mode. The procedure is summarized in Algorithm 4.

The value of m was chosen using the simple rationale that we want to keep computation as small as possible and that we might expect fewer than half of n_s to have high weights. A value of $m = n_s/4$ was tried for $n_s = 256$, but it was found that $m = n_s/8$ produced similar behaviour but with half the computational demand. In a true bimodal situation a value of $n_s/8 = 32$ would deliver 16 particles to each mode. The allowed multiple of the target half-width was chosen to be 3. It was reasoned that two targets placed side-by-side and touching, i.e. with a separation of 2 target half-widths, would not present two distinct modes; target centres with a separation of 3 half widths were more likely to return some kind of bi-modality, so the allowed separation was chosen to be that. It was recognized that *k-means* style clustering might have been a more sophisticated solution, but it would still be likely that it would have to go through similar algorithmic steps, i.e. choose the proposed centres, carry out distance comparisons etc. The pedestrian method described above yielded acceptable results without iterations.

Typical results of the process, using $m = 32$, can be seen in Figure 7.2. This case had been used to illustrate the inadequacy of the weighted mean in Section 5.4.3. The particle set shown in Figure 7.2(a) is the one associated with the target moving towards the lower right of the figure, with weights shown in Figure 7.2(c). The process noise variance was the same as that in Figure 5.10(b). It can be seen in Figure 7.2(a) that some of the particles have been associated with the other crossing target. In this case the likelihood distributions for each of the crossing targets splits into two clear modes. The dominant mode is indicated in green, the subsidiary one is indicated in red.

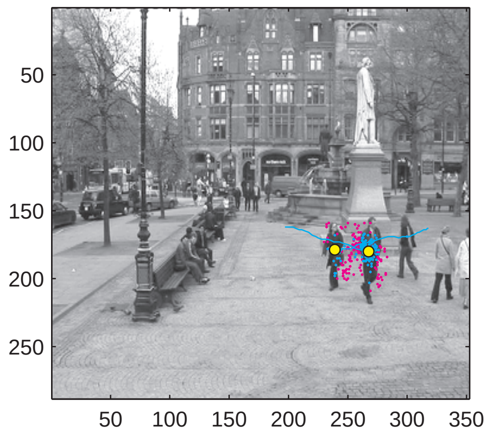
Given that we have assumed that the likelihood distribution splits into a dominant and subsidiary mode it remains to choose which of the two to assign to the tracker. If there are no particles associated with a subsidiary mode then there is the simple solution: the track position is the dominant mode. If there are two modes then it is necessary to make a decision about which mode to assign to the track.

At the end of the preceding tracking step, each track was extrapolated to make a prediction of the next position. The decision process consisted of calculating, for each mode, the probability that it belonged to the predicted position of the tracker, and the probability that it did NOT belong to the predicted positions of the other trackers.

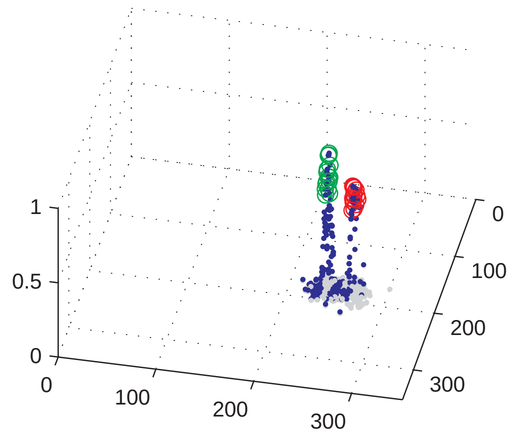
If at time t the position vector of the main mode is represented by $\bar{\mathbf{x}}_A$, and that of the subsidiary mode is represented by $\bar{\mathbf{x}}_B$, then for a track position $\mathbf{x}_{t-1}$, a

Algorithm 4 Extract dominant and subsidiary mode

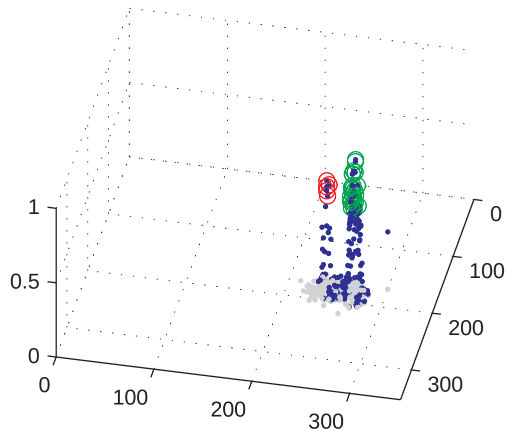
```
**Function def:** $[\bar{\mathbf{x}}_A, \bar{\mathbf{x}}_B] = \text{getmodes}\left(\{\mathbf{x}^i, w^i\}_{i=1}^{n_s}, m, \text{alloweddistance}\right)$

**Stage 1**: call sort function to sort the particles according to the weights

$\left[\{\mathbf{x}^j, w^j\}_{j=1}^{n_s}\right] = \text{quicksort\_descend}\left(\{\mathbf{x}^i, w^i\}_{i=1}^{n_s}\right)$

**Stage 2**: allocate particles to modes

select first representative of first mode:
$\{\mathbf{x}, w\}_A \leftarrow \{\mathbf{x}^1, w^1\}$

allocate the rest of the top $m$ weights:
**for** $j = 1$ to $m$ **do**

  calculate the distance to the first mode
  $d_A \leftarrow \|\mathbf{x}^j - \mathbf{x}_A\|$

  **if** $d_A <$ alloweddistance **then**
    add the particle to the set of Mode A particles
    $\{\mathbf{x}, w\}_A \leftarrow \{\mathbf{x}, w\}_A + (\mathbf{x}^j, w^j)$

  **else if** $\nexists\,\text{mode}\,\text{B}$ **then**
    assign the first representative of the second mode
    $\{\mathbf{x}, w\}_B \leftarrow \{\mathbf{x}^j, w^j\}$

  **else if** $\exists\,\text{mode}\,\text{B}$ **then**
    calculate the distance to the second mode
    $d_B \leftarrow \|\mathbf{x}^j - \mathbf{x}_B\|$

    **if** $d_B <$ alloweddistance **then**
      add the particle to the set of Mode B particles
      $\{\mathbf{x}, w\}_B \leftarrow \{\mathbf{x}, w\}_B + (\mathbf{x}^j, w^j)$
    **end if**
  **end if**
**end for**

**Stage 3**: calculate the weighted mean for each mode

$\bar{\mathbf{x}}_A \leftarrow \text{E}\left[\{w\mathbf{x}\}_A\right]$
$\bar{\mathbf{x}}_B \leftarrow \text{E}\left[\{w\mathbf{x}\}_B\right]$
```



(a) crossing similar targets



(b) tracker 2, moving towards left in the figure, dual modes



(c) tracker 1, moving towards right, dual modes

Figure 7.2: Sort and count tracking for crossing similar targets

predicted position $\hat{\mathbf{x}}_t$, and j other trackers:

$$P(\mathbf{x}_t = \bar{\mathbf{x}}_A | \mathbf{x}_{t-1}) = P(\bar{\mathbf{x}}_A | \hat{\mathbf{x}}_t) \prod_j \left[1 - P(\bar{\mathbf{x}}_A | \hat{\mathbf{x}}_t^j)\right] \tag{7.5}$$

and

$$P(\mathbf{x}_t = \bar{\mathbf{x}}_B | \mathbf{x}_{t-1}) = P(\bar{\mathbf{x}}_B | \hat{\mathbf{x}}_t) \prod_j \left[1 - P(\bar{\mathbf{x}}_B | \hat{\mathbf{x}}_t^j)\right] \tag{7.6}$$

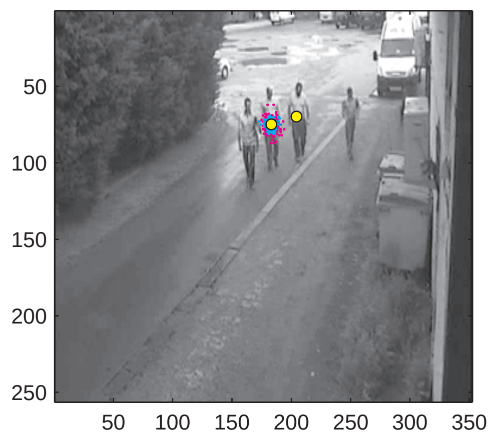
where

$$P(\cdot | \hat{\mathbf{x}}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\rho_E^2}{2\sigma^2}\right) \tag{7.7}$$

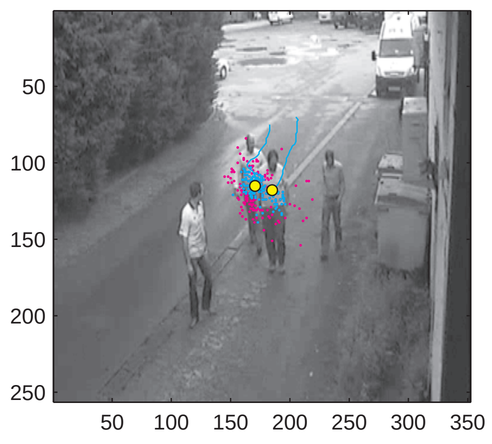
with ρ_E being the Euclidean separation i.e.

$$\rho_E = ||\hat{\mathbf{x}} - (\cdot)|| \tag{7.8}$$

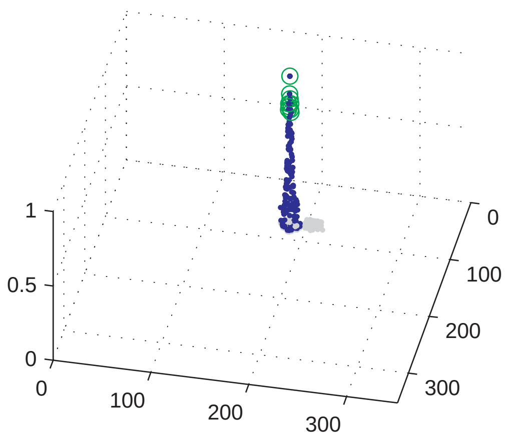
The value of the standard deviation σ was taken to be that of the process noise for



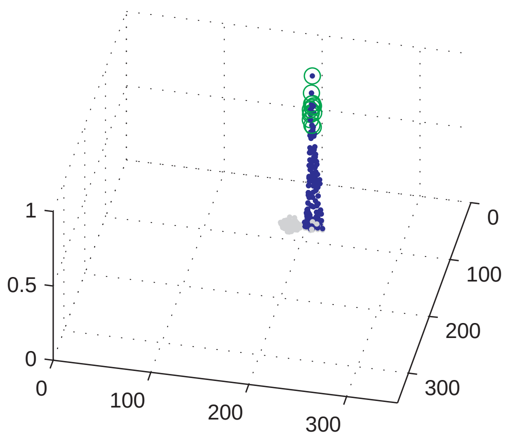
(a) initial frame



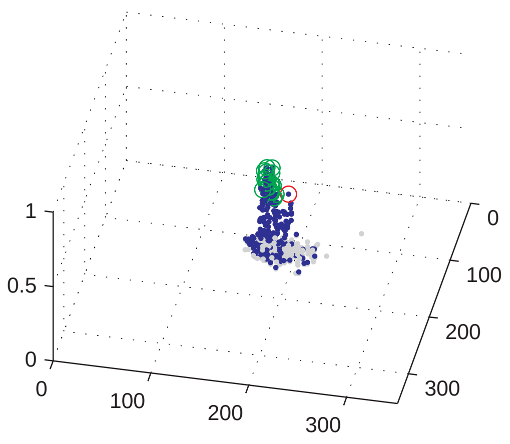
(b) frame 31



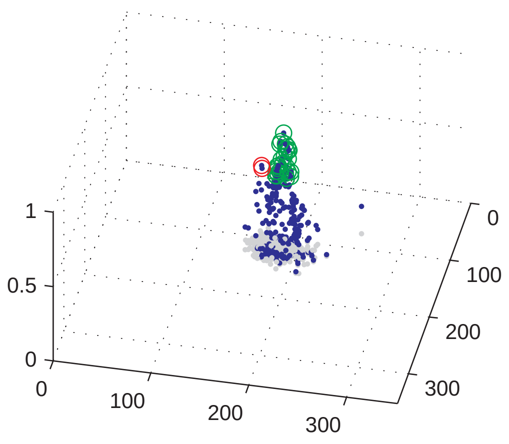
(c) tracker 1, weights at the first frame



(d) tracker 2 weights at the first frame



(e) tracker 1 weights in frame 31



(f) tracker 2 weights in frame 31

Figure 7.3: Tracker mode identification by sort and count

the respective tracker, i.e. one quarter of the target bounding box in pixels.

The position vector of the mode with the largest probability was then assigned to the tracker, i.e.

$$\mathbf{x}_t = \max\left(p(\mathbf{x}_t = \bar{\mathbf{x}}_A|\mathbf{x}_{t-1}), p(\mathbf{x}_t = \bar{\mathbf{x}}_B|\mathbf{x}_{t-1})\right) \tag{7.9}$$

In the case shown in Figure 7.2 the dominant modes were chosen for each tracker and the integrity of their crossing tracks was maintained.

The approach was also applied to the challenging sequence described in Section 7.2.3. The results are shown in Figure 7.3. The rate of track loss was investigated as reported in Section 7.2.3. The behaviour of the Mode Selection approach was significantly better than that of the Driessen approach, returning a ground truth point loss rate of less than 1% under the same conditions. This compares with the ground truth point loss rate of 12% for the Driessen MAP in this selected difficult sequence.

7.3 Re-sampling

Particle set re-sampling was described in Section 3.4.2 as a method of maintaining the number of effective particles by reducing the statistical variance of the weights. The process presents, however, a computational bottleneck because it cannot be started until all the weights have been calculated. It is necessary to compensate for the bottleneck by making sure that the re-sampling method chosen is computationally efficient. Equation (3.61) indicates that if the weight variance is carefully controlled it might be possible to reduce the size of the particle set n_s. But if the number of particles is reduced then the process becomes sensitive to statistical sampling variance with the consequence that it cannot be guaranteed that the reselected particle set fairly represents the underlying pdf. Figure 7.4(a) shows an extreme case with a small sample of ten particles in which the multinomial re-sampling step has missed the dominant weight.

Recent studies have looked at both the statistical and computational aspects of particle filter re-sampling [148, 149]. The common approaches considered were: multinomial, stratified, systematic and residual re-sampling. The first three use the cumulative sum of the weights as in Algorithm 1 in Section 3.4.2 but differ in the way the sampling numbers u are generated. The fourth case, residual re-sampling, has two parts. In the first part a proportion of samples are deterministically selected and the second part uses one of the first three types to complete the process.

In multinomial re-sampling a set of n_s ordered uniformly distributed random

Algorithm 5 Produce ordered random numbers

```
$\left[\left\{u^{(k)}\right\}_{k=1}^{n_s}\right] = \text{orderedrandom}\,(n_s)$

Purpose:
Generation of $n_s$ ordered uniformly distributed random numbers
in the open interval $(0,1)$

Generate a random number:
$\tilde{u} \sim \mathcal{U}(0,1)$

Produce the upper value of the set:
$u^{(n_s)} \leftarrow \tilde{u}^{1/n_s}$

Generate the rest of the ordered set:

for $k = n_s - 1$ to 1 in steps of $-1$ do

  $\tilde{u} \sim \mathcal{U}(0,1)$
  $u^{(k)} = u^{(k+1)}\tilde{u}^{1/k}$

end for
```

numbers $u^{(k)}$ is produced as described in Algorithm 5. The ordering simplifies computation during the search of the cumulative weight distribution. The re-sampled weights are determined as in Algorithm 1 in Section 3.4.2.

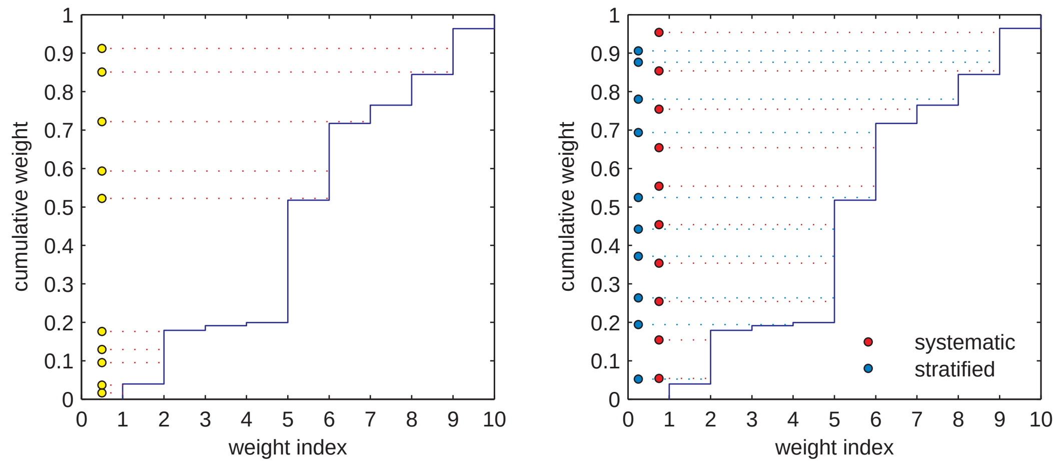
Stratified re-sampling subdivides the range for $u^{(k)}$ into n_s intervals and dithers the values within the intervals.

$$u^{(k)} = \frac{(k-1) + \tilde{u}^{(k)}}{n_s}, \text{ with } \tilde{u}^{(k)} \sim \mathcal{U}[0,1)$$

In systematic re-sampling the range is subdivided as in the stratified case but only the first value is dithered, the rest of the values are distributed uniformly beyond this first value:

$$u^{(k)} = \frac{(k-1) + \tilde{u}}{n_s}, \text{ with } \tilde{u} \sim \mathcal{U}[0,1)$$

In residual re-sampling the particle multiplicity m^i, i.e. the number of copies of the original state $\mathbf{x}^i$, is calculated such that for a total particle set n_s, $m^i = \lfloor \tilde{w}^i n_s \rfloor$ where $\lfloor \cdot \rfloor$ denotes rounding down of the product. The truncation results in a reduced particle set. Experiments carried out by Hol et al. [148] show that this method returns approximately $n_s/2$ deterministically, the remainder has to be made up by implementing one of the other schemes. Once the overall multiplicity is determined



(a) an illustrative example of multinomial re-sampling, in which the dominant weight has been missed

(b) systematic and stratified re-sampling

Figure 7.4: Comparison of re-sampling processes: examples using 10 weights

there is an additional computation step of assigning particle states to their new labels. An illustration of the use of stratified and systematic re-sampling, using a limited set of 10 weights is given in Figure 7.4(b). It can be seen that the multiplicity of the re-sampled weight indices is similar in both cases. Figure 7.5 illustrates that for a large number of particles the heavily skewed original set of weights shown in Figure 7.5(a) produces similar re-distributions, as shown in Figure 7.5(b), with a selection of re-sampling methods.

Re-sampling quality is defined by Hol et al. [148] in terms of the distance between the expectation of the state calculated using the originally weighted set,

$$\mathrm{E}_p[\mathrm{g}] = \sum_{i=1}^{n_s} g(\mathbf{x}^i)\tilde{w}^i, \tag{7.10}$$

and that calculated using the re-sampled states, i.e.:

$$\mathrm{E}_{\hat{p}}[\mathrm{g}] = \frac{1}{n_s}\sum_{i=1}^{n_s} g(\mathbf{x}^{i*}) \tag{7.11}$$

where p represents the distribution of the original sampled states, $\hat{p}$ represents the distribution of the re-sampled states $\mathbf{x}^{i*}$, and $g(\cdot)$ is an arbitrary function.

In general the studies suggest that systematic re-sampling is the preferred option. Hol et al. [148] report that in their implementation systematic returned the lowest computational effort for particle sets with n_s ranging from 1 to 4000. In terms of resampling quality and computational complexity, systematic and residual methods were favourable in comparison to multinomial re-sampling. Douc et al. [149]

Algorithm 6 RSR re-sampling

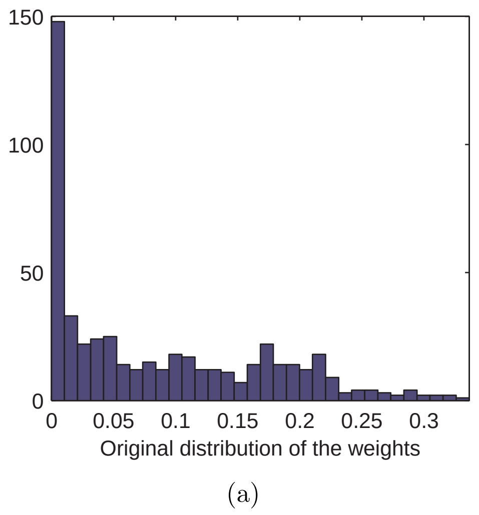
```
**Function definition:** $\left[\{m^i\}_{i=1}^{n_s}\right] = \text{RSR}\left(n_s, \{\tilde{w}_t^i\}_{i=1}^{n_s}\right)$

Purpose: Generation of an vector of multiplicities $\{m^i\}_{i=1}^{n_s}$ at time instant $t$, $t > 0$

Generate a random number $u^0 \sim \mathcal{U}[0, \frac{1}{n_s}]$

**for** $i = 1$ to $n_s$ **do**
    $m^i \leftarrow \lfloor(\tilde{w}_t^i - u^{i-1}) \cdot n_s\rfloor + 1$
    $u^i \leftarrow u^{i-1} + \frac{m^i}{n_s} - \tilde{w}_t^i$
**end for**
```

concluded that for practical applications of Sequential Monte Carlo methods residual, stratified and systematic methods were generally found to provide comparable results but systematic was often preferred because it is the simplest method to implement. Arulampalam et al. [69] also took the view that systematic is preferred on the grounds of implementation simplicity and minimization of the Monte Carlo variation. But from a computational perspective, all the above methods incorporate a **while** loop. This type of structure can be inconvenient to implement at machine level because of the need to accommodate an unspecified number of iterations, and the overall processing time will vary with the weight statistics; this can present difficulties for real-time applications with a frame rate constraint.

Bolić et al. [145] re-examined re-sampling processes from an implementation complexity perspective. They developed an algorithm, referred to as residual systematic re-sampling (RSR), based upon residual re-sampling but avoiding the second iteration to make up the set residue. Their approach is shown in Algorithm 6 (note: the authors make the unfortunate choice of using the symbol 'i' to signify particle multiplicity rather than particle index, and 'm' was used for the particle index; in the algorithm description given here the notation is reversed to aid readability).

This residual method produces a re-sampling that lies somewhere between the systematic and stratified methods illustrated in Figure 7.4(b). The target for re-sampling is to produce a sample multiplicity with the probability of reselected index proportional to the particle weight i.e. $P(i^j) \propto \tilde{w}^i$. With a total number of samples n_s this makes the target multiplicity $m^i \propto \tilde{w}^i n_s$. The following argument illustrates the rationale of the RSR approach. The first line within the 'for' loop is considered, and rewritten, to give:

$$m^i \leftarrow \left\lfloor \tilde{w}_t^i n_s - u^{i-1} n_s \right\rfloor + 1 \qquad (7.12)$$



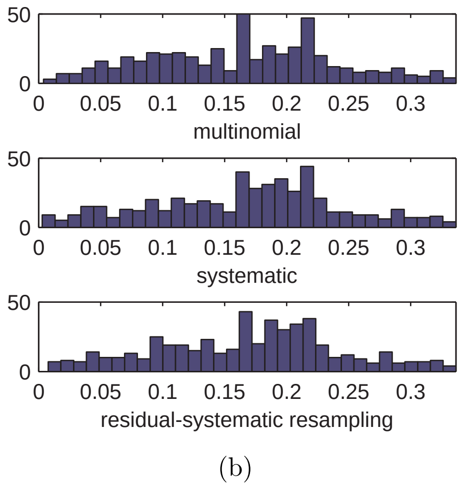


Figure 7.5: Comparison of weight distribution before (a) and after re-sampling (b)

The process starts with a random number, u^0, lying between 0 and $1/n_s$. If the initial weight is zero then the maximum value within the brackets is zero and the minimum is -1. This means that the multiplicity for $\tilde{w}_t^i = 0$ will be either 0 or 1. If $\tilde{w}_t^i > 0$ then the multiplicity becomes the value expected from deterministic re-sampling i.e. $\tilde{w}_t^i n_s$, but with a random decrement, and the $+1$ produces a rounding up. The second line in the loop is a negative feedback step, correcting an overestimate. If it is rewritten as:

$$u^i \leftarrow u^{i-1} + \frac{m^i - \tilde{w}_t^i n_s}{n_s} \tag{7.13}$$

then it can be seen an n^{th} of the difference between the calculated multiplicity and the target multiplicity is added to the random number for the next step. The overall effect is a residual re-sampling with a fractional residue correction at each step. This avoids the need for a second iteration with its troublesome **while** loop. There is, however, still the need for the residual re-sampling step of allocating the states to their multiplicities, and a requirement of a division at each step. If the number of samples n_s is chosen to be an integer power of 2 then the division can be implemented through simple register shifts. An example of the outcome of an RSR re-sampling is shown in the lower histogram of Figure 7.5(b).

All of the methods have similar outcomes, timings and computational complexity, but the RSR method is distinctive in that it delivers computational convenience for real-time applications.

7.4 Discussion

The first part of the chapter considered the recent claim to a provably correct MAP estimator for particle filters. The approach was tested against challenging sequences in which targets with very similar representations either moved closely in parallel paths or had crossing paths. It was found, using repeated runs of the sequences, that the method did not prevent track loss in general. A practical alternative was presented. This alternative simply pulled out highest weight subsets and assumed that they represented either one or two modes. The mode centres were found using the weighted means of the subsets. The mode nearest to the predicted track point was then considered, taking account of the predicted track points of nearby competing trackers, and the most probable one was chosen. Experiment showed that this was more reliable than the ‘provably correct’ one.

The final step of the particle filter, re-sampling, was also looked at in depth. Commonly accepted methods were rejected and the recently proposed residual systematic approach was favoured due to its computational convenience.

Chapter 8

Tracker initialization

8.1 Introduction

The work up to this point has shown that arbitrary objects can be tracked successfully using independent particle filters and a structured histogram object representation. In the examples presented the trackers were initialized manually. The operation of the trackers was not sensitive to the position of the initial tracking point, the particle filters would follow the objects successfully if the initial tracking box was placed reasonably centrally on the target of interest.

It has been noted [4] that automatic tracker initialization is not a simple task and many significant tracking studies assume that the targets of interest can be identified by some other process or detection module. The detection process is, of course, integral in radar style detect-before-track (DBT) systems. In those approaches the processes start with a measurement set and focus on assigning the measurements to targets. Target 'birth' is accommodated by looking for consistent measurements that can not be accounted for by the current tracks.

DBT is also the basis of some approaches to non-radar multi-object tracking. The Koller-Meier Condensation extension and the PHD filter involved 'mode growth' around new measurements, BraMBLe dynamically extended the multi-target state using a predetermined set of object labels and looking in the image for the newly appearing evidence of known representations, Okuma's boosted particle filter incorporated a trained classifier to find new targets, etc. Those systems work well in the DBT context, but can be computationally costly to implement. For example, Maggio et al. [113] report that around 67% of the computational resources in their change detection based PHD filter tracker were used in the detection step.

The histogram based tracking approach developed in this work is track-before-detect (TBD). The tracking process is, itself, the measurement step. In the context of this work the particle filter is essentially a search of sub-regions of the image into which the target motion model suggests that the object might move. The measurement consists of finding the strongest evidence for the targets within those regions. In the DBT approaches newly appearing targets were identified where they conformed to some predefined all-target representation; in this TBD approach targets are not sought in terms of some all-target representation, they are initially identified in terms of being other than background, having a general shape close to that of the objects of interest, and having movement. Once detected each target being tracked has its own separate identity and representation. In fact it is the individualized representation that helps the trackers to deal with occlusion and recovery after temporary loss of track. The tracking part is not the difficult one, the main problem is that of initially locating the trackable object and then finding a suitable central point from which a good representation can be drawn.

The workload associated with the initialization search can be eased to some extent by limiting the regions within the image that targets can be expected to appear. In a security or CCTV surveillance context there are generally perimeters, tripwires or other boundaries defining regions of interest. Disturbances at those boundaries would trigger tracker initialization. The aim would be to track anything that violates the boundaries and only subsequently classify it as being of interest dependent upon the way the path develops.

The computational constraints associated with this work rule out some possible initialization approaches. The target frame rate of 5fps is too low for optical flow type approaches. Detection of specific shapes using identification tools based on SIFT [150], HOG [58], or classifiers in general, calls for pre-training and can be computationally demanding. Methods that rely on the extraction of large numbers of histograms of background regions are rejected because of the time and memory demands of constructing an integral histogram additional to that used for foreground object tracking.

Instead the work focuses on those broad characteristics of motion, general shape, and being other than background. Two experimental methods are described. The first has characteristics similar to those of the Beleznai et al. [47] detection but is simpler in its implementation, the second is a novel method that draws on aspects of the particle filter itself when a motion trigger is activated in predefined regions in the camera view. The first method calls for the use of a background image so the chapter builds upon Chapter 2 and starts with a description of the background modeling approach developed in this work.

8.2 Background image for initial detection

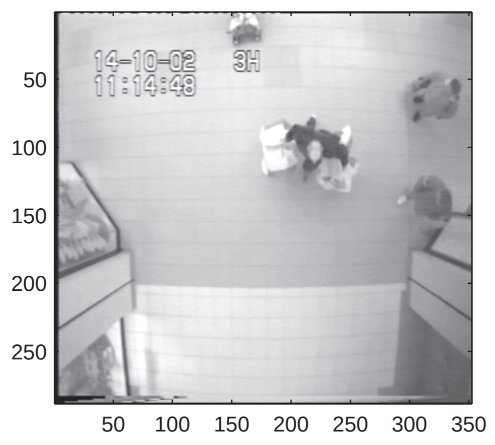
Chapter 2 presented a brief survey of commonly cited background modeling techniques. The focus of the survey was to consider the methodology and look for aspects of the approaches that might transfer well to the constrained conditions of the fixed point processing with camera embedded software. Out of the techniques presented the Approximating Median had some attractions: it would be simple to implement, requiring only background pixel value increment or decrement to follow differences between foreground and background. The median based approaches are based upon the assumption that the background values are seen more often than foreground ones. The assumption does not hold well if there is heavy traffic in the field of view: in such cases the background image becomes polluted with residues of the passing objects. The approach also responds slowly, taking a long time to find background at startup. Figure 8.1 shows the performance of the Approximating Median over 260 frames of the Overhead sequence. Figure 8.1(c) shows that there is still evidence of initial objects after the first 100 frames of processing. By the 260^{th} frame the initial objects have disappeared but evidence of traffic trails has begun to emerge.

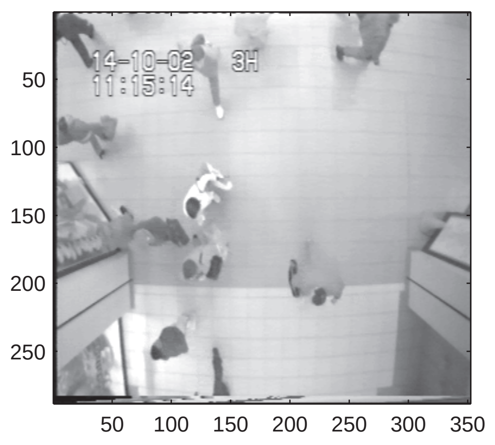
It was concluded in Chapter 2 that the counting based Multi-Modal Mean had useful characteristics. It was, in effect, an efficient approximation to the MOG. The approach developed for this work has characteristics similar to that of the Multi-Modal Mean but is implemented in a different way. It works with the assumption of a single mean pixel value rather than a set of means, it works with monochrome images rather than colour components, and instead of counting the number of occurrences of matches to the mean values out of a given number of frames it simply looks for pixel value consistency over a consecutive small sequences of frames. It is a dual background approach: pixels that remain stable for a short run of frames become part of a short-term background, the short-term background is then used in a recursive update of longer-term background. The method is described in Algorithm 7.

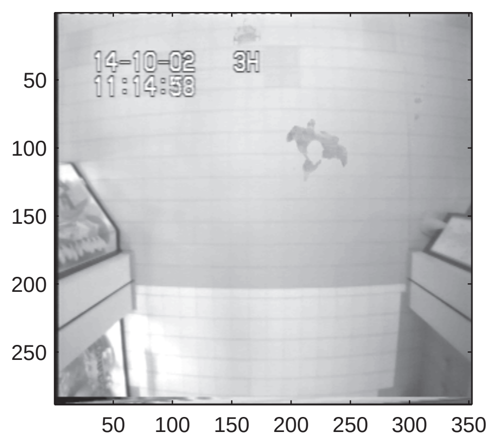
The long-term background image $I_{B_{LT}}$ is initialized with the first frame I_0. Each pixel is then subsequently examined for temporal consistency. If the absolute difference between the pixel intensities in successive frames is less than a pre-set threshold $\theta_{\Delta t}$ then a counter, C, associated with the pixel is incremented. A pixel intensity temporal mean is developed, starting from $\mu = 0$, through the accumulation of fractions of the pixel intensity values at each step. If the accumulation is set for a continuous sequence of C_{max} frames then the fraction to be added at each step is I/C_{max}. If the intensity varies beyond the threshold then both the counter and the pixel mean are re-set to zero. If a continuity pixel count reaches C_{max}, indicating

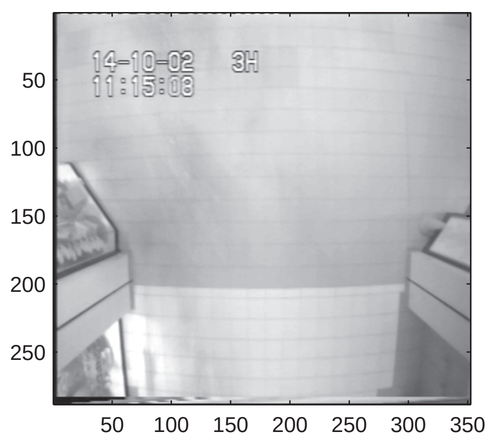
Algorithm 7 Mosaic of short term means

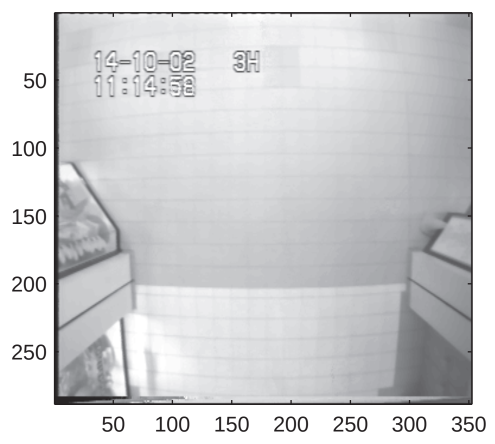
```
**Function definition:**
$[I_{B_{LT}}, I_{B_{ST}}, C_t, G_t, \mu_t] = mmean\left(I_t, I_{t-1}, I_{B_{LT}}, I_{B_{ST}}, C_{t-1}, G_{t-1}, \mu_{t-1}, \theta_{\Delta t}, C_{\max}\right)$

**for** $x = 1$ to $cols$ **do**
  **for** $y = 1$ to $rows$ **do**

    $I_{\Delta t}(x, y) \leftarrow |I_t(x, y) - I_{t-1}(x, y)| < \theta_{\Delta t}$

    **if** $I_{\Delta t}(x, y) = 1$ **then**

      $C_t(x, y) \leftarrow C_{t-1}(x, y) + 1$
      $\mu_t(x, y) \leftarrow \mu_{t-1}(x, y) + I_t(x, y) / C_{max}$

    **else**

      $C_t(x, y) \leftarrow 0$
      $\mu_t(x, y) \leftarrow 0$

    **end if**

    **if** $C_t(x, y) = C_{max}$ **then**

      $G_t(x, y) \leftarrow G_{t-1}(x, y) + 1$
      $C_t(x, y) \leftarrow 0$
      $I_{B_{ST}}(x, y) \leftarrow \mu_t(x, y)$
      $\mu_t(x, y) \leftarrow 0$

      **if** $G(x, y) <= 8$ **then**

        $I_{B_{LT}}(x, y) \leftarrow \frac{G(x,y)}{G(x,y)+1} I_{B_{LT}}(x, y) + \frac{1}{G(x,y)+1} I_{B_{ST}}(x, y)$

      **else**

        $I_{B_{LT}}(x, y) \leftarrow \frac{7}{8} I_{B_{LT}}(x, y) + \frac{1}{8} I_{B_{ST}}(x, y)$

      **end if**

    **end if**

  **end for**
**end for**
```


(a) frame 1


(b) frame 260


(c) Approximating Median, 100th frame


(d) Approximating Median, 260th frame


(e) Mosaic of Means, 100th frame

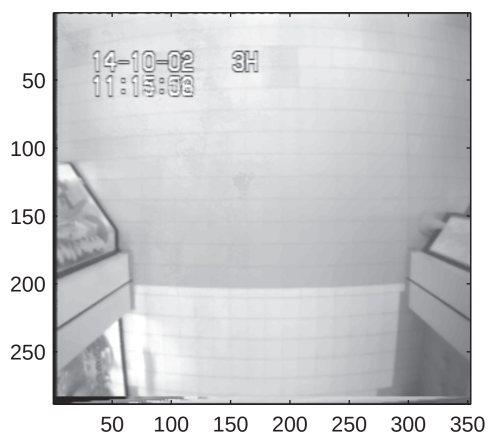

(f) Mosaic of Means, 260th frame

Figure 8.1: Comparison of Approximating Median and Mosaic of Means background development. It can be seen that by the 100th frame the Mosaic of Means has cleared the foreground content of frame 1. By the 260th frame the Mosaic of Means remains clear, whereas the Approximating Median shows signs of traffic contamination.

that the pixel intensity variation has been less than $\theta_{\Delta t}$ for a continuous run of C_{max} frames, then the mean value is assigned to the short term background $I_{B_{ST}}$, the counter and the mean are then reset. In the initial 'burn-in' period a 'group' count G is maintained for each pixel. This counts the number of continuous short term background runs that have occurred since the start. The group count is used to fractionally develop the long term background $I_{B_{LT}}$. Once the group count has reached $G = 8$ it defines the end of the burn-in sequence for that pixel and G remains constant. This means that the long-term background is updated with the constant fractional contributions of $7/8^{ths}$ the previous long-term background and $1/8^{th}$ of the latest short term background. The process seeks out temporally stable pixels, background update is suspended during heavy traffic.

If the pixels show stability for a reasonable period of time their contribution grows in the long term background. The example shown in in Figure 8.1(e) indicates that, with this approach, evidence of objects in the initial frame has been removed, and in Figure 8.1(f) that there is much less traffic contamination than in the Approximating Median.

8.3 Mean-shift based object detection

The mean-shift based object detection approach of Beleznai et al. [47] was outlined in Section 2.4.4. It was suggested in the conclusion to Chapter 2 that a similar approach might be useful for identifying objects entering the field of view.

The mean-shift process involves an iterative ascent towards the local mode of a density distribution [43]. The density distribution in the Beleznai approach is represented by the pixel values in the image $I_{\Delta B}$ formed by unthresholded subtraction of the background image and the current frame.

$$I_{\Delta B} = |I_t - I_{B,t}| \tag{8.1}$$

Given a region of image of width Δx and height Δy centred at a location (x, y), the centre of mass coordinates $(\hat{x}, \hat{y})$ are determined by:

$$\hat{x} = \frac{\sum_{x\in\Delta x} x I_{\Delta B}(x, y)}{\sum_{x\in\Delta x} I_{\Delta B}(x, y)} \tag{8.2}$$

$$\hat{y} = \frac{\sum_{y\in\Delta y} y I_{\Delta B}(x, y)}{\sum_{y\in\Delta y} I_{\Delta B}(x, y)} \tag{8.3}$$

Each mean-shift step in the iterative ascent involves moving the centre of the region from its current position (x, y) to the centre of mass $(\hat{x}, \hat{y})$ of the region.

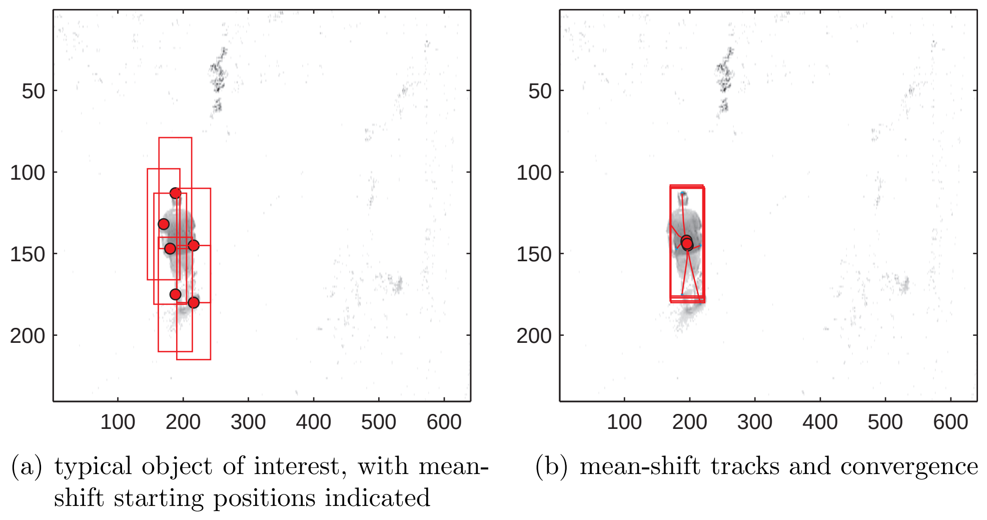
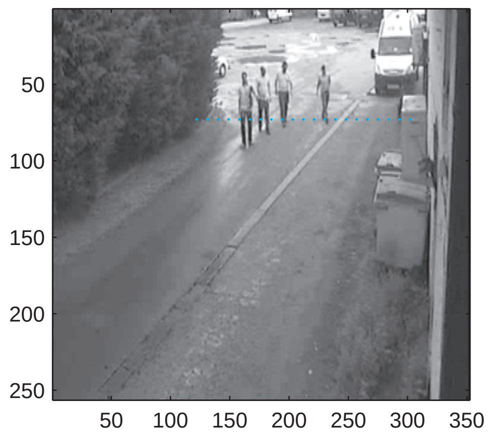


Figure 8.2: Illustrative Beleznai mean-shift start points and convergence

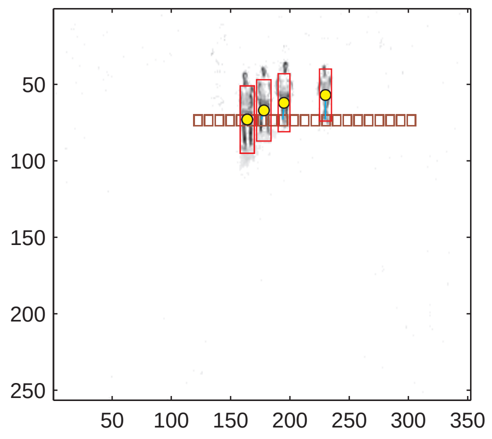
Figure 8.2(a) shows a typical set of mean-shift start points around the edges of a pedestrian image, together with the expected bounding box rectangular regions used for the process. Figure 8.2(b) shows the tracks taken on the mean-shift steps as the rectangles moved to convergence. It was stated by Beleznai et al. that only 3 or 4 steps were generally necessary for convergence. The start points were determined by a cycle of finding the maximum in the difference image $I_{\Delta B}$, setting it to 1 and re-scaling the rest of the image proportionately. A region having half the dimensions of the expected object bounding box was reset to 0 at the found point and the difference image was then re-scanned to find the next maximum value. The search was repeated and eventually stopped when the magnitude of the found maximum dropped below a threshold. The expected object bounding box size at a given row in the image was determined using the simple calibration of determining the expected sizes at low and high row values and then linearly scaling between those extremes.

Fast extraction of the centres of gravity was facilitated by calculating three integral images, one for each of $I_{\Delta B}$, $xI_{\Delta B}$ and $yI_{\Delta B}$, so that each component in Eqs.(8.2) and (8.3) could be determined by simple integral image look-up operations.

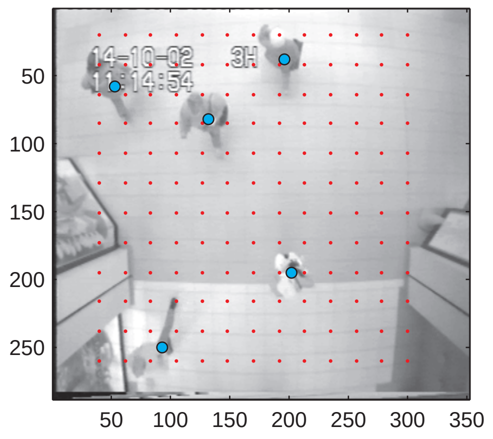
The approach developed in this thesis was similar in the use of the three integral images, the bounding box scaling and aspects of the iterative processes but had different elements aimed at computational economy. The mean-shift start points were not determined by the exhaustive process of searching out and accumulating maxima in the difference image, start points were preset along lines, perimeters or grids in the image. Mean-shift 'triggers' were activated through the placement of square detector 'pads' centred on each point. The pads had both width and height equal to the expected width of the target object bounding box at the pad position. The pad size, however, was not critical: in some cases pads could be smaller than



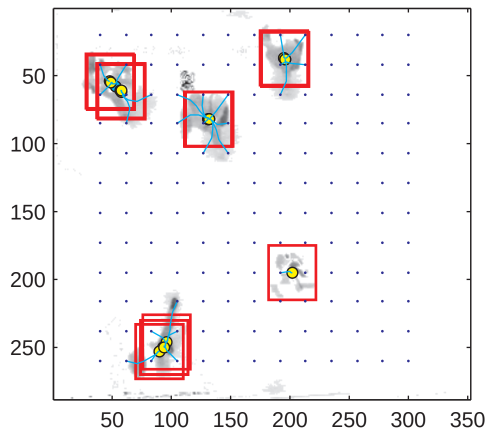
(a) Rear Building sequence, tripwire



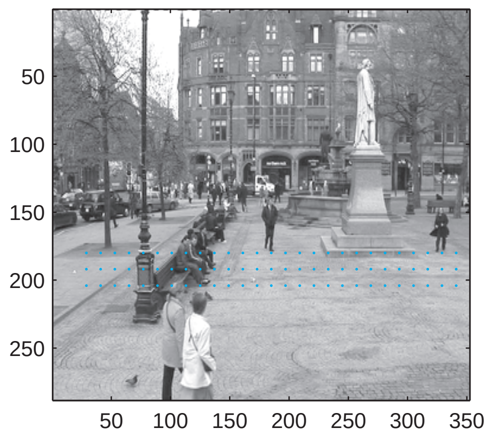
(b) feature image and detections



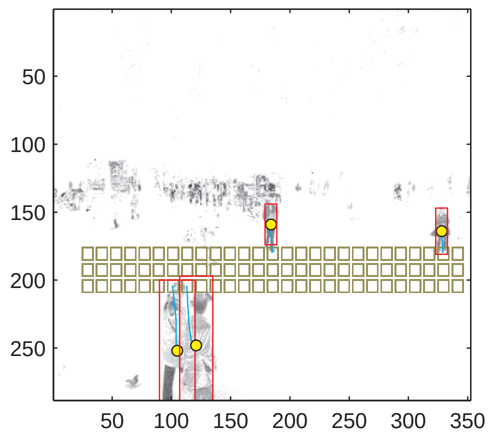
(c) Overhead sequence, grid



(d) feature image and detections



(e) Albert Sq. sequence, mini-grid



(f) feature image and detections

Figure 8.3: Pad based mean-shift detection

the expected target width and still trigger successfully.

The image used for the mean-shift steps also differed. A binary image was made by thresholding the background subtracted image:

$$I_{\Delta B\theta} = |I_t - I_{B,t}| > \theta \tag{8.4}$$

The background differences image was then pixel-wise multiplied by the thresholded image to produce the feature image $I_{\Delta f}$ to be used for the mean-shift process:

$$I_{\Delta f} = I_{\Delta B} \cdot * I_{\Delta B\theta} \tag{8.5}$$

The result was that small differences were set to zero and those above the threshold were retained.

A range of options were available for triggering the pad, the essence being to detect change. The trigger could occur as a result of threshold crossing by the sum across the pad of background subtraction differences, the sum of temporal differences, the mean of the differences, the sum of binary indicators of differences etc. Experimentation suggested that temporal differences were more useful than the others. The subtraction of successive frames provided some immunity to general illumination changes and the differences in the pad regions tended to be small in the event of lateral camera movement. The low frame rates meant that the number of differences across the pad could be large when an object moved through it. In the examples shown here, the triggers were based upon the mean temporal difference $\hat{d}_t$ across the pads, and a requirement that the mean was rising, i.e.

$$\hat{d}_t = \frac{1}{n_P} \sum_{x,y \in P} |I_{P,t}(x,y) - I_{P,t-1}(x,y)| \tag{8.6}$$

where P represents a pad, $I_{P,t}$ is the pad image at time t, and n_P is the number of pixels in the pad. The trigger condition was then:

$$\hat{d}_t > \theta_d \wedge (\hat{d}_t - \hat{d}_{t-1}) > 0 \tag{8.7}$$

where θ_d is a difference threshold.

When a pad was triggered an expected object bounding box sized rectangle was initialized at the triggered pad centre and mean-shifted to the object from that point. In perspective views the size of the mean-shifting rectangle changed to match the expected target size at the current row position. The mean-shift iterations were preset to a fixed number of steps rather than progressing to convergence. It was found that presets of 4 or 5 steps were adequate. It was also found that the shifts

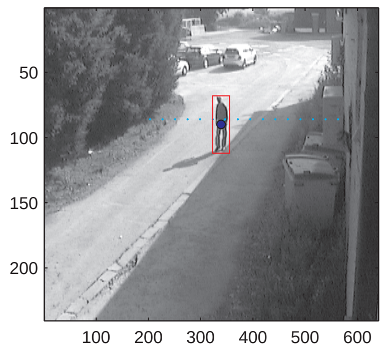


Figure 8.4: Mean-shift detection with target shadow

converged to the object irrespective of whether they were triggered by the arriving object itself or the trailing temporal difference 'ghost'.

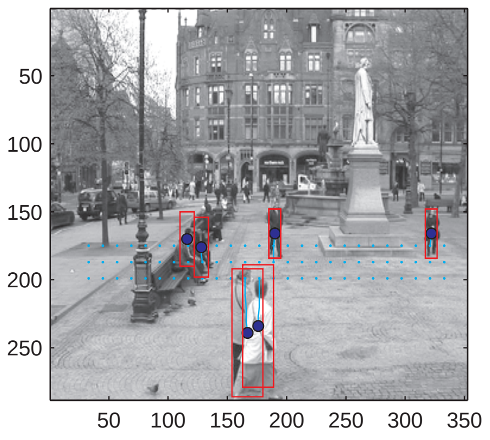
Examples of the detection process can be seen in Figure 8.3. Figure 8.3(a) shows a typical industrial scene with a single tripwire of pads. Figure 8.3(b) shows the corresponding thresholded feature image and pad triggered mean-shifts. The detections continue for a sequence of frames as the targets move through the tripwire pads. There is a range of options for tracker initialization: the first detection could initialize a tracker with subsequent detections judged to be from the same object being suppressed, multiple trackers could be initialized and merged, initialization could occur after a series of consistent detections has occurred etc.

It can be seen that the detection bounding boxes do not necessarily fit the targets entirely. This turned out to be not a problem: the initial tracking histogram was based upon a region within the objects with dimensions half that of the bounding box, it was generally extracted from representative pixels within the perimeters of the objects.

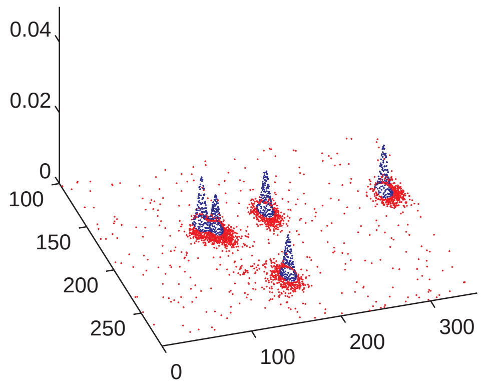
Figures 8.3(c) and 8.3(d) illustrate the detection process in the Overhead sequence. In this example a grid of detect regions is used but a single rectangular arrangement of centres could be used as a sensitive perimeter. Figures 8.3(e) and 8.3(f) show detections in a perspective view. In this case a simple mini-array of centres is adequate: foreground targets trigger the pads with their heads, those further back in the scene trigger them with their feet.

Figure 8.4 shows a simple tripwire arrangement and a target with a significant shadow. The tripwire is set up to detect pedestrians and hence aims to fit a vertical target shaped rectangle in the mean-shift process. The search for the maximum in the difference image moves the rectangle onto the target and is not significantly affected by the shadow. In contrast, a blob based detection system would return a bounding box encompassing both object and shadow.

Figure 8.5 shows the mean-shift end points being used as input to a PHD filter.



(a) mean-shift detections



(b) PHD filter on the detections

Figure 8.5: Mean-shift detection as a source for a PHD filter

As discussed in Section 4.7, there is little advantage in the use of the PHD filter because of the need for further processing to extract the states, and the collapse of the modes in the absence of detections. It is suggested in this thesis that it would be better to initialize conventional single particle filter trackers at the detection points.

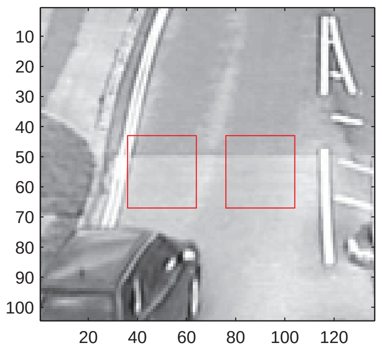
8.4 Histogram based detection

A simpler but surprisingly effective alternative to the mean-shift detection is to use histogram information. This has the computational attraction of removing the requirement of a full-frame background image and drawing on the resources already available for the histogram based tracking. The triggers use the same pad approach as described in the previous section, using a rising mean of pad temporal differences. The pads can be placed singly at known entry points, in tripwire style arrays, perimeters or grids. Simple tripwires or single placements tended to be appropriate. Figure 8.6 shows an example of a pair of pads being used with the AVSS 2007 i-LIDS Challenge data-set [10]. The original sequence ran at 25fps but every fifth frame was extracted to produce the target 5fps. Figures 8.6(a) and 8.6(b) represent successive frames at the reduced rate.
The general approach is to:

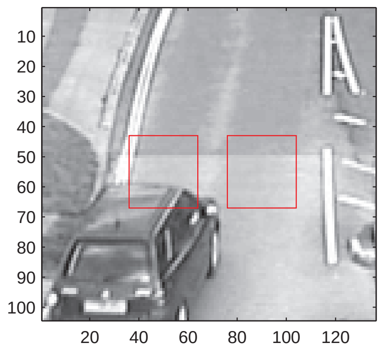
(i) look for temporal change in the pad region;

(ii) in the absence of change make a 'probably background' histogram of the pad region;

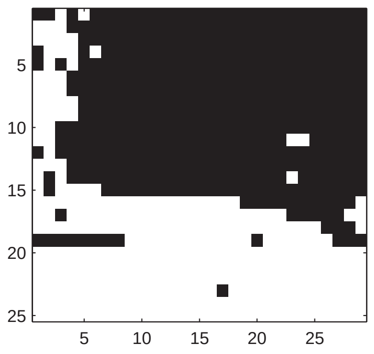
(iii) if change then find the change pixels and construct a 'probably foreground' histogram;



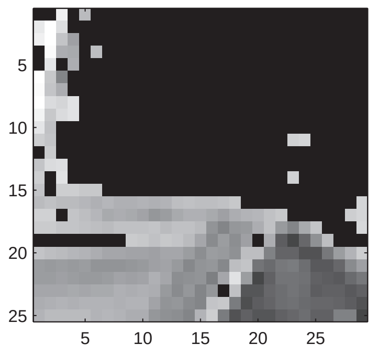
(a) pad region at time $t-1$, $I_{P,t-1}(x,y)$



(b) pad region at time t, $I_{P,t}(x,y)$



(c) pad(1) temporal difference mask $I_{P\Delta t\theta}(x,y)$



(d) masked pad(1) difference image $I_M(x,y)$

Figure 8.6: Trigger frames for pad histogram based detection

(iv) look around the pad for further evidence of 'probably foreground';

(v) find the most likely centre of the 'probably foreground'.

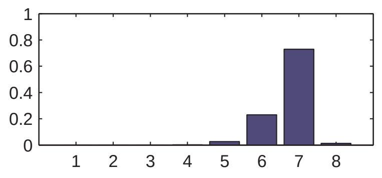
8.4.1 Constructing an object indicator histogram

In the absence of a trigger a normalized 8-bin 'probably background' grey scale histogram $\mathbf{q}_{B,t}$ is extracted from the pad pixels. An integral histogram of the whole image is made at each frame, to be used for the particle filter tracking, so this 'pad only' background histogram can be extracted without much extra computation. The histogram is used to recursively update a 'time averaged probably background' histogram $\hat{\mathbf{q}}_{B,t}$:

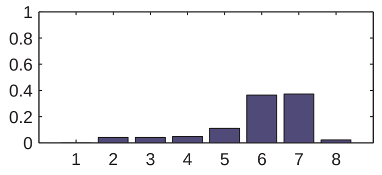
$$\hat{\mathbf{q}}_{B,t} = (1-\alpha)\hat{\mathbf{q}}_{B,t-1} + \alpha\mathbf{q}_{B,t} \tag{8.8}$$

with the update fraction α having a value typically of the order 0.2.

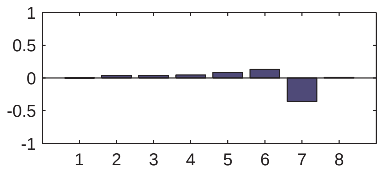
When a trigger condition occurs an 8-bin histogram $\mathbf{p}_t$ is made of the pad image. This contains information about both background and object. The pad image histogram $\mathbf{p}_t$ is used together with the time averaged probably background histogram, $\hat{\mathbf{q}}_{B,t}$, to make an 'indicative object' proposal histogram $\tilde{\mathbf{q}}_{o,t}$. The first step is to



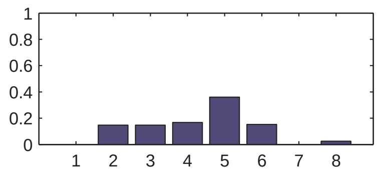
(a) time averaged probably background $\hat{\mathbf{q}}_{B,t}$



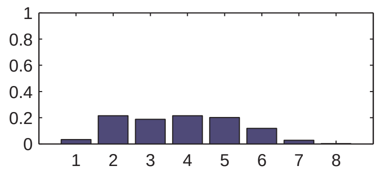
(b) pad image histogram $\mathbf{p}_t$



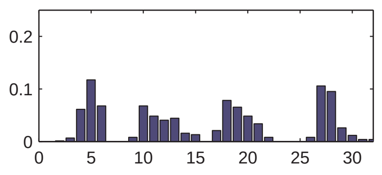
(c) histogram differences $\mathbf{q}_{\Delta,t}$



(d) proposal $\tilde{\mathbf{q}}_{o,t}$



(e) 8-bin tracking histogram $\mathbf{q}_t$



(f) four-quadrant histogram

Figure 8.7: Histograms used in pad proposal histogram construction

construct a difference histogram $\mathbf{q}_{\Delta,t}$ by subtracting the time averaged probably background histogram from the current pad histogram:

$$\mathbf{q}_{\Delta,t} = \mathbf{p}_t - \hat{\mathbf{q}}_{B,t} \tag{8.9}$$

Negative bin values in the difference histogram indicate pixel value ranges that are more representative of the background rather than the foreground. Figures 8.7(a) to 8.7(c) show a set of typical histograms associated with this stage of the process.

Next, a difference masked pad image $I_M(x, y)$, shown in Figure 8.6(d), is made by pixel-wise multiplication of the pad thresholded temporal difference image $I_{P\Delta t\theta}(x, y)$ with the pad image $I_{P,t}(x, y)$ i.e.

$$I_M(x, y) = I_{P\Delta t\theta}(x, y) \cdot * I_{P,t}(x, y) \tag{8.10}$$

The masked image contains pixels associated with the object, shadows, ghosts, and zero values associated with the static region. The masked image is grey level sliced using the bin boundaries of the n = 8 bin histogram to produce a binary pad image $I_{Pb}(x, y)$ for each slice u:

$$I_{Pb}(x, y) = \delta\left(b\left(I_M(x, y)\right) - u\right) \quad \forall(x, y) \in P \tag{8.11}$$

where $b\left(I_M(x, y)\right)$ is a function that compares the grey scale value $I_M(x, y)$ against the bin boundaries and returns the bin number associated with it; $\delta(\cdot)$ is the Dirac delta function. The lower boundary of the first bin is set to 1 rather than 0 out of the range $[0, 255]$; this removes the mask static pixel contribution.

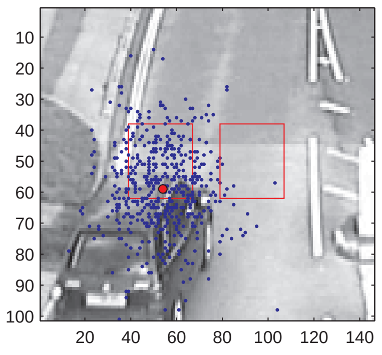
The bins of the proposal histogram $\mathbf{q}_{o,t}$ are set to be the sums of the corresponding binary slice $I_{Pb(u)}$ if the associated difference histogram bin $\mathbf{q}_{\Delta,t}(u)$ is positive, otherwise they are set to zero:

$$\mathbf{q}_{o,t} = \begin{cases} \sum\limits_{x,y\in P} I_{Pb(u)}(x, y) & \text{if } \mathbf{q}_{\Delta,t}(u) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8.12}$$

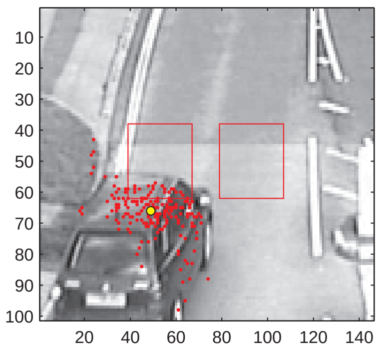
The ‘indicative object’ histogram is then normalized:

$$\tilde{\mathbf{q}}_{o,t} : \tilde{q}_{o,t}(u) = \frac{q_{o,t}(u)}{\sum\limits_{v=1}^{n} q_{o,t}(v)} \quad u = 1, ..., n \tag{8.13}$$

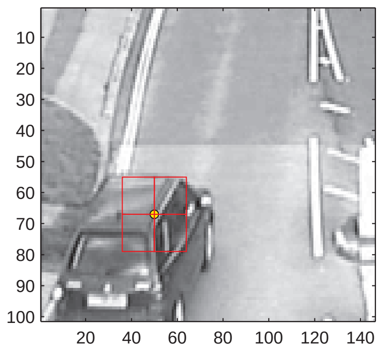
The proposal histogram, shown in Figure 8.7(d), indicates the grey scale values that are more characteristic of the part of the object entering the pad region than the



(a) Particle prior around difference image centre of gravity



(b) High weight particles and weighted mean



(c) four quadrant region centred on mean position

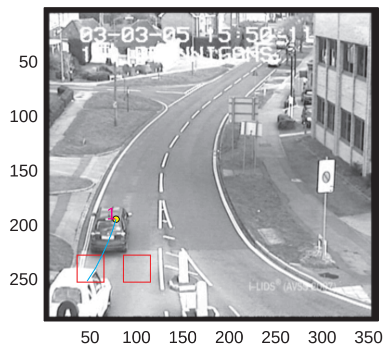
Figure 8.8: Particle filter type search around the pad and initial four quadrant tracking region

background. This proposal histogram and the centre of gravity of the pad pixel-wise difference image are used as the starting points of a search for a more representative histogram of the incoming object. Even though the proposal histogram is not a true representation of the object histogram, it is likely to be closer to it, in terms of standard feature distance measures, than to histograms calculated from background in the region of the pad after the triggering event. The proposal histogram does not include bins that are dominant in the probably background histogram.

8.4.2 Finding a histogram to track

A particle filter type search is used on the region around the pad in order to get a better representation of the object. A prior is constructed by adding Gaussian random values to the centre of gravity $\mathbf{x}_0 = (x_0, y_0)$ of the thresholded pad difference image $I_{P\Delta t\theta}(x, y)$ i.e.

$$\left\{\mathbf{x}^i = \mathbf{x}_0 + \boldsymbol{\nu}^i\right\}_{i=1}^{n_s} \tag{8.14}$$


(a) Tracker progress after initialization

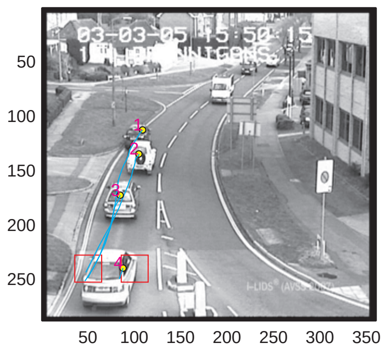

(b) Successive initializations and tracks

Figure 8.9: Initialized trackers in the i-LIDs sequence

where $\boldsymbol{\nu}^i \sim \mathcal{N}(0, \Sigma)$. The standard deviations σ of the components of the distribution are taken to be half of the largest dimension of the detection pad. A typical particle prior centred on the pad difference image centre of gravity is shown in Figure 8.8(a).

Histograms $\mathbf{p}^i$ are extracted, using the integral histogram, at each of the n_s particle prior locations. They are given a weight in terms of the histogram intersection distance and linear likelihood as described in Chapter 6. The alternative, if the computational constraints are not stringent, is to use the Bhattacharyya distance and Gaussian weighting. Particles with weights above a chosen threshold are extracted and their weighted mean position is taken to be the detect point for tracker initialization. A weight threshold of $\theta_w = 0.2$ was used. An example of high weight particles and their weighted mean is shown in Figure 8.8(b).

8.4.3 Tracker initialization and subsequent management

The final step of the detection process is to extract a tracking histogram at the weighted mean position of the high-weight particles. An 8-bin tracking histogram, $\mathbf{q}_t$, extracted at the object location indicated in Figure 8.8(b), is shown in Figure 8.7(e). The histogram is extracted from a rectangular region, centred on the weighted mean position, with dimensions half those of the expected object bounding box at that point. For more effective tracking, a four-quadrant grey scale histogram as described in Section 5.5.1, would be extracted. The four-quadrant tracking histogram for this sequence, extracted from the region shown in Figure 8.8(c), is shown in Figure 8.7(f).

The particle filter initial velocities are taken to be zero at the initialization step, they pick up the speed of the object after a couple of iterations. Tracker progress for the object initialization described above can be seen in Figure 8.9(a). Subsequent

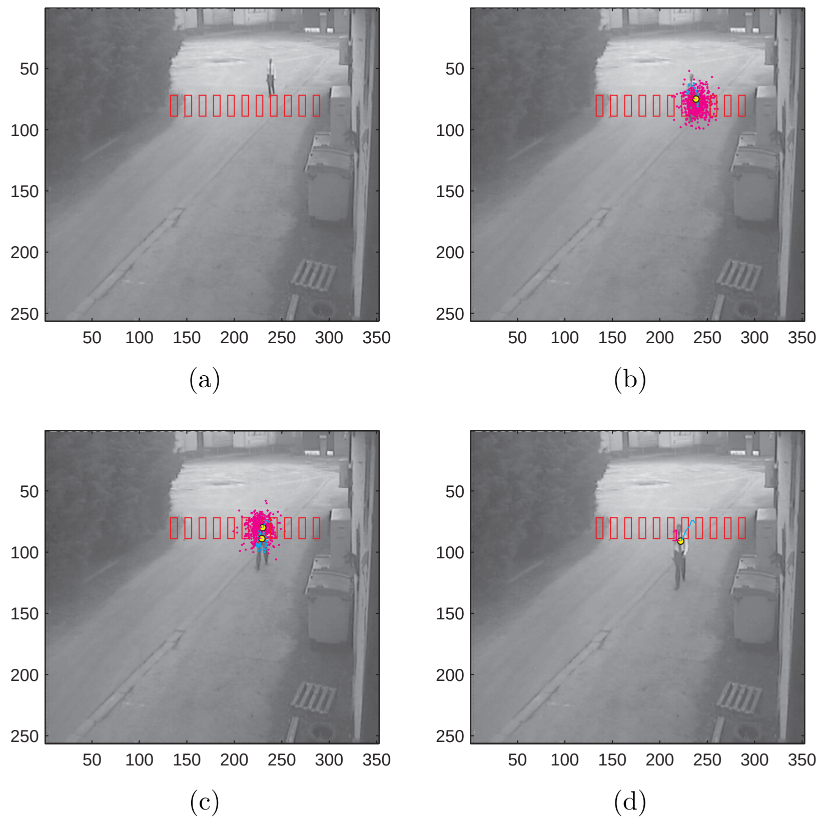


Figure 8.10: Tracker initialization with multiple triggers

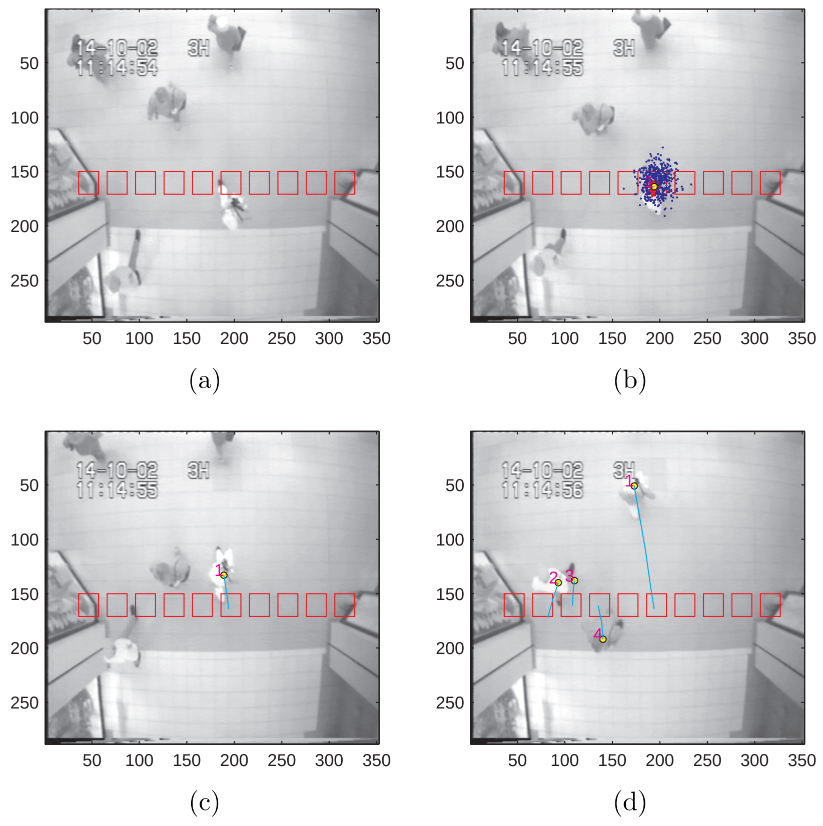


Figure 8.11: Tracker initialization, Overhead sequence

trackers initialized by a sequence of detections can be seen in Figure 8.9(b).

In the event of multiple detections of a single object as it passes through a pad the trackers are allowed to follow the object for a couple of frames. If trackers are close to each other, have similar velocities and similar histograms, then they are merged by taking the mean positions of the tracker end points, the mean velocities at those end points, and the average of the two histograms.

Figures 8.10 and 8.11 show further examples of pad histogram based tracker initialization. Figure 8.10 shows a pedestrian triggering an initialization in successive frames with the trackers merging into a single one. Figure 8.11 shows a set of initializations in the Overhead sequence. In Figure 8.11(d) it can be seen that a tracker is initialized on the foot of one of the shoppers; the foot is successfully tracked in subsequent frames. Additional unwanted trackers of this kind are less likely to occur with a mean-shift style initialization, the simple model shape would shift to cover the expected shape.

The number of allowed initializations at any time is determined by the computational capacity available. In typical applications it is expected that up to five trackers might be active at any time and that up to three might appear at the initialization step. Trackers would be 'signed off' when they progress beyond a perimeter and hence would free up resources for new trackers. In the case of detector grids, newly established trackers are assigned an initialization 'veto' region having half the dimensions of the object bounding around the proposed next position. Any new detections within a veto region are ignored.

8.5 Discussion

Two simple approaches to object detection and tracker initialization have been described. One was an extension of a previously published method, the other was novel and was tailored to make use of resources already available as part of the tracking process. It was found that a simple counting based background construction method was adequate for the mean-shift style approach and could cope with small variations in background illumination. The pad histogram based approach did not even require the development of a full frame background image, focusing only on identification of 'probably background' pixels within the pad regions. With appropriate choice of trigger thresholds both approaches were found to be effective in tracker initialization.

Consideration was given to combining the two approaches. In this case the triggered mean-shift rectangles would move from the trigger points to the local mode of the background subtraction differences. At the end point a histogram style search would then be carried out in an attempt to identify foreground pixels in the event of the end point not being reasonably central to the object. But in practice it turned out that the extra searches were not really necessary, the centres of the mean-shifted rectangle generally gave good tracker initialization points. Both the methods were effective in their own right, but a combination of them suggested no advantage.

The mean-shift style detection system was also considered as a source of measurements for a PHD style tracker, or as points for measurement based proposal densities along the lines of the boosted particle filter work of Okuma et al., as described in Section 4.5.2. But the adequacy of the independent particle filter tracker initializations, and the subsequent tracking behaviour, suggested that there was little to be gained from the additional computational burden.

Chapter 9

Conclusions and further work

9.1 Review of aims and findings

The aim of the work was to look closely at practical techniques associated with detection and tracking of moving objects in surveillance video with a view to eventual real-time implementation through camera embedded software. The field of computer vision linked to surveillance type applications is relatively young, growing alongside impressive hardware developments over the last two decades. It is characteristic of newly developing academic areas, with technical and possibly significant commercial potential, that they are idealistic and optimistic. It is also characteristic that theoretical explorations linked to a new and promising paradigm develop along diverse evolutionary pathways, many of which turn out to be limited, leaving only a subset capable of thriving in the environment of the time. Applications such as face detection and ANPR have developed well, are now established, and show commercial success. Areas such as pedestrian tracking and security related behaviour analysis are still in the development stages. The boundaries, limitations and constraints become clearer as the field matures. The outcomes of explorations in this thesis suggest that there are practical and implementable possibilities for object detection and tracking in CCTV video.

The work started out with a number of self-imposed constraints linked to the search for systems capable of being developed as camera embedded software sharing processing time on a fixed point visual signal processor. It aimed to deal with monocular imaging with a frame rate no greater than 5fps. It aimed to avoid approaches dependent upon mathematical operations like floating point division, extraction of square roots, calculation of Gaussian quantities, open-ended iterations etc. It recognized the need to be not dependent on good quality chromatic information, i.e. to work with what is essentially low resolution monochromatic video.

The study contributed the following findings with regard to particle filter tracking [15]:

- commonly used histogram similarity measures are related, and that satisfactory tracking can be achieved using the most computationally economic measure;
- there is no advantage in seeing the Gaussian likelihood standard deviation as a tuneable design parameter, if the distance measure is normalized to the range [0, 1] then the standard deviation should be set at $\sigma = 0.2$;
- a computationally economic linear likelihood can be used in place of the traditional Gaussian.

The traditional approaches to state estimation were rejected. It was shown that the weighted mean state estimation was inadequate if large particle filter process noise was used. The use of the MAP, or maximum weight, estimates could also lead to track loss in situations where similar targets moved closely or their tracks crossed. It was shown that:

- a novel sorting and counting approach, to indicate the dominant modes in the multi-modal *pdf*, coupled with a probabilistic link to predicted states, was able to maintain track in the case of closely moving similar objects.

The problem of tracker initialization was considered. It was concluded that existing object detection approaches would not offer the required computational efficiency for use in the given processing context. Two approaches were developed for initial target detection:

- a simplification of Beleznai [47] mean-shift object detection, to use pre-defined grids or lines of points as mean-shift start points;
- a novel histogram-based 'pad' trigger, that draws upon on the same resources as the trackers [17].

A summary overview of the complete work is given in the following sub-sections.

9.1.1 Approaches to object detection

An exploration of approaches to object detection and representation was carried out. They were grouped into those dependent upon background modeling and subtraction, those that were based upon temporal frame differencing, those that sought to make sense of blob patterns by fitting target models, and those that did not depend upon background subtraction but instead focused on the detection of salient characteristics of the objects of interest. It was recognized that an identifying characteristic of objects deemed trackable in surveillance video is that they are not background, so some attention was given to the methodology of background image development. Most of the object detection approaches, whilst theoretically sound, were based upon PC style processing with a dependence on floating point calculations and hence were seen to be not good candidates for the intended development.

9.1.2 Consideration of theoretical foundations

Much of the published work in the field of object tracking is described and justified in terms of Bayesian probability. The fundamentals of Sequential Bayesian Estimation were explored. The Kalman filter is recognized as the optimal device for tracking objects that can be represented in terms of linear process and measurement with Gaussian uncertainty. An interpretation of the filter equations was given to illustrate how they combine measurement and prediction uncertainties so that a measurement directed correction to the predicted state can be made. An examination of the theoretical foundations of the sub-optimal, but more applicable, particle filter was carried out. The particle filter provides a method that can be used in the more realistic non-linear process and measurement steps likely to be encountered in video based object tracking.

9.1.3 Multi-target tracking

The extension of Bayesian methods to multi-target tracking was explored. The methods had strong and well developed foundations in the field of radar and sonar type object tracking. It was of interest to see if the methodologies developed in that area would be productively transferable to multi-object tracking in video. It was recognized that because of their provenance, the methods were focused more towards resolving the data association problem, given similar detection signals, than to the management of multiple tracks in which each object might have its own distinct identity.

Other multiple object methods that were more closely linked to object tracking in video concerned themselves with issues like the resolution of occlusions, analysis in terms of world view representations of detected objects, or manipulations to penalize likelihoods where object tracks might cross. Some looked to see if there was any advantage in the Bayesian recursion of a complete multi-target state, but there was no clear evidence that such approaches were superior to the use of multiple independent trackers.

The Probability Hypothesis Density filter had been suggested as a possible foundation for multi-target tracking in surveillance type applications. Its mode of operation was looked at in a little detail. It was recognized as basically a computationally intensive clutter filter that was still dependent on having a separate object detection stage and a multi-frame data association step in order to resolve lost target and occlusion issues.

9.1.4 Implementation of the particle filter

With none of the multi-target approaches showing any attractive theoretical or practical aspects, attention returned to looking in detail at the implementation of the single target particle filter. A grey scale histogram representation was chosen. The integral histogram was used for fast extraction of representations at particle positions. With a frame rate of 5fps temporal differencing allowed stationary background components of the image to be ignored and successful particle filter based tracking of distinctive targets was demonstrated in a small range of scenarios. It was seen that the objects could be successfully tracked through small occlusions. The approach of using a simple multi-part histogram was adopted and the subsequent improved localization of the tracked object was illustrated. It was seen that the standard weighted mean estimate of state had its inadequacies when the process noise was not controlled well and paths of targets with similar appearance crossed.

9.1.5 Choice of similarity distance and particle likelihood

The next step was to look more closely at the operational components of the particle filter itself. It was of interest to see if the performance of the trackers would be compromised if the components were simplified for fixed-point embedded software implementation. The components of interest were the histogram dissimilarity distance and the measurement likelihood function, together with their parameters. The standard dissimilarity measure was taken to be the Bhattacharyya distance and the standard likelihood was taken to be Gaussian in form. Alternative distance measures such as the Matusita distance, the Histogram Intersection distance, the chi-squared distance etc. had all been chosen by other researchers. It was shown that there was little to choose between the distances: the Matusita distance was just an alternative expression of the Bhattacharyya distance, the others could be seen to have a link to the Bhattacharyya distance and simply introduce a bit of further scatter in the randomization of the particle set. The Histogram Intersection distance was shown to be adequate and was adopted because of its computational simplicity.

An important element was seen to be the choice of the value of the standard deviation if the Gaussian likelihood was used. The value was generally seen, in published work, to be a tunable parameter in the particle filter implementation without its central importance being considered. A wide range of values had been reported in the literature. It was shown that if the chosen distance measure ranged in value between 0 and 1 then the value of the parameter should be chosen to be approximately 0.2.

With the understanding of the choice of the Gaussian parameter, it was recognized that a linear likelihood would suffice in place of the Gaussian itself. It was demonstrated that there was no difference in tracking performance between a histogram representation Bhattacharyya based Gaussian likelihood and one with the computationally more attractive Histogram Intersection based linear likelihood. This shifted the attention to an examination of the methods for state estimation and particle re-sampling.

9.1.6 State estimation

Recent work had argued correctly that the weighted mean state estimate was inadequate and claimed a theoretically sound MAP approach. Experimentation showed that in the context of histogram based tracking in grey scale video the proposed MAP approach also showed deficiencies. An alternative approach was developed in which the dominant modes of the likelihood distribution were identified and the one closest to the predicted position of the relevant tracker was chosen, even though it might not be the largest one. The findings suggested that the mode selection approach gave a better performance than both the weighted mean and the theoretically sound MAP approach.

9.1.7 Particle re-sampling

Alternative approaches to re-sampling were considered. Whilst researchers in the field tend to find theoretical justification for their own choice of re-sampling method, in the work associated with this thesis it was found that there was little to distinguish between the approaches. One of them, however, offered a computational advantage over the others in that it was not dependent on the need for a WHILE loop, and hence that method was adopted.

9.1.8 Object detection for tracker initialization

Two possible approaches to tracker initialization were suggested. The first one was a target bounding box model-fitting approach based upon a mean-shift localization.

It had some similarities to the work of Beleznai et al. [47] but differed in the way the initial points in the mean-shift tracks were chosen. Instead of searching for local intensity maxima the start points are preset as either a set of grid points or as points along a line. Even though the target model, a rectangle, was simple, it turned out to be adequate for the task by returning a satisfactory near central position estimate for objects of interest. Using the detected centres as input to a multi-target PHD filter was considered but the additional demands of the filter suggested that there was no advantage in its use.

The mean-shift approach, being essentially 'blob' based, needed the generation of a background image. A counting based background image generation method was developed.

The second detection approach was designed to draw on the resources already available for particle filter tracking. It did not need the production of a background image. It used image intensity change sensitive 'pads' and a local histogram of pad background pixels built when the pads report little change from frame to frame. A pad change trigger then initiated a search around the pad for foreground pixels from which a trackable histogram could be inferred.

9.2 Further work

Effective tracker initialization remains an important area of investigation. The approaches have to be simple in order to be implementable within the frame rate and computational constraints imposed by a fixed-point processor of the kind described in Section 1.2. The descriptions of the two approaches described in Chapter 8 illustrated the concepts and can be regarded as starting points for further work. As with initialization approaches in general, they are prone to occasional initialization on non-target disturbances. False alarms tend to have characteristic signatures in the developing tracks. For example, the spacing between false alarm track points tends to vary unpredictably whereas true track points indicate a regular object velocity. Strategies for suppression of false initializations need to be developed.

Whilst optical flow based methods were rejected on the basis of the low frame rate, there is still some opportunity for motion based possible target detection through the use of the motion vector component of MPEG coding. Image compression for storage is one task carried out by the processor alongside any analytics addition. Motion vectors extracted from the compression process might be useful features that could be grouped to indicate regions of the image linked to trackable objects.

The work was essentially preparatory to hardware implementation, it remains to investigate the realization and performance of the proposals on the processor itself. It is not a trivial task, each processor design presents unique programming constraints and advantages. The need for unforeseen processor specific algorithmic compromises and approximations will emerge at the coding stages. But there are some further general processor independent aspects that might be considered. For example, one general processor difficulty lies with the randomization of the particle proposals. It is recognized that the randomization is a central element of the Monte-Carlo representation of the probability distribution functions, but it would be of interest to investigate the use of predefined, not necessarily uniform, ziggurat style grid patterns for the placement of the particle prior.

It was concluded that the multi-target tracking task could be dealt with by treating each tracker as independent, initializing a separate tracker with its own representation signature for each object. The problem of the best management of resources for multi-trackers has not been considered and is an aspect that needs to be developed further. One approach would be to allow only a fixed number of trackers to be in action at any time, with newly appearing objects being assigned a tracker when one becomes available, with trackers signing off when they move beyond a perimeter or the boundaries of the image. An alternative might be to not track all the objects at every available frame but have a 'sampling' approach in which the objects might be looked at every available frame early in the tracks but less frequently as the tracks develop. In effect the behaviour of the trackers during frames when they are ignored would be similar to that of trackers moving through occlusion.

Appendix A

Video sequences linked to the figures

The accompanying DVD contains a set of AVI files showing short animated sequences associated with figures from the thesis.

The filenames are:

Chapter 4

Fig 4.1 Overhead PHD.avi

Chapter 5

Fig 5.5 Overhead.avi
Fig 5.5 Additional Overhead.avi
Fig 5.7 Overhead motion.avi
Fig 5.7 Additional Overhead motion.avi
Fig 5.8a Square.avi
Fig 5.8b PETS2006.avi
Fig 5.8c iLIDS.avi
Fig 5.8d PETS2001 van.avi
Fig 5.9 Overhead occlude.avi
Fig 5.10 Large process failure.avi

Chapter 7

Fig 7.2 Mode select.avi

Chapter 8

Fig 8.3 Overhead mean shift.avi

Fig 8.3 Square mean shift.avi

Fig 8.4 Yard detection shadow.avi

Fig 8.9 Pad iLIDS.avi

Fig 8.9 Pad Overhead.avi

Fig 8.9 Pad Yard.avi

Appendix B

Publications

P.Dunne and B. Matuszewski. Choice of similarity measure, likelihood function and parameters for histogram based particle filter tracking in CCTV grey scale video. *Image and Vision Computing*, 29(2-3):178-189, 2011

P.Dunne and B. Matuszewski. Histogram-based detection of moving objects for tracker initialization in surveillance video. *International Journal of Grid and Distributed Computing*, 4(3):71-78, 2011

Choice of similarity measure, likelihood function and parameters for histogram based particle filter tracking in CCTV grey scale video

Peter Dunne[a,b,*], Bogdan Matuszewski[b]

[a]*AD Group, Daresbury Park, Daresbury, Warrington, WA4 4HS, UK*
[b]*Applied Digital Signal and Image Processing Research Centre (ADSIP), School of Computing, Engineering and Physical Sciences, University of Central Lancashire, PR1 2HE, UK.*

Abstract

The choice of particle filter dissimilarity distance measures and likelihood functions is considered in the context of object tracking in grey scale CCTV video. The geometrical interpretation of the Bhattacharyya coefficient and distance is reviewed and the relationships between the Bhattacharyya, Matusita, histogram intersection and χ^2 distances are examined. It is argued that as long as the likelihood function satisfies certain criteria its analytical form is not critical in the stated tracking context. This is demonstrated through an experimental comparison between the use of the standard Bhattacharyya distance/Gaussian likelihood combination and the potentially computationally simpler histogram intersection distance/triangular likelihood combination in particle filter tracking sequences. It is shown that the differences between the approaches are marginal when the likelihood criteria are applied. Whilst the analysis was focused on a specific application and context, we suggest that the findings will be of value to particle filter tracking in general.

Keywords: particle filter, similarity measure, likelihood, grey scale, histogram intersection

1. Introduction

Video object tracking for commercial and surveillance applications can be required for situations ranging from real-time focus of attention tasks, such as alerting CCTV control room operatives to possible intrusion in a restricted area with indication of the point of entry and subsequent path, to longer term data collection for statistical analysis such as collecting pathway patterns, learning about navigational strategies in stores, monitoring the lengths of queues etc.

The tracking task consists of identifying regions in the image frames corresponding to an object of interest, using some method to characterize the object and then following an updated characterization from frame to frame.

The method used to characterize the object is dependent upon the available computational resources and the quality of the image at each frame. It can range from fitting shape models to the target objects to representing the objects in terms of their colour or edge signature. A useful overview of approaches is given by Yilmaz et al [1].

The particle filter has become an established tool for the job. It is a recursive Bayesian estimation technique that maintains multiple motion hypotheses, is capable of handling non-Gaussian probability distributions and recovering tracks that have been interrupted by occlusion. It has been researched extensively in the last decade and a range of refinements and levels of sophistication have emerged [2, 3]. Examples illustrating the variety of applications of the filter include broadcast sport video sequences tracking players using shape and colour [4], occupational therapy research tracking the motion of persons descending a stairwell [5], multi-feature tracking of faces and pedestrians [6], colour based tracking of pedestrians and soccer players [7], and the classification of commuters in a railway station [8].

The motivation for the work described in this paper was linked to the development of analytics components to run as camera embedded software using

*Corresponding author
Email addresses: pdunne@uclan.ac.uk (Peter Dunne), bmatuszewski1@uclan.ac.uk (Bogdan Matuszewski)

the Chipwrights range of single instruction multiple data fixed point visual signal processors [9]. The processors have a primary role of dealing with image capture, intensity correction, compression, sensor multiplexing etc. but there is spare processing capacity for additional tasks.

The expected CCTV applications included situations in which the colour information might be weak [10, 11] e.g. with people with dark or plain clothing in poorly illuminated indoor public spaces or in outdoor environments under sodium or infra-red lighting etc. We reasoned that approaches shown to work with grey scale would be transferable to polychromatic systems, but ones developed to work with colour might not transfer to monochrome very well. The additional need for computational economy, and a commercial requirement of backward compatibility with existing monochrome installations, led us to consider working with only the incoming luminance channel of the image data and use grey scale histograms for object characterization.

There have been suggestions that such histograms might not be adequate for the task [12]. Our preliminary investigations confirmed the findings that the grey-scale histogram approach produced tracks that did not correlate closely with the frame-to-frame mean squared difference matching of the target regions. The investigations suggested, however, that grey-scale signature tracking produced target correspondence adequate for the surveillance type applications and was worth exploring further.

We re-examined the relationship between types of histogram comparison dissimilarity measure in order to determine the one most appropriate for the application. Recognizing that the variance of the standard particle filter likelihood function is generally taken to be a design parameter we reviewed choices of its value and developed an argument that upper and lower limits can be put on that choice.

We carried out experiments to show not only that adequate grey scale tracking could be achieved but that a combination of a simple likelihood function and histogram comparison measure can perform just as well as the more traditional Gaussian/Bhattacharyya combination.

We begin, in section 2, by reviewing the basic theoretical structure of the particle filter. In section 3 we consider a selection of examples from published work to illustrate the range of parameter values used with typical dissimilarity measures and likelihood functions. This sets the context for the analysis to follow. In sections 4 and 5 the nature of the Bhattacharyya distance measure and its role in the Gaussian likelihood function is considered. Distance measures other than the Bhattacharyya distance are briefly considered in section 6, but the main focus of that section is the relationship between Bhattacharyya distance and the less commonly used histogram intersection distance. The discussion links the Bhattacharyya distance to the simpler sum of absolute difference based L_1 distance and other proposed distance measures. Section 7 looks at the use of a simple triangular likelihood function in place of a Gaussian. The experimental investigations into the use of the simplified approaches in place of the standard approaches are described and discussed in section 8 and prospects for computational economy are considered in section 9. The findings are discussed in section 10 and conclusions are drawn in section 11.

2. The particle filter

The particle filter is based upon the use of a possibly non-linear state transition model subject to additive noise ν_k, in which a current state $\mathbf{x}_k$ depends on the immediately preceding state $\mathbf{x}_{k-1}$:

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \nu_k) \tag{1}$$

and the use of a measurement model:

$$\mathbf{z}_k = h(\mathbf{x}_k, n_k) \tag{2}$$

in which a current measurement is taken to be a possibly non-linear function of the current state with measurement noise n_k.

The objective of the Bayesian tracker is to recursively estimate the state $\mathbf{x}_k$ of each tracked object by constructing the posterior probability density function (pdf) $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ where $\mathbf{z}_{1:k}$ is a sequence of measurements from k time steps. The particle filter approach approximates the pdf with a set of possible states$\{\mathbf{x}_k^i\}_{i=1}^{Ns}$, or 'particles', and associated weights $\{\omega_k^i\}_{i=1}^{Ns}$ such that the posterior pdf can be approximated as:

$$p\left(\mathbf{x}_k|\mathbf{z}_{1:k}\right) \approx \sum_{i=1}^{N_s} \omega_k^i \delta\left(\mathbf{x}_k - \mathbf{x}_k^i\right) \tag{3}$$

where $\delta(.)$ is the Dirac delta function.

The recursion element occurs in the sequential update of the particle weights as the pdf develops from p$(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ at time k-1 to p$(\mathbf{x}_k|\mathbf{z}_{1:k})$ at time k. It can be shown [3] that:

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{z}_{1:k})} \tag{4}$$

where $q(.)$ is an importance density from which samples $\left\{\mathbf{x}_k^i\right\}_{i=1}^{Ns}$ are drawn.

If the importance density is taken to be the transition prior $p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)$ then the update equation reduces to:

$$w_k^i \propto w_{k-1}^i p(\mathbf{z}_k|\mathbf{x}_k^i) \tag{5}$$

If the proposals are re-sampled according to their original normalized weight at k-1 then w_{k-1}^i is reset to $\frac{1}{N_s}$, where N_s is the number of samples, and the proportionality becomes:

$$w_k^i \propto p(\mathbf{z}_k|\mathbf{x}_k^i) \tag{6}$$

where $p(\mathbf{z}_k|\mathbf{x}_k^i)$ is a measurement based likelihood function.

In histogram based particle filter trackers it is common to assume that the distribution of histogram dissimilarity measures D is Gaussian. The Gaussian measurement likelihood function has the form:

$$p(\mathbf{z}_k|\mathbf{x}_k^i) \propto \exp\left(-\frac{{D_i}^2}{2\sigma^2}\right) \tag{7}$$

where σ is the standard deviation of the distribution of distance measures. The measurement noise n_k is characterized by this distribution.

Once the likelihood of the states has been determined the final estimated state can be returned as either a weighted mean:

$$\mathrm{E}\left[\mathbf{x}_k|\mathbf{z}_{1:k}\right] \approx \sum_{i=1}^{N_s} \tilde{\omega}_k^i \mathbf{x}_k^i \tag{8}$$

where $\tilde{w}_k^i$ is the normalized weight:

$$\tilde{w}_k^i = \frac{w_k^i}{\sum\limits_{i=1}^{N_s} w_k^i} \tag{9}$$

or as a maximum a posteriori (MAP) estimate:

$$\mathbf{x}_k^{map} = \arg\max_{\mathbf{x}_k} p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \arg\max_{\mathbf{x}_k} w_k^{(i)} \tag{10}$$

although it has been argued that this maximum weight estimator might not be a fair approximation to the MAP [13].

A re-sampling process ensures that particles with high weights are preferentially reproduced and propagated to the next recursion step at the expense of those with low weights. The process is carried out in order to prevent the degeneration of the particle set into one in which only a small proportion of the particles end up carrying most of the weight.

The behaviour of the particle filter is determined by the choices of the state transition function (1), the magnitude of its associated process noise ν_k, and in the case of a Gaussian likelihood (7) the choice of distance measure D and the likelihood standard deviation σ.

3. Choice of Gaussian likelihood standard deviation in published work

There has been a good range of high quality work relating to colour and grey-scale intensity or intensity gradient histogram based particle filter tracking during the last decade. The use of a Gaussian likelihood measure (7) has become standard and use of the Bhattacharyya distance [14] as the dissimilarity measure D is common, although other measures are used. The Bhattacharyya distance returns a value in the range [0,1].

As the numerical value of the distribution standard deviation σ is generally taken to be a tuneable design parameter we can expect to see a range of reported values dependent on the choice and range of the distance measure. Out of a large body of work in the field we have selected some representative examples in order to make the point. A recent approach [7], using the Bhattacharyya distance measure reports using $\sigma = 0.8$ in one example and $\sigma = 0.6$ in others. Another [6], using a multiple feature approach and the Bhattacharyya distance, reports using $\sigma = 0.09$ for the colour component.

Early examples of Bhattacharyya distance colour based particle filter tracking [15, 16, 17] expressed the likelihood Eq.(7) in the form $\exp(-\lambda D^2)$ with $\lambda = 20$. This corresponds to a value of $\sigma = 0.16$. A more recent example [5] uses $\lambda = 50$ to give $\sigma = 0.1$. Lu et al. [4] use a HOG distance measure in a Gaussian likelihood with $\lambda = 10$ to give $\sigma = 0.22$.

In a slightly different approach, [18] uses $\exp(-(1-\alpha)D^2)$, with $\alpha = 0.6$ giving $\sigma = 1.12$. It is common, however, to see reported work in which numerical values for the likelihood parameters are not given [19, 20, 21, 22, 23].

Considering the reported variation of Gaussian likelihood parameters in the context of colour/grey-scale histogram based tracking involving the Bhattacharyya distance, and the apparently crucial choice of the parameters as design components, it is worthwhile re-examining the role of the likelihood standard deviation and its relationship to the distance measure in order to get better guidance on the choice of design values.

4. The Bhattacharyya coefficient and distance

If two n-bin histograms are represented as $\mathbf{p} = \{p_u\}_{u=1\ldots n}$ and $\mathbf{q} = \{q_u\}_{u=1\ldots n}$, and normalized such that $\sum_{u=1}^{n} p_u = 1$ and $\sum_{u=1}^{n} q_u = 1$, then the Bhattacharyya *coefficient* is defined as $\rho[\mathbf{p},\mathbf{q}] = \sum_{u=1}^{n} \sqrt{p_u}\sqrt{q_u}$. It has a geometrical interpretation [24] as the cosine of the angle between the n-dimensional unit vectors $\left(\sqrt{p_1}, \ldots.., \sqrt{p_n}\right)^{\mathrm{T}}$ and $\left(\sqrt{q_1}, \ldots.., \sqrt{q_n}\right)^{\mathrm{T}}$.

The Bhattacharyya *distance* is a dissimilarity measure. It is defined [14] to be $B = \sqrt{1-\rho[\mathbf{p},\mathbf{q}]}$. This formulation ensures that the distance between two identical histograms has $B = 0$ and dissimilar histograms have values $0 < B \leq 1$.

The upper limit of 1 (rather than $\sqrt{2}$) occurs because the histogram bin values cannot be negative. This means that the unit vectors will have no negative components, the maximum angle between the two unit vectors will be $90deg$ and hence $B_{\max} = \sqrt{1-0}$.

5. Choice of the likelihood standard deviation for particle weighting and resampling

The particle filter state transition equation (1) and its associated noise $\nu_{\mathbf{t}}$ propagates the particle representation of the probability density function from the last frame into the current one.

In normal operation a proportion of the particles will end up in the region of the target position with the remainder spread into places that the target might reach with changes of speed and direction. In the ideal case this would return a set of particle distance measures spread across the full range [0,1]. In practice, with occlusion or changes in scene illumination, the distribution of particle distances can cover less than the full range and show either a positive or negative skew. A typical particle distance frequency distribution from real data is shown in figure 1. The Gaussian weighting for that set, using a likelihood standard deviation of $\sigma = 0.2$, together with curves for $\sigma = 0.1$ and 0.3, is also shown.

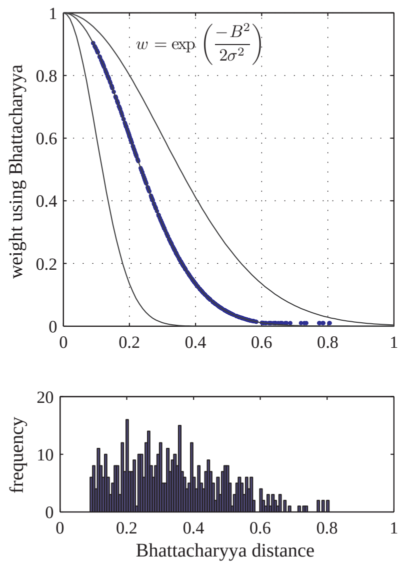


Figure 1: Gaussian weighting of the Bhattacharyya distance measure for a typical set of 512 particles from a tracking sequence. The standard deviation of the distribution is set to $\sigma = 0.2$. The curves to the left and right of the central one have standard deviations of $\sigma = 0.1$ and $\sigma = 0.3$ respectively. The lower figure shows the frequency distribution of the distances

The particle weighting has two functions: it provides values for the estimate of the object state through the weighted mean, and it provides feedback for the evolution of the particle set. The process of re-sampling with replacement from the weighted particle set removes particles that are not very representative of the target state and produces multiple copies of those that are more representative. The weighting of the particle set changes the effective size of the set as unrepresentative particles end up making diminished contributions to the calculations at each stage. The effective particle number can be estimated [3, 25] using:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_s} (\tilde{w}_k^i)^2} \tag{11}$$

where $\tilde{w}_k^i$ is the normalized weight of the ith particle at the kth step as defined in (9), and N_s is the number of particles.

In some particle filters the effective particle number is used to trigger re-sampling if N_{eff} falls below a given threshold. For example Doucet et al. [25] used a threshold of $N_s/3$ in their early simulations. It is common, however, for re-sampling to be implemented at each iteration irrespective of the value of N_{eff}. Even if it is not used to trigger re-sampling N_{eff} can be used to give an indication of the degree of degeneracy of the filter.

We can relate the effective particle number to the standard deviation of the Gaussian likelihood weighting function (see Appendix A) for the ideal case in which the dissimilarity measures are uniformly spread across the full [0,1] range. We start with $w^i = \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right)$ where l_i has N_s uniformly incremented discrete values from 0 to 1. This leads to the normalized weight of:

$$w^i = \frac{2}{N_s\sigma\sqrt{2\pi}} \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right) \tag{12}$$

Using this value for the weight in equation (11) leads to

$$N_{eff} = \sqrt{\pi}\sigma N_s \tag{13}$$

for $\sigma \leq 0.3$.

For $\sigma = \{0.1, 0.2, 0.3\}$ we get $\sqrt{\pi}\sigma = \{0.18, 0.35, 0.53\}$, producing N_{eff} of approximately one fifth, one third and one half respectively of the particle set.

It is also useful to refer to the Bhattacharyya coefficient and consider the angle between the target vector and vectors from the particle set. For $\sigma = \{0.1, 0.2, 0.3\}$ the 3σ Bhattacharyya distances correspond approximately to angles between feature vectors of $\theta = \{25, 50, 79\}deg$ respectively. For a uniform distribution of distances, using a Gaussian with $\sigma = 0.1$ will result in an effective particle set of size $N_{eff} \sim N_s/5$ focusing on feature vectors within $25deg$ of the target vector. This looks attractive at first sight but, in practice, distribution skew will mean that there will be many fewer than $N_s/5$ contributing. Re-sampling of such a diminished set is likely to result in loss of diversity amongst the particle population. We suggest that such a value of σ is likely to make the filter have difficulty tracking the target.

If we look at the other extreme with $\sigma = 0.3$ we are likely to have an effective sample size greater than $N_s/2$ but with significant contribution from feature vectors that are at angles up to $80deg$ from the target vector. If N_s is small then the large angle feature vectors could have too great an undesirable influence on the weighted mean and there is a significant chance of the particle cloud diverging.

The argument points to the suggestion that $\sigma = 0.2$ is a compromise value. This will maintain a particle set consisting of feature vectors within $50deg$ of the target vector, it will produce an effective sample size of $N_s/3$, and drawing from 60% of the distance axis it will be reasonably tolerant to skew of the weight frequency distribution in the particle set. Our experimental work, reported in section 8, supports this choice. It is interesting to note that Doucet et al. [25] opted for an N_{eff} threshold of $N_s/3$ to trigger re-sampling in their particle filter simulations.

6. Distance measures other than Bhattacharyya

A general class of measures for distances between d-dimensional real valued vectors $\mathbf{a}, \mathbf{b}$, referred to as the L_r norm, has the form [26]:

$$L_r(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{n} |a_i - b_i|^r\right)^{1/r} \tag{14}$$

As we are using the normalized histograms $\mathbf{p}, \mathbf{q}$ as the feature vectors the vector components have to be the square roots of the bin values for the vectors to have unit length, hence we expect to see the equivalent L_r norm in the form:

$$L_r(\mathbf{p}, \mathbf{q}) = \left(\sum_{i=1}^{n} |\sqrt{p_i} - \sqrt{q_i}|^r\right)^{1/r} \tag{15}$$

The Matusita distance between two probability distributions $\mathbf{p} = (p_\mathbf{1}, .., p_\mathbf{n})^\mathbf{T}$ and $\mathbf{q} = (q_\mathbf{1}, .., q_\mathbf{n})^\mathbf{T}$ is defined as [27]

$$M(\mathbf{p}, \mathbf{q}) = \left(\sum_{i=1}^{n} (\sqrt{p_i} - \sqrt{q_i})^2\right)^{1/2} \tag{16}$$

and hence can be seen as an L_2 norm.

The Matusita distance is directly related to the Bhattacharyya distance. The relationship is discussed in [24] although the authors state the Matusita distance as the square of the original definition in [27]. We work with the original definition. By expanding the square under the summation in (16) and gathering the terms it can be shown that $B = M/\sqrt{2}$. The link between the Bhattacharyya *coefficient* and the Matusita distance can be seen in terms of the triangle cosine rule: the Bhattacharyya *coefficient* is the cosine of the angle between the vectors, the Matusita distance is the magnitude of the difference between the vectors, the Bhattacharyya *distance* is then a simple fraction of that distance.

We can use the Matusita distance to bring out a relationship between the Bhattacharyya distance, the L_1 norm, and the less often used histogram intersection distance. Histogram intersection is defined as [28]:

$$\cap(\mathbf{p},\mathbf{q}) = \frac{\sum_{u=1}^{n} \min(p_u, q_u)}{\sum_{u=1}^{n} q_u} \tag{17}$$

The measure calculates the commonality (the intersection) between the two histograms $\mathbf{p}, \mathbf{q}$. The measure is normalized by the number of pixels in the target histogram in order to obtain a fractional match value between 0 and 1. In our case of normalized histograms the denominator in (17) is unity. For a perfect match between normalized histograms $\cap(\mathbf{p},\mathbf{q}) = 1$. To get a dissimilarity measure we take the complement of the intersection to get the histogram intersection distance H:

$$H = 1 - \cap(\mathbf{p},\mathbf{q}) \tag{18}$$

It can be shown [28] that for normalized histograms the intersection distance H is linked to an L_1 type distance formed using the simple sum of absolute differences of the bin values:

$$1 - \cap(\mathbf{p},\mathbf{q}) = \tfrac{1}{2}\sum_{u=1}^{n} |p_u - q_u| = \tfrac{1}{2}L_1 \tag{19}$$

Intersection has been used, for example, in the context of logo identification in video sequences [29], spatial histograms for region based tracking [30], via the L_1 form in 'Bag-of-Features' face matching [31] and in multi-segment histogram matching for particle filter tracking of crossing targets [32].

We link the Bhattacharyya and histogram intersection distances by comparing the square of the Matusita distance and this binwise absolute difference form of the intersection:

$$\begin{aligned} M^2 &= \sum_{u=1}^{n} \left(\sqrt{p_u} - \sqrt{q_u}\right)^2 \\ &= \sum_{u=1}^{n} \left(\sqrt{p_u} - \sqrt{q_u}\right)\left(\sqrt{p_u} - \sqrt{q_u}\right) \qquad (20) \\ L_1 &= \sum_{u=1}^{n} |p_u - q_u| = \sum_{u=1}^{n} |\sqrt{p_u}^2 - \sqrt{q_u}^2| \\ &= \sum_{u=1}^{n} |\left(\sqrt{p_u} - \sqrt{q_u}\right)| \left(\sqrt{p_u} + \sqrt{q_u}\right) \qquad (21) \end{aligned}$$

We compare equations (20) and (21). For a given set of histogram bin differences as indicated by a single Bhattacharyya distance associated with $\left(\sqrt{p} - \sqrt{q}\right)$, there will be a range of L_1 values determined by alternative combinations of component bin values ($0 \le p \le 1, 0 \le q \le 1$), that give rise to a set of given differences. The alternative combinations supply the factor $\left(\sqrt{p} + \sqrt{q}\right)$ in (21)

Figure 2 shows a comparison of the calculated histogram intersection distance measure (H) against the Bhattacharyya distance (B) for a set of 512 particles from a typical particle filter step. It can be seen that the scatter of the H measures about the line $H = B$ is reasonably compact. An understanding of the form of the scatter can be found in Appendix B.

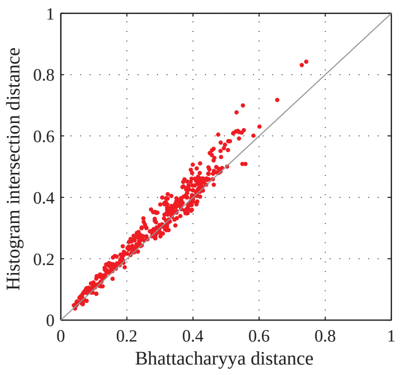


Figure 2: Illustration of the scatter of histogram intersection distance values plotted against the Bhattacharyya distance for a typical set of particles

Figure 3 shows the effect of using the H measure with a Gaussian likelihood function with $\sigma = 0.2$. The Bhattacharyya distance is plotted on the horizontal axis and the weight associated with the corresponding H distance is plotted on the vertical axis.

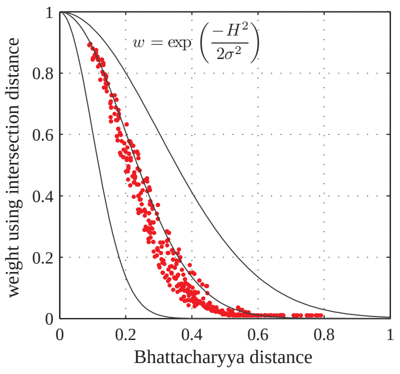


Figure 3: Illustration of the effect of using the H distance with a Gaussian likelihood function, $\sigma = 0.2$.

It can be seen that the degree of scatter shown in figure 2 translates to a scatter about the chosen Gaussian and is within the limits associated with $\sigma = 0.1$ and $\sigma = 0.3$.

This analysis suggests that if we take into account statistical averaging associated with using a weighted mean, or *pdf* mode finding based upon weighted particle densities, the use of the computationally simpler histogram intersection distance might produce outcomes that are not much different from those obtained using the Bhattacharyya distance.

The χ^2 measure has also been used as a distance measure to compare histograms [8, 33]. For our histograms $\mathbf{p}, \mathbf{q}$ the measure would be [34]:

$$\chi^2(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} \frac{(p_i - q_i)^2}{p_i + q_i} \tag{22}$$

We can analyze the relationship between this measure and the Bhattacharyya distance using an approach like that for the histogram intersection distance. The numerator in (22) expands in a way similar to that in (21) except that the factors are squared:

$$\chi^2(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} \frac{\left(\sqrt{p_i} - \sqrt{q_i}\right)^2 \left(\sqrt{p_i} + \sqrt{q_i}\right)^2}{p_i + q_i} \tag{23}$$

The first factor in the numerator is similar to the Matusita component of the sum. By combining values of $0 \leq p \leq 1$ and $0 \leq q \leq 1$ it can be shown that the factor $\left(\sqrt{p} + \sqrt{q}\right)^2 / (p + q)$ varies between 1 and 2 with most of the values at the upper end of that range. The argument is presented in Appendix C. The factor results in values of χ^2 being greater than the corresponding Bhattacharyya values and having a scatter similar to that in the intersection case. The effect, using experimental particle data, is shown in figure 4.

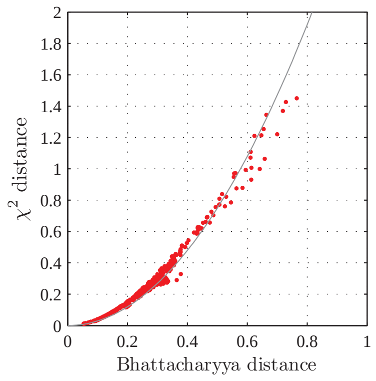


Figure 4: Comparison between χ^2 distances corresponding to Bhattacharyya distances for a typical set of particles from a tracking sequence

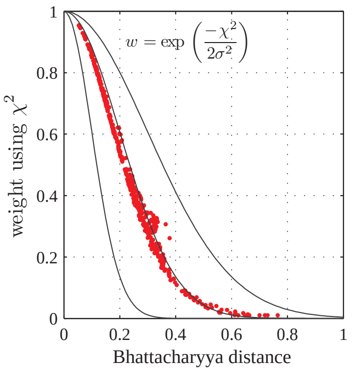


Figure 5: Scattered distribution of χ^2 distance weightings with a Gaussian likelihood, using $\sigma = 0.35$

The curvature associated with the plot in figure 4 is due to the fact that χ^2 is linked to the square of the Matusita distance in (23). An example plot of χ^2 vs B^2 is given in figure C.12 in Appendix C. Experiment suggests that the functional relationship between the χ^2 values and the Bhattacharyya values can be approximated by $\chi^2 = 3B^2$.

If χ^2 is substituted for B^2 in the Gaussian likelihood then a standard deviation of $\sigma = 0.35 \approx \sqrt{3} \times 0.2$ should produce a distribution similar to one with a Bhattacharyya distance and standard deviation of $\sigma = 0.2$. A plot of such a χ^2 likelihood weighting is shown in figure 5. The Bhattacharyya distances are plotted as abscissae and the weights produced using χ^2 with $\sigma = 0.35$ are plotted as ordinates. The scatter is less than in the histogram intersection distance case but the computational demand is a little greater.

We can conclude from this analysis that for the purposes of estimation there is little to gain in choosing between the Bhattacharyya, Matusita and χ^2 distances. The first two differ only by a simple numerical factor, the first and the third differ approximately by such a factor. Their differences in the context of particle filter calculations can be compensated for by adjusting the value of the standard deviation in the likelihood function. The histogram intersection distance, however, offers potential computational economy through avoidance of the need to extract square roots. It does produce a scatter of likelihood weight values but they are generally within a range corresponding to the use of Bhattacharyya distances and a Gaussian likelihood with standard deviations between the useful limits of $\sigma = 0.1$ to $\sigma = 0.3$. As this standard deviation range would produce an acceptable reweighting, dependent upon the prior spread of the distance values, we might expect that given sufficient particles distributed reasonably symmetrically around the target location a position estimate would be returned not far from one calculated using the Bhattacharyya distance with a standard deviation of $\sigma = 0.2$. The behaviour of the histogram intersection distance suggests that it might produce particle filter tracking that is not much different from that produced with the Bhattacharyya distance.

7. Likelihood functions other than Gaussian

The basic role of the likelihood function is to give preferential weight to proposed states that are close to the target state and to suppress proposals that are far from the target state. We suggest that the choice of a Gaussian likelihood function is due to the common assumption of normally distributed measurement error. It is of interest to note that the appropriateness of the underlying Gaussian error model has been questioned by some authors [35, 36, 37] who point out that a heavier tailed Cauchy distribution might be more appropriate in some situations. An example of the use of a Cauchy distribution in the context of particle filter tracking is given in [38].

Some particle filter applications approximate the behaviour of such a distribution by adding a constant tail to the likelihood function [39]. The role of the 'tail' is to maintain the distribution through re-sampling in the event that the target object is not observed in an image. In such cases the distance measure would be large and the corresponding weights would be near zero. Allowing a small but non-zero weight for large distances ensures survival of the particle set at the re-sampling stage.

Whilst it might indeed be the case that measurement error is best described by either a Gaussian or a Cauchy distribution, we propose that there is nothing particularly special about the form of the particle filter likelihood function other than it must maintain the positive evolution of the particle set by preferentially selecting low distance vectors and suppressing vectors with a large distance.

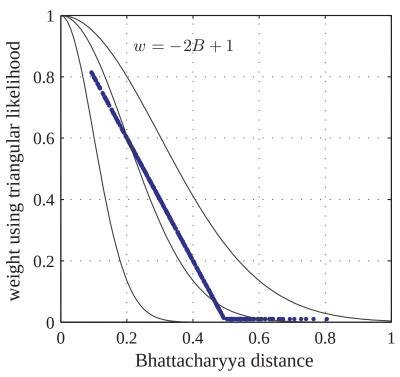


Figure 6: Weighting a Bhattacharyya distance function using a triangular likelihood given by $w = -2B + 1$ for $B < 0.5$, $w = 0.01$ for $B \geq 0.5$

In practice a Gaussian weighting is likely to be implemented via a look-up table. In order to illustrate the non-critical form of the likelihood function we propose that a simple triangular function of the form $w = -2B + 1$ might be used as a selection device. An example of such a triangular likelihood with a constant tail is shown in figure 6. In this case the function drops to zero at $B = 0.5$ and the tail returns the weight $w = 0.01$ for $B \geq 0.5$.

When the histogram intersection distance is used in place of the Bhattacharyya distance the associated scatter re-distributes the weights in a similar way to that seen in figure 3. The effect is shown in figure 7. The combination of histogram intersection distance and the simple triangular likelihood function produces a scatter of weightings that are not much different to those of the Gaussian/intersection combination seen in figure 3.

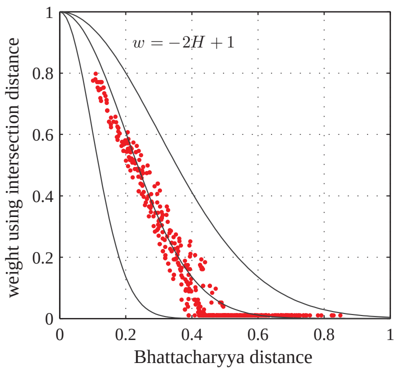


Figure 7: Weighting a histogram intersection distance function using a triangular likelihood given by $w = -2H + 1$ for $H < 0.5$, $w = 0.01$ for $H \geq 0.5$. Bhattacharyya distance is plotted on the horizontal axis but the weight is calculated using the H value corresponding to that Bhattacharyya distance.

8. Experiments

We explored the proposal that the analytical form of the likelihood function is not critical in the context of grey scale histogram based particle filter tracking by comparing tracks produced using the standard Bhattacharyya distance/Gaussian likelihood combination against ones produced using the simple histogram intersection distance/triangular likelihood combination. We chose to use the histogram intersection distance rather than χ^2 because whilst the scatter of distance measures appeared to have similar consequences for each of the approaches the intersection distance was computationally more attractive.

Targets were tracked using an elementary SIR particle filter [3]. Static background elements were suppressed by the use of a temporal frame difference based mask. We worked with four image sequences each of which presented its own challenges to the tracker.

The first image sequence consisted of an overhead view of shoppers walking through a mall. The appearance of the shoppers changes significantly as they move from the bottom of the scene to the top due to a mixture of object deformability, viewing angle and varying lighting conditions.

The second sequence is a daytime outdoor oblique view of pedestrians. The pedestrians often occlude each other. In the example shown the tracked pedestrian passes partly behind two foreground pedestrians having a different grey scale signature and then completely behind one with a similar signature.

The third was an outdoor view of traffic taken from the AVSS 2007 i-LIDS Vehicle Challenge data set [40]. In this sequence there is some slight camera movement and illumination changes.

The fourth sequence is taken from the PETS 2006 data set [41]. It is an indoor scene and the experiment tracks a pedestrian passing close to others of similar appearance.

In each of the sequences a selected object was tracked 100 times using the Gaussian likelihood/Bhattacharyya distance combination and then 100 times using the alternative combination. The frame rates were set to 5fps. The mean and variance of the 100 measurements of each track point position was determined for each combination. The mean track from the first likelihood/distance combination was then compared with the second. Investigations were carried out using smaller and larger statistical sample sizes (between 5 and 1000 track sequences) but the outcomes were only significantly different, as expected, for small sample sizes (e.g. < 10 measured tracks). The behaviour of most of the targets in each sequence was investigated and the outcomes for the ones reported here were typical of all.

The target starting position was manually initialized in order to ensure comparability of the resulting tracks. A single rectangular region with dimensions half those of the bounding box of the target object was tracked using a normalized 8-bin greyscale histogram. For example, the tracking rectangle seen in figure 8(a) had dimensions 17×17 pixels.

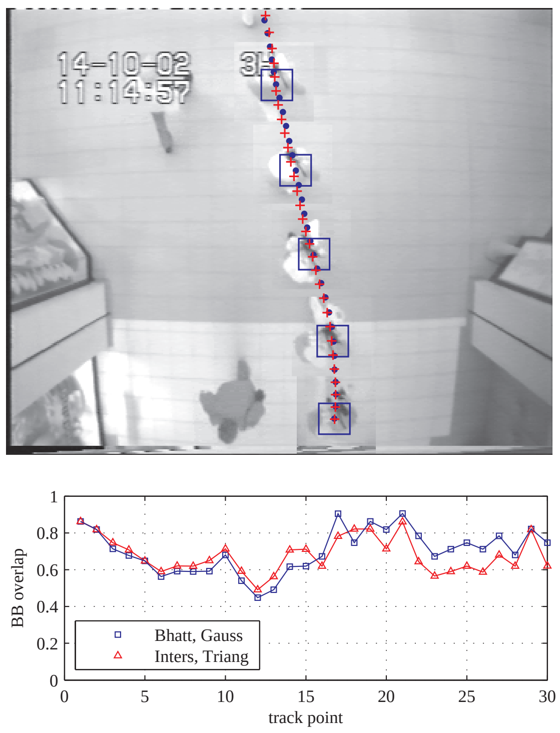
The histograms were extracted using the integral histogram method [42].

A first order state transition step was used with the state vector $\mathbf{x} = [x\ \dot{x}\ y\ \dot{y}]^{\mathrm{T}}$. The number of particles used was $N_s = 512$.

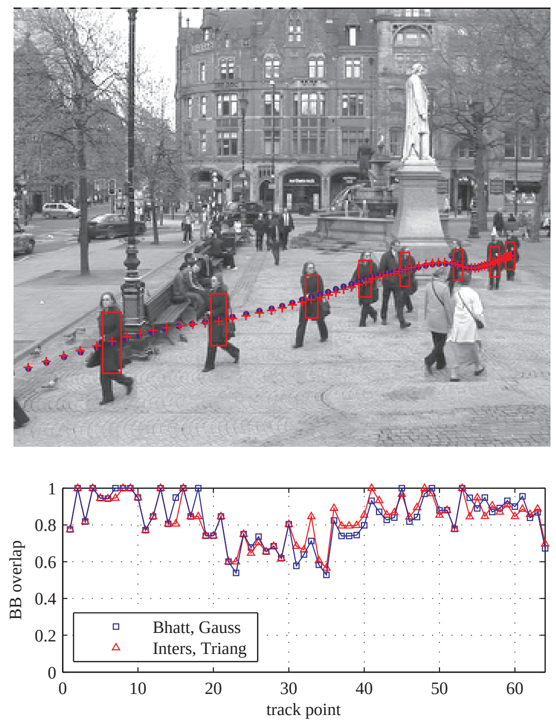
In the cases where the view was at an oblique angle the size of the rectangle was adjusted using a linear function of the target position in the image in order to retain its relative size in proportion to the size of the target.

The standard deviation of the additive Gaussian process noise (ν_t in eq. (1)) was set to be $0.2 \times$ the length of the side of the tracking rectangle for both the spatial and velocity components of the state vector. This choice produced a prior particle spread appropriate to the typical velocities of the targets in the sequence.

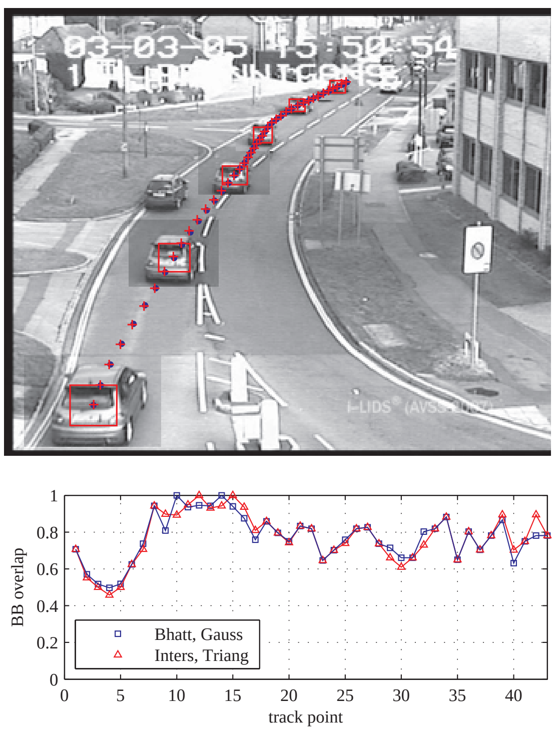
The template histogram $\mathbf{q}$ was updated at each frame using $\mathbf{q}_t = (1-\alpha)\mathbf{q}_{t-1} + \alpha\bar{\mathbf{p}}$ where $\bar{\mathbf{p}}$ was the



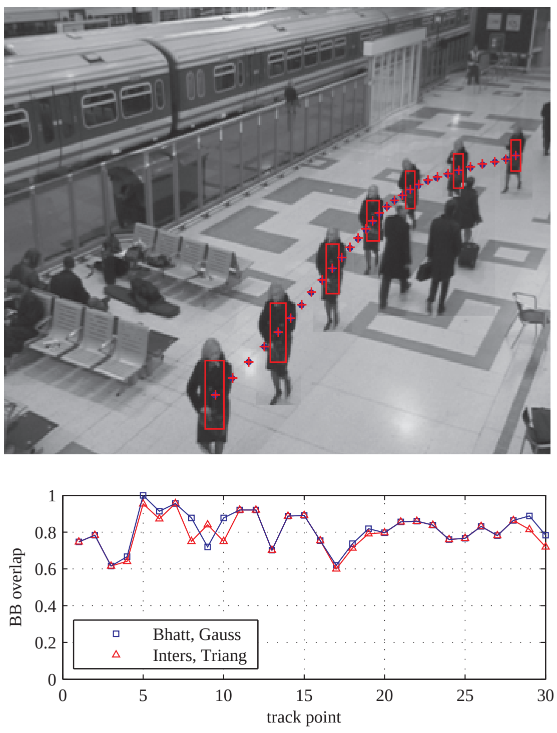
(a) Overhead



(b) Pedestrians



(c) i-LIDS



(d) PETS

Figure 8: Composite images and ground truth to tracker bounding box overlap data for the sequences. Blue (·) points - Gaussian likelihood\Bhattacharyya distance, red (+) points - triangular likelihood\histogram intersection distance

histogram extracted at the current target position. The value of α was set at 0.2 for all sequences.

The measurement noise, i.e. the standard deviation of the likelihood Gaussian function, was kept constant at $\sigma = 0.2$. The likelihood distributions were given a 3σ 'tail' value of 0.01 for all likelihood values that were initially below that value. The posterior particle set was updated at each step using standard stratified re-sampling.

Track smoothing was carried out by adjusting the position components of $\mathbf{x}_{t-1}$ and $\mathbf{x}_{t-2}$ at each time step t by a recursive procedure that consisted of deriving and averaging velocities from measured positions at t-3 to t to produce an adjusted $\hat{\mathbf{x}}_{t-2}$ and $\hat{\mathbf{x}}_{t-1}$. The resulting comparison of the tracking approaches can be seen in figure 8. Both the Bhattacharyya/Gaussian and intersection distance/triangular tracks are shown for each sequence. Snapshots of the targets together with occluding objects at periodic track points have been overlaid so that the tracks can be seen in context.

The correspondence of the tracks with the geometrical ground truth of the tracked objects was investigated. For each sequence the ground truth was determined manually in each frame as the centroid of the bounding box of the objects. The tracks for each of the two approaches were compared to the ground truth using the bounding box spatial overlap [43], defined as the overlapping area $A(GT_k, ST_k)$ between the ground truth bounding box GT and the tracking system bounding box ST in frame k:

$$A(GT_k, ST_k) = \frac{Area(GT_k \cap ST_k)}{Area(GT_k \cup ST_k)} \qquad (24)$$

This measure returns a value of 1 when the track corresponds exactly with the ground truth and 0 when bounding boxes do not overlap at all. The overlaps for each approach are plotted beneath the relevant track images in figure 8. It can be seen that in general the overlap is between 0.6 and 1.0 giving sufficient correspondence for tracking in surveillance contexts. It can also be seen that the track points associated with each of the two approaches are closer to each other than to the geometrical ground truth. It is to be expected that mean position of the tracked grey scale feature would not necessarily correspond to the geometrical centroid of the tracked object due to a combination of object deformation, perspective and shading effects. The closeness of the overlap for the two tracking approaches indicates that there is no significant difference between them in terms of the tracks returned.

9. Computational economy

The differences in computational economy between the approaches described above come down to the calculation of $w = \exp\left(-\frac{D^2}{2\sigma^2}\right)$ with $D^2 = 1 - \sum_1^n \sqrt{pq}$ in the Bhattacharyya case, $D^2 = \sum_1^8 \left(\sqrt{p} - \sqrt{q}\right)^2$ in the Matusita case; or a calculation of $w = -2D + 1$ with $D = 0.5 \sum_1^n |p - q|$ in the histogram intersection case.

Irrespective of the particular hardware architecture involved, it is likely that the number of processor operations involved in the calculation of the exponential term is likely to be greater than in the linear case: the linear calculation can be carried out with one left shift and an addition. We compared the impact of the distance measures D using a Chipwrights CW5521 fixed point development simulator running at 300MHz [9] to carry out each distance calculation for a fixed number of times. For 10000 8 bin histogram calculations the times were 400ms for the Matusita calculation, 153 ms for Bhattacharyya and 40ms for the intersection distance. Similarly, 10000 32 bin calculations took 1780, 614 and 166ms respectively. The histogram intersection distance approach ran between 3 and 4 times faster than the Bhattacharyya one, and 10 times faster than the Matusita one. The combination of this result with the expectation of fewer processor operations for the linear calculation compared to the Gaussian one leads us to suggest a potential economy of time and resources (e.g. processing power, memory required for a fast look-up table implementation of the Gaussian likelihood function etc.) if the simpler combination is used. Such economy is valuable in the context of real-time embedded processing in a multitasking environment.

10. Discussion

The overall aim was to re-examine the components of a basic particle filter with the aim of finding a suitable low complexity approach that would work adequately with grey scale video. The findings of Khalid et al.[12] might be interpreted as suggesting that the intensity information in grey scale images is not a sufficient feature for Bhattacharyya distance based tracking; we have produced a reasoned demonstration that adequate grey scale tracking with a particle filter can be achieved. We agree that the filter does not track the minimum mean-squared difference position of the target object, but it does track the grey scale signature well enough for surveillance type applications.

A review of distance measures and their interrelationships satisfied us that the Bhattacharyya distance and the related Matusita distance had a clear interpretation in terms of the magnitude of the difference between histogram based vectors in a Euclidean feature space. In comparison to these measures others, like the histogram intersection distance and χ^2, had a less clear fundamental interpretation in this context and produced a scattering of distances in comparison to the Bhattacharyya distances.

Following our understanding of effects of choosing a particular Gaussian standard deviation and a normalized distance measure we proposed that the use of the potentially computationally advantageous histogram intersection distance, combined with a triangular likelihood function, might produce results comparable to those obtained using a Bhattacharyya measure/Gaussian likelihood combination. The application of our proposal was supported by our experimental findings. In general the mean track points for the two approaches stayed well within the boundaries of the objects being tracked. It is true that they deviated from the geometrical centroids of the objects but the results were still within acceptable limits for the purpose. A comparison of the worst case differences, in the overhead sequence tracks, is shown in figure 9

We argued that a well conditioned Gaussian likelihood particle filter should have a process stage that returns distance values across the full range [0,1]. For such a case we can use equation (13) to relate the effective particle number N_{eff} to the Gaussian standard deviation. We interpret the broad range of likelihood standard deviations, discussed in section 3, as indicating choices of process noise parameters that delivered ranges of similarity distance appropriate for particle re-weighting and resampling in those cases. We suggest, however, that a Gaussian likelihood particle filter being operated with normalized distance measures and values of $\sigma < 0.1$ is likely to have a process stage consistently returning a high proportion of distances in the lower end of the [0,1] range in order to maintain a reasonable experimental value of N_{eff}. This will not be spreading the prior particles sufficiently to accommodate extreme behaviour of the target. A particle filter being operated with normalized distances and values of $\sigma > 0.3$ might be returning distance measures across the full range, with $> 50\%$ of the particles contributing to N_{eff}, but will allow undue influence from histogram feature vectors that are very different from the target vector. It is likely that it would end up spreading the particles too widely.

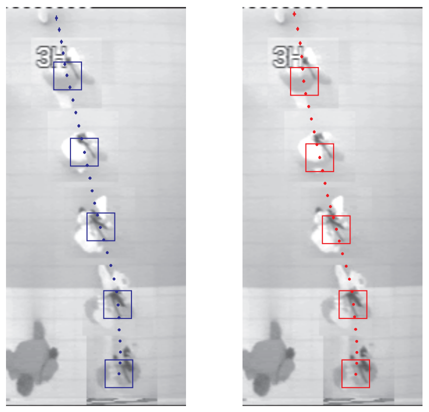


Figure 9: Illustrating the track differences for Gaussian/Bhattacharyya (left) and triangular/histogram intersection (right)

11. Conclusions and further work

This work was focused towards producing adequate tracking of objects using a single feature in a computational resource constrained environment. We have shown that particle filter approach adequate for surveillance purposes can be carried out using grey scale video. We examined the choice of feature distance measure and particle filter likelihood functions to see if mathematically simple approaches would be adequate for the task. Our analysis of the Bhattacharyya, Matusita and χ^2 distances showed that they can be expressed as simple numerical multiples of each other and there is no fundamental reason to prefer one over the others in this context.

With respect to the choice of likelihood function and parameters, a correctly operating particle filter process stage would place particles at the expected target positions in the next frame but it should also place a proportion of the prior in low probability regions in order to respond to extreme behaviour of the target. If the distance measure for such a distribution of particles is designed to return values spread across the full range [0,1] then for likelihood based resampling a function behaving like a Gaussian with a standard deviation of $\sigma \sim 0.2$ and a tail at the 3σ point should deliver adequate tracking. We have argued that a triangular function with a constant tail is adequate for the task.

The closeness of the experimental results obtained with the Bhattacharyya distance/Gaussian combination to those obtained with the histogram intersection distance/triangular likelihood combination suggests that the forms of the likelihood function and distance measure are not critical in the tracking situations examined. The simpler combination, however, offers potential computational economy in systems where resources might be constrained.

We regard this as a first step in reducing the number of tuneable elements in a particle filter for use in the stated context. Directions for further development include extending the analysis to the control of variables in polychromatic and multi-feature tracking. Colour histograms are often constructed as a concatenation of bins for each component and hence become vectors for use with a Bhattacharyya distance, the arguments above should still apply in such cases. Such an approach though does not model dependencies between different colour components represented in real scenes. It might be therefore of value to calculate separately the dissimilarity for each colour component and model explicitly the interdependence between them using a multi-variable likelihood function. It might be valuable to see if such approach would improve tracker performance operated on the polychromatic data. In both the colour case and that of multi-feature tracking (e.g.using combinations of intensity histograms, edge and/or shape information etc.) it would be of interest to explore the degree and impact of the resultant likelihood scatter associated with the use of the histogram intersection distance measure with triangular approximation likelihood function applied to each component.

12. Acknowledgments

The authors would like to thank Wei Wang, Richard Patel and Laurence Firth of AD Group for their helpful discussion and contributions in the development of this work. They would also like to acknowledge the insightful and positive feedback of the paper reviewers.

Appendix A. Relating effective sample size to the standard deviation of the Gaussian weighting function

We are using weights of $w_k^i = \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right)$ where l has N_s discrete values in the range [0,1]. We consider $\sigma \leq 0.3$ so that we can assume that the area under the Gaussian in the range [0,1] is approximately equal to the area in the range [0,∞].

The area under a Gaussian curve is given by:

$$\int_{-\infty}^{+\infty} \exp\left(\frac{-l^2}{2\sigma^2}\right) dl = \sigma\sqrt{2\pi} \tag{A.1}$$

which is recognizable as the standard normalizing coefficient for a Gaussian distribution.

For the likelihood weighting we are only interested in the positive half of a Gaussian curve with area given by:

$$\int_{0}^{+\infty} \exp\left(\frac{-l^2}{2\sigma^2}\right) dl = \frac{\sigma\sqrt{2\pi}}{2} \tag{A.2}$$

In the case of the discrete distribution of Gaussian weights an element of area dA is given by:

$$dA_i = w_k^i \cdot \frac{1}{N_s} \tag{A.3}$$

In the limit of large N_s, the area under the weight curve is given by:

$$\sum_{i=1}^{N_s} dA_i = \sum_{i=1}^{N_s} w_k^i \cdot \frac{1}{N_s} = \frac{\sigma\sqrt{2\pi}}{2} \tag{A.4}$$

so that:

$$\sum_{i=1}^{N_s} w_k^i = \frac{N_s\sigma\sqrt{2\pi}}{2} \tag{A.5}$$

The normalized weights are then given by:

$$\tilde{w}_k^i = \frac{2}{N_s\sigma\sqrt{2\pi}} \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right) \tag{A.6}$$

The particle filter effective weight is given by:

$$N_{eff} = \frac{1}{\sum\limits_{i=1}^{N_s} \left(\tilde{w}_k^i\right)^2} \tag{A.7}$$

We can write the denominator of (A.7) as:

$$\sum_{i=1}^{N_s} \left(\tilde{w}_k^i\right)^2 = \frac{4}{{N_s}^2 \sigma^2 2\pi} \sum_{i=1}^{N_s} \left(w_k^i\right)^2$$

If $w_k^i = \exp\left(-\frac{{l_i}^2}{2\sigma^2}\right)$ then $\left(w_k^i\right)^2 = \exp\left(-\frac{{l_i}^2}{\sigma^2}\right)$ Equating the area under the squared weight curve with that under the half Gaussian we get:

$$\int_0^{+\infty} \exp\left(\frac{-l^2}{\sigma^2}\right) dl = \frac{\sigma\sqrt{\pi}}{2} = \sum_{i=1}^{N_s} \left(w_k^i\right)^2 \cdot \frac{1}{N_s}$$

so that:

$$\frac{N_s \sigma \sqrt{\pi}}{2} = \sum_{i=1}^{N_s} \left(w_k^i\right)^2 \tag{A.8}$$

and the denominator of equation (A.7) becomes:

$$\sum_{i=1}^{N_s} \left(\tilde{w}_k^i\right)^2 = \frac{4}{{N_s}^2 \sigma^2 2\pi} \cdot \frac{N_s \sigma \sqrt{\pi}}{2}$$

leading to:

$$N_{eff} = \sqrt{\pi} \sigma N_s \tag{A.9}$$

Appendix B. Understanding the histogram intersection distance scatter

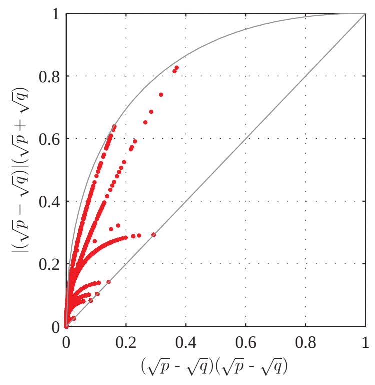


Figure B.10: Plot of histogram intersection distance components against the corresponding Bhattacharyya distance components for 512 x 8 $\{p, q\}$ bin pairs.

Figure B.10 shows the relationship between the histogram intersection type component plotted against the corresponding Bhattacharyya type component as described by equations (B.1) and (B.2) for 512 x 8 $\{p, q\}$ pairs taken from an experimental set of 512 normalized candidate 8-bin histograms $\mathbf{p}$ and the single normalized target 8-bin histogram $\mathbf{q}$.

$$2B^2 = M^2 = \sum_{u=1}^{8} \left(\sqrt{p_u} - \sqrt{q_u}\right)\left(\sqrt{p_u} - \sqrt{q_u}\right) \tag{B.1}$$

$$\begin{aligned} 1 - \mathbf{p} \cap \mathbf{q} &= \frac{1}{2} \sum_{u=1}^{8} |p_u - q_u| \\ &= \frac{1}{2} \sum_{u=1}^{8} |\left(\sqrt{p_u} - \sqrt{q_u}\right)|\left(\sqrt{p_u} + \sqrt{q_u}\right) \end{aligned} \tag{B.2}$$

The $45deg$ straight line in figure B.10 corresponds to points in the plot where either $p = 0$ or $q = 0$. The upper bounding curve corresponds to points where either $p = 1$ or $q = 1$. In the case where either of the bin values is unity the functional relationship for the curve is given by $y = 2\sqrt{x} - x$. Points between the upper and lower boundaries correspond to $0 < p < 1$, $0 < q < 1$. The lower left hand corner of the graph corresponds to points where the bin values $\{p, q\}$ are similar. The upper right hand corner of the graph corresponds to points where the bin values $\{p, q\}$ are very different. There are eight curves in figure B.10 corresponding to the eight bin values q associated with the target histogram. The points along each curve represent the 512 p values linked to each q.

The Bhattacharyya distance calculation involves summing eight abscissa values from figure B.10. The histogram intersection distance involves summing from the figure eight ordinate values corresponding to those abscissa values. However, for each abscissa there can be a number of ordinate values where the vertical line associated with that abscissa intersects with the curves. This means that each Bhattacharyya distance can be associated with a number of histogram intersection distances hence we can expect to see a scatter if one distance is plotted against the other.

The relatively compact form of the scatter in figure 2 in section 6 can be understood by recognizing that its extent is limited by bounding $45deg$ line and the upper curve in figure B.10. Also, each set of p and q values must sum to unity so most of the values tend to be concentrated in the lower left hand corner of that figure. These limitations mean that the resulting scatter of the points in figure 2 is not large. If the sums of eight ordinate values and eight abscissa values taken from figure B.10 are plotted against each other in figure 2 then the point will be placed above the $45deg$ line. The multiplication by 1/2 in the histogram intersection distance displaces the sum of points vertically downwards in that figure, and the square root in the Bhattacharyya distance calculation shifts the sums parallel to the x-axis towards the right. The overall result, as seen in figure 2, is the observed scattered distribution of the points around the $45deg$ diagonal.

Appendix C. The relationship between χ^2 and the Matusita distance

The χ^2 probability density function describes the probability of occurrence of a given value of the sum:

$$Q = \sum_{u=1}^{n} {x_u}^2 \tag{C.1}$$

for n independent values of $x \sim \mathcal{N}(0,1)$. It is used in testing the goodness of fit of experimental data to expectation. That kind of testing is based upon the recognition that for frequency binned data the distribution of

$$\sum_{u=1}^{n} \frac{(O_u - E_u)^2}{E_u} \tag{C.2}$$

where O_u and E_u represent observed and expected frequencies respectively is approximated by a χ^2 distribution.

For matched pairs of observed and expected histogram bin values p_u, q_u it can be shown [34] that the distribution of

$$X^2 = \sum_{u=1}^{n} \frac{(p_u - q_u)^2}{p_u + q_u} \tag{C.3}$$

is also approximated by a χ^2 distribution. This quantity X^2 is generally labelled as χ^2.

Whilst this statistic is used for significance/goodness of fit testing its usefulness as a distance measure can be questioned. Aherne et al. [24] point out the non-linear nature of χ^2 type distances. They replace p and q in (C.3) by $\sqrt{p}$ and $\sqrt{q}$ respectively and apply a first order Taylor approximation to the denominator to show that for small distances the χ^2 distance is twice their version of the Matusita distance (which is the actually the square of Matusita's original distance).

We can bring out a more direct link between the χ^2 measure and the Matusita distance, and hence

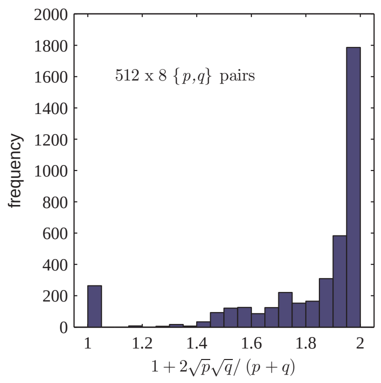


Figure C.11: Frequency distribution of the factor $1 + 2\sqrt{p}\sqrt{q}/\,(p+q)$ for $512\times8\{p,q\}$ pairs from a step in a typical particle filter track

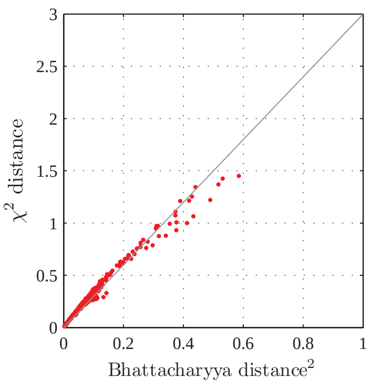


Figure C.12: Plot of χ^2 vs the square of the Bhattacharyya distance for 512 typical particles, indicating the scatter associated with the multiplicative factor. Note: the Bhattacharyya distances range from 0 to ~ 0.8 hence the maximum of the squared distances is ~ 0.6. The graph shows that the relationship between between the χ^2 and the B^2 values can be approximated by $\chi^2 = 3B^2$

the Bhattacharyya distance, in order to understand the detail of the observed scatter between the distance measures.

In equation (23) it was shown that χ^2 differs from M^2 by a factor $\left(\sqrt{p}+\sqrt{q}\right)^2/\,(p+q)$ multiplying each Matusita component $\left(\sqrt{p}-\sqrt{q}\right)^2$. Expansion of the numerator of the factor gives:

$$\begin{aligned}\frac{\left(\sqrt{p}+\sqrt{q}\right)^2}{p+q} &= \frac{p+q+2\sqrt{p}\sqrt{q}}{p+q}\\ &= 1+\frac{2\sqrt{p}\sqrt{q}}{p+q}\end{aligned}$$

In a normalized histogram the values of p or q are limited to the range [0,1]. If $p = q = 1$ or if $p \approx$

$q \approx 0$ then the component $\frac{2\sqrt{p}\sqrt{q}}{p+q} = 1$ so that

$$\frac{\left(\sqrt{p}+\sqrt{q}\right)^2}{p+q} = 2$$

If one of p or $q = 0$ then $\frac{2\sqrt{p}\sqrt{q}}{p+q} = 0$ so that

$$\frac{\left(\sqrt{p}+\sqrt{q}\right)^2}{p+q} = 1$$

Other combinations of p and q return values of the factor between 1 and 2. A typical frequency distribution of the factor for 512×8 $\{p, q\}$ pairs from a typical step of a particle filter track is shown in figure C.11. It can be seen that the factor varies between 1 and 2 with a tendency for most of the values to be at the upper end of the range. The relatively large number of entries in the lowest bin reflects the frequency of occurrence of one of p or $q = 0$. For a given set of $\{p, q\}$ pairs in a histogram Matusita/Bhattacharyya distance calculation a corresponding χ^2 distance is going to have different value due to the bin by bin contributions of the multiplicative factors. A given p, q difference can correspond to a range of p, q sums. We can expect to see a scatter of χ^2 distances for a given set of Matusita/Bhattacharyya distances. A typical scatter between the χ^2 distance and the square of the Bhattacharyya distance, using real particle filter data, is shown in figure C.12. The graph suggests that the relationship between between the χ^2 and the B^2 values can be approximated by $\chi^2 = 3B^2$

References

[1] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, ACM Comput. Surv. 38 (4) (2006) Article 13.

[2] R. Ristic, S. Arulampalam, N. Gordon, Beyond the Kalman filter, Artech House, 2004.

[3] S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188.

[4] W. Lu, K. Okuma, L. J. J., Tracking and recognizing actions of multiple hockey players using the boosted particle filter, Image and Vision Computing 27 (2009) 153–166.

[5] J. Snoek, J. Hoey, J. Stewart, R. S. Zemel, M. A., Automated detection of unusual events on stairs, Image and Vision Computing 27 (2009) 153–166.

[6] A. Maggio, F. Smerladi, A. Cavallaro, Adaptive multi-feature tracking in a particle filtering framework, IEEE Transactions on Circuits and Systems for Video Technology 17 (10) (2007) 1348–1359.

[7] J. Czyz, B. Ristic, B. Macq, A particle filter for joint detection and tracking of color objects, Image and Vision Computing 25 (8) (2007) 1271–1281.

[8] S. Kong, C. Sanderson, B. C. Lovell, Classifying and tracking multiple persons for proactive surveillance of mass transport systems, in: AVSS 07, 2007, pp. 159–163.

[9] Chipwrights, Programmable Visual Signal Processors, `www.chipwrights.com/`.

[10] British Security Industry Association, An introduction to video content analysis, `http://www.bsia.co.uk/` (2009).

[11] L. M. Fuentes, S. A. Velastin, People tracking in surveillance applications, Image and Vision Computing 24 (2006) 1165–1171.

[12] M. Khalid, M. U. Ilyas, M. S. Sarfaraz, M. S. Ajaz, Bhattacharyya coefficient in correlation of gray-scale objects, Journal of Multimedia 1 (1) (2006) 56–61.

[13] H. Driessen, Y. Boers, Map estimation in particle filter tracking, in: IET Seminar on target tracking and data fusion, 2008, pp. 41–45.

[14] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, in: CVPR 00, 2000, pp. 142–149.

[15] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, in: ECCV 02, LNCS 2350, 2002, pp. 661–675.

[16] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, D. G. Lowe, A boosted particle filter: Multitarget detection and tracking, in: ECCV 04, LNCS 3021, 2004, pp. 28–39.

[17] K. Okuma, J. J. Little, D. G. Lowe, Automatic acquisition of motion trajectories: Tracking hockey players, in: Proceedings of SPIE Internet Imaging V, no. 5304, 2003, pp. 202–213.

[18] X. Song, J. Cui, H. Zha, H. Zhao, Probabilistic detection-based particle filter for multi-target tracking,, in: BMVC 08, no. 1, 2008, pp. 223–232.

[19] K. Nummiaro, E. Koller-Meier, L. Van Gool, An adaptive color-based particle filter, Image and Vision Computing 21 (1) (2003) 99–110.

[20] Y. Cai, N. de Freitas, J. J. Little, Robust visual tracking for multiple targets, in: ECCV 06, LNCS 354, 2006, pp. 107–118.

[21] J. Czyz, B. Ristic, B. Macq, A color-based particle filter for joint detection and tracking of multiple objects, in: Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2, 2005, pp. 217–220.

[22] B. Han, L. Davis, Robust observations for object tracking, in: ICIP 05, Vol. 2, 2005, pp. 442–445.

[23] K. Kim, L. S. Davis, Multi-camera tracking and segmentation of occluded people on ground plane using search guided particle filtering, in: ECCV 06, LNCS 3953, 2006, pp. 98–109.

[24] F. J. Aherne, N. A. Thacker, P. I. Rockett, The Bhattacharyya metric as an absolute similarity measure for frequency coded data, Kybernetica 32 (4) (1997) 1–7.

[25] A. Doucet, S. Godsill, A. C., On sequential Monte Carlo sampling methods for Bayesian filtering, Statistics and Computing 10 (2000) 197–208.

[26] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification 2nd Ed., Wiley-Interscience, 2000.

[27] K. Matusita, Y. Suzuki, H. Hirosi, On testing statistical hypotheses, Annals of the Institute of Statistical Mathematics 2 (1954) 133–141.

[28] M. J. Swain, D. H. Ballard, Color indexing, International Journal of Computer Vision 7 (1) (1991) 11–32.

[29] D. Hall, F. Pélisson, O. Riff, J. L. Crowley, Brand identification using Gaussian derivative histograms, Machine Vision and Applications 16 (1) (2004) 41–46.

[30] S. T. Birchfield, S. Rangarajan, Spatial histograms for region-based tracking, ETRI Journal 29 (5) (2007) 697–699.

[31] C. Sanderson, A. Bigdeli, T. Shang, S. Chen, E. Berglund, B. C. Lovell, Intelligent cctv for mass transport security: Challenges and opportunities for video and face processing, Electronic Letters on Computer Vision and Image Analysis 6 (3) (2007) 30–41.

[32] Y. Iwahori, S. Okada, H. Kawanaka, S. Fukui, W. R. J., Particle filter based tracking for crossing of targets with similar pattern, in: IAPR Conference on Machine Vision Applications 07, 2007, pp. 307–310.

[33] M. Mason, Z. Duric, Using histograms to detect and track objects in color video, in: Applied Imagery Pattern Recognition Workshop, 2001, pp. 154–159.

[34] N. D. Gagunashvili, Pearson's chi-square test modifications for comparison of unweighted and weighted histograms for two weighted histograms, in: *XI* International Workshop on Advanced Computing and Analysis Techniques in Physics Research, Vol. 060 of PoS ACAT, 2007, pp. 1–10.

[35] G. Chen, J. R. Gott III, B. Ratra, Non-Gaussian error distribution of Hubble Constant measurements, Publications of the Astronomical Society of the Pacific 115 (2003) 1269–1729.

[36] K. M. Hanson, Bayesian analysis of inconsistent measurements of neutron cross sections, in: 25th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering, Vol. 803 of AIP Conference proceedings, 2005, pp. 431–439.

[37] T. Briegel, V. Tresp, Robust neural network regression for offline and online learning, Advances in Neural Information Processing Systems 12, MIT Press (2000) 407–413.

[38] N. Ikoma, N. Ichimura, T. Higuchi, H. Maeda, Maneuvering target tracking by using particle filter, in: Joint 9th IFSA World Congress and 20th NAFIPS International Conference, 2001, pp. 2223–2228.

[39] M. Isard, Condensation - conditional density propagation for visual tracking, International Journal of Computer Vision 29 (1998) 5–28.

[40] Home Office Scientific Development Branch, i-LIDS dataset for AVSS 2007, sequence: AVSS PV medium, `http://www.elec.qmul.ac.uk/staffinfo/andrea/spevi.html`.

[41] PETS, 2006 benchmark data: frames s1-t1-c.00107-00391, `http://www.cvg.rdg.ac.uk/PETS2006/data.html`.

[42] F. Porikli, Integral histogram: A fast way to extract histograms in cartesian spaces, in: CVPR 05, Vol. 1, 2005, pp. 829–836.

[43] F. Yin, D. Makris, S. Velastin, Performance evaluation of object tracking algorithms, in: PETS 07, 2007, pp. 17–24.

Histogram-based Detection of Moving Objects for Tracker Initialization in Surveillance Video

Peter Dunne

AD-Group, Daresbury Park, Daresbury,Warrington, WA4 4HS, UK
Applied Digital Signal and Image Processing Research Centre, School of Computing Engineering and Physical Sciences, University of Central Lancashire,
Preston PR1 2HE, UK
pdunne@ad-holdings.co.uk

Bogdan J. Matuszewski

Applied Digital Signal and Image Processing Research Centre, School of Computing Engineering and Physical Sciences, University of Central Lancashire,
Preston PR1 2HE, UK
bmatuszewski1@uclan.ac.uk

Abstract

We present an approach to localized object detection that is not dependent upon background image construction or object modeling. It is designed to work through camera embedded software using spare processing capacity in a visual signal processor. It uses a localized temporal difference change detector and a particle filter type likelihood to detect possible trackable objects, and to find a point within a detected object at which a particle filter tracker might be initialized.

Keywords: *object modelling, particle filter, tracker initialization*

1. Introduction

Moving object detection and tracking is central to a range of security and business intelligence surveillance video applications. The information derived from the processes can be used to alert security camera monitoring staff to potential trespass or rule violation in sensitive areas. In the case of business intelligence the information can be used to gather positive statistical information relating to customer behaviour.

The approach described in this paper is one of a range of analytics components being developed for use with the Chipwrights range of visual signal processors(VSPs) [1] as camera embedded software to provide low and intermediate level data for subsequent user analysis and interpretation. The VSPs have a primary role of dealing with image capture, intensity correction, compression, camera multiplexing etc., but there is spare processing capacity available for analytics tasks. The component that we describe is a computationally economic method of object detection for counting and possible tracker initialization.

Recent developments, for example [2,3,4], integrate initialization into the overall management of the trackers. They draw upon aspects such as 3D colour representations, spatiotemporal pixel homogeneity, optical flow etc. in order to identify objects within the images. Typically tracker initialization is carried out when detected objects cannot be accounted for by existing tracked targets yet satisfy criteria relating to size, shape, visibility, non-occlusion etc. But sophisticated approaches like those require computational commitments and feature information that are not available in our case.

We aim for grey scale image processing as artificial illumination such as fluorescent and sodium lighting can result in what is essentially monochrome footage. To accommodate both the multitasking requirements of the VSP and multiple object tracking we aim to process 5 frames per second(fps).

Some recent developments contain aspects that are closer to our requirements. For example Girisha and Murali [5] describe a method for segmenting motion objects from background that is based upon temporal frame differencing. They look at the correlation between pixel values at a given location over three sequential frames in order to determine if they represent background or belong to a foreground object. Ko et al.[6] describe a segmentation technique in which the background at each pixel location is represented by a histogram of values in the region surrounding that location. Foreground is detected using a thresholded Bhattacharyya distance between the current histogram at the pixel location and that of the temporally recursive updated background distribution.

We have avoided supervised classification approaches because of their likely demands on the processing capacity of the single VSP. In addition, our preference is for a generalized potential target detector. We avoid full frame background modeling and use temporal frame differencing to trigger our detector. As the process is not blob based we avoid difficulties associated with morphological processing.

We track objects using a grey scale histogram based particle filter and use the integral histogram approach [7] in order to extract multiple histograms efficiently. Our histogram based detector uses the same resources as the particle filters used for subsequent tracking hence keeping overall computation costs low. Rather than searching the whole image for tracker initiation at each frame we focus on expected target entry locations.

2. Method

2.1. Event trigger

The detector, which we refer to as a 'pad', covers an image region with dimensions half that of the bounding box of the expected target objects. Pads can be placed singly at appropriate places in an image or can be placed as arrays to form a tripwire type detection. Typical pad placement can be seen in Figure 1. The event trigger is based upon the absolute differences between pixel values in the current frame and the corresponding ones in the preceding frame. With a static camera the differences are generally small in the absence of objects and significant when an object moves in the pad region.

The first step in the process is the construction of the thresholded pad temporal difference image $\boldsymbol{D}_t$.

$$\boldsymbol{D}_t(x,y) = \left(\left| \boldsymbol{P}_t(x,y) - \boldsymbol{P}_{t-1}(x,y) \right| > \theta_D \right) \quad \forall (x,y) \in \boldsymbol{P} \qquad (1)$$

where $\boldsymbol{P}_t$ is the pad image at time t and θ_D is a pixel difference threshold. A typical $\boldsymbol{D}_t$ is shown in Figure 1(c). With the low frame rate the pixel value changes from frame to frame can be significant.

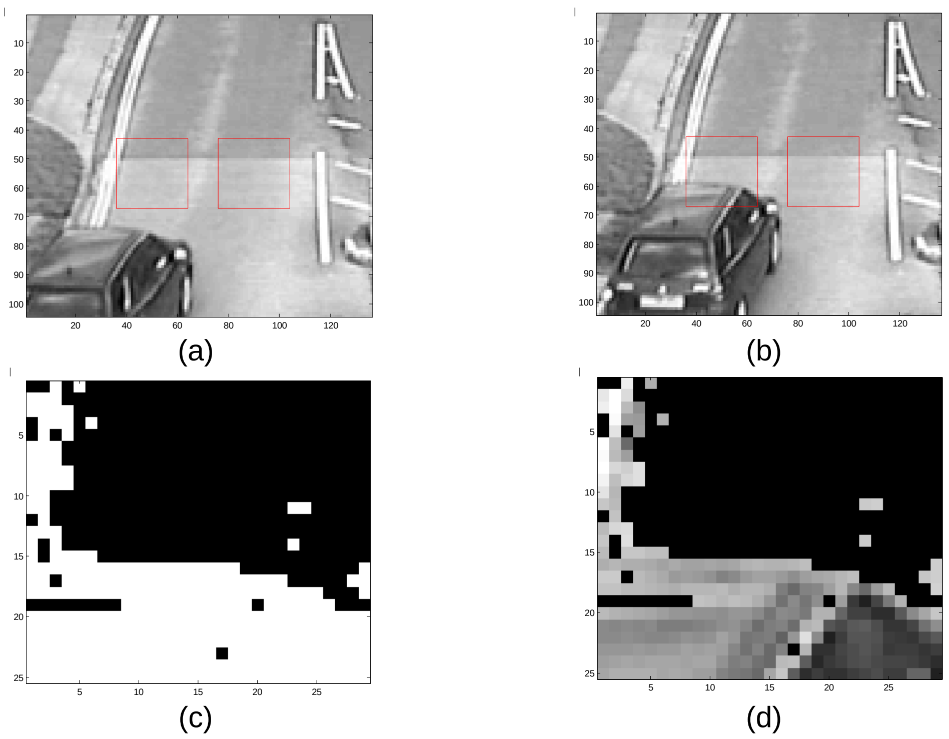


Figure 1. (a) pads 1 and 2 in frame at (t-1) (b) pads 1 and 2 in frame at (t) (c) D_t, pad(1) temporal difference mask (d) M_t, difference masked pad(1) image

The mean difference $\hat{d}_t$ across the pad is calculated:

$$\hat{d}_t = \frac{1}{N} \sum_{x,y \in \boldsymbol{P}} \boldsymbol{D}_t(x,y) \tag{2}$$

where N is the number of pixels in the pad image. A trigger occurs when $\hat{d}_t$ rises through a mean differences threshold θ_D and the mean differences are increasing:

i.e.
$$\hat{d}_t > \theta_d \wedge (\hat{d}_t - \hat{d}_{t-1}) > 0 \tag{3}$$

A range of alternative trigger conditions are possible, the essence is to detect change in the pad difference image.

The video sequence shown in the figure is from the AVSS 2007 i-LIDS Challenge dataset [8]. The images were downsized by a factor of 2 and we took every 5th frame to simulate the target 5fps rate.

2.2. Extracting object information

In the absence of a trigger we make a normalized 8 bin 'probably background' grey scale histogram $\boldsymbol{q}_B^t$ of the pad image at time t. An integral histogram of the whole image is made at each frame, to be used for the particle filter tracking, so this background histogram can be extracted without much extra computation. The histogram is used to recursively update a time averaged 'probably background' histogram $\tilde{\boldsymbol{q}}_B^t$:

$$\tilde{\boldsymbol{q}}_B^t = (1-\alpha)\tilde{\boldsymbol{q}}_B^{t-1} + \alpha \boldsymbol{q}_B^t \tag{4}$$

where α has a value typically of the order 0.2.

When a trigger condition is met we make a histogram $\boldsymbol{p}^t$ of the pad image. This contains information about both background and object. We use this together with $\tilde{\boldsymbol{q}}_B^t$ to make an 'indicative object' proposal histogram $\boldsymbol{q}_o^t$. The first step is to construct a difference histogram $\boldsymbol{q}_d^t$ by subtracting the averaged 'probably background' histogram from the current pad histogram:

$$\boldsymbol{q}_d^t = \boldsymbol{p}^t - \tilde{\boldsymbol{q}}_B^t \tag{5}$$

Negative bin values in the difference histogram indicate pixel value ranges that are more representative of the background rather than the foreground. Figures 2(a)-(c) show a set of typical histograms associated with this stage of the process.

Next, a difference masked pad image $\boldsymbol{M}_p$ (Figure 1(d)) is made by pixelwise multiplication of the pad thresholded temporal difference image with the pad image i.e.

$$\boldsymbol{M}_p(x,y) = \boldsymbol{D}_t(x,y)\cdot\boldsymbol{P}_t(x,y) \quad \forall(x,y)\in\boldsymbol{P} \tag{6}$$

The masked image will contain pixels associated with the object, shadows, ghosts, and zero values associated with the static region. The masked image is grey level sliced using the bin boundaries of the *n* = 8 bin histogram to produce a binary pad image $\boldsymbol{B}_p^u$ for each slice *u*:

$$\boldsymbol{B}_p^u(x,y) = \delta(b(\boldsymbol{M}_p(x,y)) - u) \quad \forall(x,y)\in\boldsymbol{P}, \quad u = 1,\dots,n \tag{7}$$

where $b(\boldsymbol{M}_p(x,y))$ is a function that compares the grey scale value $\boldsymbol{M}_p(x,y)$ against the bin boundaries and returns the bin number associated with it; δ is the Dirac delta function. The lower boundary of the first bin is set to 1 rather than 0 out of the range [0, 255]. This removes the mask static pixel contribution. The bins of the proposal histogram $\boldsymbol{q}_o^t$ are set to be the sums of the corresponding binary slice if the associated difference histogram bin $\boldsymbol{q}_d^t$ is positive, otherwise they are set to zero:

$$\boldsymbol{q}_o^t(u) = \begin{cases} \sum_{x,y\in\boldsymbol{P}} \boldsymbol{B}_p^u(x,y) & \text{if } \boldsymbol{q}_d^t(u) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The 'indicative object' proposal histogram is then normalized:

$$\tilde{\boldsymbol{q}}_o^t : \tilde{q}_o^t(u) = \frac{q_o^t(u)}{\sum_{v=1}^{n} q_o^t(v)} \qquad u = 1, \ldots, n \tag{9}$$

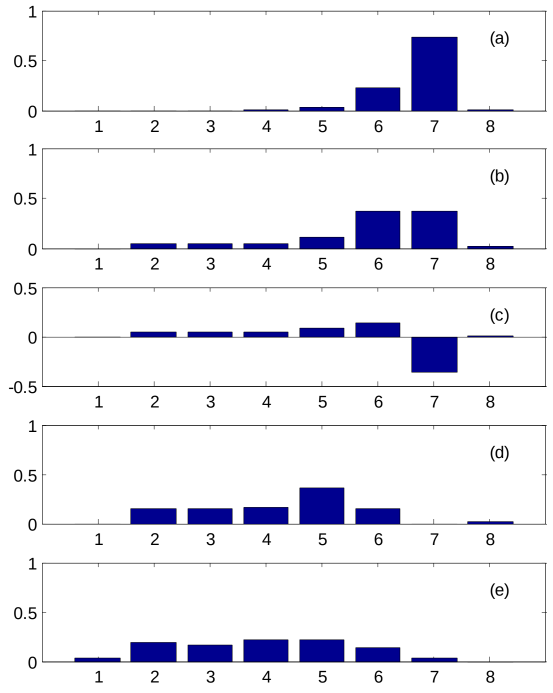


Figure 2. Histograms: (a) averaged probably background $\tilde{q}_B^t$, (b) pad image $\boldsymbol{p}^t$, (c) differences $\boldsymbol{q}_d^t = \boldsymbol{p}^t - \tilde{\boldsymbol{q}}_b^t$, (d) proposal $\tilde{q}_o^t$, (e) tracking histogram $\boldsymbol{q}_o^t$ extracted at the detect point

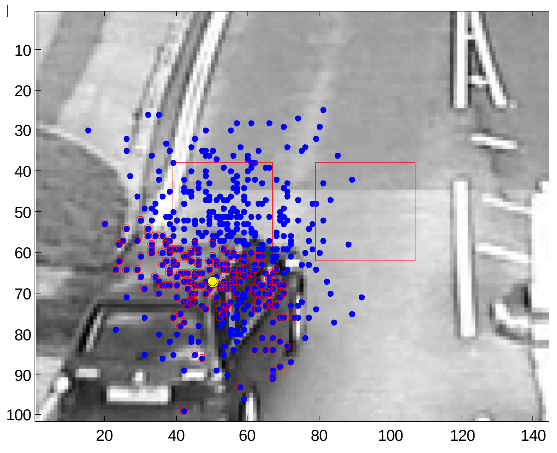
The proposal histogram, shown in Figure 2(d), indicates the grey scale values which are more characteristic of the part of the object entering the pad region than the background. We use this proposal histogram and the centre of gravity of the pad pixelwise difference image as the starting points of a search for a more representative histogram of the incoming object. Even though the proposal histogram is not a true representation of the object histogram, it is likely to be closer to it, in terms of the Bhattacharyya distance, than to histograms calculated from background in the region of the pad after the triggering event. The proposal histogram does not include bins that are dominant in the probably background histogram.

2.2. Updating object information

We use a particle filter type search of the region around the pad to get a better representation of the object. A prior is constructed by adding Gaussian random values to the centre of gravity $\boldsymbol{x}_0$ of the pad difference image $\boldsymbol{D}_t$:

$$\boldsymbol{x} = \boldsymbol{x}_0 + \nu \quad \text{where} \quad \boldsymbol{x} = (x, y) \quad \text{and} \quad \nu_i \sim \mathrm{N}\,(0, \sigma) \tag{10}$$

The standard deviation σ of the distribution is taken to be half of the largest dimension of the detect pad. A typical particle prior is shown in Figure 3(a).


(a) detect

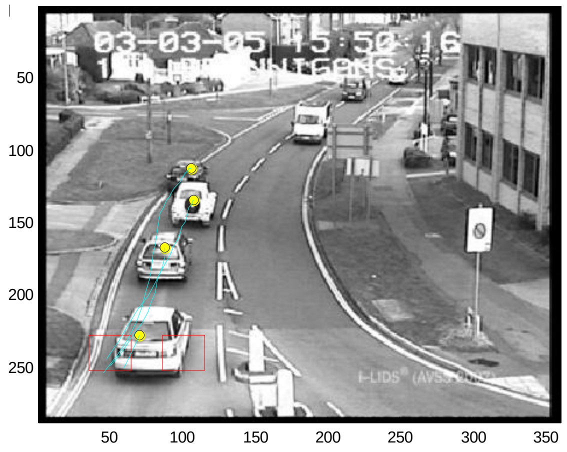

(b) merged tracks

Figure 3. (a) triggered pad, showing the prior particle set (blue), the selected high weight set (red) and the object proposed detection(yellow) (b) typical merged tracks from detect points

Histograms $\boldsymbol{p}^j$ are extracted using the integral histogram at the j particle locations. The particles are given a Gaussian weight w^j in terms of their Bhattacharyya distance S from the 'probably object' histogram $\tilde{\boldsymbol{q}}_o^t$ to give a posterior distribution i.e.

$$S^j = \sqrt{1 - \sum_{u=1}^{n} \sqrt{p_u^j \tilde{q}_{o,u}^t}} \tag{11}$$

and

$$w^j = \exp\left(-\frac{S^{j2}}{2\sigma^2} \right) \tag{12}$$

The standard deviation of the weighting distribution is set at $\sigma = 0.2$ [9].

The object locations $\boldsymbol{x}_{object}$, shown in Figures 3 and 4, are obtained using a weighted mean of the distribution of particle locations:

$$\boldsymbol{x}_{object} = \sum_{j=1}^{N} \tilde{w}^j \boldsymbol{x}^j \tag{13}$$

where $\tilde{w}^j$ is the normalized weight of the j^{th} particle:

$$\tilde{w}^j = \frac{w^j}{\sum_{i=1}^{N} w^i} \tag{14}$$

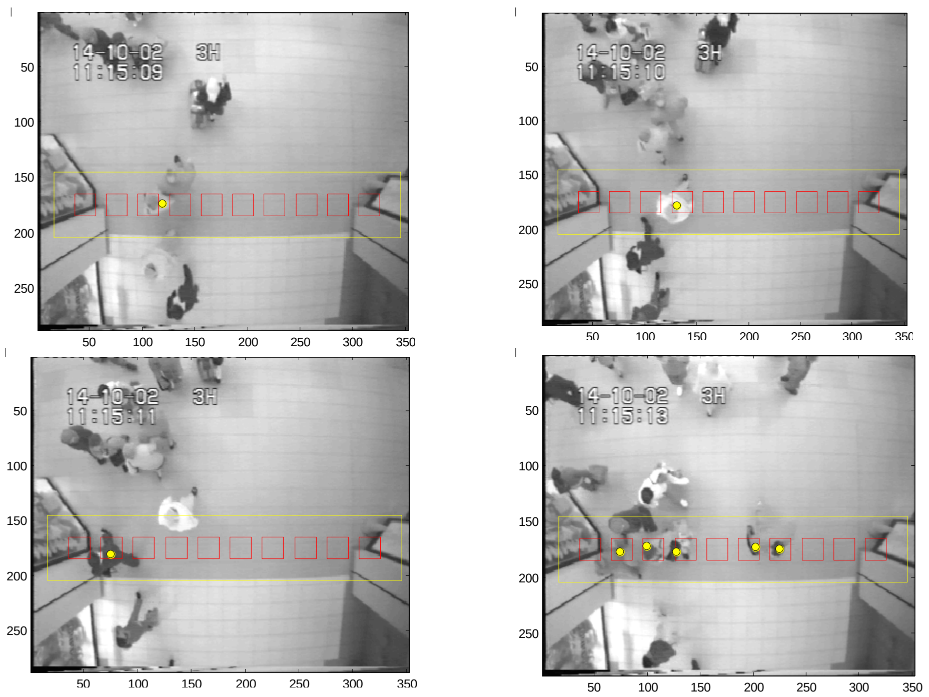


Figure 4. Examples of typical detections of pedestrians in a shopping mall, using an overhead camera

The final step of the detection process is to extract a tracking histogram at the object location and initialize a particle filter tracker at that point. An 8-bin tracking histogram $\boldsymbol{q}_o^t$ extracted at the object location in Figure 3(a) is shown in Figure 2(e). The histogram is extracted from a rectangular region with dimensions half those of the expected bounding box at the object location. In some cases, for tracking purposes, the rectangular region is split into four quadrants and a 32 bin histogram is constructed by concatenating four 8-bin histograms from those quadrants. This gives a tracking histogram with some indication of the spatial layout of the pixel values representing the foreground.

The objects are tracked using a simple SIR particle filter [10]. The particle state is described by the state vector $\boldsymbol{s}_t = [x, v_x, y, v_y]^T$. The particle filter initial velocities are taken to be zero but they pick up the velocity of the object within a couple of iterations. Typical tracks are shown in Figure 3(b).

3. Implementation

The implementation of the method is illustrated in two situations presenting different difficulties. The first one has an oblique view with slight camera movement, rigid objects and some slight illumination variation. The second with a stable camera, constant illumination, and non-rigid objects having various shapes and appearances.

Figure 3 shows a two pad array being used with the i-LIDS Challenge footage. In practice the pads can produce more than one trigger per object if passing objects are in contact with the pads for a number of frames. There are a range of heuristic strategies that can be employed to deal with trigger multiplicity. One approach is to disable the pad after the first trigger and then re-enable it once the tracker reports that the objects has cleared the area. However it is often the case that the second trigger gives a more representative histogram of the object so it can be useful to disable after an immediate second one. In the case of target overlaps, due to closely moving traffic for example, a re-enabled pad would trigger on the following object and initiate a new tracker. In such a situation a blob based system would have to assume occlusion and fit multiple models; the pad and histogram approach is capable of identifying separate objects within a blob and initiating independent trackers. An alternative is to keep the pads active and track all triggers independently; trackers that appear to have similar positions, velocities and tracker histograms are then merged into one. A typical tracker combination outcome is shown in Figure 3(b).

Figure 4 shows the use of an array of pads with an overhead camera in a shopping mall scenario. The target objects in the mall example can challenge impositions of a pedestrian model: shopping trolleys, luggage and other baggage can present a variety of shapes to the system. The potential of the approach can be seen in this sequence: with appropriate choice of trigger threshold the shopping mall sequence detected 22 out of 23 objects, returned 3 false positives (triggering on temporal difference ‘ghosts’) and 1 false negative. This returns values of precision, sensitivity and F-score [11] of the order 0.9. The heuristics are in continuing development.

4. Conclusion

We have presented a computationally economic approach to object initial detection for counting and possible tracker initialization. It is one of a number of approaches developed to use spare processing capacity for embedded analytics in intelligent cameras. It has potential for development in contexts such as vehicle or pedestrian traffic density indication, or as a tracker initializer in situations such as tripwires or perimeter violation detection. The method does not demand the use of a target model nor does it require the development of a full background image or classifier training. It works with moderate quality monochrome footage and can be used in a range of contexts.

References

[1] Chipwrights, "Programmable Visual Signal Processors", www.chipwrights.com

[2] E. Maggio, M. Taj, and A. Cavallaro, "Efficient Multi-Target Visual Tracking using Random Finite Sets", IEEE Transactions on Systems and Circuits for Video Technology, 2008, pp1016-1027.

[3] A. Bugeau and P. Peréz, "Track and Cut: Simultaneous Tracking and Segmentation of Multiple Objects with Track Cuts", EURASIP Journal on Advances in Signal Processing, 2008, pp1-14.

[4] Z. Zhang, P. L. Venetianer, and A. J. Lipton, "A Robust Human Detection and Tracking System using a Human-Model-Based Camera Calibration", 8th International Workshop on Visual Surveillance, 2008,

[5] R. Girisha and S. Murali, "Segmentation of Motion Objects from Surveillance Video Sequences using Partial Correlation", 6th IEEE International Conference on Image Processing, 2009,

[6] T. Ko, S. Soatto, and D. Estrin, "Background Subtraction on Distributions", 10th European Conference on Computer Vision, 2008, pp276-289.

[7] F. Porikli, "Integral Histogram: A Fast Way to extract Histograms in Cartesian Spaces", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, pp829-836.

[8] Home Office Scientific Development Branch, "i-LIDS Dataset for AVSS 2007 Sequence: AVSS pv medium", 2007, www.elec.qmul.ac.uk/staffinfo/andrea/spevi.html

[9] P. Dunne and B. J. Matuszewski, "Choice of Similarity Measure, Likelihood Function and Parameters for Histogram-based Particle Filter Tracking in CCTV Grey Scale Video", Image and Vision Computing, 2011, pp178-189.

[10] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for On-Line Non-Linear/Non-Gaussian Bayesian Tracking", IEEE Transactions on Signal Processing, 2002, pp174-188.

[11] A. Baumann, M. Boltz, and J. Ebling et al., "A Review and Comparison of Measures for Automatic Video Surveillance Systems", EURASIP Journal on Image and Video Processing, 2008, Article ID 824726, 30 pages, doi:10.1155/2008/824726

References

[1] British Security Industry Association. An introduction to video content analysis. http://www.bsia.co.uk/, 2009.

[2] A. W. Senior, L. Brown, A. Hampapur, C.-F. Shu, R. S. Feris, Y.-L. Tian, S. Borger, and C. Carlson. Video analytics for retail. In *IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 423–428, 2007.

[3] V. Gouaillier and A.-E Fleurant. Intelligent video surveillance: Promises and challenges. Technical report, CRIM and Technopôle Defence and Security, 2009.

[4] K. Cannons. A review of visual tracking. Technical Report CSE-2008-07, Department of Computer Science and Engineering, York University, Canada, 2008.

[5] E. Trucco and K. Plakas. Video tracking: A concise survey. *IEEE Journal of Oceanic Engineering*, 31(2):520–529, 2006.

[6] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):Article 13, 2006.

[7] Dedicated Micros. http://www.dedicatedmicros.com.

[8] Chipwrights. Programmable Visual Signal Processors. www.chipwrights.com/.

[9] Yole Développment. Uncooled IR cameras & detectors for thermography and vision report. http://www.yole.fr/iso_upload/News/2010/UncooledIR_March2010.pdf.

[10] Home Office Scientific Development Branch. i-LIDS dataset for AVSS 2007, sequence: AVSS PV medium. http://www.elec.qmul.ac.uk/staffinfo/andrea/spevi.html.

[11] PETS. 2001 dataset 1. `http://www.cvg.cs.rdg.ac.uk/PETS2001/pets2001-dataset.html`.

[12] PETS. 2006 benchmark data: frames s1-t1-c.00107-00391. `http://www.cvg.rdg.ac.uk/PETS2006/data.html`.

[13] V. Bruce, P. R. Green, and M. A. Georgeson. *Visual Perception*. Psychology Press, 4th edition, 2004. ISBN 978-1-84169-238-8.

[14] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[15] P. Dunne and B. Matuszewski. Choice of similarity measure, likelihood function and parameters for histogram based particle filter tracking in CCTV grey scale video. *Image and Vision Computing*, 29 (2-3):178–189, 2011.

[16] H. Driessen and Y. Boers. MAP estimation in particle filter tracking. In *IET Seminar on Target Tracking and Data Fusion*, pages 41–45, 2008.

[17] P. Dunne and B. Matuszewski. Histogram-based detection of moving objects for tracker initialization in surveillance video. *International Journal of Grid and Distributed Computing*, 4(3):71–78, 2011.

[18] L. M. Fuentes and S. A. Velastin. People tracking in surveillance applications. *Image and Vision Computing*, 24:1165–1171, 2006.

[19] R. Collins, R. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa. A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Pittsburgh, PA, May 2000.

[20] A. Piscaglia, A. Cavallaro, M. Bonnet, and D. Douxchamps. High level descriptors of video surveillance sequences. In *4th European Conference on Multimedia Applications*, pages 316–331, 1999.

[21] A. Cavallaro and T. Ebrahimi. Video object extraction based on adaptive background and statistical change detection. In *SPIE Visual Communications and Image Processing*, pages 465–475, 2001.

[22] A. Cavallaro and T. Ebrahimi. Change detection based on color edges. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 2, pages 141–144, 2001.

[23] A. Cavallaro and T. Ebrahimi. Interaction between high-level and low-level image analysis for semantic video object extraction. *Eurasip Journal on Applied Signal Processing*, 6:786–797, 2004.

[24] N. McFarlane and C. Schofield. Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8(3):187–193, 1995.

[25] A. Manzanera and J. C. Richefeu. A new motion detection algorithm based on $\Sigma - \Delta$ background estimation. *Pattern Recognition Letters*, 28(3):320–328, 2007.

[26] J. Vijverberg, M. J. H. Loomans, C. J. Koeleman, and P. H. N. de With. Global illumination compensation for background subtraction using gaussian-based background differences modeling. In *6th IEEE International Conference on Video and Signal Based Surveillance*, pages 448–453, 2009.

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

[28] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach. Real-time video analysis on an embedded smart camera for traffic surveillance. In *IEEE Real-time and Embedded Technology and Applications Symposium*, pages 1080–1812, 2004.

[29] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–767, 2000.

[30] P. W. Power and J. A. Schoonees. Understanding background mixture models for foreground segmentation. In *Image and Vision Computing, New Zealand*, pages 267–271, 2002.

[31] M. Isard and J. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *International Conference on Computer Vision*, volume 2, pages 34–41, 2001.

[32] S. Apewokin, B. Valentine, S. Wills, S. M. Wills, and A. Gentile. Multimodal mean adaptive background for embedded real-time surveillance. In *IEEE Embedded Computer Vision Workshop*, pages 1–6, 2007.

[33] T. Ko, S. Soatto, and D. Estrin. Background subtraction on distributions. In *European Conference on Computer Vision*, pages 276–289, 2008.

[34] F. J. Aherne, N. A. Thacker, and P. I. Rockett. The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetica*, 32(4):1–7, 1997.

[35] P. Noriega, B. Bascle, and O. Bernier. Local kernel color histograms for background subtraction. In *International Conference on Computer Vision Theory and Applications*, volume 1, pages 213–219, 2006.

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

[37] H. Grabner, P. M. Roth, M. Grabner, and H. Bischof. Autonomous learning of a robust background model for change detection. In *9th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 39–46, 2006.

[38] Y. Freund and R. Schapire. A short introduction to Boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, 1999.

[39] S. Cheung and C. Kamath. Robust background subtraction with foreground validation for urban traffic video. *EURASIP Journal on Applied Signal Processing*, 14:2330–2340, 2005.

[40] Y. Yoo and T. Park. A moving object detection algorithm for smart cameras. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop*, pages 1–8, 2008.

[41] A. Bugeau and P. Peréz. Detection and segmentation of moving objects in highly dynamic scenes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

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

[43] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[44] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[45] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413, 2004.

[46] Z. Zhang, P. L. Venetianer, and A. J. Lipton. A robust human detection and tracking system using a human-model-based camera calibration. In *Eighth International Workshop on Visual Surveillance*, 2008.

[47] C. Beleznai, B. Frühstück, and H. Bischof. Human tracking by fast mean shift mode seeking. *Journal of multimedia*, 1(1):1–8, 2006.

[48] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[49] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, 2001.

[50] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[51] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.

[52] G. Montiero, P. Peixotot, and U. Nunes. Vision-based pedestrian detection using Haar-like features. *Robotica*, 24:46–50, 2006.

[53] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, 2001.

[54] B. Wu and R. Nevatia. Detection of multiple partially occluded humans by Bayesian combination of edgelet based part detectors. In *Tenth IEEE International Conference on Computer Vision*, pages 90–97, 2005.

[55] D. M. Gavrila. A Bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1–14, 2007.

[56] H. G. Barrow, J. M. Tennebaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: two new techniques for image matching. In *International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.

[57] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 1–8, 2007.

[58] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition.*, volume II, pages 886–893, 2005.

[59] K. Nummario, E. Koller-Meier, and L. van Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99–110, 2003.

[60] J. Czyz, B. Ristic, and B. Macq. A particle filter for joint detection and tracking of color objects. *Image and Vision Computing*, 25:1271–1281, 2007.

[61] E. Maggio and A. Cavallaro. Multi-part target representation for color tracking. In *IEEE International Conference on Image Processing*, pages 729–732, 2005.

[62] W. Lu, K. Okuma, and J. J. Little. Tracking and recognizing actions of multiple hockey players using the boosted particle filter. *Image and Vision Computing*, 27:189–205, 2009.

[63] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *7th European Conference on Computer Vision, Lecture Notes in Computer Science*, volume 2350, pages 661–675, 2002.

[64] S. T. Birchfield and S. Rangarajan. Spatial histograms for region based tracking. *ETRI Journal*, 29:697–699, 2007.

[65] C. Yang, R. Duraiswami, and L. Davis. Fast multiple object tracking via a hierarchical particle filter. In *10th IEEE International Conference on Computer Vision*, pages 212–219, 2005.

[66] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV 04*, LNCS 3021, pages 28–39, 2004.

[67] Y. Iwahori, S. Okada, H. Kawanaka, S. Fukui, and R. J. Woodham. Particle filter based tracking for crossing of targets with similar pattern. In *IAPR Conference on Machine Vision Applications*, pages 307–310, 2007.

[68] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[69] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[70] S. Maskell and N. Gordon. A tutorial on particle filters for on-line nonlinear/non-Gaussian tracking. In *IEE Workshop on Target Tracking: Algorithms and Applications*, pages 1–15, 2001.

[71] P. S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, 1979.

[72] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005. ISBN 0-262-20162-3.

[73] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 0-387-31073-8.

[74] A. Doucet, S. Godsill, and Andrieu C. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[75] N. J. Gordon, D. J. Salmon, and A. F. Smith. Novel approach to nonlinear/non-Gaussian bayesian state estimation. In *IEE Proceedings-F*, volume 142(2), pages 107–113, 1993.

[76] M. K. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

[77] N. Oudjane, C. Musso, and F. LeGland. *Sequential Monte Carlo Methods in Practice*, chapter 12, pages 247–269. Springer, 2001.

[78] J. Kotecha and P. Djuric. Gaussian particle filtering. *IEEE Transactions on Signal Processing*, 51(10):2592–2601, 2003.

[79] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1144–1149, 2000.

[80] G. W. Pulford. Taxonomy of multiple target tracking methods. *IEE Proceedings Radar Sonar and Navigation*, 152(5):291–304, 2005.

[81] S. S. Blackman. *Multiple-Target Tracking with Radar Applications*. Artech House, 1986.

[82] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, 1979.

[83] I. J. Cox and S. L. Hingorani. 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(2):138–150, 1996.

[84] J. Vermaak, S. Maskell, M. Briers, and P. Pérez. A unifying framework for multi-target tracking and existence. In *IEEE 8th International Conference on Information Fusion*, pages 250–258, 2005.

[85] C. Rasmussen and G. D. Hager. Probabilistic data association methods for tracking multiple and compound objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):560–576, 2001.

[86] R. Karlsson and F. Gustafsson. Monte Carlo data association for multiple target tracking. In *IEE International Seminar on Target Tracking: Algorithms and Applications*, volume 1, pages 13/1–13/5, 2001.

[87] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *International Journal of Robotic Research*, 22(2):99–116, 2001.

[88] A. Almieda and J. Almieda. Real-time tracking of multiple moving objects using particle filters and probabilistic data association. *Automatika*, 36:39–48, 2005.

[89] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[90] E. B. Koller-Meier and F. Ade. Tracking multiple objects using the Condensation algorithm. *Robotics and Autonomous Systems*, 34(2,3):93–105, 2001.

[91] M. Marrón, M. A. Sotelo, and J. C. Garcia. Tracking multiple and dynamic objects with an extended particle filter and an adapted k-means clustering algorithm. In *5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, volume 267, 2004.

[92] M. Marrón, J. C. Garcia, M. A. Sotelo, D. Fernández, and D. Pizarro. "XPFCP": An extended particle filter for tracking multiple and dynamic objects in complex environments. In *International Conference on Intelligent Robots and Systems*, pages 2474–2479, 2005.

[93] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision*, 39(1):57–71, 2000.

[94] J. Vermaak, A. Doucet, and P. Pérez. Maintaining multi-modality through mixture tracking. In *9th IEEE International Conference on Computer Vision*, volume 2, pages 1110–1116, 2003.

[95] K. Okuma, J. J. Little, and D. G. Lowe. Automatic acquisition of motion trajectories: Tracking hockey players. In *Proceedings of SPIE Internet Imaging V*, volume 5304, pages 202–213, 2004.

[96] Y. Cai, N. de Freitas, and J. J. Little. Robust visual tracking for multiple targets. In *ECCV 06*, LNCS 354, pages 107–118, 2006.

[97] Z. Khan, T. Balch, and F. Dallaert. Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model. In *IEEE/RSJ International Conference on Robots and Systems*, volume 1, pages 254–259, 2003.

[98] Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. In *European Conference on Computer Vision*, volume 4, pages 279–290, 2004.

[99] K. Smith, D. Gatica-Perez, and J. Odobez. Using particles to track varying numbers of interacting people. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 962–969, 2005.

[100] M. Morelande, C. Kreucher, and K. Kastella. A Bayesian approach to multiple target detection and tracking. *IEEE Transactions on Signal Processing*, 55(5):1589–1604, 2007.

[101] M. R. Morelande, C. M. Kreucher, and K. Kastella. Multiple target tracking with a pixelized sensor. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume V, pages 585–588, 2005.

[102] C. Kreucher, K. Kastella, and A. Hero III. Multitarget tracking using the Joint Multitarget Probability Density. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1396–1414, 2005.

[103] R. Mahler. Why multi-source, multi-target data fusion is tricky. In *IRIS National Symposium on Sensor and Data Fusion*, volume 1, pages 135–153, 1999.

[104] R. Mahler. Multitarget moments and their application to multitarget tracking. In *Proceedings of the Workshop on Estimation,Tracking, and Fusion: A Tribute to Yaakov Bar-Shalom*, pages 134–166, 2001.

[105] B. Vo, S. Singh, and A. Doucet. Sequential Monte Carlo implementation of the PHD filter for multi-target tracking. In *IEEE Sixth International Conference on Information Fusion*, pages 792–799, 2003.

[106] H. Sidenbladh. Multi-target particle filtering for the probability hypothesis density. In *6th International Conference on Information Fusion*, pages 800–806, 2003.

[107] N. Ikoma, T. Uchino, and H. Maeda. Tracking of feature points in image sequence by SMC implementation of PHD filter. In *Annual Conference of the Society of Instrument and Control Engineers*, pages 1696–1701, 2004.

[108] M. Tobias and A. Lanterman. Probability hypothesis density - based multi-target tracking with bistatic range and Doppler observations. In *36th South-eastern Symposium on Systems Theory*, pages 2205–209, 2004.

[109] D. Clark and J. Bell. Bayesian multiple target tracking in forward scan sonar images using the PHD filter. *IEE Proceedings on Radar, Sonar Navigation*, 152(5):327–334, 2005.

[110] Y. Wang, J. Wu, A. Kassim, and W. Huang. Tracking a variable number of human groups in video using probability hypothesis density. In *IEEE International Conference on Pattern Recognition*, pages 1127–1130, 2006.

[111] C. D. Haworth, Y. de Saint-Pern, D. Clarke, E. Trucco, and Y. R. Petillot. Detection and tracking of multiple metallic objects in millimetre-wave images. *International Journal of Computer Vision*, 71(2):183–196, 2007.

[112] E. Maggio, E. Piccardo, C. Regazzoni, and A. Cavallaro. Particle PHD filtering for multi-target visual tracking. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 1101–1104, 2007.

[113] E. Maggio, M. Taj, and A. Cavallaro. Efficient multi-target visual tracking using random finite sets. *IEEE Transactions on circuits and systems for video technology*, 18(8):1016–1027, 2008.

[114] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. Brooks/Cole Publishing, 1999. ISBN 053495393X.

[115] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 142–149, 2000.

[116] F. Porikli. Integral histogram: A fast way to extract histograms in Cartesian spaces. In *CVPR 05*, volume 1, pages 829–836, 2005.

[117] M. S. Venegas, J. Knebel, and J. Thiran. Multi-object tracking using the particle filter algorithm on the top-view plan. In *European Signal Processing Conference*, volume 1, pages 285–288, 2004.

[118] A. Criminisi, I. Reid, and A. Zisserman. A plane measuring device. *Image and Vision Computing*, 17(8):625–634, 1999.

[119] R. Pflugfelder and H. Bischof. Online auto-calibration in man-made worlds. In *Digital Image Computing:Techniques and Applications*, pages 519–526, 2005.

[120] B. Merven, F. Nicolls, and G. de Jager. Auto camera calibration method for person tracking applications. In *13th Scandinavian Conference on Image Analysis*, pages 91–100, 2003.

[121] A. D. Bagdanov, A. Del Bimbo, F. Dini, and W. Nunziati. Improving the robustness of particle filter-based visual trackers using online parameter adaptation. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 218–223, 2007.

[122] E. Maggio and A. Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *IEEE Signal Processing Society International Conference on Acoustics, Speech and Signal Processing*, pages 221–224, 2005.

[123] E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[124] A. Maggio, F. Smerladi, and A. Cavallaro. Adaptive multifeature tracking in a particle filtering framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(10):1348–1359, 2007.

[125] J. Snoek, J. Hoey, J. Stewart, R. S. Zemel, and A. Mihailidis. Automated detection of unusual events on stairs. *Image and Vision Computing*, 27:153–166, 2009.

[126] X. Song, J. Cui, H. Zha, and H. Zhao. Probabilistic detection-based particle filter for multi-target tracking,. In *BMVC 08*, volume 1, pages 223–232, 2008.

[127] J. Czyz, B. Ristic, and B. Macq. A color-based particle filter for joint detection and tracking of multiple objects. In *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 217–220, 2005.

[128] B. Han and L. Davis. Robust observations for object tracking. In *ICIP 05*, volume 2, pages 442–445, 2005.

[129] K. Kim and L. S. Davis. Multi-camera tracking and segmentation of occluded people on ground plane using search guided particle filtering. In *ECCV 06*, LNCS 3953, pages 98–109, 2006.

[130] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification 2nd Ed.* Wiley-Interscience, 2000.

[131] K. Matusita, Y. Suzuki, and H. Hirosi. On testing statistical hypotheses. *Annals of the Institute of Statistical Mathematics*, 2:133–141, 1954.

[132] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[133] D. Hall, F. Pélisson, O. Riff, and J. L. Crowley. Brand identification using Gaussian derivative histograms. *Machine Vision and Applications*, 16(1):41–46, 2004.

[134] C. Sanderson, A. Bigdeli, T. Shang, S. Chen, E. Berglund, and B. C. Lovell. Intelligent CCTV for mass transport security: Challenges and opportunities for video and face processing. *Electronic Letters on Computer Vision and Image Analysis*, 6(3):30–41, 2007.

[135] S. Kong, C. Sanderson, and B. C. Lovell. Classifying and tracking multiple persons for proactive surveillance of mass transport systems. In *AVSS 07*, pages 159–163, 2007.

[136] M. Mason and Z. Duric. Using histograms to detect and track objects in color video. In *Applied Imagery Pattern Recognition Workshop*, pages 154–159, 2001.

[137] N. D. Gagunashvili. Pearson's Chi-Square test modifications for comparison of unweighted and weighted histograms for two weighted histograms. In XI *International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, volume 060 of *PoS ACAT*, pages 1–10, 2007.

[138] G. Chen, J. R. Gott III, and B. Ratra. Non-Gaussian error distribution of Hubble Constant measurements. *Publications of the Astronomical Society of the Pacific*, 115:1269–1729, 2003.

[139] K. M. Hanson. Bayesian analysis of inconsistent measurements of neutron cross sections. In *25th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, volume 803 of *AIP Conference proceedings*, pages 431–439, 2005.

[140] T. Briegel and V. Tresp. Robust neural network regression for offline and online learning. *Advances in Neural Information Processing Systems 12, MIT Press*, pages 407–413, 2000.

[141] N. Ikoma, N. Ichimura, T. Higuchi, and H. Maeda. Maneuvering target tracking by using particle filter. In *Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, pages 2223–2228, 2001.

[142] F. Yin, D. Makris, and S. Velastin. Performance evaluation of object tracking algorithms. In *PETS 07*, pages 17–24, 2007.

[143] G. Kitagawa. Monte-Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[144] J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamical systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[145] M. Bolić, P. M. Djurić, and S. Hong. Resampling algorithms for particle filters: a computational complexity perspective. In *EURASIP Journal on Applied Signal Processing*, volume 2004:15, pages 2267–2277, 2004.

[146] A. Naeem, T. Pridmore, and S. Mills. Managing particle spread via hybrid particle filter/kernel mean shift tracking. In *British Machine Vision Conference*, 2007.

[147] H. Driessen and Y. Boers. MAP estimation in non-linear non-Gaussian dynamic systems. *submitted to IEEE Transactions on Aerospace and Electronic Systems*, 2010.

[148] J. D. Hol, T. B. Schön, and F. Gustafsson. On resampling algorithms for particle filters. In *IEEE Nonlinear Statistical Signal Processing Workshop*, pages 79–82, 2006.

[149] R. Douc, O. Capp, and E. Moulines. Comparison of resampling schemes for particle filtering. In *4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 64–69, 2005.

[150] D. J. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.