Mean-Shift Tracking for Surveillance: Evaluations and Enhancements

A thesis submitted to the University of Dublin

in fulfilment of the requirements for the degree of

Doctor in Philosophy

Darren James Caulfield

March 2011

Declaration

Darren James Caulfield

11 March 2011

Abstract

Mean-shift tracking is a technique for following an object of interest as it moves through a video sequence. It is a gradient ascent approach that models the image region to be tracked by its colour histogram. In this thesis, we apply mean shift in the domain of surveillance in order to track people as they walk through a scene. Our objectives are to evaluate the performance of the technique and subsequently to introduce modifications which make the method more *robust*, i.e., more likely to follow a designated target through an entire video sequence.

We first compare mean shift to a standard template matching approach. The latter is found to be much more reliable, rarely losing track of its target, and so its performance serves as a baseline against which to measure the effects of our subsequent modifications to the basic mean-shift method.

In an effort to improve the reliability of mean shift, we employ an existing technique – the use of multiple-part models – to introduce a degree of spatial structure into its histograms, mimicking one of the strengths of template matching. We further extend the method by exploiting background models of the scene, another widely used modification. Our innovation of combining the two enhancements, while not enabling mean shift to reach the performance of the template matching tracker, increases its reliability considerably.

There are several parameters associated with any tracker. In the case of using the mean-shift technique in the surveillance domain, we seek the optimal choices for the colour space in which it operates and the size of its model histograms, among other parameters. Once again, the evaluations allow us to improve the performance of the method.

The greatest advantage of mean shift over other tracking techniques is arguably its computational efficiency (deriving from its gradient ascent nature), even if this comes at the expense of lower robustness. However, we succeed in developing a tracker based on normalised cross-correlation (the similarity measure at the heart of template matching) that also uses a gradient ascent optimisation strategy. The new approach is both fast and significantly more reliable than mean shift, calling into question the use of the latter technique.

We also define an algorithm for verifying that the output of a tracker is trustworthy.

Unlike our previous evaluations, the *track validation* approach does not require the use of ground truth data, and provides a semi-automated means of assessing the performance of any tracking method. It also allows our new gradient-based tracker to update its model in response to appearance changes in the target, further increasing the reliability of the method.

Acknowledgements

Firstly, I would like to thank my supervisor Ken Dawson-Howe, who introduced me to computer vision as an undergraduate. His guidance and encouragement over the years have been invaluable.

To the many current and former members of the Graphics, Vision and Visualisation (GV2) group in Trinity College, I offer my sincere gratitude. Wesley Cooper, Barry McCullagh, Mirko Arnold, Fernando Vilariño and David Eadie deserve special mention for sharing their insights into vision.

This work was financed in part by the Embark Initiative, operated by the Irish Research Council for Science, Engineering and Technology (IRCSET) and funded by the State under the National Development Plan.

Finally, my deepest thanks go to my family and friends, who have supported me in so many ways throughout the PhD.

Darren James Caulfield

Trinity College Dublin
March 2011

Contents

List of Figures

List of Acronyms

BG	background
CAVIAR	Context Aware Vision using Image-based Active Recognition
CC	cross-correlation
CVML	Computer Vision Markup Language
EMD	earth mover's distance
FG	foreground
IQR	interquartile range
MS	mean shift
NCC	normalised cross-correlation
pdf	probability density function
PETS	Performance Evaluation of Tracking and Surveillance
SSD	sum of squared differences
ViPER	The Video Performance Evaluation Resource
XML	Extensible Markup Language

Chapter 1

Introduction

Tracking objects as they move through video sequences is one of the most basic and most important tasks in computer vision. It serves as the foundation for numerous higher-level applications in many domains, including surveillance, augmented reality and motion capture. In this thesis we assess the performance of gradient-based tracking techniques, and in particular the mean-shift method. A number of experiments are performed in which people are tracked, using mean shift, as they walk through a scene. Quantitative evaluations lead us to introduce various modifications to mean-shift tracking.

Our objectives are to undertake a detailed investigation into the performance of mean shift, and to determine how the method can best be applied to the tracking of pedestrians in surveillance videos. The results of our experiments allow us to specify the optimal parameter settings and structure for the tracker. Ultimately, we find that the technique is simply not as capable, whether in its basic or modified form, as simpler approaches. We develop, for example, a gradient ascent template-based tracker that is significantly more reliable than mean shift while having similar computational efficiency. We conclude that the use in the surveillance domain of mean-shift tracking, in its present form, must be questioned.

1.1 Motivation

Reliable tracking of objects in videos is of great use in a number of areas in computer vision. In this section we review some of the applications that employ object tracking, and we describe the main difficulties that are faced by any tracking technique.

1.1.1 Applications

Moeslund *et al.* [105] identify three distinct application classes for motion capture (and hence object tracking): surveillance, control and analysis. *Surveillance applications* are concerned primarily with the monitoring of people. For example, we may wish to count the number of people in a group [129], or to study the overall flux of a crowd [3], perhaps to detect congestion or other dangerous situations. Tracking individuals within a larger group is one way of accomplishing such tasks. However, it may be desirable to detect the specific activities that are occurring, perhaps in order to notify a security guard of suspicious behaviour, for example, loitering [13]. Studying other types of human motion, such as how customers move around shops [63], also depends on being able to track people.

Control applications relate to the interaction between humans and computers. The *Eye-Toy*, which is similar to a webcam, tracks a user's movements, allowing them to play games on Sony's PlayStation console.[1] Controlling a computer by means of hand gestures also typically requires the use of tracking [119]. In the field of surgery, virtual objects can be inserted into a video stream in such a way that they appear to be a part of the scene [111]. Real-time, robust tracking of landmarks is essential for these *augmented reality* systems to work convincingly.

Analysis applications, which also employ object tracking techniques, typically process large amounts of video. For example, systems that track a person's joints allow doctors to diagnose problems with gait [177], while algorithms for following players can enable trainers to find means of improving a team's performance [6]. Other uses of tracking include video annotation [26] and content-based video retrieval [22]. The emerging area of car control also requires object tracking, whether for lane following [56] or for collision avoidance [4].

1.1.2 Difficulties in tracking

For a variety of reasons, tracking an object through video is a challenging task [179]. Firstly, the imaging process itself, whereby a three-dimensional scene is depicted in two dimensions, results in a significant loss of information: the depth of a pixel can no longer be directly measured. The limited image resolution, the small number of bits used to represent each pixel, noise introduced during image acquisition and artifacts caused by compression all serve to lower the quality of the video data. The objects to be tracked also give rise to difficulties: they can undergo complex motion, e.g., exhibiting variable velocity and acceleration, that is not well described by a given model; in the case of people, they display non-rigid motion and articulated motion; and they may not always contrast sharply with the background. An object's appearance can also change over time because of variations in scene illumination,

[1]Sony PlayStation website: `http://uk.playstation.com/`

and its apparent size depends on how close it is to the camera. Partial or full occlusions also prove very challenging when attempting to follow an object. Finally, it may be necessary for the system to operate under real-time constraints, ruling out certain effective but slow tracking techniques.

1.2 Objective

This thesis serves as an investigation into the mean-shift tracking technique. Mean shift is a gradient-based method that uses a histogram to represent the image region that it wishes to track. Its gradient ascent, or *hill climbing*, nature makes it computationally efficient, which, along with its relatively straightforward implementation, has made it one of the most popular tracking techniques of the last decade.

Our objective is to learn more about the performance of mean shift and to develop methods for its improvement. We wish to answer the following questions about the technique:

- How does the performance of mean shift compare to other data-driven tracking approaches such as template matching? We are interested in the robustness (how often it loses track of its target) and the accuracy (whether it is correctly positioned on the object) of each method, so that we can put the performance of mean shift in context.

- Is it possible to improve the robustness and accuracy of mean shift by imposing some spatial structure on the tracker? Such an approach would bring the method closer in operation to template matching, while retaining its gradient ascent nature, which allows it to be computationally efficient.

- How does the choice of various parameter settings affect the performance of the mean-shift tracker? For example, do some colour spaces yield more robust tracking than others? Does the type of spatial smoothing kernel used have an influence on the tracker's accuracy?

- Instead of using only the initial target model to track an object through a video sequence, can we develop strategies to allow us to update the model in a reliable fashion, so that the tracker can better cope with changes in the appearance of the object? Such changes are most commonly caused by the variation in lighting encountered over a scene, by the articulation of a person's limbs and by the movement of their clothing.

1.3 Approach

We have chosen pedestrian tracking as the application domain for the thesis. As discussed in section 1.1.1, many areas of computer vision, including surveillance, activity recognition and video analysis, are largely concerned with the tracking of people. We limit the scope of the research by specifying the attributes that our dataset of video test sequences should possess. For example, we operate on videos that are recorded by a fixed camera that is placed some distance above the ground. The latter constraint allows us to exploit the perspective effect in order to estimate a person's height when given their position in the image. The videos in the publicly available CAVIAR[2] and PETS 2007[3] datasets meet these requirements, and so they are used in all of our experiments. (Some example frames from the sequences are shown in figure 1.1.) We expect our trackers to cope with targets that have low contrast with the background and that undergo significant brightness and appearance changes. It is assumed that the trackers have been initialised correctly by an external process. And although the people to be tracked are never occluded, the trackers must contend with "distractors" – objects with a similar appearance moving in the vicinity of the target.

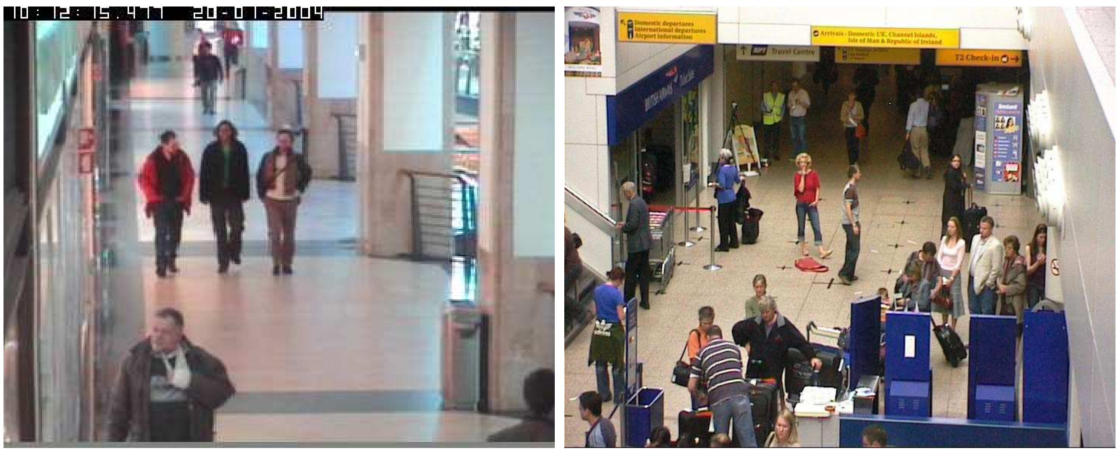


Figure 1.1: *Typical frames from the CAVIAR and PETS video sequences on which our tracking methods will operate*

1.3.1 Mean-shift assessment and other similarity measures

We begin our research into mean shift by using it to track a specified target in each of 21 CAVIAR and PETS videos that comprise our dataset. Each sequence has data associated

[2]The CAVIAR datasets come from the EC Funded CAVIAR project/IST 2001 37540, found at `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`

[3]The PETS 2007 datasets can be found at `http://www.cvg.rdg.ac.uk/PETS2007/`

with it which specifies, for every frame, the location of the person to be tracked. This ground truth data allows us to undertake a quantitative assessment of mean shift: at every frame we compare the location of the tracker to the ground truth location of the target. The performance of the mean-shift technique is measured in two principal ways. Firstly, we define *robustness* as the number of targets in the dataset which the method is successfully able to track without wandering away to an incorrect part of the video frame. Secondly, the *accuracy* of the tracker represents the degree to which its location matches the centre of the target, as specified by the ground truth data. (Two tracking methods may successfully follow a target, i.e., not wander away to an incorrect location, but one may be significantly more accurate than the other.)

Having determined the performance of the mean-shift technique on our video test set, we next compare it to a variety of other data-driven trackers. Firstly, we evaluate a histogram-based tracker which uses the Bhattacharyya coefficient as its similarity measure – the same measure that is at the heart of the mean-shift method. But, unlike the gradient ascent nature of mean shift, the Bhattacharyya tracker uses a brute-force search strategy. The comparison between the two methods allows us to isolate the effect of gradient ascent optimisation on mean-shift tracking. We find that it is beneficial to the technique, reducing the chances of a "distractor" causing the method to lose track of its target. Two other brute-force trackers are also evaluated alongside mean shift in this fashion. The first, a histogram-based tracker that uses the *earth mover's distance* [136] as its similarity measure, is found to have performance very similar to that of the Bhattacharyya method. However, the second, a normalised cross-correlation (NCC) template matching tracker, proves to be significantly more robust than the other methods tested. It thus provides the baseline against which all later modifications to mean shift are compared.

1.3.2 Multiple-part models and background exclusion

We hypothesise that the effectiveness of the NCC tracker is due, at least in part, to its encoding of spatial structure within the image template, something which the histograms of mean shift are lacking. This leads us to develop a mean-shift tracker that captures a degree of the spatial arrangement of the colours that make up an object we wish to follow, and that simultaneously exploits background models of the scene to improve the reliability of tracking. We evaluate our combined multiple-part–background exclusion method in the same manner as in the previous experiments. It is found that multiple-part models on their own are detrimental to the tracker's performance; but when used together with background exclusion, our new mean-shift tracker is more robust than the basic version.

1.3.3 Tracker parameters

Having achieved an improvement in the performance of mean shift, we next seek to determine the precise effect of the many parameters associated with the method. Our objective is to find the optimal settings for these parameters in the context of pedestrian tracking. We use our standard evaluation framework to assess the influence on tracking robustness of the colour space used, the size of the target and candidate histograms, the threshold on the convergence condition and the type of spatial kernel employed by the tracker. The results confirm that the use of multiple-part models in combination with background exclusion is the single most effective way of improving the performance of mean shift. We should also avoid single-channel colour spaces but make use of small histograms.

1.3.4 Beyond mean shift and verifying tracker output

Unfortunately, even with optimal settings for the parameters of the mean-shift technique, it remains significantly less robust than the brute-force NCC tracker. Indeed, we develop a gradient-based version of NCC that is as reliable as its brute-force counterpart and that has computational efficiency comparable to mean shift. With the emergence of this new method, we conclude that there is little reason to use mean shift, at least in the domain of pedestrian tracking.

Finally, we develop an algorithm for determining if the output returned by a data-driven tracking technique is trustworthy. Our approach – *track validation* – is to compare the paths that result from tracking the target forwards in time and then backwards in time through the video sequence. A significant difference between the two indicates that the tracker lost its target at some point during the video. In such cases, we shorten the sequence until the paths match, i.e., until the tracks are *validated.* This has the advantage of allowing us to update the model used for tracking to a new one taken from the end of the subsequence, since we know that the tracker followed its target correctly. The updates let the model reflect changes in the object's appearance, and so make the next phase of tracking more likely to succeed.

1.4 Contributions

The thesis makes a number of contributions to the field of gradient-based tracking. Firstly, we provide a comprehensive assessment of the performance of mean-shift tracking on a large test set of video sequences, counting the proportion of the targets that the technique is able to follow successfully and measuring its positional accuracy during tracking. We also compare mean shift against other histogram-based methods, and against a template matching approach (NCC), using the same dataset of videos. We show that NCC is significantly more robust than mean shift, which is in turn more robust than a similar histogram-based tracker that does not use a gradient ascent approach.

Secondly, we develop a tracker based on mean shift that unifies two existing elements – multiple-part models and background exclusion – in order to introduce spatial structure into the method, and we evaluate its performance. The new method is found to be more robust than trackers that use either element in isolation. The results also reveal that multiple-part models on their own result in a decrease in tracking performace compared to the basic mean-shift technique.

Our third contribution is a thorough investigation of the effect of changing various parameters associated with mean shift: the colour space in which it operates, the size of its histogram, the threshold on the convergence condition and the type of spatial kernel used by the tracker. Our experiments confirm that the use of multiple-part models and background exclusion is the single most effective way of improving the performance of the mean-shift method. Further performance gains can be achieved by using two- or three-channel colour spaces that encode luma (brightness) information, and by using small histograms.

Next, we develop a gradient-based normalised cross-correlation tracker, which has the advantage of being both computationally efficient and robust. We determine that its robustness matches that of brute-force NCC and that its execution speed is comparable to that of mean shift.

Our final contribution is the development of a *track validation* algorithm – a mechanism for verifying that the tracker's output is trustworthy. It also provides us with a means of updating the model of the tracked object safely in order to accommodate appearance changes that it may undergo. Our tests show that it is a very effective filter on a tracker's output: it very rarely accepts an incorrect trajectory for a target.

1.5 Outline of the thesis

In this section we provide a brief overview of the structure of the thesis. Figure 1.2 is a schematic representation of the framework we use to evaluate and improve mean-shift tracking. The four primary components – tracking methods, spatial structure, tracker parameters and track validation – are described in greater detail in the following sections and correspond to the research presented in chapters 4 to 7. The evaluation methodology, discussed in chapter 3, is the same for all of the components.

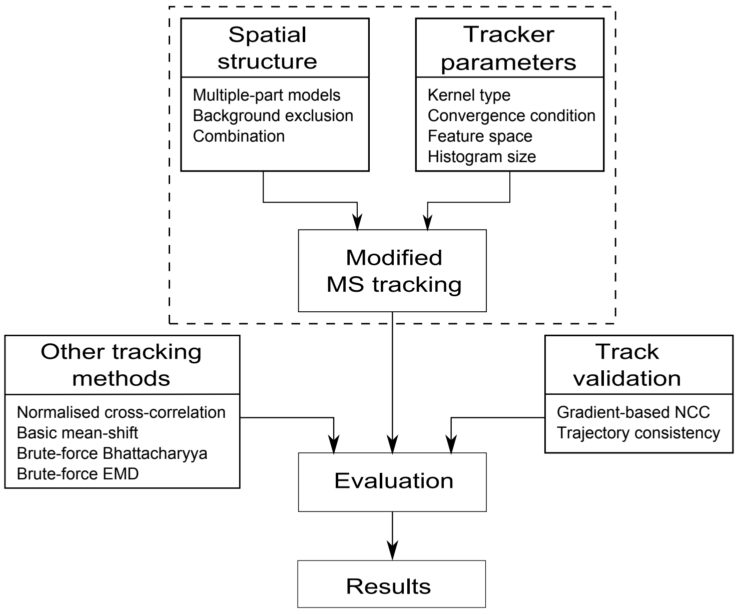


Figure 1.2: *The tracking evaluation framework, as implemented in this thesis*

Chapter 2: Pedestrian tracking: a review

In chapter 2 we present a review of the state of the art in object tracking, with a particular focus on techniques and systems related to following pedestrians. The various types of control strategies that are commonly used in tracking are described, and we discuss the different classes of image features that serve as inputs to the methods. Data-driven techniques, including mean shift and its variations, are also reviewed.

Chapter 3: Overview of datasets and metrics

Chapter 3 describes in detail the datasets and metrics that are used in all of the subsequent experiments. We give the formulas for the per-frame metrics we employ, and discuss how the aggregate (per-sequence) measures are calculated.

Chapter 4: Target representation and optimisation strategies

If we are to evaluate the effect of changes we make to mean-shift tracking, it is first necessary to know the performance of the basic technique, before any modifications have been introduced. In chapter 4 mean shift is used to track a designated target through each of several video sequences. Various accuracy metrics are recorded, and the number of times that tracking is lost is also determined. We also evaluate the performance of two brute-force histogram-based trackers and a normalised cross-correlation tracker. The last of these approaches serves as a baseline against which all enhancements to the mean-shift technique are compared.

Chapter 5: Tracking with multiple-part models

Chapter 5 presents a technique for unifying two elements in a single mean-shift tracker. Multiple-part models capture a degree of the target's spatial structure while background exclusion limits the influence of distractors on the tracker.

Chapter 6: Effect of tracker parameters

Mean-shift tracking has many parameters, both implicit and explicit, that affect its performance. In chapter 6 we perform a series of experiments to determine the most effective set of parameters for general person tracking with the mean-shift technique.

Chapter 7: Gradient-based NCC and track validation

Chapter 7 presents our *track validation* algorithm – a means of verifying that the output of a tracker is trustworthy. It also allows us to update the object model safely as the target undergoes appearance changes. We derive a gradient ascent version of normalised cross-correlation, which we use together with the algorithm. The new tracker is found to be both robust and computationally efficient.

Chapter 8: Conclusions

In the final chapter we discuss the overall performance of mean shift, and draw several conclusions about the method and about other data-driven tracking techniques. A number of avenues for future research are also suggested.

1.6 Publications to date

- D. Caulfield and K. Dawson-Howe. Direction of Camera Based on Shadows. In *Irish Machine Vision and Image Processing Conference*, pages 216–223, 2004.
- D. Caulfield and K. Dawson-Howe. Evaluation of multi-part models for mean-shift tracking. In *International Machine Vision and Image Processing Conference*, pages 77–82, 2008.

Chapter 2

Pedestrian tracking: a review

The last fifteen years have seen an explosion of interest in visual tracking. The continued improvement in computer hardware has allowed video sequences to be processed on desktop machines, resulting in the development of many new algorithms. In this chapter we build on previous surveys [105, 179] and review the state of the art in pedestrian tracking, by far the most common application of the technique in the literature. The mean-shift method – the focus of this thesis – is placed in the context of other tracking approaches in order to describe our subsequent research.

Visual tracking, or simply *tracking*, can be defined as "estimating the state sequence of a target from an observed image sequence [...], given a description of the target in some predefined form" [37]. It may or may not include an initialisation process, whereby an object to be tracked is first identified using techniques such as background subtraction or object detection. Determining the state of an object can be limited to finding its position in the image, or, in the case of pedestrian tracking, it can extend to *pose estimation*, where the objective is to recover the precise position and orientation of a person's limbs. In this review we are not concerned with pose estimation or other high-level video analysis such as action recognition (i.e., interpreting the intent behind a person's movements). Instead we concentrate on the control strategies and low-level image features that are used in tracking algorithms.

A variety of applications have been proposed for pedestrian tracking [105]. In the area of surveillance, tracking is an essential element in systems that count people and monitor crowds. It is also a prerequisite for most activity recognition techniques (e.g., detection of loitering [13]) and for behaviour analysis (e.g., studying the movement patterns of shoppers [63]). Tracking is used in the sports domain to follow football players around the pitch [98]. A number of challenges arise when creating systems of this type [179]. Using only a single camera results in a loss of three-dimensional information; we must work with a perspective projection of the scene. Noise in the image and the limited brightness range of the sensor lead to video sequences of less-than-ideal quality. The targets themselves (i.e., people) display non-rigid, articulated motion with varying speeds and accelerations. People can occlude one another, either partially or completely, and the illumination can vary over the scene and over the course of the sequence. Robust tracking methods are therefore necessary to overcome these difficulties.

It is possible to look at tracking from two angles: control strategies and low-level image features. In the former view, tracking is seen as a top-down, or model-driven, process: the state of the tracker is first evolved according to some (motion) model, before being refined by data from the image. Alternatively, we can consider the various types of low-level features that are used for tracking. Features extracted from the image or video would constitute the *observation model* in a top-down tracker. However, they can also be used by systems that employ no explicit state and observation models – so-called bottom-up, or data-driven, trackers. Control strategies and feature types are orthogonal; the choice of one is largely independent of the other. Drawing an analogy with pattern recognition, control strategies can be viewed as classifiers, with the tracking features serving as input. In our research, therefore, we seek more robust image features to improve the performance of tracking, but we use the same basic classifier/algorithm (mean shift) throughout so that we can make a fair evaluation of the effect of the new features.

The present chapter is structured as follows. Section 2.1 describes the general structure of pedestrian tracking systems and reviews some of the more well-known approaches. In section 2.2 we examine various tracking control strategies such as Kalman filtering and particle filtering, and discuss the real-world constraints and initialisation techniques (background subtraction and pedestrian detection) that are commonly used. The different classes of low-level image features are described in section 2.3, while section 2.4 reviews data-driven tracking in detail. As the focus of the thesis, mean shift is placed in the context of other gradient ascent and bottom-up methods so that possible enhancements and innovations can be identified.

2.1 Visual tracking

Of the many applications of visual tracking that are found today, perhaps the biggest proportion relates to the monitoring of people. The increased interest in security over the last decade has spurred research into systems capable of tracking individuals as they move through areas observed by video cameras. Given their prominence in the literature, we will use such systems as the basis for our investigations into the mean-shift tracking technique.

Figure 2.1 depicts a structure that is common to many single-camera pedestrian tracking systems. The video stream to be processed comes either from a recorded sequence or a live camera. The former approach, or the buffering of part of the stream, allows the system to delay making decisions until further data has been seen, which can sometimes clarify the nature of an event and improve the robustness of the tracking. Many systems assume that the camera is fixed and exploit background subtraction to identify pixels belonging to foreground objects. The various approaches to background modelling are reviewed in section 2.2.5. The "blobs" that result from this process will not always have a one-to-one correspondence with individuals to be tracked; a group of people walking together may appear as one blob, and, conversely, a person may be split into multiple blobs if his or her appearance is similar to that of the background. It is therefore necessary to employ various techniques to generate person *hypotheses* from the foreground pixels. Direct detection of human-like shapes in the image is one approach; alternatively, we can attempt to detect specific parts of the human body, e.g., heads or feet (section 2.2.5).

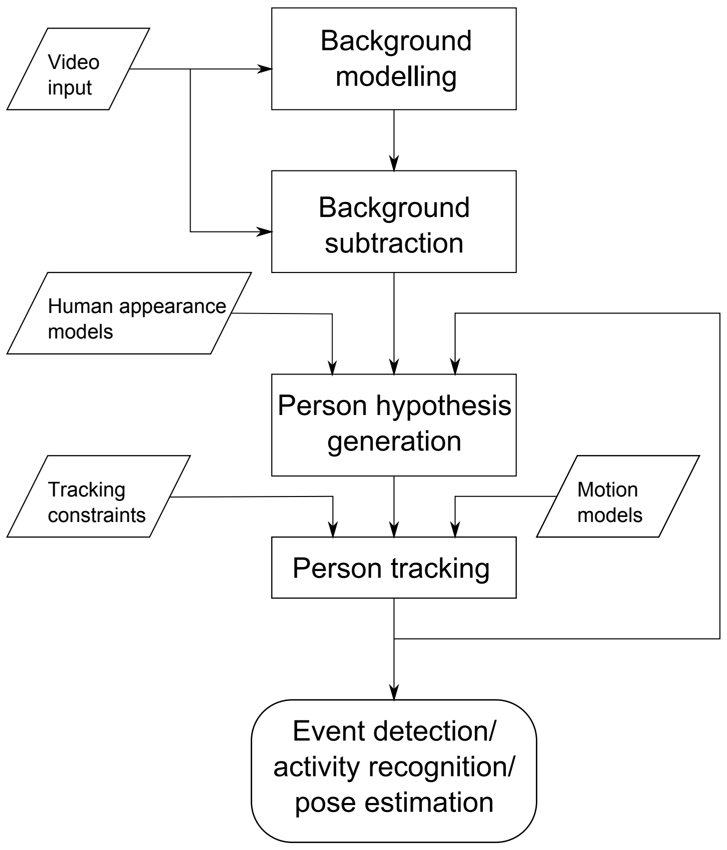


Figure 2.1: *The high-level structure of many single-camera pedestrian tracking systems*

Once we have hypotheses of the locations of people in a given video frame, we can attempt to track them through the sequence. Tracking and detection are complementary: a putative human detection is used to initialise a tracker, which then attempts to verify the hypothesis by tracking the target over time. Once initialised the tracker can continue to operate even in the absence of further detections; it effectively "bridges the gap" between possibly intermittent detections. The two stages, detection and tracking, operate in a continuous loop. Various types of features are used for tracking: points, motion, contours, etc. These are reviewed in section 2.3.

Two further sources of information can be used by the tracker. Firstly, motion models are employed to provide predictions of the object's next location so that the search space can be reduced in size. Section 2.2 describes the various control strategies (Kalman filtering, particle filtering, etc.) used by single- and multiple-target trackers. Secondly, real-world constraints can be brought to bear on the tracking problem (section 2.2.4). These could, for example, encode the requirement that different points on the same object should display similar motion over time; or that tracking an object forwards in time and then backwards in time should yield similar trajectories.

The structure shown in figure 2.1 is a very general depiction of single-camera tracking. Most systems can be mapped onto this layout. However, not all of the components and information sources shown will be present in a given system. Furthermore, we limit the scope of this thesis to determining people's trajectories, specifically using mean-shift tracking techniques (section 2.4). We also apply certain constraints to these trajectories (section 2.2.4) in order to validate the output of the tracker. Higher-level processing, such as event detection, activity recognition or pose estimation, is not considered in the present work.

The following section reviews the operation of some of the most well-known single-camera tracking systems found in the literature.

2.1.1 Pedestrian tracking systems

Over the last fifteen years many systems have been developed for tracking multiple people through a scene. They can be assessed in terms of the difficulties they are designed to cope with, e.g., shadows, busy scenes, inter-person occlusions, changing lighting conditions, low object–background contrast, rain or snow, and low-resolution imagery.

The "closed-world tracking" system of Intille [69] was one of the first to follow the movements of multiple people. An overhead camera records activities in the *KidsRoom*, tracking children as they run. Background subtraction is used to identify moving blobs, and these are matched over time using various features including colour, size, velocity and current and past position. Children can enter or leave the room, but only through a single door, in order that the system has an accurate count of the number of people to track. The authors identify various limitations of the system. Background subtraction does not always segment the children correctly, and object features cannot be updated while two blobs are merged into one. Also, since a *greedy* matching algorithm is used (where objects are matched one at a time, starting with the best match score), identity switches can arise. The matching decisions at every frame are also binding; there is no mechanism to review questionable matches using later video data.

Many of the difficulties faced by Intille remain the same for later multiple-person tracking systems. The data association problem – determining which objects match between frames – becomes very difficult in busy scenes where it is not trivial to segment merged blobs into individuals. For this reason, many earlier systems operate on footage with only a small number of people present at any one time. More-recent approaches attempt to tackle the challenge by combining segmentation and tracking. Below we review a representative sample of multiple-target tracking systems, ordered approximately chronologically.

- $\mathbf{W^4}$: The W^4 system of Haritaoglu *et al.* [64] seeks to answer questions about people's activities, including "**what** they are doing, and **where** and **when** they act", as well as determining **who** is who after an occlusion has occurred. A background model is generated by median filtering the the video sequence over time. Morphological operations are used to clean up the foreground blobs. Tracking of isolated people is achieved by looking for overlapping bounding boxes in successive frames. More precise estimation of a person's motion is accomplished by brute-force correlation of silhouettes between the current and previous frames. The system detects possible human head locations by looking for peaks in the vertical histograms of foreground blobs, which is useful for estimating the number of people in a group. However, it relies on the camera having a very oblique view of the scene, in contrast to the KidsRoom system. *Temporal texture templates*, which encode the average intensity of a person template over time, are used to re-identify the different people after occlusion. A coarse segmentation of groups into individuals is performed with the aid of the head detector. By using a *cardboard model* of people, their heads, torsos, feet, legs and hands can be located. Analysing the relative positions of the body parts allows various postures to be identified, while a lack of symmetry in the silhouette over time can indicate that the person is carrying an object.

- **Reading People Tracker:** Siebel and Maybank's system [148] combines Active Shape Models from the earlier *Leeds People Tracker* [7] with a region tracker. Temporal median filtering is again used to generate the background model. Various heuristics are employed to track blobs over time, even when two nearby blobs merge into a single one, or when low contrast with the background causes a person to be split into multiple blobs. People that become stationary are *temporarily* included in the background model to aid motion detection. An approach similar to W^4 is used to detect head candidates, which are used along with the region tracker to initialise Active Shape Models (ASMs). The shape of each model is refined by searching for edges close to the initial spline. Kalman filtering with second-order motion models is used to track the ASMs. Significant occlusion causes difficulties in the initialisation and updating of the person contours.

- **MCMC-based tracking:** The systems described above have all treated occlusions as exceptional events, after which "normal" tracking of an isolated individual can resume. In contrast, Zhao and Nevatia [185] regard occlusions as a common event to be dealt with on an ongoing basis. Again they use background subtraction and exploit the ground plane assumption to determine the apparent size of objects in the image. A three-ellipsoid model is used to capture the approximate shape of people, while a histogram summarises their appearance. Markov Chain Monte Carlo (MCMC) methods are used to update the parameters of the various person hypotheses; the updates explicitly account for inter-object occlusions. A temporal prior rewards trajectory smoothness and connectivity, and a constant-velocity motion model is employed. Mean-shift tracking is also used to generate short trajectories. New person hypotheses are generated from head candidates, using both the approach of W^4 and edge analysis to look for the the characteristic Ω shape of heads and shoulders. The extensive use of prior knowledge (about the scene and the general appearance of people) yields robust tracking even when occlusions are common.

- **Tracking with crowd segmentation:** Rittscher *et al.* [132] tackle the problem of segmenting foreground blobs made up of many people into individuals. In busy scenes such blobs contain a high degree of ambiguity, particularly in the centre of a crowd. *Greedy* approaches segment objects from the blob one at a time, starting with the "easiest" object. However, by not considering all of the image evidence together, the technique can make mistakes. Rittscher instead performs a global optimisation on features extracted from the foreground blob. The overall segmentation maximises a "merit function" and is more likely to find the correct interpretation of the image data. The crowd analysis module is combined with existing tracking techniques [141, 184] to follow multiple people through the scene.

2.1.2 Top-down and bottom-up tracking

The systems described in the previous section can all be viewed in terms of the structure of figure 2.1: person hypotheses are generated by a variety of means, and these hypotheses are tracked over time. The systems can also be differentiated by the sophistication of their occlusion handling strategies. Earlier approaches typically wait for the objects to separate before resuming tracking. Recently, much research has been focused on maintaining tracking during the occlusion. In all such systems, however, we can distinguish between the high-level knowledge that is employed and the low-level features used as input to the algorithms.

High-level knowledge can include background models of the scene, general models of the objects to be tracked (for example, an Active Shape Model of a human) and motion models that encode information about an object's dynamics. Such knowledge is often referred to as *context* or *top-down information*, and can simplify the tracking task. In section 2.2 we review the most popular types high-level knowledge used in pedestrian tracking, including various background modelling techniques and object detection approaches. We also look at the tracking control strategies employed to fuse this prior information with the incoming image measurements.

Low-level features usually serve as the raw data to be used by the high-level control strategies. Feature points, motion vectors and edges are examples of low-level data that can be extracted from images and videos. *Bottom-up* or *data-driven* tracking approaches use such features without very much high-level knowledge of the task at hand. For example, mean-shift tracking uses histograms to follow an image region through a video sequence, but it usually has no prior model for the object's expected motion. Section 2.3 reviews the most common types of image features, which can be used in either top-down or bottom-up tracking. In section 2.4 we focus specifically on mean shift and other data-driven tracking techniques.

2.2 Tracking control strategies

Many types of high-level knowledge can be applied to people tracking. We can model the typical walking patterns of humans, placing limits on their speed and acceleration in order to make better predictions about their state – most commonly their position – in the next video frame. If we have a stationary camera, we can create a background model to represent the scene; this helps us to detecting moving objects. In the present section we review principled approaches to state estimation, including Kalman filtering and particle filtering. We also describe some of the real-world constraints that can be brought to bear on tracking problems, and look at two techniques used to initialise tracking: background subtraction and direct object detection.

2.2.1 Kalman filtering

Perhaps the most well-known method in control theory as applied to computer vision, *Kalman Filtering* [77] estimates the current state of a dynamical system given its previous state and new observation data. In the context of visual tracking, the state to be estimated is typically the image location of the object that we wish to follow. However, the state may also include other attributes, such as the object's size or its pose. Observations can be in the form of hypothesised object detections, or they can come from lower-level feature detectors. As an example of the former approach, if we are tracking a person we might use the outputs of the Viola-Jones pedestrian detector [165] as our observation model. The Kalman filter assumes that both the state transition model (often called the *motion model* in visual tracking) and the observation model are linear functions of the state. The noise in both cases is modelled by a normal distribution.

When the linearity assumptions are violated, the Kalman filter can perform poorly. The *Extended Kalman Filter* (EKF) [5] allows both the motion and the observation models to be non-linear. The functions are approximated by a Taylor series expansion around the current estimate. However, the EKF can also perform poorly if the models are highly non-linear. *Unscented Kalman Filtering* (UKF) [75] picks a set of sample points ("sigma points") around the mean and propagates them through the models. This allows the mean and covariance of the state to be estimated more accurately than with the EKF.

The Kalman filter has also been modified to deal with noise that is not normally distributed. *Bias-aware Kalman Filters* (BAKF) [53, 41] can estimate the prediction bias provided that an unbiased subset of observations is available. The *Ensemble Kalman Filter* (EnKF) [47] can be used for high-dimensional problems where direct calculation of the covariance matrix is not feasible. A collection of samples is processed by the system, making the method somewhat similar to particle filtering (reviewed in the next section), but with assumption that all distributions are normal. *Gaussian Sum Filters* (GSF) [54] represent the state density as a mixture of Gaussians in order to approximate non-normal distributions.

2.2.2 Particle filtering

Kalman filtering and most of its extensions assume that the state of the system has a Gaussian distribution. *Particle Filtering* is a class of tracking algorithms that represent the state density by a set of samples, each with an associated weight that defines its importance. This allows non-Gaussian distributions to be accurately modelled, provided that we use sufficient samples. Maskell and Gordon present a review of the method and its variations [96]. The original particle filter algorithm [57] used a scheme now known as *Sequential Importance*

Sampling (SIS) to update the particles according to the motion model and in response to the observed data. The new samples are drawn from the *proposal distribution.* When the transition prior, i.e., the motion model, is used for this purpose, the procedure is referred to as *bootstrapping.* The *CONDENSATION algorithm* [70] is an application of bootstrapping to the tracking of contours through a video sequence.

As with Kalman filters, a number of extensions to the basic algorithm have been proposed. *Sampling Importance Resampling* (SIR) [43] implements the SIS algorithm, but adds a resampling step that eliminates samples having low weights. This procedure reduces the problem of "sample impoverishment", in which many of the samples contribute no useful information to the estimation of the system state. A further weakness of particle filters is seen with higher-dimensional state spaces, which will be very sparsely populated with samples, potentially resulting in poor state estimation. The *Auxiliary Particle Filter* [125] draws samples that are more likely to be close to the true state, provided that the process noise is small.

2.2.3 Multiple-target tracking

The state estimation methods of the previous sections are applicable to the tracking of single targets through a sequence. However, when there are multiple targets in the scene, and we wish to track one or more of them, we must consider the problem of *data association*: determining which object measurement corresponds to each of the predicted states [5]. Difficulties can arise because of false detections, missed detections and objects being in close proximity to one another.

Deterministic correspondence methods typically define a cost for each potential prediction–measurement assignment, with the goal being to find a set of assignments that minimises this cost. Various real-world constraints can be encoded in the costs; for example, an object should not change its position significantly between one frame and the next, and its velocity should be reasonably stable [69]. The *Hungarian Method* [80] finds the optimal solution in polynomial time. Rittscher *et al.* [132] use another global optimisation method, this time based on expectation maximisation, to segment crowds of people into individuals. "Greedy" algorithms, on the other hand, make assignments one after another, with cheaper assignments made before more expensive ones. The solution is not guaranteed to be optimal, but it has been shown to work well in the multiple-person tracking systems of Wu and Nevatia [169][170]. Correspondence matrices can also be resolved by *ad hoc* methods in order to deal with objects appearing in and disappearing from the scene, and foreground blobs splitting and merging [36, 100, 76].

Statistical correspondence methods are analogous to Kalman and particle filters, but, aside from tracking multiple targets, they also perform data association in a statistical framework. The *Joint Probabilistic Data Association Filter* (JPDAF) [23] associates multiple targets in the current frame with multiple targets in the previous frame. It assumes that the number of objects remains constant. The *Multiple Hypothesis Tracker* (MHT) [131] can accommodate missing measurements, e.g., in the case of an occlusion, and thus overcomes this limitation. It also uses several frames to determine the most likely tracks for a given set of measurements. Cox and Hingorani [34] develop a more efficient version of the algorithm. An alternative approach, the *Probabilistic MHT* (PHMT) [155] avoids an exhaustive search of correspondences by assuming that the associations are statistically independent. In contrast to the "hypothesise-and-test" nature of MHTs, Ryoo and Aggarwal [137] present their "observe-and-explain" technique for tracking multiple people through occlusions with greater efficiency.

A number of methods for tracking more than one target using particle filters have been developed; they belong to the class of "stochastic sampling" techniques. Pérez *et al.* [122] construct a joint likelihood term in order to track multiple objects using a particle filter. The Viola–Jones detector [164] is used in combination with a *mixture particle filter* [163] by Okuma *et al.* [114] in their *Boosted Particle Filter* to track many ice hockey players around a rink. Yang *et al.* [175] use Haar-like features and edge orientation histograms to track several objects. However, interactions are not explicitly modelled, and so only short occlusions can be handled. Zhao and Nevatia [185] also use Markov Chain Monte Carlo (MCMC) methods to track multiple targets, even as they interact and undergo extended occlusion. Each point in the state space is a particular configuration of the people. Smith *et al.* [149] use *trans-dimensional* MCMC methods [58], which improve the efficiency of the sampling. In both of the latter systems, the number of objects being tracked is allowed to change over time.

2.2.4 Constraints used in tracking

Various types of high-level knowledge can be brought to bear on a tracking problem. Knowing the number of objects in the scene *a priori*, for example, or having their apparent size in the image specified can greatly simplify the task. Human dynamics are constrained in a number of ways that can be exploited by a tracking system to solve the data association problem. Yilmaz *et al.* [179] list some heuristics that are most applicable when objects are represented by a set of points. *Proximity* and *maximum velocity* require that a given point does not move more than a certain distance between one frame and the next. The threshold is informed by the speed of people's movements in the scene and how much of the image they occupy (i.e., the zoom of the camera). The *smooth velocity change* assumption limits the acceleration that an object can undergo, while multiple points belonging to the same target should display *common motion*, i.e., follow similar trajectories. (In three-dimensional tracking, the stronger *rigid motion* constraint may be more appropriate.) Common motion has been used to segment crowds of people into individuals [147, 129, 18] and to track football players [98]. Since it is not necessary for all of the points belonging to an object to be observed in a given frame, the technique is able to cope with partial occlusions. The *Particle Video* approach of Sand and Teller [138] is similar to these feature tracking methods, but it yields a denser collection of points. Adjacent particles obey the constraint of following similar paths, provided that they are part of the same object.

Real-world knowledge about object *trajectories* can also prove useful in tracking. Given templates of an object at the start and end of its path through the scene, Sun *et al.* [156] are able to determine the full, end-to-end trajectory in spite of occlusions. They extract short track segments from the entire video sequence that *may* correspond to the object. These segments are only accepted if they can be joined together in space-time to form a smooth path. Wu *et al.* [171] extend the KLT tracker [145] to incorporate a new "time-reversibility constraint". A point is only retained if tracking it in both the forwards and backwards directions yields a similar answer. Like the original KLT method, the new approach only operates on pairs of frames at any one time. In contrast, the "recurrent tracking" of Pan *et al.* [116] looks at the entire trajectory to determine if an object has been followed correctly. The target is first tracked forwards in time in the normal manner. At the end of the sequence, the model is reinitialised with the image region corresponding to the tracker's final location, and the object is then tracked backwards in time. In the case of accurate tracking, the forwards and backwards trajectories will match each other. However, a lost track or inaccurate tracking will result in the model being reinitialised with a "bad" image region, and the forwards and backwards trajectories will be very different.

We extend this approach in chapter 7: whenever a tracking failure is detected, we shorten the video sequence and repeat the process until the trajectories match. Our *track validation* algorithm then attempts to follow the object through the remainder of the sequence using an updated model of the target taken from the point of reinitialisation. This strategy allows the tracker to accommodate significant changes in the object's appearance.

2.2.5 Initialisation

Before tracking can commence properly, the target model must be initialised. If we are concerned only with the effectiveness of the tracking technique itself, it may suffice to initialise the target by hand. However, developing an autonomous system requires that we use automated methods. These can be divided into two main classes: background subtraction techniques and class-specific object detection approaches. In the following sections we review the most influential methods in each class.

Background subtraction

Background subtraction can serve as a very useful means of locating moving objects in a scene. However, it must contend with numerous difficulties, including changes in lighting, camera jitter and low object–background contrast. Several reviews of popular techniques have been conducted in the past [124, 25, 130]. We follow the structure of Moeslund [105] and examine the techniques in terms of representation, classification, update and initialisation of the background.

- **Background representation**: The majority of background models treat each pixel and its neighbours independently. The *Pfinder* system of Wren *et al.* [168] models a pixel as having a unimodal Gaussian distribution. Each pixel in the background model is updated by a weighted contribution from the current frame. The goal is to allow the model to adapt to slow illumination changes in the scene. In order to cope with pixels having a non-Gaussian distribution, Stauffer and Grimson introduced the *mixture-of-Gaussians* (MoG) approach [150]. The model is intended to incorporate effects commonly found in outdoor scenes, e.g., shadows, reflectance and repetitive object motion. Elgammal *et al.* [45] use kernel density estimation (KDE) to represent each pixel by its previous N values. In performing background subtraction, they compare a pixel to nearby pixels in the background model, and not just to its corresponding pixel. The aim is to cope with the effects of camera jitter and swaying trees in the background. Oliver *et al.* [115] apply an eigenspace decomposition to the spatio-temporal volume of the video sequence (or at least some portion of it). The background is modelled by the most descriptive eigenvectors extracted from the volume. By projecting the input image onto the eigenspace, the moving objects are detected. Since the representation captures the global changes in the sequence, the effects of illumination variation are reduced.

  In moving beyond the unimodal background model, the ability of the more advanced techniques (MoG, KDE) to disregard uninteresting motion is arguably gained at the expense of lower sensitivity to objects that we do wish to detect. To lessen this undesirable effect, a number of authors have developed techniques for background subtraction in dynamic scenes. Monnet *et al.* [107] model pixels as auto-regressive processes in order to predict their subsequent values. The method is effective at suppressing false-positive detections of motion. Zhong and Sclaroff [186] also use an auto-regressive model, but employ a robust Kalman filter to update the current appearance of a dynamic texture, such as a moving escalator or the surface of the sea. Foreground objects moving against these textures can then be detected. Tian and Hampapur [158] combine temporal differencing and optical flow to find "salient motion" – objects moving in a consistent direction for a period of time.

- **Background classification:** Pixels belonging to moving targets are typically identified by subtracting the current frame from the background model. However, this foreground mask will usually contain many pixels that do not belong to moving objects of interest. These are caused by uninteresting motion in the background, camera jitter, lighting changes and reflections. Often there will also be "false negatives" – pixels that ideally would be part of the foreground mask but which are too similar to the background model to be considered as such.

  The most common approach to dealing wish such erroneous pixel classifications is to postprocess the foreground mask with various filters [35, 45, 100, 184]. The *opening* morphological operation can be used to remove isolated noise pixels, while *closing* is often used to fill holes within objects. Median filtering with a large mask can be seen as intermediate between the two [40] and its execution time is independent of the mask size used [123]. Markov Random Fields have been used as an alternative to the above techniques to enforce the constraint that adjacent pixels are likely to belong to the same class, i.e., foreground or background [139, 142].

- **Background updating:** Most of the methods described previously are designed to cope only with slow changes in the appearance of the scene. However, more drastic changes occur, for example, when the sun appears from behind a cloud, or when a light is switched on or off in an indoor setting. The *Wallflower* system of Toyama *et al.* [160] uses three types of information – pixel-, region- and frame-level data. The last of these can be used to detect sudden, global changes in the scene, at which point a new background model can be initialised or a previously-built model can be used. A similar approach is used by Javed *et al.* [71]. In their codebook model, Kim *et al.* [79] update all of the codewords in response to global illumination changes.

  When a background modelling system is not required to operate online – i.e., to make immediate classifications as each new frame arrives – we can use data from both the past and the future to improve the results. The *Visual Surveillance and Monitoring*

(VSAM) system of Collins *et al.* [27] delays decisions by one second in order to distinguish between moving foreground objects, objects that enter the scene and stop, and slow illumination changes. Figueroa *et al.* [51] also use past and future values so that they can apply their morphological levelling operation to a pixel's brightness history. The technique provides effective background modelling for football sequences undergoing significant brightness changes.

- **Background initialisation:** Early background modelling approaches required the scene to be empty at the moment of initialisation. This constraint often proves impossible to satisfy in real scenarios, leading to methods that can initialise a model from video sequences containing moving objects. In the W^4 system of Haritaoglu *et al.* [64], the median of each pixel's history during a learning phase is used as the background model. The resulting synthetic image will be free of moving objects provided that every pixel spends at least 50% of its time in the background state. Eng *et al.* use the same approach in their drowning detection system [46].

  The technique of Gutchess *et al.* [60] does not assume that the background is visible for more than half of the length of the training sequence. Instead, it locates short intervals where a pixel has a stable value, and chooses the one having the lowest average motion (which is measured by optical flow). Wang and Suter [166, 167] also find short, stable subintervals in the video sequence, but their approach is to choose the interval having the highest ratio of data points to variance. Both of these approaches are capable of initialising empty background models from busy scenes.

The particular method used for background modelling and updating will usually be determined by the nature of the video data to be processed. For example, in indoor scenes there is usually no need to account for slow lighting changes, but cast shadows may be an important issue. Similarly, sophisticated techniques for background initialisation are only required if the scene is never observed empty of objects. In any case, background subtraction can only serve as one part of a larger system. When multiple people are to be tracked, occlusions will lessen the value of the approach. However, the foreground mask can still prove useful in generating hypotheses for the locations of people in the scene. In chapter 5 we combine a background model with a multiple-part tracker to improve the robustness of the mean-shift technique.

Detection and segmentation

In certain scenarios it is not possible to use background subtraction to initialise a tracker. Very busy scenes and moving cameras can make the application of the method difficult, if not impossible. Direct, class-specific object detection can serve as an alternative initialisation mechanism in these cases. There has also been much recent interest in methods capable of segmenting crowds of people into individuals. These techniques are usually tightly integrated with multiple-object tracking. Here we review the most popular approaches to person detection and crowd segmentation.

Papageorgiou *et al.* [117] presented a framework for detecting objects in static scenes. They used wavelet basis functions as features and trained a support vector machine (SVM) classifier to detect a particular class of object. Results for both face and pedestrian detection are shown. Gavrila [55] detects pedestrians with a two-stage approach. First, candidate regions are identified using chamfer matching on a distance-transformed edge image. Next, the candidates are verified or rejected using a radial basis function (RBF) classifier. In contrast, Mohan *et al.* [106] take a bottom-up approach and find people in images by combining individual detectors for the head, left arm, right arm and legs. Another data-driven technique, the cascaded classifier of Viola and Jones [164] detects faces in still images. Each stage in the cascade uses a collection of simple Haar-like features, which can be generated very quickly. In a later paper [165], simple motion features are extracted from successive video frames so that the method can be used for detecting pedestrians. A class of powerful image features, *Histograms of Oriented Gradients* (HOGs) are used by Dalal and Triggs [38], together with a linear SVM, to build an effective person detector. In subsequent work [39], they combine HOGs with histograms of optical flow features to find people in video sequences. Leibe *et al.* [81] employ a hybrid top-down–bottom-up approach that uses local cues (a codebook of learned image patches) and global cues (edge-based human shape templates) to find people in images even when they overlap and partially occlude one another. Wu and Nevatia also combine high-level and low-level information to locate people in images; the algorithm fuses the responses of multiple edgelet-based body part detectors [170] into a single person hypothesis.

As explained in section 2.1, detection and tracking can be regarded as complementary processes that operate in a loop. Person detection serves to initialise a tracker, which proceeds to follow its target through the sequence. Further detections may occur, but tracking can continue even in their absence. However, the tracker is itself prone to errors, sometimes losing its target. In such cases, the detector can be used to correct the tracking process before it fails. Indeed, it is sometimes possible to track a person by simply detecting them in (almost) every frame – a *tracking-as-detection* approach [152].

In crowded situations where occlusions are common, direct person detection may not be possible. It is necessary instead to segment the crowd into individuals. The later tracking systems of section 2.1.1 (those of Zhao *et al.* [185] and Rittscher *et al.* [132]) follow this approach, as do other researchers. Lin *et al.* [87], for example, use “foot candidates” to estimate appearance models and segmentations simultaneously using a greedy optimisation scheme. By combining low-level edge features and higher-level cues (such as closed boundaries), Sharma and Davis [87] detect and segment people in static images. To find humans in a crowd, Rodriguez and Shah [133] learn a codebook of person shapes, which are then used to “vote” for a particular pose hypothesis. The segmentation is refined by searching for nearby object contours. A specific head-and-shoulders detector is instead used by Tu *et al.* [162] to generate person hypotheses. These are combined with motion and colour cues in an expectation maximisation algorithm to find a consistent segmentation of the crowd.

As with background modelling, detection and segmentation approaches cannot solve the tracking problem on their own. Occlusions in particular cause difficulty for most person detectors. However, we can use the techniques to generate person hypotheses, which can be verified or rejected by the subsequent tracking stage.

2.2.6 Summary of control strategies

This section has reviewed the main types of high-level information that are brought to bear on visual tracking problems. Models of the typical dynamics of humans can be encoded in a state-estimation framework, such as Kalman filtering or particle filtering, to predict the next location of the person to be tracked. When multiple people are present in the scene, methods that account for interactions and that can solve the data association problem are required. Such approaches can be classified as deterministic (e.g., the Hungarian Method) or statistical (e.g., the Probabilistic Multiple Hypothesis Tracker). Real-world constraints also play a large role in object tracking. For example, if a person is represented by a number of points, those points should display common motion as the sequence proceeds.

Initialisation of tracking also relies on high-level information. It is frequently accomplished by means of background subtraction, but direct person detection can also be used. In either case, hypotheses about the locations of people in the scene are generated, which the tracker subsequently attempts to verify or reject. The two processes – tracking and hypothesis generation – operate in a loop.

Aside from the various sources of high-level information just described, tracking requires that the target(s) be represented by a set of features; the following section reviews various feature types, including points, motion, edges and regions. However, there is also a class of *data-driven* tracking algorithms (section 2.4) that can follow objects while making little use of top-down information. Many of the same types of features are used by these methods. Although such bottom-up approaches, e.g., the mean-shift technique, can be embedded in top-down tracking systems, doing so is not essential to their operation.

2.3 Features used for tracking

The control strategies of the previous section are used to determine the location of the tracked object(s), given a collection of measurements from the current frame. These measurements are described as *features*, and they can be placed into different classes – points, motion-based, contours, regions – depending on the information they attempt to encode.

Objects can be represented in a number of different ways, which usually amounts to a scheme for grouping features. Shape-based representations can take the form of a set of points, a geometric primitive (e.g., an ellipse or rectangle), a contour, a silhouette, an articulated model (perhaps using cylinders in 3D, or ellipses in 2D) or an object skeleton. Appearance-based representations capture the pixel values *inside* the object boundaries in different ways: as probability density functions (using a mixture-of-Gaussians model, kernel density estimation or histograms) or as templates. *Active Appearance Models* [33] capture both shape and appearance characteristics jointly.

In the following sections we review the different types of features used in tracking. The most well-known examples in each class are described.

2.3.1 Points

Feature point, or interest point, detectors find parts of an image that have a distinctive structure in their locality [179]. The term *point* is used somewhat loosely because such such features can extend over a reasonably large area of the image; Cannons [19] refers to them as *discrete features.* A *descriptor* can be associated with a detected feature point in order to characterise the region around the point. Aside from tracking, such descriptors have applications in stereo matching [97], object recognition [92] and image retrieval [101].

An investigation of the performance of several popular detectors and descriptors was performed by Mikolajczyk and Schmid [102, 103]. A desirable property of a descriptor is high *repeatability* (the likelihood that a point will be detected at the corresponding location in two similar images of a scene) in spite of image rotation and changes in scale, viewpoint and illumination.

Detectors

One of the earliest interest point detectors is that of Moravec [109]. Patches in the image that are dissimilar to nearby regions (as measured by their sum of squared differences) are considered to be “corners”. Instead of operating on the image directly, Harris and Stephens [65] use image gradients. Corners are thought of as image patches with a large gradient in every direction. The *KLT tracker* [145] finds points using a method very similar to the

Harris detector and tracks them using the Lucas–Kanade optical flow method [93]. The *SUSAN* corner detector [153] evaluates a function in a circular region around the point to be tested. It measures the proportion of the circle's area that has brightness similar to the centre pixel. Corners are image points where the function attains a local minimum. Rosten and Drummond's *FAST* feature detector [134, 135] also tests a circular region. It looks for contiguous pixels that are all lighter or all darker than the centre pixel, and uses a decision tree for efficiency.

The detectors described above typically perform poorly (in terms of repeatability) when the scale of the image changes. The *Harris-Laplace* detector [104] uses a scale-space approach [88] to find the "characteristic scale" of a feature point, thus making the method scale invariant. Scale-space implementations typically apply Laplacian-of-Gaussian (LoG) filters of different sizes to the original image, where each successive level in the scale space blurs the image to a greater degree. (LoG filters are often approximated by Difference-of-Gaussian filters [91].) *Affine-invariant* feature detectors can also cope with rotation and shearing, and so can be used to accommodate (approximately) viewpoint transformations. Mikolajczyk's *Harris-affine* detector [104] starts by identifying scale-invariant Harris-Laplace features and then iteratively adapts the shape of the point neighbourhood. *Maximally Stable Extremal Regions* (MSERs) [97] are also affine-invariant. The detector finds blobs whose area changes slowly in response to a change in the brightness threshold applied to the image.

Descriptors

The most well-known of the image *descriptors* is the *Scale-Invariant Feature Transform* (SIFT) [92]. It uses an image pyramid to locate keypoints that are minima or maxima of their neighbours in both scale and space. Points that have low contrast or that are poorly localised along an edge are discarded. An orientation is assigned to each keypoint based on its local image gradient. To generate a descriptor, 16 separate orientation histograms are calculated in a 4×4 neighbourhood around each keypoint. Each histogram has 8 bins, which are all concatenated to form a 128-element vector (the descriptor). A nearest-neighbour approach is used for keypoint matching, and the search is made efficient using *kd-trees* [108]. *Random Sample Consensus* (RANSAC) [52] is used together with SIFT to implement robust object recognition. Many ideas from the descriptor, including strong brightness normalisation, are used in *Histograms of Oriented Gradients* (HOGs) [38], which were first applied in the area of pedestrian detection. *Speeded Up Robust Features* (SURF) [8] can be calculated more quickly than SIFT features, and achieve a similar level of repeatability. The recent CenSurE (Center Surround Extrema) descriptor [2] is also very efficient to calculate, and has been applied to the task of visual odometry.

The evaluation methodologies used in the area of feature descriptors, particularly HOGs, influence our research into mean-shift tracking (chapters 5 and 6). We perform comprehensive, quantitative assessments of the effect of different parameter settings on the performance of the method.

2.3.2 Motion-based features

Motion can serve as a very strong cue in object detection and tracking. The simplest motion detection algorithms use image differencing, which typically identifies the boundaries of moving objects. The *Visual Surveillance and Monitoring* (VSAM) system of Collins *et al.* [27] uses three frames instead of two to make the calculation more robust. Such methods do not detect motion in the interior of objects, however, and so background subtraction is commonly employed instead (section 2.2.5). In the domain of object detection, Viola *et al.* [165] use image differencing and Haar-like features to identify moving pedestrians in video sequences. The *Motion History Images* (MHIs) of Bobick and Davis [15] capture the recent history of an image region that is in motion and can be used for action recognition.

Optical flow is a widely used technique for determining image motion. Several methods have been developed over the last thirty years. The simplest, block matching, finds the displacement that minimises some error measure between a block in the current frame and the corresponding block in the previous frame. Common error measures include normalised cross-correlation, sum of absolute differences (SAD) and sum of squared differences (SSD). The technique is computationally expensive, and so a number of gradient-based approaches have been devised. The *Lucas–Kanade* method [93] estimates the temporal and spatial image derivatives, and uses them to formulate the *image constraint equation.* To solve the equation, Lucas–Kanade assumes that the flow field is locally constant, i.e., constant within a small neighbourhood. The Horn–Schunck method [67], on the other hand, assumes that the brightness of a point does not change significantly over time – the *brightness constancy constraint.* Various other optical flow techniques have been proposed [9, 157]. Sidenbladh [146] detects moving people by combining the method of Black and Anandan [14] with a support vector machine.

A dense flow field will usually contain a number of incorrect vectors. For this reason, the *KLT tracker* [145] only calculates flow vectors at Harris corners. Although relatively few motion vectors are returned, they are more reliable than those from dense methods. KLT features have been used used to identify motion in a number of applications, including crowd counting and segmentation [18, 129].

2.3.3 Edges and contours

Segmentation-based and contour-based tracking methods exploit the tendency for most objects to contrast with the background. Their boundaries therefore give rise to edges in the image, which can be used to segment the object from the scene.

Segmentation

Segmentation is normally applied to static images. The mean-shift approach [30] clusters points in a joint colour–spatial space, resulting in a partition of the image into compact regions of homogeneous colour. The results are sensitive to the settings used for the various parameters. Alternative methods represent each image pixel by a node in a graph. Edges linking adjacent pixels have associated costs which depend on their colour, brightness and texture similarity. The *graph cuts* technique [172] finds the partition that minimises the cost of cutting these edges, giving rise again to homogeneous regions. *Normalised cuts* [144] attempts to reduce the problem of oversegmentation by considering the weights of all edges in the graph, and not just those involved in the cut. Graph-based segmentation approaches have been used in place of morphology to refine the results of background subtraction [139, 142]; the real-valued foreground mask is converted to a binary mask in which isolated noise pixels are suppressed and holes are filled in.

Object detection is commonly performed by matching edge templates of the target class against edge images of the current frame – a form of shape-based segmentation. Zhao and Davis [181] apply chamfer matching to a distance-transformed edge image in order to detect the head-and-shoulders contours of people. Leibe *et al.* [81] use a similar approach as the verification step in their system for segmenting individuals in crowded scenes.

Contour tracking

In contrast to segmentation, contour-based tracking methods operate over a sequence of frames, attempting to identify the object boundary at each time step. The CONDENSATION algorithm [70] fits a pre-defined spline to edges extracted from the image. A particle filter is used to update the state variables. Chen *et al.* [24] use an ellipse as the contour to track a person's head. Edge intensity and colour are used as measurements. The contour state is estimated by a Hidden Markov Model, whose transition probabilities are themselves estimated by a Joint Probabilistic Data Association Filter (JPDAF).

Explicit contour representations, such as the splines and ellipses of the above approaches, cannot handle splits and merges of objects. Techniques that attempt to minimise some measure of *contour energy* are often used to overcome this difficulty. The contour energy is specified as the sum of regularisation constraints (the contour must be continuous and its curvature must fall within certain bounds) and appearance-based constraints (the contour should favour strong image edges or texture and colour transitions). The *Active Shape Model* [7, 148] uses this approach to track people walking through a scene.

It is common for control points to be used to provide an explicit representation of the contour. Alternatively, the object shape can be defined implicitly by means of level sets (similar to a distance transform applied to the contour). Bertalmío *et al.* [10] use optical flow at the object boundary to evolve the level set. Yilmaz *et al.* [180] do not use temporal information, but instead gather image statistics from around the object's boundary, which is initially marked by hand. Other contour-tracking methods use background subtraction to perform the initialisation step.

2.3.4 Regions

Whereas contour-based approaches are concerned with the boundaries of an object, region-based tracking techniques represent the appearance of the target *within* the boundary. Template matching is one of the most common methods in this category. Lewis [84] presents a number of optimisations that can be applied when calculating the normalised cross-correlation (NCC) of a template and an image. To further lower the computational cost, Schweitzer *et al.* [140] develop a method that uses the "integral images" of Viola and Jones [164] to find a fast approximation to NCC. However, basic template matching of this kind can only accommodate translational image motion. In contrast, the approach of Hager and Belhumeur [61] extends the Lucas–Kanade optical flow method to account for scaling and rotation. It also incorporates "basis images" into the formulation to cope with changes in the scene illumination.

In the area of human–computer interaction, the tracking of people's heads is a popular application of region-based techniques. Birchfield [11] combines two scores – a gradient measure for the head boundary and a colour measure for the hair and skin – to follow a person's head on screen. The combined similarity measure is evaluated in a brute-force fashion in a neighbourhood around the previous head location. Fieguth and Terzopoulos [50] instead approximate the head by five rectangular regions. A similarity score (based on the mean colours of these regions) is evaluated at only eight neighbouring image positions, with the tracker moving to the location with the highest score. Partial occlusion is handled by selectively dropping some of the rectangles from the comparison. Jepson *et al.* [72] treat a region to be tracked as a combination of stable, quickly changing and outlier pixels. Using the example of a person's face, the pixels around the nose area will change relatively slowly, while the eyes and mouth can transform their appearance quickly; outliers are the result of occlusions. Identification of the stable pixels allows a dynamic object to be tracked over time.

Bradski's *CAMSHIFT* system [17] was one of the first to use the mean-shift technique [29] for tracking. It uses a single-channel (hue) histogram to represent the face region, since human skin pixels are tightly clustered in hue space. After the new face location has been found, image moments are used to update the size and the orientation of the region. The version of mean-shift tracking used in this system is not the same as that developed later by Comaniciu *et al.* [31, 32]. Specifically, Bradski does not use spatial kernels in his approach. We will discuss the standard mean-shift technique and its enhancements in section 2.4.1.

2.3.5 Summary of tracking features

In contrast to the top-down information exploited by many tracking systems, features serve as the image measurements used by algorithms that estimate an object's location in the current frame. It is convenient to group features into different classes depending on the image and video information they encode. Point-based methods attempt to find distinct, small regions of an image. Ideally, these points will be detected again in subsequent frames, and so tracking becomes a problem of "joining the dots" – associating the correct points with one another over the course of the video sequence. It is also useful to characterise the appearance of the image patch around a given point. Such *descriptors* provide more information for a point matching algorithm to exploit.

Image motion is another strong cue that is often used in tracking. Optical-flow-based methods typically operate on a pair of images at a time, attempting to determine the flow vectors for every image pixel. Alternatively, point- and motion-based features can be combined, so that a motion vector is only returned for an image location that is distinct in appearance (a feature point). The resulting motion field is sparser but less noisy than standard optical flow. It is also possible to summarise the movement in an image region over longer periods of time by using Motion History Images (MHIs), feature tracking or the Particle Video approach [138].

By segmenting each video frame into its constituent objects, we can follow a designated target over time. Image edges form the basis of most such methods; they can arise from transitions in colour, brightness or texture. Graph-based algorithms use edges to define the *cost* of a given segmentation. The more homogeneous the regions in a partitioned image are, the lower the cost will be. It is also possible to specify a particular shape of contour in advance and track it through the sequence. The contour has a greater affinity for strong image edges, and its shape is allowed to evolve, within limits, over time. Active Appearance Models and the CONDENSATION algorithm are two popular techniques in this class.

The internal appearance of an image region is another useful feature for tracking. Descriptions of the image patch range from the detailed (e.g., a template) to the summary (e.g., a histogram). They can characterise various low-level image attributes including colour, intensity, texture or image gradients. Tracking entails finding the region in the local neighbourhood that best matches the model. Template-based representations typically require a brute-force search strategy, whereas histograms allow the use of gradient ascent methods such as mean shift. However, in chapter 7 we develop a template-based tracker that uses gradient ascent optimisation. It is both robust and computationally efficient.

2.4 Data-driven tracking

As explained in section 2.2.6, there is a class of tracking algorithms that make little use of high-level, contextual information. Such *data-driven* methods use the image features on their own to determine the next location of the tracker. We look first at the mean-shift technique and enhancements that have been made to it in order to accommodate transformations besides simple translation. Next we review approaches that incorporate spatial structure within the mean-shift histogram and that exploit background models of the scene to improve the tracker's robustness. Variations on mean-shift and data-driven trackers that do not represent image regions using histograms are also discussed.

2.4.1 Mean-shift tracking

Mean-shift tracking [31, 32] tries to find the area of a video frame that is *locally* most similar to a previously initialised model. The image region to be tracked is represented by a histogram. A gradient ascent procedure is used to move the tracker to the location that maximises a similarity score between the model and the current image region. (A detailed mathematical description of the method is given in chapter 4.) The technique is computationally very efficient, and usually converges in three or four iterations. The original version was designed to recover the translation and scaling that an object had undergone. However, Collins [28] observed that the approach for determining scale changes in an object is sometimes unreliable. He proposed an explicit scale-space approach for tracking objects whose size varies over time. Further extending the flexibility of gradient ascent tracking, Zivkovic and Kröse's method [187], based on expectation maximisation, is capable of following regions undergoing translation, rotation and changes in scale and aspect ratio. The algorithm of Nguyen *et al.* [113] can also recover the scale of the object by treating the pixel coordinates of the target as latent variables to be estimated, again by an expectation maximisation algorithm.

Regions undergoing affine transformation are handled by Guskov's tracker [59], which can also accommodate illumination changes.

One limitation of the previous methods is that the inter-frame displacement of the object must be small; otherwise, the tracker will lose its target. Initialising multiple trackers around the expected location of the object can overcome the difficulty, and is the approach taken by Porikli and Tuzel [127]. Adam *et al.* [1] observe that the *integral histogram* [126] allows comparisons of histograms to be made at every location in a large subimage at modest computational cost. While not a gradient ascent approach, their *FragTrack* technique yields similar results to mean-shift tracking, with the advantage that fast and partially occluded objects can be followed. In a similar vein of maintaining multiple hypotheses, Shen *et al.* [143] seek to avoid *local* maximums of the histogram similarity surface, finding instead the global maximum. They begin with an oversmoothed version of the surface and find its mode. By reducing the bandwidth of the tracker, other modes appear, but the tracker continues to converge towards the highest one. The approach is termed *annealed mean shift*, by analogy with other annealing methods. It limits the influence of distractors – objects with a similar appearance to the target – on the tracker.

2.4.2 Multiple-part models and background exclusion

Histograms, which are used by mean-shift tracking as the target and candidate models, suffer from a lack of spatial structure, making them less discriminative than other object representations. Various authors have formulated mean-shift approaches with multiple-part models, where the region to be tracked is divided into a number of subregions. The goal is to encode a degree of spatial structure in the model while retaining the computational efficiency delivered by the gradient ascent nature of mean shift. Dong *et al.* [173] partition the region to be tracked into concentric circles and derive a mean-shift optimisation formula for the multiple-part model. In contrast, Maggio and Cavallaro [94] first divide the ellipse into quadrants and then into rings. Their approach allows the subregions to overlap. Parameswaran *et al.* [118] use four vertically stacked rectangles to form the image region to be tracked. Each rectangle has its own spatial kernel at the centre. All three methods demonstrate an improvement in tracking accuracy over the basic approach.

When a background model of the scene is available, mean-shift tracking can benefit from its use. Zhao and Nevatia [185] incorporate a "background exclusion" term into the technique. Their mean-shift tracker favours image regions that are similar to the object model (as before), but that are also dissimilar to the corresponding region in the background model. The approach is analogous to background subtraction, but it does not require a separate image differencing step.

An ellipse is a poor approximation of the shape of a person to be tracked, and consequently the model will contain data from pixels that belong to the background. Yilmaz [178] addresses this problem by using an asymmetric kernel, defined by level sets, that matches the shape of the object. It is initialised by hand in the first frame, and is more suited to rigid objects. Jeyakar *et al.* [73] also reduce the influence of background pixels on the mean-shift calculation. However, their approach requires that the target be seen in isolation, surrounded by background pixels, with no other objects nearby. The method of Leichter *et al.* [82] assumes that objects have distinct boundaries. Three kernels – one each for the object, boundary and background – are used to limit the influence of non-object pixels. (The boundary kernel uses strong image gradients as its cue.) The approach allows for safer updating of the model as the video progresses.

In chapter 5 we develop a method to combine multiple-part models with the background exclusion constraint in a single mean-shift tracker. And in chapter 6 we study the effect of using different kernel types on tracking performance.

2.4.3 Variations on mean shift

The standard mean-shift tracker uses the Bhattacharyya coefficient to compare the model and target histograms. A number of methods have been proposed that use alternative similarity measures, model representations and optimisation techniques for tracking. Elgammal *et al.* [44] develop a joint feature–spatial distribution to encode both the structure and the appearance of the target. Setting the bandwidth parameter to zero causes the method to reduce to sum-of-squared-differences tracking (i.e., template matching). At the other extreme, as the bandwidth tends towards infinity, the approach is equivalent to a histogram-based tracker. The Kullback–Leibler distance is used to compare the model and target distributions. In later research, Yang *et al.* [174] create a new similarity measure based on this joint feature–spatial representation. It is shown to be more discriminative in higher dimensions than the Bhattacharyya coefficient. They also develop a mean-shift procedure that can operate with the measure. Real-time tracking is achieved by employing the *improved fast Gauss transform* (IFGT) [176]. However, Leung and Gong [83] observe that only a small number of pixels from the model and target are necessary for robust tracking using the new measure. Their "random sampling" approach makes it unnecessary to apply the IFGT. Another similarity score, the *earth mover's distance*, is used by Zhao *et al.* [182] for person tracking. The model and target distributions are summarised by *signatures* in order to allow an efficient, differential method to be used for optimisation.

Zhao's work is just one example where gradient ascent techniques besides mean shift have been used for tracking. Hager *et al.* [62] recast the histogram-based tracking equations in matrix form and derive a Newton-style optimisation procedure. It converges more quickly than the traditional mean-shift approach. The use of matrices also allows for an analysis of the situations that give rise to tracking difficulties. For example, empty histogram bins in the models can lead to instability in gradient ascent methods. Furthermore, the choice of spatial kernel determines which kinds of image motion the tracker can detect. Fan *et al.* [48] use this notion of "kernel observability" to improve the method's robustness. Multiple kernels are combined in a single tracker to increase the chances that at least one of them can correctly recover the object's motion. In later work [49] the same authors develop a technique to quantify how reliably a given image region can be tracked. It is shown that, ideally, the centres of mass of the various colours that make up the region should be evenly distributed around its centre. They also derive a gradient ascent approach for moving a tracker from a "bad" to a "good" location before initialisation. Exploring other optimisation techniques, Liu and Chen [90] compare the *trust-region* method to mean shift. They find that the former typically converges to a better local maximum of the similarity surface, making the tracker less susceptible to distractors.

2.4.4 Alternatives to histograms

Not all data-driven trackers use histograms or distributions to represent the target. It is common to use the image patch directly, or a transformed version of it (e.g., wavelet coefficients), as the model to be tracked. Template matching with a brute-force neighbourhood search (section 2.3.4) can then be employed to find the object in the next frame. (In chapter 7 we show that gradient ascent optimisation can be used instead.) Looking at other target representations, the *spatiograms* of Birchfield and Rangarajan [12] encode a degree of spatial structure of the object. Aside from the count of pixels having a particular colour (as in a standard histogram), the spatiogram also records the mean and covariance of the co-ordinates of the those pixels. A mean-shift approach is used to track image regions. Huang's *correlogram* [68] also stores information about the spatial distribution of pixels: for a pair of colours u and v, it records how likely it is that pixels of those two colours are found at a given distance from one another in an image region. Zhao and Tao [183] use a cut-down version of the correlogram to recover both the position and the orientation of an object using an extension to the mean-shift algorithm.

2.4.5 Summary of data-driven tracking

The present section has reviewed various data-driven tracking techniques. These algorithms use only a limited amount of high-level information, and instead employ the image features directly to follow their target. Arguably the most popular method in this class, mean-shift tracking is a gradient ascent approach that iteratively moves to the most similar region in the local neighbourhood. In its basic form, mean shift is only effective at determining translational motion in a video sequence; extensions are required to allow it to accommodate changes in scale, image rotation and affine transformations. By running several nearby trackers simultaneously, the algorithm can also cope with fast object motion.

Mean shift represents the image region to be tracked by a histogram, which discards all spatial structure from the object, reducing its discriminative power. Multiple-part models can be used instead, which allow the tracker to retain some of the region's spatial information while still operating as a gradient ascent approach. In addition, it is possible to incorporate background models of the scene, when available, into mean shift in order to make it more robust to distractors.

We have also reviewed gradient ascent tracking methods that use alternative similarity measures and region representations to those used by the mean-shift technique (i.e., the Bhattacharyya coefficient and histograms, respectively). Joint feature–spatial spaces are better able to encode the spatial structure of a region than a histogram, and fast gradient ascent tracking techniques exist for this representation. Mean-shift optimisation itself can be replaced by Newton-style methods, which generally converge more quickly. Such approaches also lend themselves more easily to theoretical analysis, allowing us to determine whether the results returned by a tracker are likely to be trustworthy. Correlograms provide a further alternative to histograms; they are useful for determining the rotation that an object has undergone.

In the following chapters we evaluate the effect of modifying various elements of mean-shift tracking. We use the techniques discussed in the present section to incorporate spatial structure into the method (chapter 5), and we assess the performance of mean shift as we vary several of its parameter settings (chapter 6).

2.5 Summary

This chapter has reviewed the state of the art in pedestrian tracking. We present a schematic structure that is applicable to most single-camera tracking systems: person hypotheses are generated by a variety of means, and these image regions are tracked through the sequence. The results of the tracking stage feed back to the hypothesis generation, and so the two processes operate in a loop. We describe a number of systems, of increasing sophistication, that operate in this manner to track people through a scene.

Sources of information used in a tracking system can be classified as high-level (context, top-down) or low-level (bottom-up). In the former category we find models of typical human motion; state estimation methods such as Kalman filtering and particle filtering commonly use such models to make predictions about a person's position in the next frame of the sequence. Real-world constraints also provide high-level information for tracking. If an object is represented by a collection of points, for example, we would expect that the points should all display similar motion as the video proceeds. Different methods of initialising trackers are also discussed, among them background subtraction and direct person detection. They provide human hypotheses to be verified or rejected by the subsequent tracking stage.

Image features serve as the measurements for the higher-level state estimation processes. We review the various types commonly employed in tracking. Point-based features are associated with distinctive, compact locations in an image. Ideally, the same locations will be detected in successive frames, which reduces the tracking problem to matching the correct points to one another over time. By calculating a descriptor for the area around a given point, the matching algorithm has more information available on which to base its decisions. Motion-based features often use pairs of frames to associate optical flow vectors with every image pixel. Alternatively, we can calculate vectors only for distinctive image locations (feature points) to improve the reliability of the results. It is also possible to estimate motion for longer sections of the video sequence; the recent *Particle Video* approach is intermediate between long-range feature tracking and optical flow, for example. Strong edges in an image are another class of features that can be exploited for tracking. They can either be used as the basis of a single-frame segmentation algorithm, or they help a contour tracking method to "lock on" to its designated target. Image regions can be represented in numerous ways, ranging from a detailed template to a succinct summary such as a histogram. Furthermore, a variety of low-level image attributes can be encoded in the representation, including colour, intensity and image gradients. Tracking a region involves finding the nearby location in the image that best matches the model according to some similarity measure. Histogram-based models are amenable to gradient ascent searches, but, to date, image templates have typically employed a brute-force strategy. (However, in chapter 7 we develop a gradient-based template tracker, which is robust and computationally efficient.)

We have also reviewed a selection of data-driven tracking methods, which make little use of high-level information, instead relying on the image features directly to follow their target. As the focus of this thesis, mean-shift tracking and its various extensions are described. Techniques for incorporating spatial structure into the method have been proposed, as have ways of exploiting background models of the scene. We have also looked at alternative similarity measures and optimisation methods that can be used in gradient ascent tracking, along with replacements for the standard histogram-based representation of the image region.

Our research builds on the feature types and data-driven tracking methods described in this chapter. Section 1.5 presented our framework for evaluating and improving the performance of mean-shift tracking, describing the research that we undertake in the remainder of the thesis. We begin by comparing mean shift to other data-driven techniques in order to obtain a baseline measure of its performance (chapter 4). In chapter 5 we develop a method for unifying two constraints – multiple-part models and background exclusion – in a single tracker. Chapter 6 is concerned with the effect of various parameters on mean shift, including the type of spatial kernel and the feature space used. We improve the performance of the method by determining its optimal parameter settings via a series of experiments. However, in chapter 7, we develop a gradient-based template tracker that is both computationally efficient and more reliable than mean shift, leading us to question the use of the latter technique. We use the new tracker together with our *track validation* algorithm in order to update the object model safely to accommodate changes in the target's appearance. The approach, although not based directly on mean shift, yields robust gradient-based tracking in the surveillance domain.

Chapter 3

Overview of datasets and metrics

This chapter presents an overview of the video datasets which we will use for our experiments throughout the thesis. We limit the scope of our investigation to the tracking of pedestrians viewed by a single camera. We make use of two publicly available datasets, CAVIAR and PETS. From these we have extracted a number of tracking "scenarios" – short sequences showing a person walking unoccluded through the scene. (We have investigated the use of other publicly available datasets such as ETISEO[1] [112], but we found it unsuitable for our purposes, due to the placement of the cameras.)

Our objective in the thesis is to attempt to track the person in each video sequence using different versions of mean shift (and other tracking techniques), and to assess the performance of the trackers. We use the results of each experiment to make modifications to the mean-shift method with the aim of improving its reliability. The metrics used for performance evaluation are described in section 3.3; they provide the basis for a quantitative assessment of the trackers, both in terms of robustness (not losing track of their target) and accuracy (maintaining the correct position with respect to the target). In order to compute these metrics it is necessary to establish *ground truth* data for each scenario – the result we would expect to obtain if the trackers were performing ideally. We describe the generation of this data and how we extract it for use in our experiments.

3.1 Datasets

In each of the subsequent chapters we will conduct a series of experiments with the aim of assessing, and ultimately improving, the performance of the mean-shift tracking technique. We have selected two different datasets of video sequences in an effort to ensure that we can

[1]The ETISEO datasets can be found at `http://www-sop.inria.fr/orion/ETISEO/`

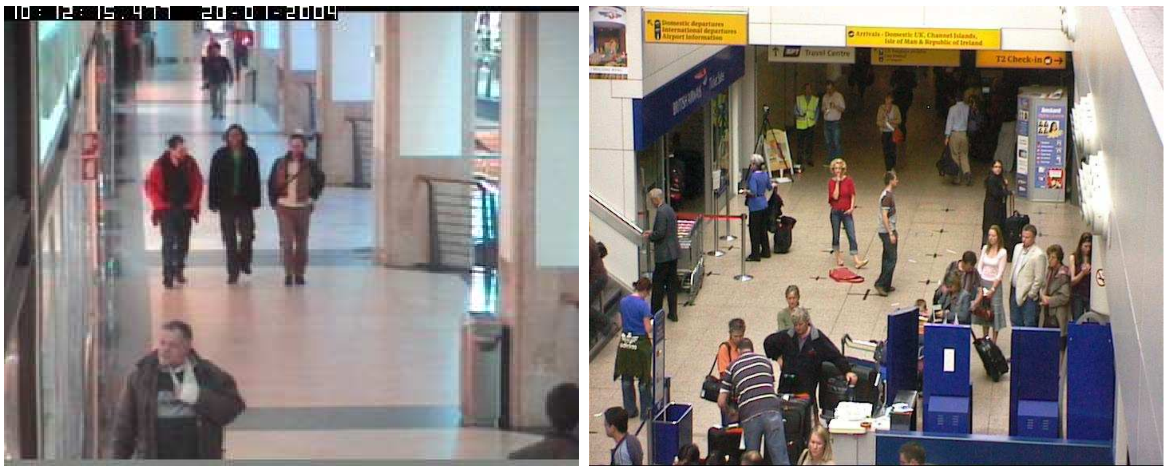


Figure 3.1: *Sample frames from the CAVIAR and PETS datasets*

legitimately draw general conclusions about tracker performance, and not merely conclusions that relate to one particular scene.

Sample frames from the CAVIAR and PETS sequences are shown in figure 3.1. The two datasets have certain important aspects in common: in both cases the camera is positioned at a height that is some way above the heads of the pedestrians; the camera also has zero "roll" – the horizon of the scene appears horizontal in the image. However, the downward tilt of the camera varies between the datasets. The common attributes are highlighted because they will allow us to determine the apparent size of an object from its position in the image: objects closer to the bottom of the image will always appear larger, and their size can be estimated according to a formula (see section 4.2).

3.1.1 CAVIAR

The CAVIAR dataset[2] includes, among other clips, a collection of video sequences of people walking along a corridor in a shopping centre. The twenty-six sequences vary in length from 12 to 150 seconds, and also exhibit significant variation in how many people are visible in the scene at any one time.

The videos represent a challenging tracking task for a number of reasons. Firstly, although we are only considering scenarios that do not feature occlusion, people walk in close proximity to each other, presenting many opportunities for the tracker to be "distracted" by alternative, incorrect targets. Secondly, the lighting varies considerably over the area of the scene. A

[2]The datasets come from the EC Funded CAVIAR project/IST 2001 37540, found at `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`

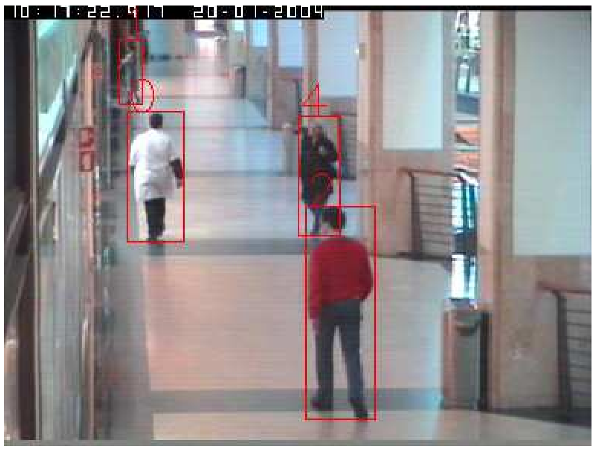


Figure 3.2: *A frame from a CAVIAR sequence with the ground truth data overlaid*

person's appearance can thus alter dramatically as they enter shadow, or as the colour of the light falling on them changes.

Aside from the reasons given above, the CAVIAR dataset is appealing because it provides full *ground truth* data for the sequences. This data is stored in XML format, using CVML – the Computer Vision Markup Language [89]. The XML files contain the bounding boxes for each person in every frame, along with a unique label. (Other data, such as that describing each person's *role* and *context* can be ignored for our purposes.) The bounding boxes will allow us to calculate a number of metrics to assess tracker performance (section 3.3). An example is shown in figure 3.2, where the ground truth data for four people is overlaid on the corresponding video frame.

3.1.2 PETS

The second set of sequences we use for our experiments are those from the PETS 2007 dataset.[3] The videos are of the inside of an airport terminal, with many people queueing or walking through the scene. Occlusions of people (by fixed objects in the scene or by other people) are very common. As with the CAVIAR videos, there are significant lighting changes, both spatially and temporally, in the videos.

The PETS 2007 datasets do not provide any ground truth data for the positions of the people walking through the scene. It was therefore necessary for us to generate ground truth for those people we wished to track. For this purpose we used the software provided by the *Video Performance Evaluation Resource* (ViPER) project[4] [42, 95]. ViPER allows the

[3]The PETS 2007 datasets can be found at `http://www.cvg.rdg.ac.uk/PETS2007/`
[4]ViPER can be found at `http://viper-toolkit.sourceforge.net/`

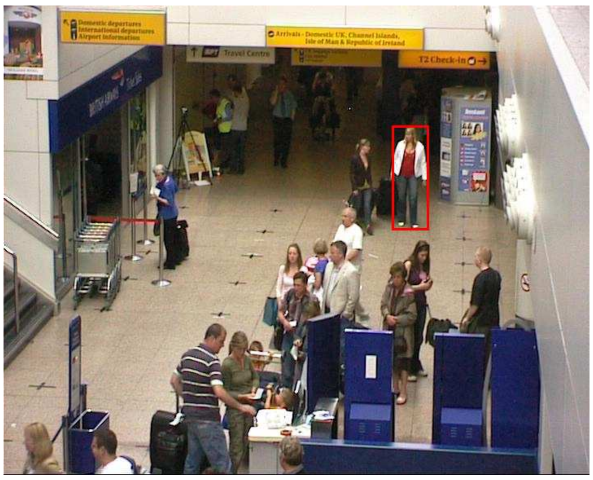


Figure 3.3: *A frame from a PETS 2007 sequence with our approximate ground truth data overlaid*

user to specify the position of a bounding box for an object in every frame of a video. We have generated *approximate* ground truth for some of the people in the PETS sequences: by dragging the rectangle to follow the object as the video is playing, we can obtain acceptable ground truth data very quickly. (An example is shown in figure 3.3. We are exploiting the "propagate" feature of the *ViPER Ground Truth Authoring Tool* to accomplish this.)

It may seem that we have introduced unnecessary inaccuracy into our methodology by not specifying the bounding boxes with high precision. However, we believe that this is not a concern, for two reasons. Firstly, even after laying down very strict guidelines for how a bounding box should be placed with respect to an object, a great deal of arbitrariness remains. For example, the bounding boxes of CAVIAR (see figure 3.2) are placed such that every pixel belonging to a person is within the rectangle. As a consequence, a large number of background pixels also fall within the region, and the width of the rectangle can change rapidly from frame to frame due to the articulation of the person's limbs. Secondly, we do not expect the precision of the bounding boxes to have a systematic effect on the results of our experiments. Kasturi [78] suggests that "high levels of consistency [of annotation] are not necessary for differentiating performance of immature technologies". Such consistency only becomes important when an algorithm approaches the performance of a human undertaking the same task. It will be readily seen, in subsequent chapters, that the various trackers have high failure rates, and can be considered "immature" for assessment purposes.

3.2 Tracking scenarios

In all of the experiments in the subsequent chapters, we attempt to track people (who do not undergo occlusion) through each of several video clips. We have isolated 14 such sequences from the CAVIAR dataset and a further 7 clips from the PETS videos. All of the sequences (which we refer to as "scenarios") are between 4 and 21 seconds long. Details can be found in table 3.1, while example frames from each scenario are shown in table 3.2.

Ideally, a broader set of tracking scenarios would be used in the evaluation of mean shift. In particular, we could consider videos exhibiting a greater variability of lighting or lower contrast between the objects and the background, or containing a time-varying background. Likewise, attempting to track an object as it undergoes a partial occlusion, or as it passes close to another object of similar appearance (a "distractor"), would give greater weight to the results of the evaluation. However, the scenarios shown here, while narrow in their scope, can be used to learn much about the performance of mean shift and other tracking techniques.

Table 3.2: *Representative frames from each of the 21 tracking scenarios in our dataset. The numbers on the left will be used to refer to the scenarios throughout the thesis.*

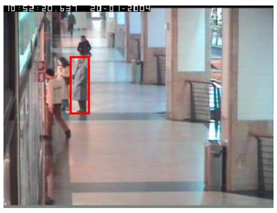
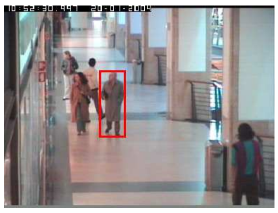
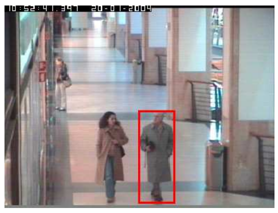
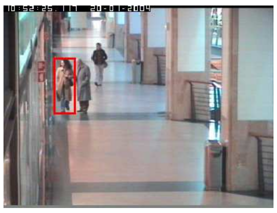
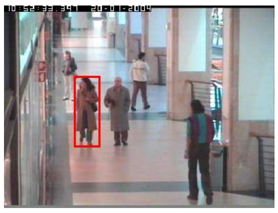
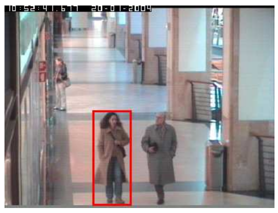
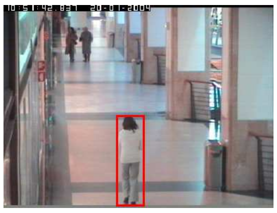
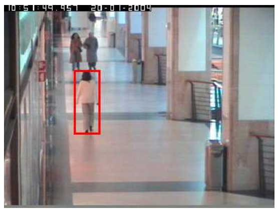
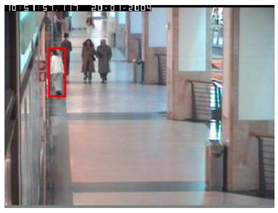
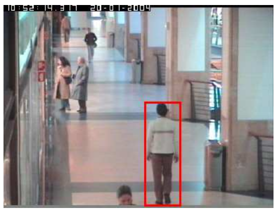
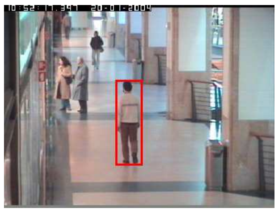
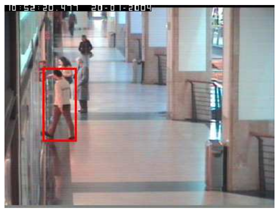
1			
2			
3			
4			

Table 3.2: *Representative frames from each of the 21 tracking scenarios in our dataset (continued)*

5			
6			
7			
8			
9			
10			
11			

Table 3.2: *Representative frames from each of the 21 tracking scenarios in our dataset (continued)*

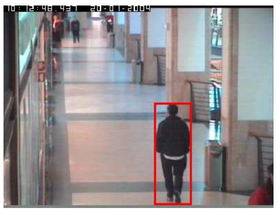
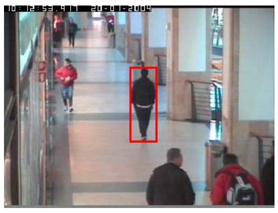
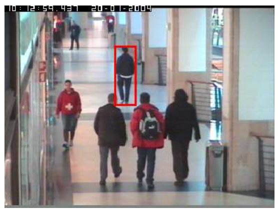
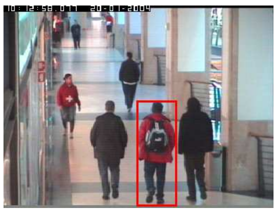
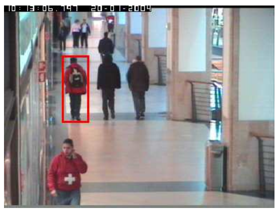
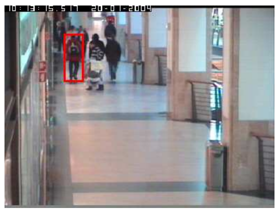
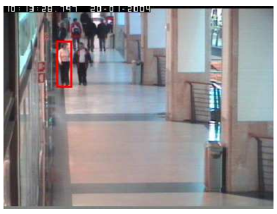
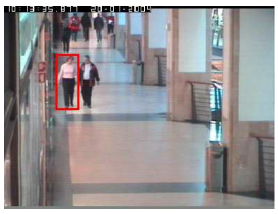
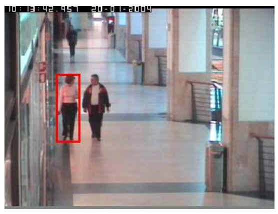
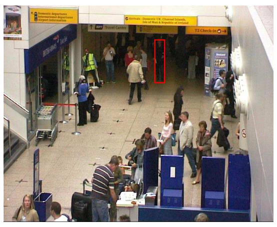
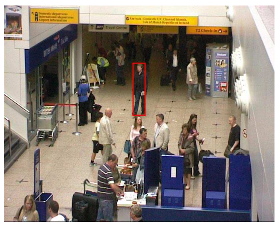
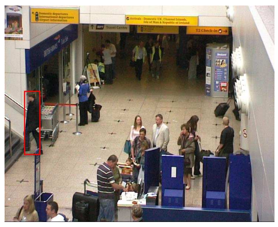
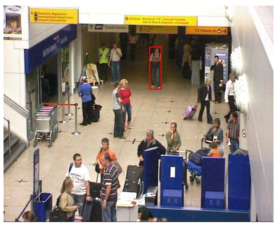
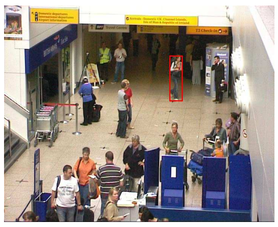
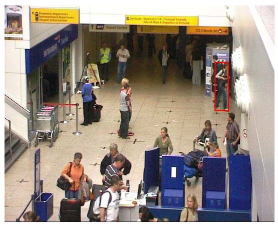
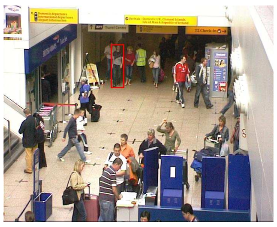
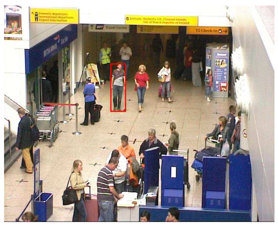
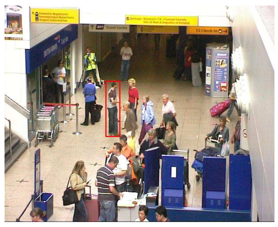
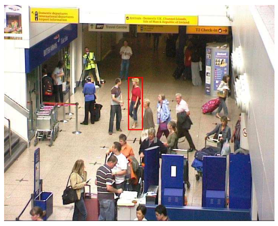
12			
13			
14			
15			
16			
17			

Table 3.2: *Representative frames from each of the 21 tracking scenarios in our dataset (continued)*

18			
19			
20			
21			

3.3 Metrics

In this section we describe the metrics that we will use for a quantitative assessment of the various trackers. These metrics are generally calculated on a frame-by-frame basis, but we can also calculate statistics from the metrics over an entire sequence that summarise a tracker's performance more succinctly. The measures that we employ – centroid distance, overlap, dice coefficient and lost track – are commonly used in the literature for assessing the performance of tracking techniques [42, 94, 110]. By choosing popular metrics, it is easier to compare the results of our evaluations to those of others. The metrics are also easy to calculate when ground truth data of the type described in section 3.1 is available – no further parameters are required. Of the four measures described below, only *lost track* is binary; it simply indicates whether or not the tracker was successful in following its target for the entire duration of the video sequence. The other three are real-valued functions of their inputs.

Dataset	Scenario	Video	Start frame	End frame	Num. frames	Description
CAVIAR	1	WalkByShop1cor	1563	2082	520	Man in grey coat
	2	WalkByShop1cor	1675	2089	415	Woman in brown coat
	3	WalkByShop1cor	618	975	358	Woman in white jumper
	4	WalkByShop1cor	1405	1559	155	Man in white jumper
	5	ShopAssistant1cor	127	259	133	Woman carrying bags
	6	ShopAssistant1cor	400	554	155	Woman in black coat
	7	ShopAssistant2cor	230	606	377	Man in black suit
	8	ShopAssistant2cor	164	603	440	Man in black jacket
	9	ShopAssistant2cor	2800	3328	529	Woman in grey jacket
	10	ShopAssistant2cor	468	792	325	Woman in denim jacket
	11	ThreePastShop2cor	212	552	341	Girl in red jumper
	12	ThreePastShop2cor	157	432	276	Man in dark clothes
	13	ThreePastShop2cor	398	834	437	Man in red jacket
	14	ThreePastShop2cor	1166	1520	355	Woman in white top and black trousers
PETS 2007	15	S03-COUPLE_SWAP_BAG_1\2	2303	2559	257	Man in black jacket and blue jeans
	16	S03-COUPLE_SWAP_BAG_1\2	916	1013	98	Girl with a backpack
	17	S03-COUPLE_SWAP_BAG_1\2	0	248	249	Man in grey top
	18	S03-COUPLE_SWAP_BAG_1\2	0	234	235	Woman in red top
	19	S03-COUPLE_SWAP_BAG_1\2	756	905	150	Man in white shirt
	20	S03-COUPLE_SWAP_BAG_1\2	2581	2730	150	Man in yellow hi-viz vest and black trousers
	21	S03-COUPLE_SWAP_BAG_1\2	2706	2892	187	Woman in white jacket

Table 3.1: *Details of each of the 21 tracking scenarios in our dataset*

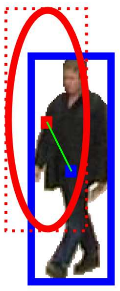


Figure 3.4: *A person, their associated bounding box (blue) and the position of a tracker (red ellipse). The centroid distance is represented by the green line.*

3.3.1 Centroid distance

For a tracker centred at (x_t, y_t) and a ground truth bounding box with centre (x_b, y_b) we define the *centroid distance* (depicted in figure 3.4) as

$$\text{dist}_{\text{centroid}} \triangleq \sqrt{(x_t - x_b)^2 + (y_t - y_b)^2}$$

In order to have a distance measure between the tracker and the ground truth that is comparable across objects of different sizes, we define the *normalised centroid distance* in terms of the width w_b and the height h_b of the bounding box:

$$\text{normalised\_dist}_{\text{centroid}} \triangleq \sqrt{\left(\frac{x_t - x_b}{w_b}\right)^2 + \left(\frac{y_t - y_b}{h_b}\right)^2}$$

3.3.2 Overlap

The proportion of the ground truth bounding box that is occupied by the tracker in a given frame is another useful measure of the tracker's accuracy [42]. This metric is referred to as the *overlap*:

$$\text{overlap} \triangleq \frac{\text{area}_{\text{common}}}{\text{area}_{\text{bounding\_box}}} \tag{3.1}$$

Note that we treat the tracker as rectangular (as opposed to elliptical) for the purposes of the calculation (see the dashed red rectangle in figure 3.4).

3.3.3 Dice coefficient

A final metric that is often used to assess the frame-by-frame accuracy of a tracker is the *dice coefficient* [42]. For two rectangles it is defined as twice their shared area divided by the sum of their areas:

$$\text{dice} \triangleq \frac{2 \times \text{area}_{\text{common}}}{\text{area}_{\text{tracker\_box}} + \text{area}_{\text{bounding\_box}}}$$

The measure is symmetric with respect to the bounding box and the tracker, and it is less prone to being skewed in cases where one of the rectangles is much larger than the other.

3.3.4 Lost tracks

Aside from measuring the accuracy of a given tracker on every frame, we also wish to know how it performs overall, i.e., over the course of an entire tracking scenario. Most importantly, we must determine when we have lost track of an object. The prevalence of such failure cases is a measure of the lack of *robustness* of a tracker.

We define a tracker to be lost as soon as its overlap with the bounding box (equation 3.1) falls below a threshold. Nascimento [110] chooses a value of 10%, while Kasturi [78] sets the threshold at 20%. Clearly, there is no consensus on the "correct" value, and we set it at 10% in our experiments. Although very "forgiving" of large tracking inaccuracies, it is constant for every tracker we use, and thus still allows fair comparisons to be made.

We also disregard any notions of a tracker "recovering" its target by chance: if the overlap falls below the threshold at *any* point during the tracker's run, it is considered lost. If the tracker should happen to follow the target again at a later stage in the run, the status of "lost track" is not overturned.

3.3.5 Aggregate metrics

The metrics described above (excluding "lost track") are all calculated on a frame-by-frame basis. However, it is often useful to summarise the accuracy of a tracker over an entire sequence by a handful of statistics. To capture the tracker's "average" performance we calculate the medians of all of the dice coefficient, overlap and normalised centroid distance values returned for a sequence. We choose the median for this purpose because its value is not strongly influenced by occasional outliers in the data. Similarly, we use the *interquartile range* (IQR) as a robust measure of the data's dispersion, in place of the classical standard deviation. Note that in calculating these statistics, we only use the frames in the sequence from before any loss of track that may occur.

We can further summarise a tracker's performance across all tracking scenarios by calculating the *median of medians*, and the *median of IQRs*, for each of the averages listed above. It allows us to compare the accuracy of two trackers at a glance.

3.4 Summary

The present chapter has described the video datasets that we will use for all of the experiments in the remainder of the thesis. It has also introduced the type of ground truth data that we will employ, and the metrics for assessing each tracker's performance. We are now in a position to compare the basic mean-shift method to other data-driven tracking techniques. In the next chapter, we attempt to track the designated targets through the 21 sequences in our dataset, using the ground truth data and the metrics to measure each tracker's robustness and accuracy.

Chapter 4

Target representation and optimisation strategies

In this chapter we assess the mean-shift tracking technique, both in terms of how it performs on several different video sequences and how its performance compares to that of other common data-driven tracking techniques. Two main aspects of the trackers are investigated. Firstly, we try different optimisation strategies – gradient ascent and brute-force – to find the best match between the target region and the various candidate regions. Secondly, we vary the representation used for the target and candidate regions – using both template-based and histogram-based approaches.

In the following sections we describe in detail each of the trackers used in the experiments, and derive a formula that allows us to vary each tracker's size to match that of the target object in the image. Next, we discuss the experimental setup, and present the results of the experiments. An analysis of the results concludes the chapter.

4.1 Trackers

This section describes the four trackers that are used in the present chapter. We start with the technique that is the focus of the thesis, namely, mean shift. Next, we describe a tracker that uses the same method as mean shift for comparing target and candidate region histograms – the Bhattacharyya coefficient – but that replaces the gradient ascent optimisation strategy with a brute-force search of the nearby image regions. The third tracker uses a different measure, the earth mover's distance, to make comparisons between targets and candidates.

The second class of trackers represents the region to be tracked by its image template, and not by a histogram summarising that region, as the previous methods do. For our fourth tracker, we use the well-known normalised cross-correlation technique, which is commonly employed in template matching.

4.1.1 Mean-shift tracking

Mean-shift tracking tries to find the area of a video frame that is both (a) most similar to a previously initialised model and (b) close to the tracker's location in the previous frame. By applying the technique to each video frame in sequence a region can be tracked over time. The method was first presented by Comaniciu in 2000 [31, 32]. The tracking begins with an object model being created from the region in the first frame. The probability density function (pdf) of the region to be tracked (the *target*) is represented by a histogram $\hat{\mathbf{q}} = \{\hat{q_u}\}_{u=1\ldots m}$ where, for each bin u and pixel i,

$$\hat{q_u} \triangleq C \sum_{i=1}^{n} k\left(\left\|\frac{\mathbf{x}_i}{h_q}\right\|^2\right) \delta\left[b\left(\mathbf{x}_i\right) - u\right] \tag{4.1}$$

In this equation k is a kernel profile that gives more weight to pixels whose locations $\mathbf{x}_i$ are closer to the centre of the target. The bandwidth h_q sets the size of the target region, which contains n pixels.[1] The Epanechnikov kernel profile is the most commonly used in the mean-shift tracker[2]:

$$k\left(x\right) = \begin{cases} l\left(1-x\right) & \text{if } x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

It is shown, along with its spatial structure, in figure 4.1. (We will explore the effect of using other kernels in chapter 6.)

Returning to equation 4.1, C is a normalising constant that ensures that $\sum_{u=1}^{m} \hat{q_u} = 1$. The binning function b maps the pixel at location $\mathbf{x}_i$ to its corresponding histogram bin. Similarly, the pdf of the candidate region $\hat{\mathbf{p}}(\mathbf{y}) = \{\hat{p_u}(\mathbf{y})\}_{u=1\ldots m}$ at location $\mathbf{y}$ is given by

$$\hat{p_u}(\mathbf{y}) \triangleq C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h_p}\right\|^2\right) \delta\left[b\left(\mathbf{x}_i\right) - u\right] \tag{4.2}$$

where h_p is the kernel bandwidth, which determines the size of the candidate region. It is useful to think of u as a colour, but the histograms could actually represent any feature space,

[1]Elliptical image regions must first be mapped onto circles before equation 4.1 can be used.

[2]As with mean-shift segmentation [30], we set the value of the normalising constant to $l = 15/(8\pi) \approx 0.59683$

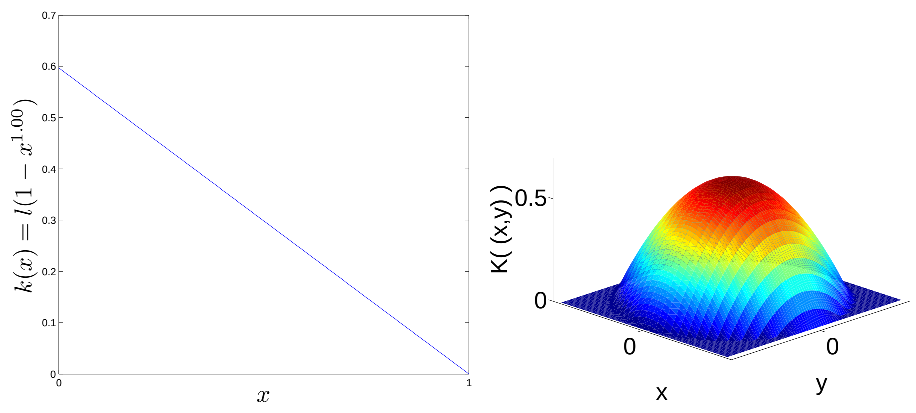


Figure 4.1: *Profile of the Epanechnikov kernel (left) and its corresponding two-dimensional spatial structure for a circular image region (right). A kernel K is related to its profile k by the formula* $K(\mathbf{x}) = k\left(\|\mathbf{x}\|^2\right)$

e.g., edge magnitudes or oriented gradients. Again, the index i ranges over each of the n_h pixels in the candidate region.

Central to the operation of mean shift is the weighting w_i for each pixel:

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q_u}}{\hat{p_u}(\mathbf{y_0})}} \delta\left[b\left(\mathbf{x}_i\right) - u\right] \tag{4.3}$$

which is derived from the Bhattacharyya similarity measure. ($\mathbf{y_0}$ is the location of the candidate region in the previous frame.) Assuming Epanechnikov kernels, the new location $\mathbf{y_1}$ for the candidate region is found simply as

$$\mathbf{y_1} = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i}{\sum_{i=1}^{n_h} w_i} \tag{4.4}$$

Mean shift is an iterative procedure, and so the above formula must be applied until convergence. The tracker is considered to have converged if the (x, y) locations returned by two successive iterations are separated by less than a particular threshold, ϵ.

4.1.2 Brute-force Bhattacharyya coefficient

The mean-shift tracker uses, at its heart, a similarity measure for comparing the target histogram $\hat{\mathbf{q}}$ and the candidate histogram $\hat{\mathbf{p}}(\mathbf{y})$ (equations 4.1 and 4.2). This measure is the

Bhattacharyya coefficient ρ:

$$\hat{\rho}(\mathbf{y}) \triangleq \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p_u}(\mathbf{y})\,\hat{q_u}} \tag{4.5}$$

In order to eliminate the effect of the gradient ascent optimisation strategy of mean-shift, we implement a second tracker that uses the Bhattacharyya coefficient directly. It evaluates the similarity measure at a number of image locations and chooses the location that yields the largest value – a *brute-force* search strategy. Our aim is to study the target *representation* used in mean-shift (a histogram) separately from its optimisation strategy (gradient ascent). We will then be able to see if mean shift is prone to becoming "stuck" on local maxima (incorrect image locations) when there may be a global maximum nearby (the correct image location).

4.1.3 Earth mover's distance

Once we remove the gradient ascent aspect from the mean-shift tracker, we are free to use any similarity measure to compare the target and candidate image regions, and not just those measures that, like Bhattacharyya, have derivatives that can be determined analytically.

In our third tracker, we wish to see if another measure for comparing two histograms can perform better than that used by mean shift. The *earth mover's distance* (EMD) [136] has been described as "the minimal cost that must be paid to transform one histogram into the other" [121]. Since it is a "cross-bin" metric, we might expect it to have greater discriminative power than the Bhattacharyya coefficient.

Our EMD-based tracker also uses a brute-force search strategy to find the optimal image location in each frame. It differs from the previous tracker only in the similarity measure it employs. We have used the *Fast EMD* software[3] from Pele [120, 121] in order to implement our approach.

4.1.4 Normalised cross-correlation

The previous three trackers all use histograms to represent the target and candidate image regions. In contrast to this, normalised cross-correlation (NCC) operates more directly on the image patches. Looking first at regular cross-correlation, we see that its value $\varphi(u, v)$ at image location (u, v) is given by

[3]The software can be found at `http://www.cs.huji.ac.il/~ofirpele/FastEMD/code/`

$$\varphi(u,v) = \sum_{x,y} f(x,y)\, t(x-u, y-v)$$

where t is the image region of the model and f is the candidate image region. (Each pixel in the target is simply multiplied by the corresponding pixel in the candidate and all the products are summed.) Unlike the kernel-weighted histograms of the previous trackers, cross-correlation is highly dependent on the spatial structure of the image.

Normalised cross-correlation [84] attempts to account for differences in the brightness distributions of the target and candidate regions. It assumes that the distributions are approximately normal, and so it subtracts their means before the multiplications and afterwards divides by their standard deviations. The NCC $\gamma(u,v)$ at image location (u,v) is given by

$$\gamma(u,v) = \frac{\sum_{x,y}\left[f(x,y) - \bar{f}_{u,v}\right]\left[t(x-u,y-v) - \bar{t}\right]}{\left\{\sum_{x,y}\left[f(x,y) - \bar{f}_{u,v}\right]^2 \sum_{x,y}\left[t(x-u,y-v) - \bar{t}\right]^2\right\}^{0.5}}$$

where $\bar{t}$ is the mean of the target and $\bar{f}_{u,v}$ is the mean of the candidate image region. The above definition of NCC can only be applied to greyscale images.

As with the previous trackers, we use a brute-force search strategy with NCC to find the best-matching image location at each frame.

4.2 Target scale

In section 3.1 we described the video datasets that will be used in our experiments. It was noted that all of the video sequences are recorded by a camera placed at a height that is some distance above the heads of the pedestrians. This was a deliberate choice; it allows us to estimate the height of a person in the image given the image row corresponding to the top of the person's head. (Collins [28] notes the inability of the basic mean-shift tracker to adapt to changes in the size of the target.) A schematic depiction of the camera arrangement is shown in figure 4.2.

The idea of exploiting the effect of perspective to determine an object's image height is not new. Lin [86] uses a support vector machine to detect the heads of people in an image, and uses their varying image size to locate the vanishing point of the scene. The final goal is to estimate the number of people in the image. Stauffer [151] shows that many attributes of objects, e.g., their radius, velocity, width and height, have an approximately linear relationship with the image row, and that the attributes tend towards a value of zero at the horizon. He provides an automated method for determining the horizon line from a video sequence. Bose [16] demonstrates that, by observing objects moving with constant

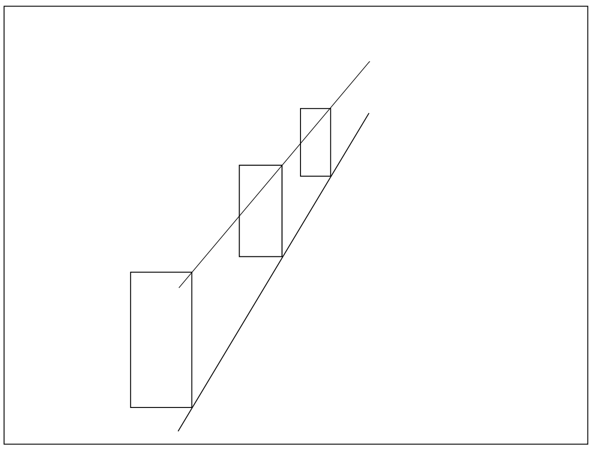


Figure 4.2: *For our raised camera arrangement, there is a simple relationship between an object's location in the image and its apparent height*

velocity, it is possible to perform a metric rectification of the scene, and thus make an object appear to have a constant size regardless of its position in the image. Finally, Hoiem [66] shows the importance of accounting for perspective to many image understanding problems, particularly object detection.

The approaches listed above all ultimately rely on finding the horizon line of the scene in order to reduce the problems of an object changing its apparent size. In our previous work, we exploited the (outdoor) shadows cast on the ground by people walking through the scene to determine the location of the horizon.[4] In the present case, we derive a formula for the image height of a person in any frame of the video given only their image height at the beginning of the sequence and the location of the horizon line. No other parameters are required. We begin by plotting the apparent heights of different people against the image rows corresponding to the tops of their heads (figure 4.3). This plot uses the ground truth data from the CAVIAR dataset (section 3.1.1); an example of the bounding boxes can be seen in figure 3.2. It is clear that the various people have different real-world heights (indicated in the plot by different colours); however, their image heights all tend towards zero at a certain image row – the horizon line. By asserting that the camera has zero roll, we can say that the horizon will appear as a horizontal line in the image. Although the plot depicts the relationship between the image row and the object height as linear, this is only an approximation, albeit one that holds very well for many camera arrangements.

Returning to figure 4.3, we wish to find an expression for h_{curr}, the height of the object in the current frame, given:

[4]The approach has been published as "Direction of Camera Based on Shadows" by D. Caulfield and K. Dawson-Howe in *Machine Vision and Image Processing Conference*, September 2004 [20].

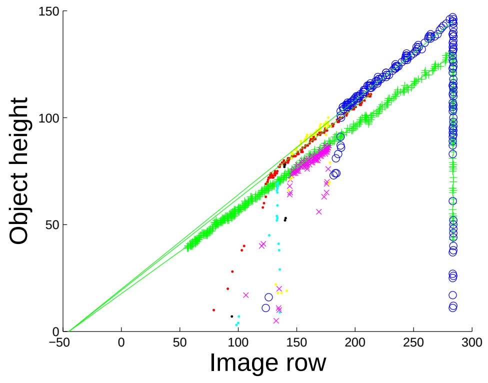


Figure 4.3: *Plots of object heights against image row. Different colours are used to represent different people. The green lines emanate from the horizon row. The collection of points in a vertical line at the right-hand side of the plot is due to certain bounding boxes being "clipped" by the bottom border of the image. Likewise, the outlier points are caused by partial occlusions.*

- its image row in the current frame, r_{curr}
- its image row and height in a previous frame, (r_1, h_1), and
- the image row of the horizon, r_{hor} (where an object's height would be 0)

We begin by observing that the three points, $(r_{hor}, 0)$, (r_1, h_1) and (r_{curr}, h_{curr}), are collinear, and thus the slopes of the line segments between any two pairs of the points are equal:

$$\frac{h_{curr} - h_1}{r_{curr} - r_1} = \frac{h_1 - 0}{r_1 - r_{hor}}$$

It is then simply a matter of making h_{curr} the subject of the above formula:

$$\begin{aligned} h_{curr} &= \frac{h_1 \left(r_{curr} - r_1\right)}{r_1 - r_{hor}} + h_1 \\ &= \frac{h_1 r_{curr} - h_1 r_1 + h_1 r_1 - h_1 r_{hor}}{r_1 - r_{hor}} \\ &= \frac{h_1 \left(r_{curr} - r_{hor}\right)}{r_1 - r_{hor}} \end{aligned} \tag{4.6}$$

Equation 4.6 gives us a way of calculating how an object's size will change over the course of a video sequence. It requires the determination of the image row of the horizon line. In the present work we specify this parameter manually for each dataset by inspecting plots like those of figure 4.3. A complete tracking system could instead use the techniques of Stauffer [151] to find the position of the horizon line automatically, but it would require the ability to detect the objects in the scene, perhaps using background subtraction.

It is noteworthy that the above equation is not limited to estimating the full (head-to-foot) image height of people in the image. It is equally applicable to the task of determining the change in apparent size of any section of an object that moves along the ground. For example, if we wish to track only the torso of a person, or their head and shoulders, we can use equation 4.6 in the same manner as before.

4.3 Experiments

We now discuss the experimental setup for the various trackers described in section 4.1. We assessed the robustness and accuracy of each of the trackers by having them follow the designated target in each of the 21 scenarios (section 3.2) in our test set. The image patch inside the ground truth bounding box in the first frame of each sequence was used to initialise a model for each tracker. The trackers then attempted to follow the object over the course of the video sequence. Metrics representing the accuracy of the trackers (section 3.3) were calculated at each frame, and aggregate metrics, e.g., the median of the dice coefficient, were generated at the end of each clip.

4.3.1 Brute-force search

All of the trackers (except for mean shift) can be regarded as operating in a *brute-force* manner, i.e., we evaluate the similarity of the target and candidate regions (according to some function) at a large number of image locations, and choose the location with the highest similarity score. This approach is in contrast to mean-shift tracking, which is a gradient ascent (sometimes called *hill climbing*) technique that typically requires only a few iterations to converge to a local maximum of the similarity surface. For each of the other trackers, we evaluate its similarity function on a grid of 11×11 image locations, giving a total of 121 evaluations. However, the image locations are not adjacent to one another – they are separated from each other by 3 pixels in the horizontal and 9 pixels in the vertical direction. This sparse grid allows us to cover a large part of the image (90 rows and 30 columns) when searching for the best-matching candidate region in each frame without the computations becoming prohibitive. Examples of the surface plots can be seen in figure 4.9 at the end of the chapter.

Each of the 121 candidate image regions will have a different width and height, depending on its position in the image. Those nearer the top of the video frame will be shorter and narrower than those nearer the bottom. We use equation 4.6 to determine the height of a patch; its width is calculated such that it has the same aspect ratio as the target region.

Having established the dimensions of a candidate rectangle, we extract it from the image. It is then resized so that its width and height match those of the target region. We have used bilinear resizing for this purpose, as it affords a reasonable compromise between preserving image quality and maintaining computational efficiency.

4.3.2 Tracker parameters

The basic mean-shift tracker (section 4.1.1) operates in the RGB colour space with a histogram size of $4 \times 4 \times 4$ bins. We set the value of ϵ, the convergence condition, to 1 pixel, and the maximum number of iterations allowed to 20. Comaniciu includes a procedure in the original mean-shift tracking algorithm [32] that attempts to ensure that the similarity score always increases between successive iterations. If a *decrease* is seen, the tracker is instead moved one half of the distance suggested by the algorithm. If the score has still not increased, it moves one quarter of the distance, and so on until an increase is seen. In our experiments, we limit these attempts at halving the distance ("half-steps") to a maximum of five.

The brute-force Bhattacharyya tracker (section 4.1.2) uses the same parameters as the mean-shift version – the gradient ascent aspect of the latter has simply been replaced by a brute-force optimisation strategy.

Brute-force search is again employed by the earth mover's distance tracker (section 4.1.3). As in the case of the previous two trackers, it operates in RGB with a $4 \times 4 \times 4$-bin histogram.

Finally, the normalised cross-correlation tracker (section 4.1.4) operates in greyscale, and does not make use of any colour information.

4.4 Results

In this section we present the results obtained from running each of the four trackers on the 21 scenarios in our test set. We look first at the *robustness* of the trackers, i.e., whether they successfully track their target for the entire duration of a video sequence. We then study the *accuracy* of the different techniques – how close they remain to the centre of the object being tracked over time.

4.4.1 Robustness

Table 4.1 shows the tracking success/failure status of each of the techniques described in this chapter when run on our 21 tracking scenarios. It is immediately clear that the normalised cross-correlation tracker (NCC) significantly outperforms all of the other techniques on these sequences, losing track in only five out of the 21 cases. An example of NCC maintaining track

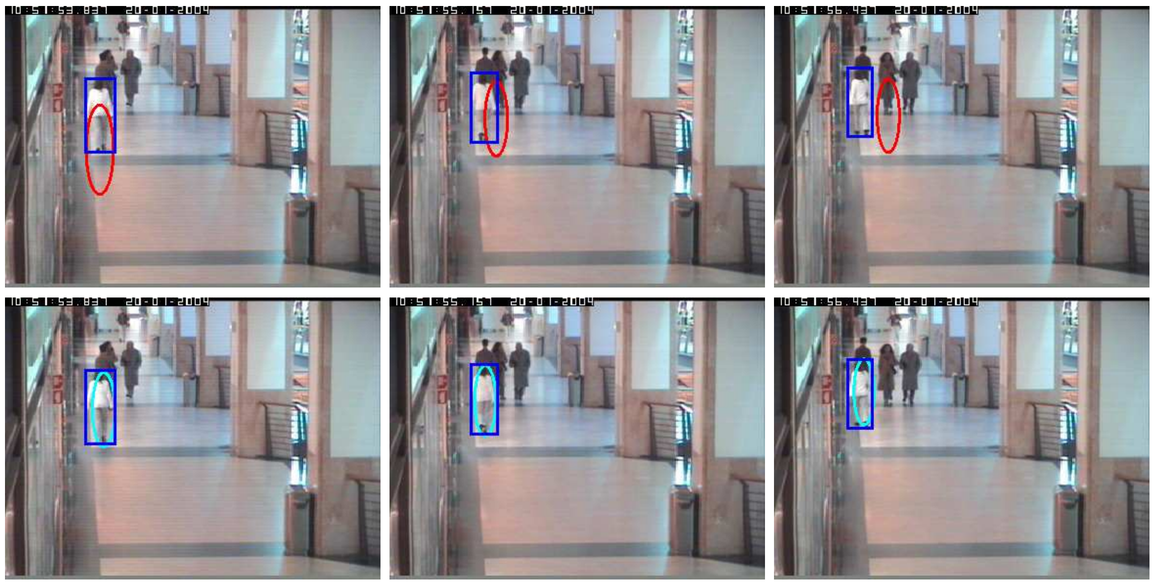


Figure 4.4: *Mean shift losing its target (top), while normalised cross-correlation successfully tracks the person (bottom), in scenario 3*

when mean shift fails to do so is shown in figure 4.4, which provides a visual summary of the performance of these trackers on scenario 3. Plots of the overlap metric for the two trackers as they operate on this sequence are shown in figure 4.5. Mean shift displays a sudden drop in the metric over the course of only a few frames near the end of the sequence, signalling the technique's loss of track. It is also seen that NCC has a consistently higher overlap score throughout the clip, indicating the greater accuracy of that tracker (which will be discussed further in the next section).

It is somewhat surprising to note (from table 4.1) that the regular (gradient ascent) mean-shift tracker appears to be more robust than the brute-force Bhattacharyya approach, with only nine lost tracks to the latter's twelve. We might have expected that the ability of the brute-force tracker to "see" the entire neighbourhood of the similarity surface would give it a better chance of following a dynamic target, especially when the tracker wanders away from the object centre temporarily. However, it seems that, counter-intuitively, the broader view of the terrain presents the tracker with more opportunities for being distracted by candidate image regions that do not correspond to the target but nonetheless have a similar histogram.

The earth mover's distance tracker (EMD), another brute-force approach, also performs worse than mean shift in terms of robustness. It appears that the EMD metric possesses even less discriminative power when comparing histograms than the Bhattacharyya metric, at least for our purposes. The similarity surfaces produced by the two measures are visually

Tracker / **Scenario**	**Mean shift**	**Brute-force Bhattacharyya**	**Brute-force EMD**	**Normalised cross-corr. grey**
1	x			x
2		x	x	
3	x	x	x	
4		x	x	x
5				
6				
7				
8				
9	x		x	x
10			x	
11		x		
12				
13			x	x
14		x	x	
15	x	x	x	
16	x	x	x	
17	x	x	x	
18	x	x	x	x
19		x	x	
20	x	x	x	
21	x	x	x	
Num. trackers lost	9	12	14	5

Table 4.1: *The success/failure of the various trackers operating on each scenario. (Failures are indicated by an 'x'.)*

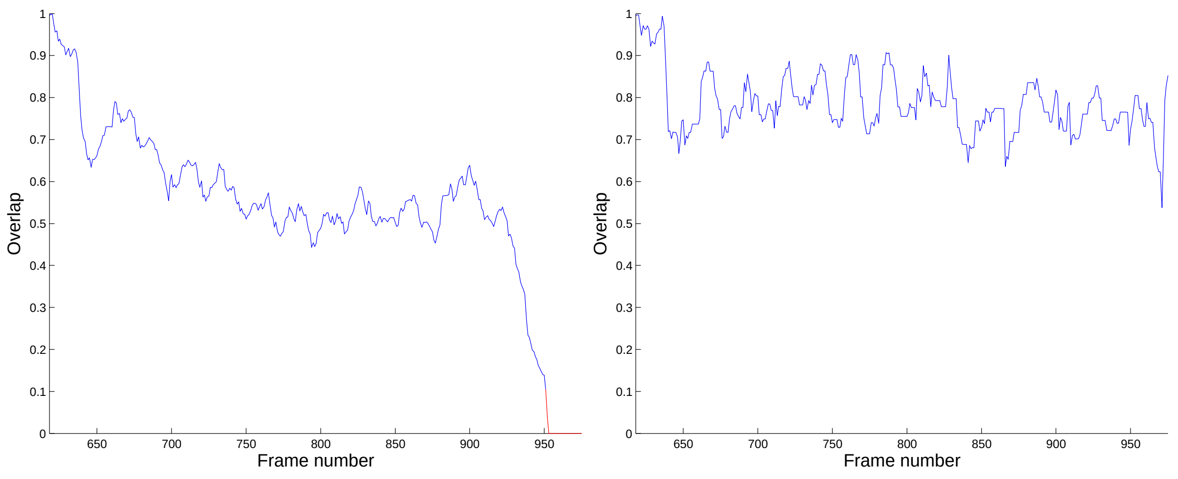


Figure 4.5: *Plots of the overlap metric for scenario 3 for the mean-shift tracker (left) and for normalised cross-correlation (right)*

very similar, and lack the sharp peak that is characteristic of NCC (see figure 4.9 later in this chapter).

Table 4.1 also reveals that the performance of the histogram-based trackers depends strongly on which of the two datasets is considered. The mean-shift and brute-force Bhattacharyya trackers display similar robustness to NCC on sequences 1 to 14 (the CAVIAR dataset). However, tracking failure is almost universal on the PETS sequences (numbers 15 to 21) for the three histogram-based approaches.

It appears that the latter sequences are inherently more challenging than those from CAVIAR. Referring to table 3.2, we can see that the PETS targets undergo more-significant appearance changes than their CAVIAR counterparts, mostly on account of the variations in lighting over the scene: transition from shade into bright daylight is common to most of the PETS sequences. The designated individuals in the PETS videos also spend a greater amount of time in close proximity to “distractors” – other people whose similar appearance may draw the tracker away from the correct target.

The disparity in results from one dataset to another emphasises the need to use test sequences that are sufficiently challenging; otherwise, differences in capability between competing tracking techniques will not be apparent. In subsequent chapters, we will again see that only by including the PETS videos can we demonstrate the improved robustness of our modified trackers.

4.4.2 Accuracy

While robustness must be the primary criterion for assessing a tracker's performance, it is important and instructive to measure also the accuracy of the technique. When we speak of *accuracy*, we are referring specifically to the distance between the centre of a tracker and the centre of the target object that it is supposed to follow. We use the metrics described in section 3.3 – dice coefficient, overlap and normalised centroid distance – to make these assessments.

In the previous section, we presented plots of the overlap metric for two trackers operating on a single video sequence (figure 4.5). We can convey the same information more succinctly, however, by using a *box-and-whisker plot*. Also known simply as a *box plot*, it summarises an entire data distribution with a number of visual elements. Firstly, there is the box, whose vertical extent represents the interquartile range of the data (with the median also marked by a line near the centre of the box). Next are the whiskers – lines extending out from the box, which depict a certain multiple of the interquartile range (1.5 times, in our case). Finally, crosses denote any data points outside of this range.

The notch in the centre of the box plot depicts the variability in the median of the data amongst different samplings. If the notches of two box plots that are beside one another do not overlap, we can say that there is a statistically significant difference between the medians of the two distributions. In this way, the box plot can serve as a visual hypothesis test – it is analogous to a t-test, but it is not as statistically rigorous.

Figures 4.6 to 4.8 show, for every tracker, box plots of the various accuracy metrics computed for scenarios 6 and 16. The results for scenario 6 are representative of the CAVIAR dataset as a whole: normalised cross-correlation is somewhat more accurate than the other trackers, as judged by the variability of the medians, regardless of the metric used. The trend is consistent across the other CAVIAR sequences. In contrast, scenario 16 is indicative of the trackers' performance on the PETS videos. (Only those frames from before the point at which a tracking failure occurred were used to calculate the metrics.) While the accuracy of NCC is unaffected, the other methods perform much more poorly than on the CAVIAR sequences. Similar results are recorded for the remainder of the PETS clips. Again, we see that the choice of dataset has a large bearing on how readily weaker tracking techniques are distinguished from more capable approaches.

Table 4.2 summarises the performance of each of the trackers over the 21 scenarios in our test set. For a given tracker we take its median overlap metric for each sequence, and then we calculate the median of these values. We perform a similar calculation for the dice coefficient and the normalised centroid distance. It will be seen that NCC records better scores than the other trackers for each of these aggregate metrics. It is not clear, however, if

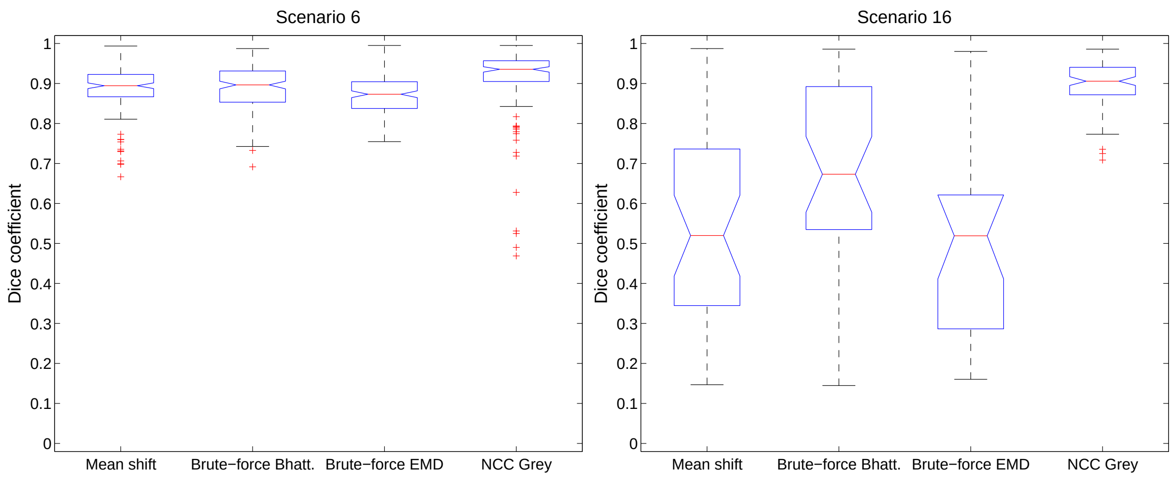


Figure 4.6: *Box-and-whisker plot of each tracker's dice coefficient for scenario 6 (left) and scenario 16 (right)*

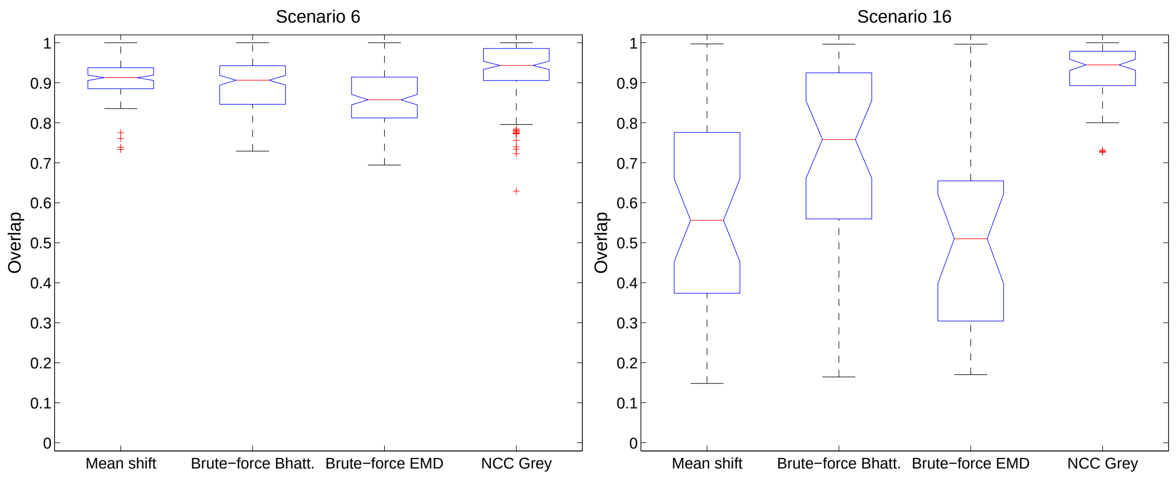


Figure 4.7: *Box-and-whisker plot of each tracker's overlap metric for scenario 6 (left) and scenario 16 (right)*

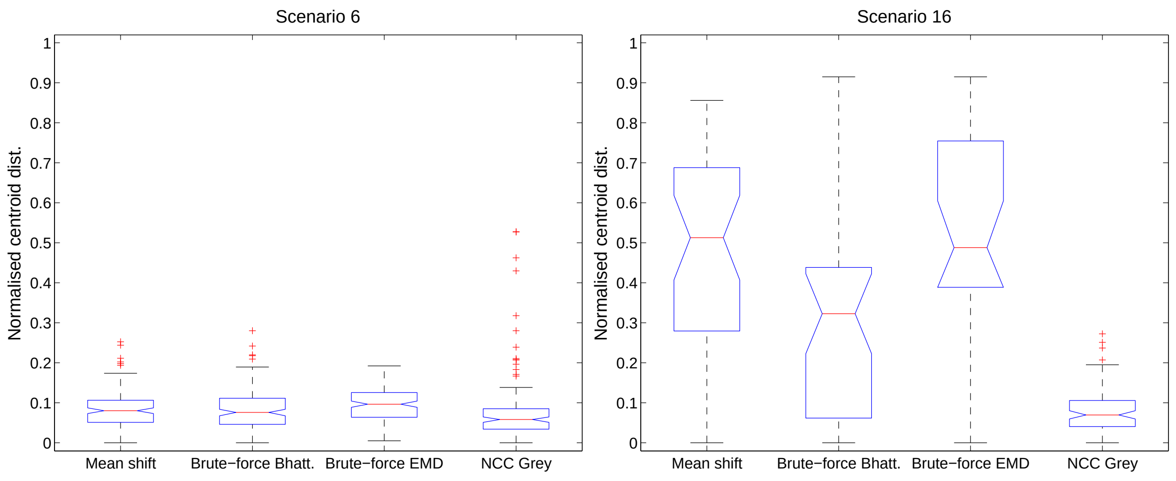


Figure 4.8: *Box-and-whisker plot of each tracker's normalised centroid distance for scenario 6 (left) and scenario 16 (right)*

we can assert that any one of the remaining trackers – mean shift, brute-force Bhattacharyya or brute-force EMD – is better than the others in terms of accuracy.

The median-of-interquartile ranges for each of the trackers is similarly recorded in table 4.3. It shows the NCC tracker has, on average, a tighter interquartile range than the other methods. The raw metrics recorded by this tracker have a lower dispersion (tighter distribution), therefore, than those of the other techniques.

With regard to the metrics themselves, figures 4.6 to 4.8 and tables 4.2 and 4.3 suggest that the trackers will be ranked similarly in terms of performance regardless of which accuracy measure we use. The NCC tracker is seen to be the most accurate, while the other three approaches all exhibit similar, lower performance, whether we use the dice coefficient, overlap or normalised centroid distance metric. In order to simplify the presentation of results in subsequent chapters, therefore, we will only use the dice coefficient to assess the accuracy of a given method; *lost track* will continue to be used to measure robustness.

4.5 Discussion

The results of the experiments indicate that NCC has at its heart much more discriminative power than either the brute-force Bhattacharyya tracker (and, by implication, mean shift) or the EMD tracker. The similarity surfaces shown in figure 4.9 support this contention.[5]

[5]A *similarity surface* is a three-dimensional plot showing, at each grid location, the value of the tracker's similarity measure – Bhattacharyya distance, earth mover's distance or normalised cross-correlation – for the target model and the candidate model at that image location.

Metric \ Tracker	Mean shift	Brute-force Bhatt.	Brute-force EMD	NCC Grey
Dice coefficient	0.72	0.77	0.80	0.93
Overlap	0.77	0.74	0.77	0.94
Normalised centroid distance	0.23	0.17	0.15	0.05

Table 4.2: *Median-of-medians for the dice coefficient, overlap and normalised centroid distance for the various trackers across all of the scenarios*

Metric \ Tracker	Mean shift	Brute-force Bhatt.	Brute-force EMD	NCC Grey
Dice coefficient	0.17	0.16	0.12	0.06
Overlap	0.18	0.15	0.15	0.08
Normalised centroid distance	0.16	0.13	0.11	0.05

Table 4.3: *Median-of-interquartile ranges for the dice coefficient, overlap and normalised centroid distance for the various trackers across all of the scenarios. Lower numbers are better in every case.*

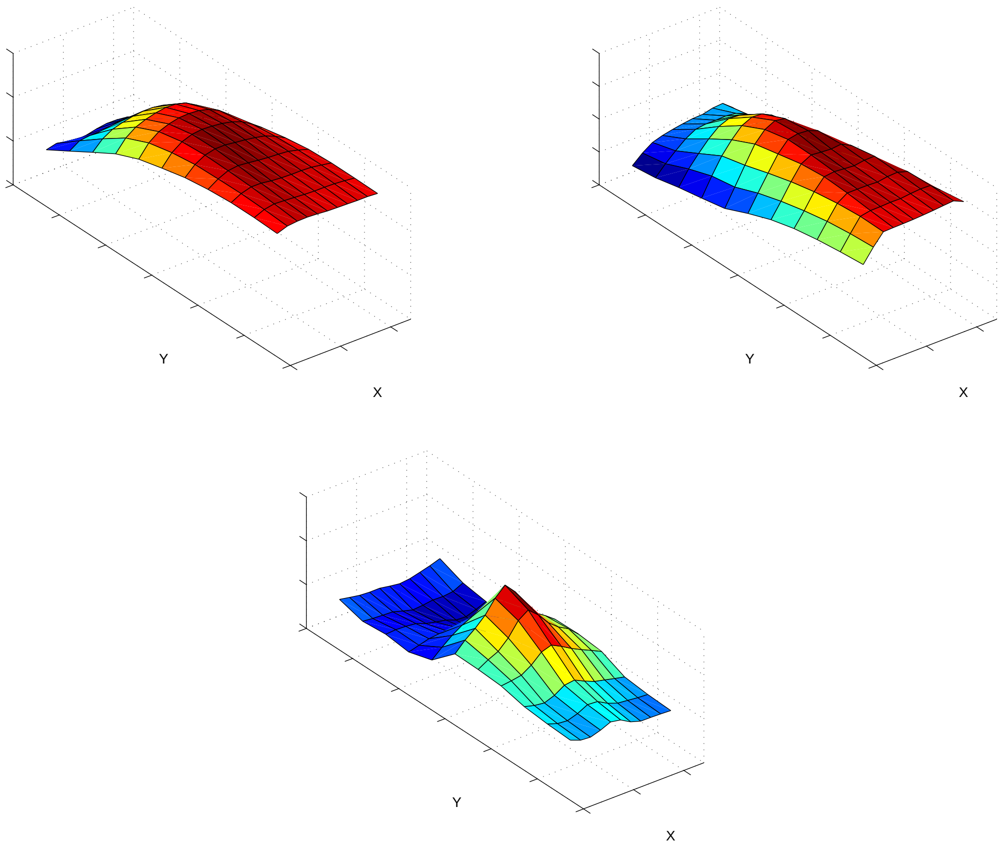


Figure 4.9: *Plots of the similarity surfaces produced by the Bhattacharyya, EMD (top) and NCC trackers (bottom) for scenario 3. The plots contain* 11×11 *datapoints and span 90 rows × 30 columns of the image.*

Only the surface for NCC displays a distinctive peak – those of the other approaches have an extended plateau near their centres, making the tracker less likely to stay at the correct location. It must also be borne in mind that NCC operates in greyscale, while the other trackers make use of colour information, making NCC's performance more remarkable.

We caution that the results presented in this chapter relate only to the relatively narrow dataset that we used for our experiments. The trends seen here, and in subsequent chapters, may not be observed with other collections of video sequences.

Tracking failures in mean shift are synonymous with a "false" local maximum developing in the similarity surface near to the correct location. The tracker will move to this location, and if the peak persists for a sufficient length of time, mean shift will wander away from its target completely and a lost track will result. Such false peaks usually arise when the target and candidate image regions have become excessively different in appearance. For example, if the person to be tracked was initially in shade but is now walking through an area of bright sunshine, the similarity score between the two image patches will be low. A higher score may in fact be achieved by comparing the target region to a dark section of the background. If the person happens to walk past such a place in the scene, mean shift may be "distracted" by the background and lose track of its target.

Two aspects of normalised cross-correlation seem to be important to that method's robustness. Firstly, unlike the histogram-based trackers, it encodes spatial structure within its model; this appears to be responsible for the peaked nature of its similarity surface. In the next chapter we introduce similar structure into the mean-shift tracker, and its surface also develops a stronger peak as a result (see figure 5.9). Secondly, the normalisation element of NCC seems to allow it to cope with brightness changes that occur in the target being tracked. In chapter 7 we develop an alternative method for accommodating appearance changes: the object model is updated periodically when we are confident that the tracker is still correctly positioned on its target.

The performance of normalised cross-correlation, as determined in the present chapter, will serve as a baseline for our further experiments. Our goal is to enhance mean-shift tracking to the point where in can compete with, and even surpass, the robustness and accuracy of NCC.

Chapter 5

Tracking with multiple-part models

The previous chapter demonstrated the lack of robustness exhibited by the basic mean-shift tracker; it regularly loses track of the target it is following. We hypothesise that at least some of this poor performance can be attributed to the absence of spatial structure encoded within the histograms employed by mean shift. To incorporate such structure, we divide the target to be tracked into several regions and apply mean shift to this new multiple-part model. We also develop a method for combining such trackers with a constraint known as *background exclusion*[1]. A number of tracking experiments are performed, and they show that the combination of the two techniques – multiple-part models and background exclusion – improves both the robustness and the accuracy of tracking when compared to the basic mean-shift approach.

5.1 Background

The mean-shift tracking technique was described in section 4.1.1. As a bottom-up, or data-driven, technique, it permits regions of an image to be tracked over time without the need to specify complex motion or appearance models. A simple colour histogram is used to encode the appearance of the object to be tracked, while a gradient ascent optimisation scheme moves the tracker to the best location in the next frame of the video sequence.

Although mean-shift tracking is popular due to its relative simplicity and computational efficiency, it suffers from a number of weaknesses: it is prone to distraction by other objects that are similar to the one being tracked, and it lacks a mechanism for encoding the spatial layout of the colours the object. A typical mean-shift similarity surface was shown

[1]An earlier version of the work in this chapter has been published as "Evaluation of multi-part models for mean-shift tracking" by D. Caulfield and K. Dawson-Howe in *International Machine Vision and Image Processing Conference*, September 2008 [21].

in figure 4.9. Owing to the lack of spatial structure encoded in the target and candidate histograms, the surface does not have a sharp peak, displaying instead a smooth plateau. It is therefore difficult for the tracker to position itself accurately on its target; a slight change in the image data is all that is required for the tracker to favour an incorrect location. In some cases, the inaccuracy can persist for several frames, causing mean shift to be drawn away from the correct target and ultimately resulting in a lost track.

To date, various researchers have attempted to address these problems. In order to enforce a particular spatial structure on the object various multiple-part models have been proposed [94, 118, 173]. Yang *et al.* [44, 174] show that data-driven tracking can be considered as a spectrum. At one end is mean shift, which imposes no spatial structure on the target, while at the other extreme we find sum-of-squared-differences (SSD) tracking and normalised cross-correlation (section 4.1.4), which are forms of rigid template matching. In this framework, mean shift with multiple-part models can be regarded as intermediate between the two endpoints: certain structure exists in the model, but it does not amount to complete rigidity. The similarity surfaces of such trackers are more peaked than those of basic mean shift, which accounts for their improved accuracy (compare figures 5.8 and 5.9 at the end of this chapter).

Zhao *et al.* [185] and Porikli *et al.* [127] have used the background exclusion constraint to make the tracker favour regions that are dissimilar to the background. This technique is applicable when we have a model of the empty scene available, perhaps generated by an automatic method [27, 35, 160]. It is analogous to *background subtraction*, but it is integrated directly into the tracker, and so issues such as thresholding do not arise.

5.2 Innovation

We have developed a mean-shift-based approach that unifies both the background exclusion constraint and multiple-part appearance models in a single tracker (section 5.5). We have also undertaken an extensive evaluation of the new tracker on several video sequences. It is compared to mean-shift trackers that use only background exclusion or multiple-part models (but not a combination of the two). The new tracker outperforms each of the existing techniques, both in terms of robustness and of accuracy (section 5.7).

The following section presents a careful derivation of the multiple-part tracker. The details of background exclusion are then provided, so that we can fully describe our approach to combining the two components in a single tracker.

5.3 Multiple-part models

The derivation of the formula for the standard mean-shift tracker starts with the *Bhattacharyya coefficient* ρ, a measure for comparing the target histogram $\hat{\mathbf{q}}$ and the candidate histogram $\hat{\mathbf{p}}(\mathbf{y})$:

$$\hat{\rho}\left(\mathbf{y}\right) \triangleq \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p_u}\left(\mathbf{y}\right) \hat{q_u}} \tag{5.1}$$

where $u = 1...m$ indexes over the bins in each. In order to use multiple-part models, we divide the region to be tracked into N subregions, where each subregion P has its own histograms $\hat{\mathbf{q}^P}$ (for the target) and $\hat{\mathbf{p}^P}(\mathbf{y})$ (for the candidate), $P = 1...N$. We can use any spatial partitioning of the region; some examples are shown in figure 5.3. Maggio [94] defines a measure ϕ for comparing two multiple-part models:

$$\hat{\phi}\left(\mathbf{y}\right) \triangleq \phi\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{P=1}^{N} \rho\left[\hat{\mathbf{p}^P}(\mathbf{y}), \hat{\mathbf{q}^P}\right] \tag{5.2}$$

It is simply the summation of the individual Bhattacharyya coefficients for each of the subregions. In order to account for the possibility of these regions having different sizes, we modify equation 5.2:

$$\hat{\phi'}\left(\mathbf{y}\right) \triangleq \phi'\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{P=1}^{N} S^P \rho\left[\hat{\mathbf{p}^P}(\mathbf{y}), \hat{\mathbf{q}^P}\right] \tag{5.3}$$

The quantity S^P represents the size of subregion P in pixels, thereby giving more weight to larger regions.

We now show how to obtain a mean-shift formula for multiple-part models. As in basic mean-shift tracking, each subregion P in the target is represented by a histogram $\hat{\mathbf{q}^P}$ with m bins:

$$\hat{\mathbf{q}^P} = \{\hat{q_u^P}\}_{u=1...m}$$

We introduce an indicator function I_R, where

$$I_R\left(\mathbf{x}_i, P\right) = \begin{cases} 1 & \text{if the pixel at location } \mathbf{x}_i \text{ belongs to region } P \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

The value in each histogram bin $\hat{q_u^P}$ is given by

$$\hat{q_u^P} = C^P \sum_{i=1}^{n} k\left(\left\|\frac{\mathbf{x}_i}{h_q}\right\|^2\right) \delta\left[b\left(\mathbf{x}_i\right) - u\right] I_R\left(\mathbf{x}_i, P\right) \tag{5.5}$$

Unlike Dong [173], where a single normalisation factor is used, we employ a different factor C^P for each subregion. The remaining quantities in equation 5.5 are the same as those of equation 4.1. The candidate histogram $\hat{\mathbf{p}^P}(\mathbf{y}) = \{\hat{p_u^P}(\mathbf{y})\}_{u=1\ldots m}$ is defined similarly:

$$\hat{p_u^P}(\mathbf{y}) = C_h^P \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h_p}\right\|^2\right) \delta\left[b\left(\mathbf{x}_i\right) - u\right] I_R\left(\mathbf{x}_i, P\right)$$

We wish to optimise equation 5.3 with respect to the tracker's position $\mathbf{y}$. As in Comaniciu's original tracker [32], we apply a Taylor series expansion to $\rho\left[\hat{\mathbf{p}^P}(\mathbf{y}), \hat{\mathbf{q}^P}\right]$, which means we must optimise

$$\sum_{P=1}^{N} S^P \left[\kappa_1^P + \frac{C_h^P}{2} \sum_{i=1}^{n_h} w_i^P k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h_p}\right\|^2\right)\right], \tag{5.6}$$

where the *weights* w_i^P are given by

$$w_i^P = \sum_{u=1}^{m} \sqrt{\frac{\hat{q_u^P}}{\hat{p_u^P}(\mathbf{y_0})}} \delta\left[b\left(\mathbf{x}_i\right) - u\right] I_R\left(\mathbf{x}_i, P\right)$$

($\mathbf{y_0}$ is the location of the candidate region in the previous frame.) Hence, after dropping the constants from expression 5.6, we must optimise

$$\begin{aligned}
&\sum_{P=1}^{N} \left[S^P C_h^P \sum_{i=1}^{n_h} w_i^P k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h_p}\right\|^2\right)\right] \\
=&\sum_{i=1}^{n_h} \left[\left(\sum_{P=1}^{N} S^P C_h^P w_i^P\right) k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h_p}\right\|^2\right)\right] \\
=&\sum_{i=1}^{n_h} \left[w_i' k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h_p}\right\|^2\right)\right]
\end{aligned}$$

This expression has the same form as that used in the basic mean-shift tracker, with the weights w_i' given by

$$w_i' = \sum_{P=1}^{N} \sum_{u=1}^{m} S^P C_h^P \sqrt{\frac{\hat{q_u^P}}{\hat{p_u^P}(\mathbf{y_0})}} \delta\left[b\left(\mathbf{x}_i\right) - u\right] I_R\left(\mathbf{x}_i, P\right)$$

To find the new location $\mathbf{y_1}$ for the tracker, we use the standard mean-shift formula:

$$\mathbf{y_1} = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i'}{\sum_{i=1}^{n_h} w_i'} \tag{5.7}$$

Equation 5.7 is applicable as long as k is the Epanechnikov kernel.

We have now derived a procedure for mean-shift tracking with multiple-part models. It is similar to the approach taken by Maggio [94], but the equations are given explicitly. We note that our definition of the indicator function I_R (equation 5.4) allows a pixel to belong to different subregions simultaneously. Therefore, we can create trackers whose subregions overlap, which permits us to be more specific about the type of spatial structure we wish to capture in the model. Finally, we treat subregions of different sizes explicitly, in contrast to both Dong [173] and Maggio. This ensures that larger subregions will have a greater influence on the tracker than smaller ones.

5.4 Background exclusion

Various authors have attempted to exploit background models of the scene to improve the performance of mean-shift tracking. Zhao *et al.* [185] and Porikli *et al.* [127] have both modified the Bhattacharyya coefficient ρ (equation 5.1) to take account of the appearance of the background:

$$\hat{L}\left(\mathbf{y}\right) \triangleq \lambda_f \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] - \lambda_b \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{d}}(\mathbf{y})\right] \tag{5.8}$$

In the definition given above, $\hat{\mathbf{q}}$ and $\hat{\mathbf{p}}(\mathbf{y})$ are the target and candidate histograms (as before), and $\hat{\mathbf{d}}(\mathbf{y}) = \{\hat{d_u}(\mathbf{y})\}_{u=1\ldots m}$ is the colour histogram of the corresponding region in the background model.

The first term represents "object attraction" and the second represents "background exclusion". It enforces the constraint that a pixel in the candidate image region should be similar to the corresponding pixel in the object model, but dissimilar to the corresponding pixel in the background model (see figure 5.1). Background exclusion is analogous to *background subtraction*, but it is integrated directly into the tracker. The factors λ_f and λ_b determine the relative influence given to the foreground and background models, respectively (and they sum to 1).

The changes introduced in equation 5.8 lead to a new definition of the weights w_i used in the mean-shift procedure:

$$w_i = \lambda_f w_i^f - \lambda_b w_i^b \tag{5.9}$$

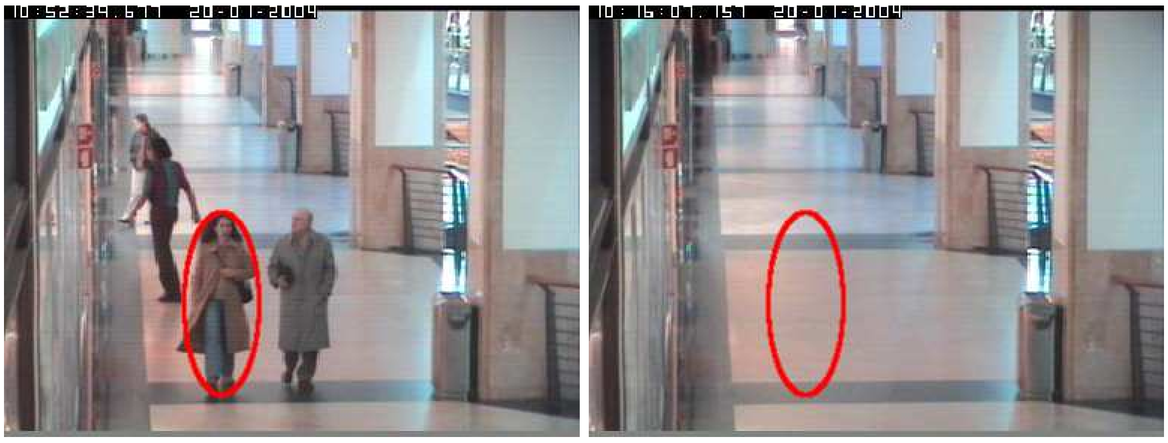


Figure 5.1: *A candidate region (left) should be similar in appearance to the object model, but dissimilar to the corresponding region in the background model (right)*

The foreground weight w_i^f is calculated as in equation 4.3, but the background weight w_i^b takes on a more complex form [185]:

$$w_i^b = \sum_{u=1}^{m} \left(\sqrt{\frac{\hat{d_u}(\mathbf{y_0})}{\hat{p_u}(\mathbf{y_0})}} \delta \left[b_f \left(\mathbf{x}_i \right) - u \right] + \sqrt{\frac{\hat{p_u}(\mathbf{y_0})}{\hat{d_u}(\mathbf{y_0})}} \delta \left[b_b \left(\mathbf{x}_i \right) - u \right] \right)$$

There are now separate binning functions b_f and b_b for the foreground and background images. We must modify the mean-shift formula, as shown by Collins [28], so that it can accommodate negative weights:

$$\mathbf{y_1} = \mathbf{y_0} + \frac{\sum_{i=1}^{n_h} \left(\mathbf{x}_i - \mathbf{y_0} \right) w_i}{\sum_{i=1}^{n_h} |w_i|} \tag{5.10}$$

In the case of the CAVIAR dataset (section 3.1.1) it was trivial to obtain a background model: the video sequences contain numerous frames showing an empty scene. However, for the PETS sequences (section 3.1.2) it was necessary to generate a background model from the videos themselves. We took the median brightness (over time) of each (x, y) pixel as the value to be used in the background model. Although the resultant image contained many imperfections, the scene it showed was sufficiently empty to allow it to be used with the background exclusion tracker (see figure 5.2).

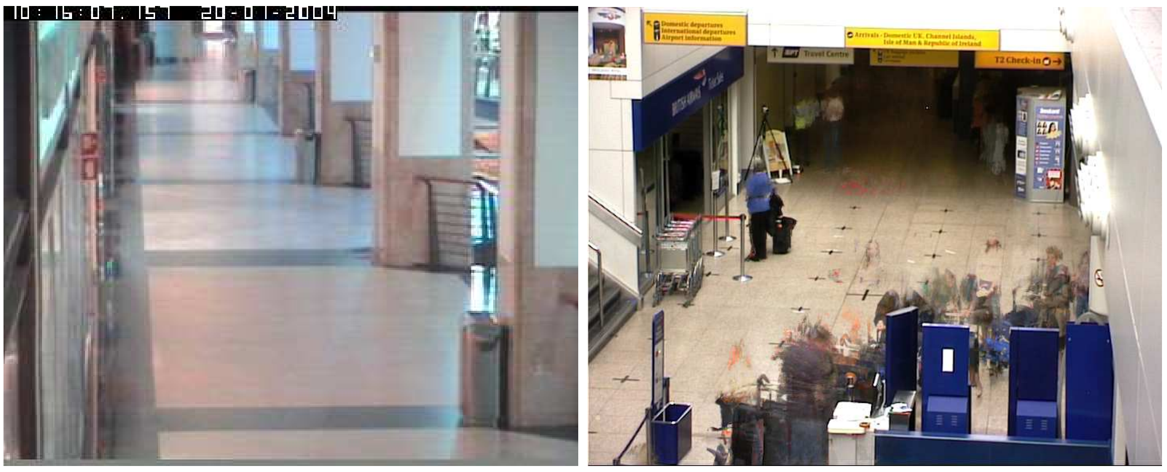


Figure 5.2: *Background scenes used for the CAVIAR (left) and PETS (right) sequences*

5.5 Combining background exclusion and multiple-part models

We propose an enhancement of the basic mean-shift tracker that is analogous to the work of Pérez *et al.* in the area of particle filters [122]. The goal is to create a mean-shift tracker with both of the following properties:

- a multiple-part appearance model: the model to be tracked should be represented by a number of histograms, so that some element of the spatial distribution of the object's colour is retained
- background exclusion: the candidate image region should look similar to the model but different to the corresponding region in the empty background scene

We achieve the above aims by combining the background exclusion tracker of Zhao [185] with the multiple-part models of Maggio [94] and Dong [173].

In Pérez's work the likelihood of a candidate region $\hat{\mathbf{p}}(\mathbf{y})$ at location $\mathbf{y}$ given the target $\hat{\mathbf{q}}$ is found using the expression

$$\exp -\lambda D^2[\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{y})] \tag{5.11}$$

The Bhattacharyya *distance* D and the Bhattacharyya *coefficient* ρ (equation 5.1) are related by the identity

$$D^2[\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{y})] = 1 - \rho[\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{y})]$$

For a multiple-part model with N subregions, expression 5.11 becomes

$$\exp -\lambda \sum_{P=1}^{N} D^2[\hat{\mathbf{q}_P}, \hat{\mathbf{p}_P}(\mathbf{y})]$$

and when a background model $\hat{\mathbf{d}}(\mathbf{y})$ is available Pérez uses

$$\exp -\lambda \left(D^2[\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{y})] - D^2[\hat{\mathbf{d}}(\mathbf{y}), \hat{\mathbf{p}}(\mathbf{y})] \right) \tag{5.12}$$

Expression 5.12 has a very similar form to equation 5.8, used by Zhao *et al.* to exploit "background exclusion" in mean-shift tracking (section 5.4). We modify Zhao's formula to allow the target, candidate and background models to have N subregions:

$$\hat{L'}(y) \triangleq \lambda_f \sum_{P=1}^{N} \rho \left[\hat{\mathbf{q}^P}, \hat{\mathbf{p}^P}(\mathbf{y}) \right] - \lambda_b \sum_{P=1}^{N} \rho \left[\hat{\mathbf{d}^P}(\mathbf{y}), \hat{\mathbf{p}^P}(\mathbf{y}) \right] \tag{5.13}$$

The weight w_i corresponding to the above definition is still given by equation 5.9:

$$w_i = \lambda_f w_i^f - \lambda_b w_i^b \tag{5.14}$$

However, the foreground weight w_i^f is now calculated over N subregions, since it is associated with a multiple-part model (section 5.3):

$$w_i^f = \sum_{P=1}^{N} \sum_{u=1}^{m} S^P C_h^P \sqrt{\frac{\hat{q_u^P}}{\hat{p_u^P}(\mathbf{y_0})}} \delta \left[b\left(\mathbf{x}_i\right) - u \right] I_R\left(\mathbf{x}_i, P\right)$$

Similarly, the new expression for the background weight w_i^b is given by

$$w_i^b = \sum_{P=1}^{N} \sum_{u=1}^{m} S^P C_h^P I_R\left(\mathbf{x}_i, P\right) \left(\sqrt{\frac{\hat{d_u}(\mathbf{y_0})}{\hat{p_u}(\mathbf{y_0})}} \delta \left[b_f\left(\mathbf{x}_i\right) - u \right] + \sqrt{\frac{\hat{p_u}(\mathbf{y_0})}{\hat{d_u}(\mathbf{y_0})}} \delta \left[b_b\left(\mathbf{x}_i\right) - u \right] \right)$$

The above expressions for w_i^f and w_i^b allow us to implement a mean-shift-based tracker that incorporates both the background exclusion constraint and multiple-part models. Again, we use equation 5.10 to move the tracker to its new location in the current frame.

5.6 Experiments

We conducted a number of experiments to assess the performance of trackers that utilise both multiple-part models and the background exclusion constraint. As in the previous chapter, we are concerned with the robustness of each tracker (whether it successfully follows its target), and also with its accuracy (how close it stays to the centre of the object). We attempt to track the designated target through each of the 21 scenarios in our dataset (section 3.2). Various accuracy metrics – dice coefficient, overlap and normalised centroid distance (section 3.3) – are recorded at each frame, and aggregate statistics derived from these measures are calculated at the end of each video sequence.

5.6.1 Tracker structure and parameters

Our derivation of mean-shift tracking with multiple-part models (section 5.3) allows us to create trackers having arbitrary spatial structure. We have implemented eight versions (depicted in figure 5.3), among which are a single-region tracker, one with four equal-sized quadrants and one with a 5×5 grid of cells. The single-region tracker is identical to the basic mean-shift method.

Each of the eight trackers – one basic and seven multiple-part – can either be run as they are or they can employ our innovation of incorporating background exclusion (section 5.5). In doing so, we double the number of trackers to be tested in the experiments.

We have retained the same tracker parameters as were used in the previous chapter: the trackers all operate in the RGB colour space, the histogram size is $4 \times 4 \times 4$ bins, the threshold on the convergence condition (ϵ) is set to 1 pixel, and a maximum of 20 iterations and 5 "half-steps" (see section 4.3.2) are permitted. The weights λ_f and λ_b, which determine the relative influence of the foreground and background models in those trackers that use background exclusion, are both set to 0.5.

5.7 Results

We now present the results of our experiments, looking first at the robustness of each tracker across all of the video sequences. An assessment of the accuracy of the different approaches is then provided.

5.7.1 Robustness

Table 5.1 summarises the performance of various trackers in attempting to follow their designated target in each of the 21 scenarios in our test set. In the top row we see the performance

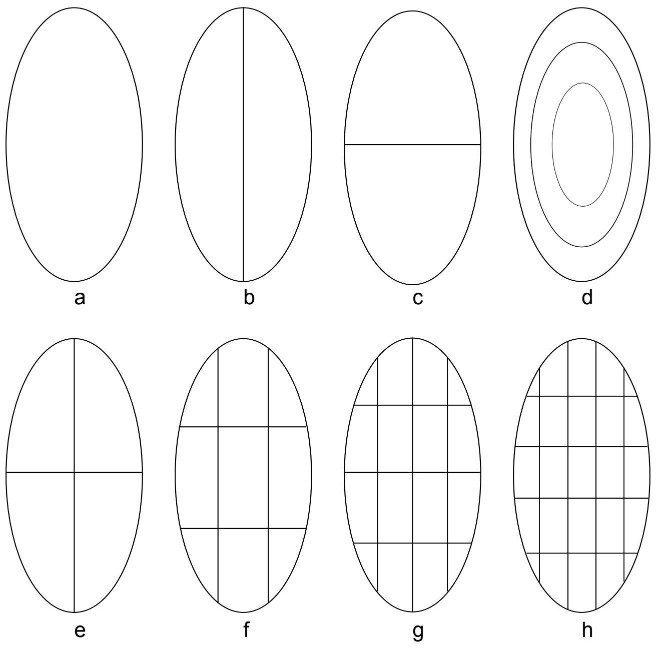


Figure 5.3: *Spatial division of the various trackers: (a) basic mean-shift (b) left-and-right (c) top-and-bottom (d) ellipses (e) quadrants (f) 9-cell (g) 16-cell (h) 25-cell*

of the basic mean-shift and the seven multiple-part trackers that do not make use of the background exclusion constraint. The number of lost tracks ranges from 9 for basic mean-shift to 16 for the 25-cell tracker.

The bottom row of the table gives the lost-track count when the background exclusion constraint is incorporated. Across the first five trackers the performance changes very little, although they are perhaps slightly more robust than their non-background exclusion equivalents. However, the final three background exclusion trackers (those with the 9-, 16- and 25-cell structures) demonstrate a significant improvement over their foreground-only counterparts. The number of lost tracks has fallen from 10 to 3 in the case of the 9- and 16-cell models, and from 16 to 5 for the 25-cell tracker.

A breakdown of the occurrence of lost tracks by video sequence is shown in tables 5.2 and 5.3, for trackers omitting and including the background exclusion constraint, respectively. As in the previous chapter, we see that the tracking failures are concentrated in the PETS dataset (scenarios 15 to 21), since its videos are more challenging than those of CAVIAR. Naturally, the disparity in performance between the two datasets will be more pronounced for techniques with a higher number of lost tracks.

The results presented in table 5.1 clearly show that the combination of multiple-part models and background exclusion can yield a mean-shift tracker that is much more robust

Tracker	Basic mean-shift	Multiple-part (left and right)	Multiple-part (top and bottom)	Multiple-part (ellipses)	Multiple-part (quadrants)	Multiple-part (9 cells)	Multiple-part (16 cells)	Multiple-part (25 cells)
Without background excl.	9	10	9	13	10	10	10	16
With background excl.	9	9	8	9	9	3	3	5

Table 5.1: *Number of lost tracks across 21 scenarios for each tracker, with and without background exclusion*

than either method in isolation. However, we note that not all multiple-part structures are equally effective in achieving this goal. It appears that the performance increases as more subregions are added to the model, but only up to a point. As we increase the region count further (beyond 16 cells in our experiments), the robustness of the tracker is reduced. It is therefore important to investigate different model structures in order to derive the greatest benefit from the technique.

Figure 5.4 shows example images of the foreground and background weights used by the combined mean-shift method. By using both the object attraction and background exclusion terms in a tracker, we can more easily follow the target through the scene.

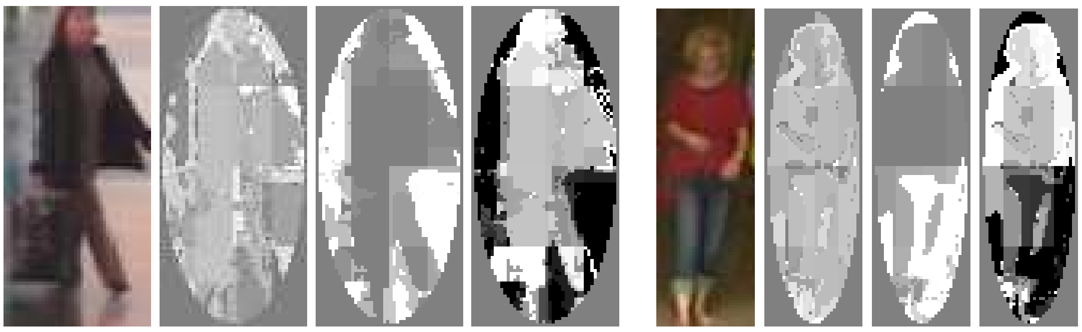


Figure 5.4: *Targets followed by the 16-cell background exclusion tracker and their corresponding foreground, background and combined weight images. Left: scenario 5; right: scenario 18.*

Tracker / **Scenario**	**Basic mean-shift**	**Multiple-part (left and right)**	**Multiple-part (top and bottom)**	**Multiple-part (ellipses)**	**Multiple-part (quadrants)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**	**Multiple-part (25 cells)**
1	x	x		x	x	x	x	x
2				x			x	x
3	x	x	x	x	x	x	x	x
4						x		x
5								x
6								
7								
8								
9	x	x	x	x	x	x	x	x
10								x
11				x				x
12								
13		x						x
14				x			x	x
15	x	x	x	x	x	x	x	x
16	x	x	x	x	x	x	x	x
17	x	x	x	x	x	x	x	x
18	x	x	x	x	x	x	x	x
19			x	x	x			
20	x	x	x	x	x	x		x
21	x	x	x	x	x	x	x	x
Num. trackers lost	9	10	9	13	10	10	10	16

Table 5.2: *The success/failure of the various non-background exclusion trackers operating on each scenario. (Failures are indicated by an 'x'.)*

Scenario \ **Tracker**	**Basic mean-shift**	**Multiple-part (left and right)**	**Multiple-part (top and bottom)**	**Multiple-part (ellipses)**	**Multiple-part (quadrants)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**	**Multiple-part (25 cells)**
1								
2								
3	x	x	x	x	x	x		x
4	x	x		x				
5								
6								
7								
8								
9								
10								
11	x				x			x
12								
13					x			
14								
15	x	x	x	x	x	x	x	x
16	x	x	x	x	x		x	x
17	x	x	x	x	x			
18	x	x	x	x	x			
19		x	x	x				
20	x	x	x	x	x			
21	x	x	x	x	x	x	x	x

Num. trackers lost	9	9	8	9	9	3	3	5

Table 5.3: *The success/failure of the various background exclusion trackers operating on each scenario. (Failures are indicated by an 'x'.)*

5.7.2 Accuracy: successful tracking

In the results that follow we use the dice coefficient (section 3.3.3) as the principal metric for assessing the accuracy of the various trackers. Here, we compare the performance of the trackers on videos where the target was successfully followed through the entire sequence. In the next section, we review the relative accuracy of the different trackers on the whole dataset, regardless of the number of lost tracks associated with each method.

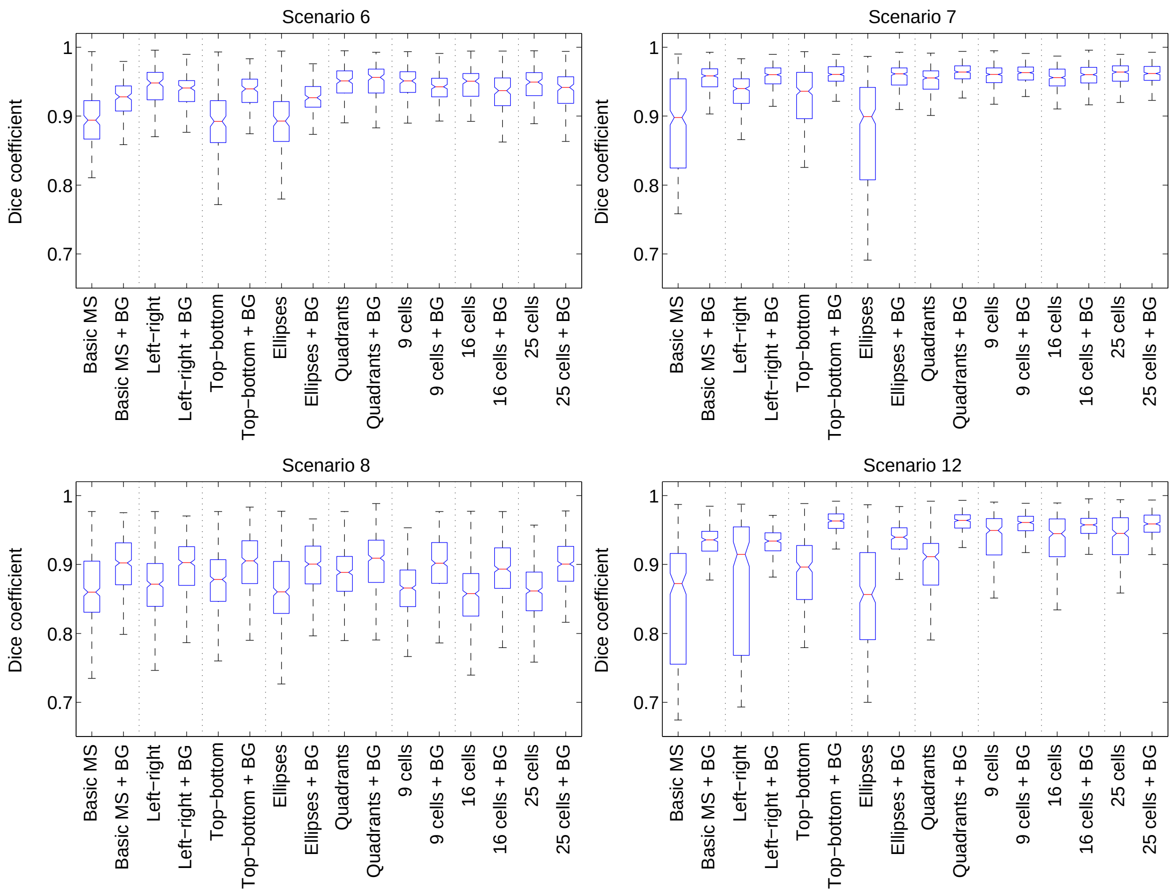


Figure 5.5: *Box plots of the dice coefficient of the various trackers operating on scenarios 6, 7, 8 and 12. The pairs of blue boxes – separated by dotted lines – show the performance without (left) and with (right) background exclusion.*

Figure 5.5 displays box plots of the dice coefficient for four separate tracking scenarios. (Tables 5.2 and 5.3 show that all of the trackers successfully followed their targets in these particular sequences – 6, 7, 8 and 12 – making them suitable for use in studying accuracy independently of robustness.) The boxes in the plots have been grouped into eight pairs, one for each multiple-part mean-shift variant, with each pair showing the performance of, firstly, a tracker that does not employ the background exclusion constraint and, secondly, the same tracker utilising the constraint.

Looking across the four scenarios, it is clear that, in general, the use of background exclusion improves the accuracy of tracking. (A similar trend is also seen in the remainder of the sequences.) The improvement is more pronounced in trackers that have fewer subregions in their multiple-part models. Some of the results shown in figure 5.5, e.g., scenarios 6, 7 and 12, also suggest that multiple-part trackers may be more accurate than the basic mean-shift method (when background exclusion is not used), especially as the number of subregions increases. In scenario 8, however, the basic technique has accuracy comparable to that of all the other (non-background exclusion) trackers, and so it is not conclusive that such a difference in performance exists between the mean-shift variants. We also note that combined multiple-part–background exclusion trackers do not appear to be any more accurate than the single-region background exclusion method: similar accuracy will be attained for any tracker incorporating the background constraint, regardless of how many subregions are in the model. We cannot assert definitively, therefore, that multiple-part models improve the accuracy of mean shift, at least for video sequences where tracking is ultimately successful.

5.7.3 Accuracy: entire dataset

The results of the previous section relate only to scenarios where all of the mean-shift variants followed their targets for the full length of the sequence. Figure 5.6 shows however that the same trends are seen even when lost tracks occur, albeit with lower average accuracy; these scenarios come from the PETS dataset. Next, we consider the entire collection of 21 videos. Table 5.4 records the median value, over all of the sequences, of the median dice coefficient for each tracker and for each video. As stated in section 3.3.5, only the frames from before the occurrence of a tracking failure contribute to the calculation of these aggregate metrics; we are attempting to assess a tracker's accuracy regardless of the ultimate outcome on a given sequence.

Looking at table 5.4, we see very different accuracy trends emerge compared to the results of the previous section. Firstly, trackers that incorporate the background exclusion constraint are more accurate than their non-background exclusion counterparts. The accuracy scores are relatively consistent for all of the background exclusion trackers, but they vary greatly for the versions of mean shift that do not exploit the constraint. For both classes of tracker (those with and without background exclusion), the use of multiple-part models improves upon the accuracy of the basic, single-region approach.

By considering the entire video dataset, we see that both background exclusion and multiple-part models, on their own, improve the accuracy of mean-shift tracking. When the

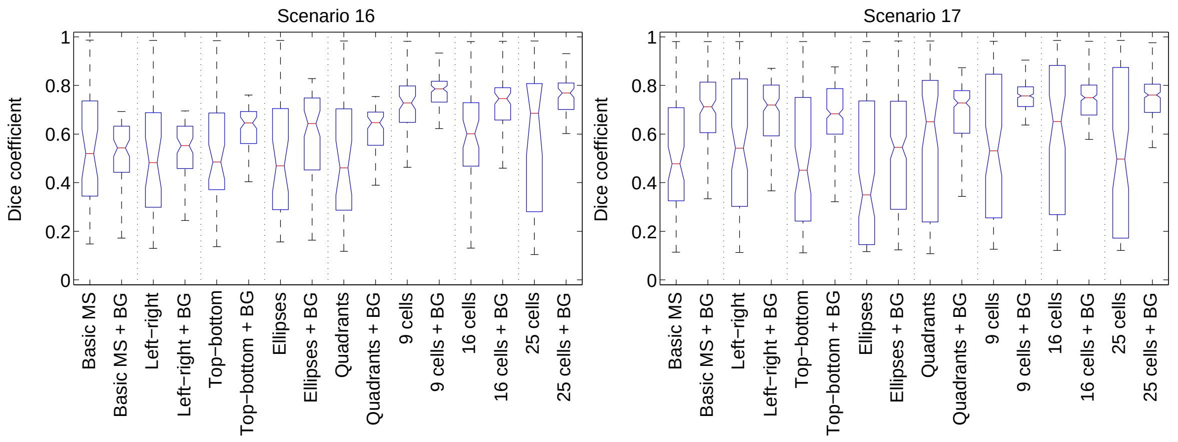


Figure 5.6: *Box plots of the dice coefficient of the various trackers operating on scenarios 16 and 17.*

two elements are combined, a further increase in accuracy is achieved. It is surprising that these trends only become apparent when we assess the performance of mean shift on all of the video sequences, whether or not tracking was ultimately successful. It is likely that low accuracy precedes many tracking failures; in such situations the tracker may be "struggling" to follow its target correctly for some time, being positioned some distance from the centre of the object. Eventually, mean shift drifts away from the object completely and a lost track is recorded, along with a low accuracy score.

Tracker	**Basic mean-shift**	**Multiple-part (left and right)**	**Multiple-part (top and bottom)**	**Multiple-part (ellipses)**	**Multiple-part (quadrants)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**	**Multiple-part (25 cells)**
Without background excl.	0.72	0.80	0.76	0.75	0.86	0.87	0.86	0.81
With background excl.	0.87	0.90	0.90	0.88	0.91	0.89	0.89	0.89

Table 5.4: *Median-of-medians of the dice coefficient across 21 scenarios for each tracker, with and without background exclusion*

5.8 Discussion

In this chapter we have taken two existing enhancements to mean-shift tracking – multiple-part models and background exclusion – and combined them in a single tracker. We assessed the performance of each technique by attempting to track objects through a number of video sequences. Neither multiple-part models nor background exclusion, on their own, provided any definitive benefit to the robustness of tracking, although each of them did improve the accuracy of mean shift. However, the combination of the two techniques, together with the appropriate spatial structure for the object model, yielded a tracker that is significantly more robust than any of the others.

The reasons for the combined tracker's improved performance are not immediately obvious. However, by looking at plots of the similarity surfaces in the neighbourhood of the basic and multiple-part trackers, we discover the relative influences of the object attraction and background exclusion terms (equation 5.13) on the robustness of each method. We have used two variants of mean shift that employ the background exclusion constraint to track the target in scenario 7 (figure 5.7). For the single-region tracker (figure 5.8), only the similarity surface associated with the background exclusion term is noticeably peaked; the tracker's lack of discriminative power is seen in the plateau of its object attraction surface. In contrast, the 16-cell tracker (figure 5.9) displays similarity surfaces which are both peaked, especially in the case of the object attraction term. It is therefore less likely that a tracker combining multiple-part models and the background exclusion constraint will drift away from the location of the target, whereas a tracker using only one or other of these elements may be drawn towards an incorrect position in the image.

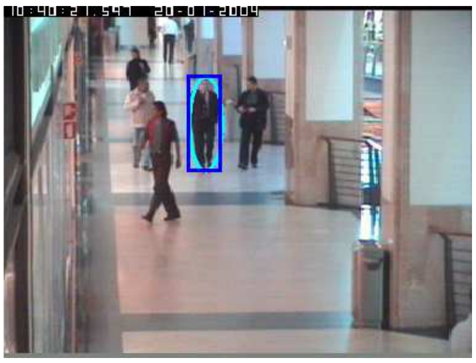


Figure 5.7: *Person to be tracked in scenario 7*

Figures 5.8 and 5.9 also raise the possibility of using different parameter settings for the two components of the combined mean-shift method – object attraction and background

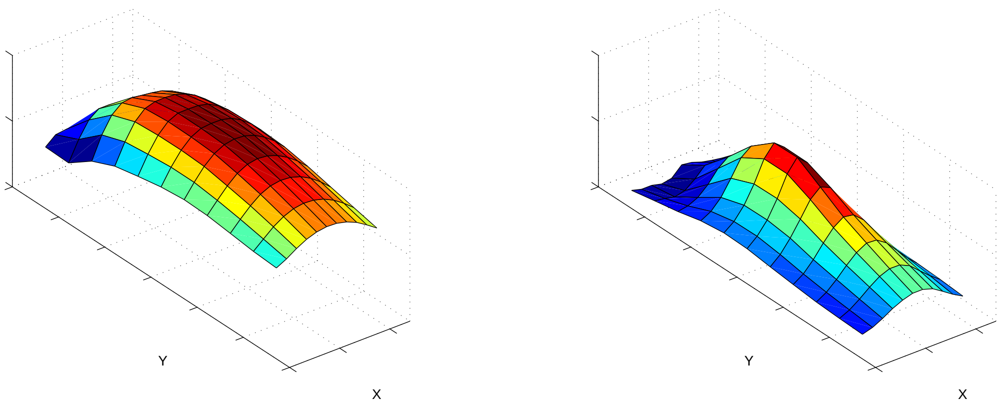


Figure 5.8: *Similarity surfaces for the single-region background exclusion tracker operating on scenario 7: object attraction component (left) and background exclusion component (right)*

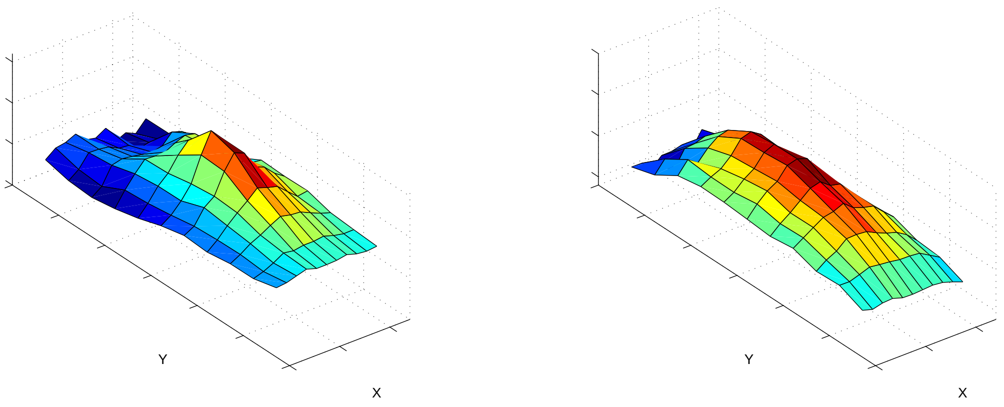


Figure 5.9: *Similarity surfaces for the combined 16-cell tracker operating on scenario 7: object attraction component (left) and background exclusion component (right)*

exclusion. For example, we could employ a 16-cell tracker for the object attraction element, while background exclusion could be achieved with the single-region version of mean shift.

Likewise, the values for λ_f and λ_b in equation 5.14, which determine the relative influence given to the foreground and background models, could be adjusted to favour one component of the tracker over the other. However, it is difficult to know in advance the ideal choices for the above parameters that will yield the best tracking result for a given video sequence. Further experiments would be required to determine values that work well in most cases. Alternatively, by sampling the similarity surfaces very sparsely during tracking, we could quickly assess their "peakedness", and adjust the settings whenever a surface appears to contain a plateau, and hence an indication that tracking is becoming difficult.

The multiple-part models that we have used (figure 5.3) have very generic structures: they do not correspond well with the limb articulations of people being tracked, for example. Parameswaran [118] developed a multiple-part mean-shift tracker that attempts to account for the typical shape and movements of pedestrians. The subregion corresponding to the lower legs – the bottom quarter of the tracker – is given less weight than the others because its appearance varies the most over time. A further possibility is to use a *cardboard model* [64] of the person being tracked, where the locations of their head, limbs, hands and feet are recorded explicitly. One subregion of the mean-shift tracker could be used to follow each such body part. However, the approach would need to allow for one limb being occluded by another.

Finally, we note that our combined multiple-part–background exclusion tracker can be used only when a background model of the scene is available. This makes it less flexible than normalised cross-correlation, which does not require any such model but which still displays impressive robustness and accuracy. In the next chapter we will investigate the effect of various parameters on the behaviour of mean shift. Our goal is to increase its performance, even in the absence of a background model, so that we can have a fast gradient ascent method that competes on robustness with normalised cross-correlation.

Chapter 6

Effect of tracker parameters

The previous chapter demonstrated the possibility of improving the performance of mean-shift tracking by incorporating spatial structure in the target and candidate histograms, and by exploiting background models of the scene. We now investigate whether the robustness and accuracy can be further improved by adjusting the various parameters associated with the technique. The colour space in which the tracker operates is first considered. Next we look at the effect of the number of histogram bins, and the size of the bounds, ϵ, that is used to decide when mean shift has converged to a local maximum. Finally, the type of spatial kernel employed to construct the target and candidate models is varied in order to study the influence it has on tracking.

6.1 Colour space

There are few comprehensive studies of the effect of the colour space on the performance of mean-shift tracking. Maggio *et al.* [94] evaluate eight different colour spaces on a test set containing nine video sequences. They conclude that RGB outperforms all of the other spaces tested: it was the only one that allowed all nine targets to be tracked successfully, and it also recorded the best average accuracy score.

Aside from Maggio's work, the colour space for most research into mean-shift tracking is chosen somewhat arbitrarily. In this section, we assess the performance of mean shift in several different spaces as it attempts to track the targets in each of the 21 video sequences in our test set. The aim is to determine whether certain colour spaces are particularly suited to mean-shift tracking, and, conversely, whether others should be avoided due to poor performance. We evaluate not only the basic mean-shift tracker but also the multiple-part trackers (both with and without background exclusion) developed in the previous chapter.

Num. channels \ **Base colour space**	**RGB**	**YUV**	**HSV**	**Normalised rgb**
3	RGB	YUV	HSV	
2	RG, RB, GB	YU, YV, UV	HV, HS, SV	rg, rb, gb
1	R, G, B	Y	H, S	

Table 6.1: *Colour spaces used in the experiments in this section*

A version of normalised cross-correlation, adapted to work with multiple-channel images, is also tested to provide a baseline against which mean shift can be compared.

6.1.1 Experiments

Table 6.1 shows the colour spaces used in our experiments. They are all derived from one of four "base" colour spaces: RGB, YUV, HSV or normalised rgb. We create variants of these by removing either one or two of the colour channels. Consequently, all of the colour spaces that we consider are either one-, two- or three-dimensional. (Note that we have not generated all possible single-channel spaces for our experiments: U and V are not used, for example.)

Normalised rgb, from which the popular rg-chromaticity space [154] is obtained, attempts to eliminate the effect of brightness from the RGB colour space. The data for the three channels – r, g and b – is calculated according to the following formulas:

$$r = \frac{R}{R+G+B}; \qquad g = \frac{G}{R+G+B}; \qquad b = \frac{B}{R+G+B}$$

For all of the mean-shift-based trackers used in this experiment, ϵ (the convergence condition) was set to 1.0, the histograms contained 16 bins per colour channel, and the Epanechnikov kernel was employed. We have run four mean-shift trackers that do not include the background exclusion constraint: the basic (single-part) method, and the 4-quadrant, 9-cell and 16-cell multiple-part trackers (section 5.3). A further four trackers – having the same multiple-part arrangements – that do exploit background exclusion (section 5.4) are also tested. Using a large selection of trackers makes it easier to discover trends that may be present in the data.

As in chapter 4, we use an alternative method as a baseline against which to compare the performance of mean shift. Previously, normalised cross-correlation (NCC, section 4.1.4) was employed for this purpose. However, we must modify the technique to accommodate two- and three-channel image data. Whereas the standard NCC $\gamma(u, v)$ of an image patch f and a template t at location (u, v) is given by:

$$\gamma\left(u,v\right)=\frac{\sum_{x,y}\left[f\left(x,y\right)-\bar{f}_{u,v}\right]\left[t\left(x-u,y-v\right)-\bar{t}\right]}{\left\{\sum_{x,y}\left[f\left(x,y\right)-\bar{f}_{u,v}\right]^{2}\sum_{x,y}\left[t\left(x-u,y-v\right)-\bar{t}\right]^{2}\right\}^{0.5}},$$

the new measure is found according to:

$$\eta\left(u,v\right)=\frac{1}{n}\sum_{c}\sum_{x,y}\left[f\left(x,y,c\right)-\bar{f}_{u,v,c}\right]\left[t\left(x-u,y-v,c\right)-\bar{t}_{c}\right],$$

where c is the colour channel to be processed and n is the number of pixels in the image patch. It is the akin to the summation of the normalised cross-correlations of the individual channels, although the division by the standard deviation has been omitted. (We found that this step had little effect on performance, and so we removed it for efficiency reasons.)

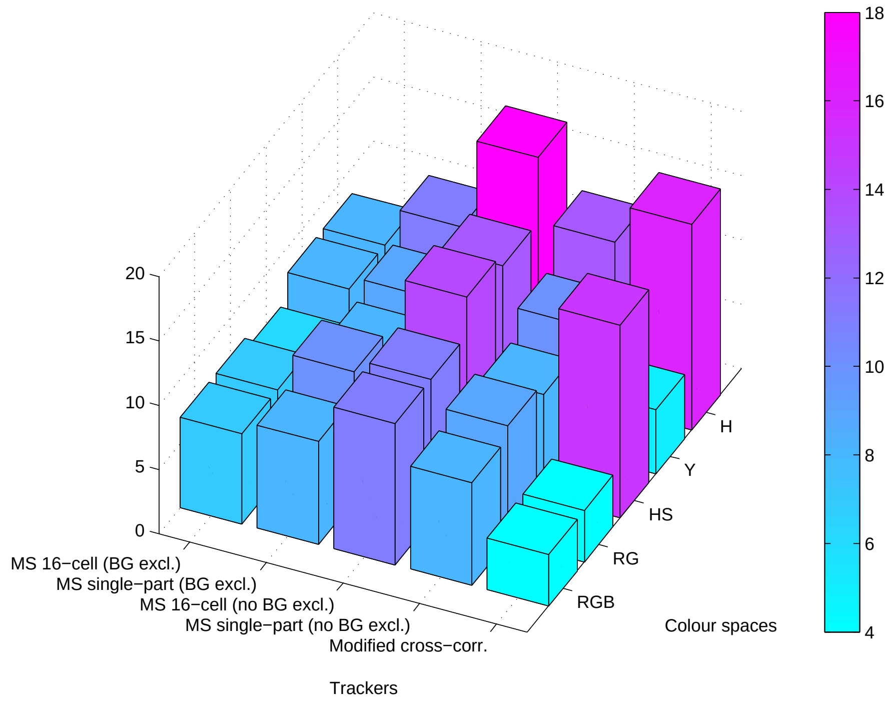
Other approaches to multiple-channel NCC can also be developed. For example, we could treat each pixel as a (one-, two- or three-dimensional) vector and replace the multiplication in the above formula by the dot product. Alternatively, we could apply *principal components analysis* [74] to the image data before using the formula. In both cases, we would be taking better advantage of information that is correlated across the colour channels.

As before, a brute-force search strategy is used with the correlation-based tracker.

6.1.2 Results

We have tested each of the 21 colour spaces shown in table 6.1 by recording the number of lost tracks that occurred when using the various trackers on our dataset of 21 CAVIAR and PETS video sequences (section 3.1). As before, we regard this as a measure of the trackers' *robustness*. We have also used the dice coefficient (section 3.3.3) as the basis for assessing the *accuracy* of the trackers across the colour spaces: the median dice coefficient for a single video sequence and colour space is calculated, and the mean of this value across all sequences (MMDC) serves as our summary statistic. In order to present the results in an accessible form, we include only certain trackers and colour spaces in the barcharts of figure 6.1. The results for a single- and a multiple-part mean-shift tracker (with 16 cells) – both with and without background exclusion – are shown, along with those for the modified cross-correlation technique. We have selected the best-performing colour space from each of five categories (explained below) for inclusion in the barcharts. Detailed results for all of the trackers and colour spaces can be found in appendix A, tables A.1 and A.2.

Looking at the counts of lost tracks in the first diagram of figure 6.1, it is apparent that none of the selected colour spaces greatly outperforms all of the others across the various tracker types. Instead, we seek trends in the data from which we attempt to extract more



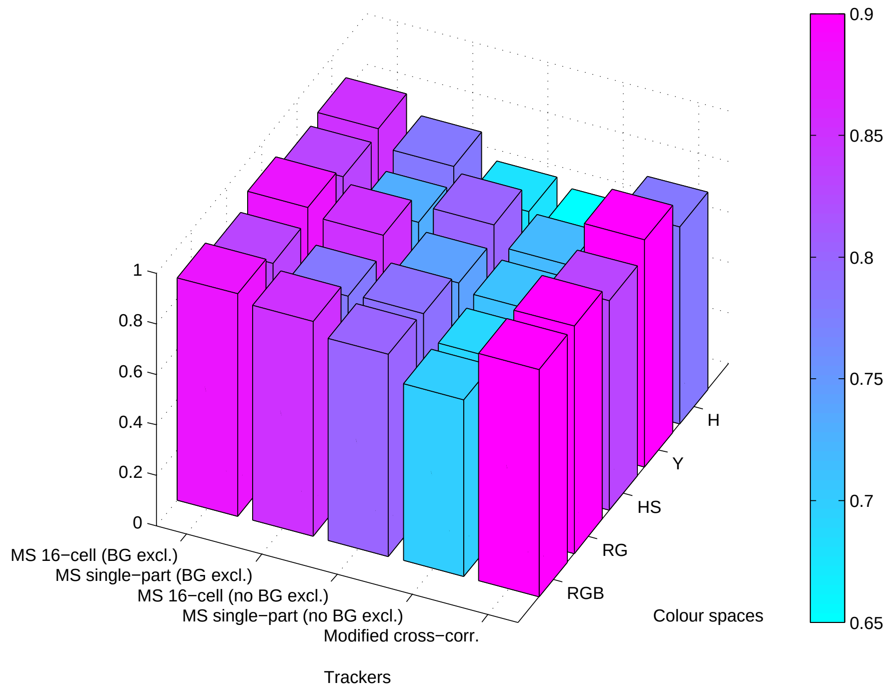


Figure 6.1: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various colour spaces and trackers*

general principles concerning the use of colour spaces. For example, the results show that, for most of the spaces, single-part mean-shift trackers do not become more robust (having fewer lost tracks) if we incorporate the background exclusion constraint. However, background exclusion does improve the *accuracy* (increases the MMDC) of single-part trackers. In contrast, for multiple-part methods both the robustness and accuracy are improved by the use of background exclusion.

Comparing single- and multiple-part trackers, we see that the latter are somewhat more accurate, regardless of whether background exclusion is employed or not. However, this gain in accuracy does not always translate into improved robustness: both background exclusion and multiple-part models are needed for an improvement in the lost-track count to be seen. Indeed, the 16-cell tracker performs very badly without the background exclusion constraint. This finding is consistent with the results of the previous chapter, and here we see that it holds across the various colour spaces.

We have placed each of the 21 colour spaces tested into one of five categories, based on the number of channels in the space and its performance when used with the modified cross-correlation tracker. (The groupings are clearly seen in the first column of table A.1, appendix A.) The colour space judged the best-performing in each category – across all trackers – was used as the representative for that category; these are the spaces for which results are displayed in figure 6.1.

It is interesting to note that the first group of two-channel colour spaces in table A.1 – SV, HV, RG, RB, GB, YV and YU – all encode image brightness, in the sense that at least some of the *luma* information (the Y in YUV) is retained. In contrast, virtually all of the brightness information has been removed from the remaining two-channel colour spaces: HS, UV, rg, rb and gb – only *chrominance* data is present. A similar distinction can be drawn for single-channel spaces. There is a marked difference in performance, as measured by the lost-track count of the modified cross-correlation tracker, depending on which of these colour space groups we use. On the basis of this observation, we suggest that it is important to incorporate brightness information, in some form, into correlation-based tracking. Indeed, the data also indicates that multiple-part mean-shift trackers should either use such colour spaces or exploit the background exclusion constraint. It is noteworthy that most image and video encoding schemes (e.g., JPEG and MPEG) store colour information at a much-reduced resolution, whereas the brightness data is stored at full resolution.

Returning to the results shown in figure 6.1, we can see that the multiple-part mean-shift tracker without background exclusion performs poorly in all colour spaces. The weakness is particularly pronounced in the HS, Y and H spaces, where the lost-track count is significantly higher than for any other mean-shift tracker. Of the five colour spaces shown, the single-channel hue space (H) causes the greatest difficulty for mean shift; only the multiple-part tracker with background exclusion performs well in this situation.[1] If we disregard the results associated with the hue space, however, there is no obvious systematic variation in the results across the different colour spaces, either in terms of robustness or accuracy. Mean-shift trackers using RGB or RG seem to operate the most consistently, regardless of the use of multiple-part models or background exclusion. We can also see that trackers which combine these two elements – multiple-part models and background exclusion – perform well in all of the colour spaces tested.

As discussed previously, the performance of our modified cross-correlation tracker depends strongly on whether or not the colour space used encodes brightness information. In figure 6.1 we can see that the tracker performs very poorly in the HS and H spaces. (Table A.1 reveals similar poor performance whenever brightness data is not available.) However, aside from these colour spaces the correlation-based tracker is far more robust than any of the mean-shift methods, in agreement with the results of normalised cross-correlation in chapter 4.

6.2 Histogram size

The second mean-shift parameter that we consider in our experiments is the number of bins to be used in the construction of the target and candidate histograms. Fewer bins will reduce the computation needed to perform tracking, but it will make the target model less discriminative. Conversely, a larger number of bins will result, on average, in fewer data points falling in a given bin's range, and will make calculations using such data more susceptible to noise. In addition, a greater number of bins will be empty; the mean-shift procedure requires that the results of calculations involving empty candidate bins be discarded (to avoid division by zero). We study the effect of the number of bins on tracking performance to determine which of these considerations is the most important.

In our experiments, we have again used the Epanechnikov kernel and set ϵ to 1.0. Three of the colour spaces from the previous experiment were used for the tests: RGB, RG and HSV. The number of bins *per dimension* was initially set to 2, and was successively doubled until it reached 32.

The histogram bins in our experiments are all of uniform width: each one spans the same number of greylevels/colour levels for a given histogram size. Further experiments could be performed in which each bin's size is different, having been determined by certain criteria. For example, we might wish to ensure that approximately equal numbers of pixels fall into

[1]The circular nature of the hue space does not affect mean shift since it never uses data from different histogram bins in its calculations. Circular colour spaces may reduce the performance of correlation-based tracking, however.

each of the bins. Other approaches to adaptive histograms have been pursued by Town [159] and Li [85].

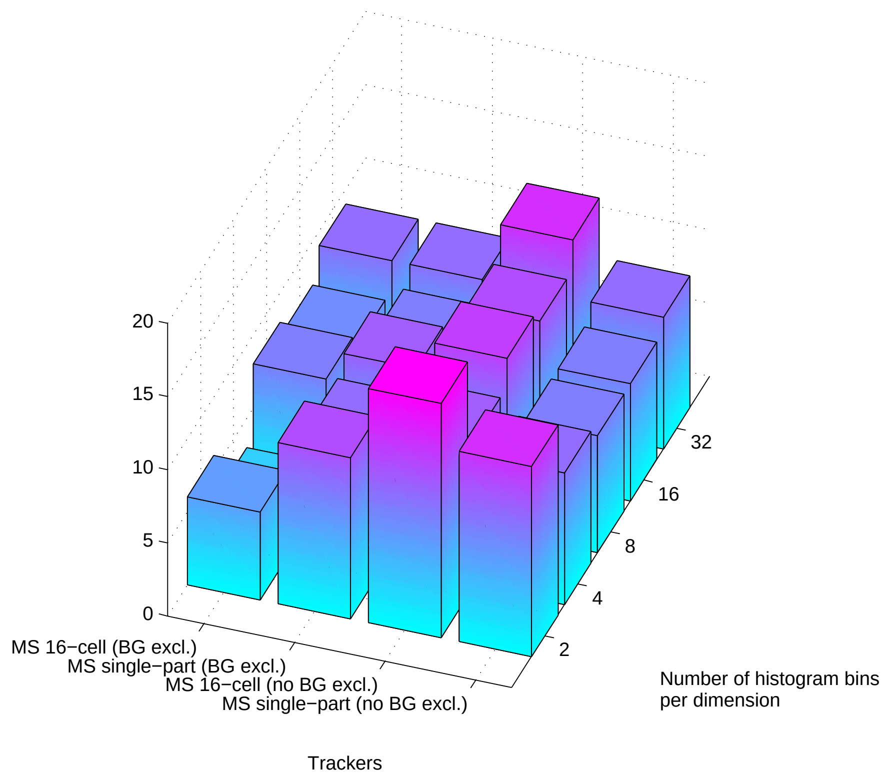
The single- and multiple-part (16-cell) trackers – both with and without background exclusion – were used for the experiments. We employed the same metrics as before – the count of lost tracks and the mean of the median dice coefficient (MMDC).

6.2.1 Results

The results of the experiment on the histogram size for the RGB colour space are presented in figure 6.2. (Further results for the RG and HSV colour spaces can be found in appendix A, figures A.1 and A.2.) It is apparent that the number of lost tracks does not display any systematic trend as the number of histogram bins is increased, regardless of the type of tracker or colour space used. Nevertheless, certain parameter settings are associated with lost-track counts that are either significantly higher or significantly lower than what is typical for the given tracker. For instance, the first diagram of figure 6.2 reveals that the single- and multiple-part trackers that do not incorporate background exclusion perform very badly when we use only two bins per dimension. (The same result holds for the RG and HSV spaces; see figures A.1 and A.2). Conversely, a low bin count – two or four bins per dimension – is associated with very robust tracking (few lost tracks) when we use the tracker that combines background exclusion and multiple-part models.

As with the results of section 6.1.2, we can see that the type of tracker has a significant influence of the counts of lost tracks. In all of the colour spaces tested, the multiple-part tracker that does not exploit the background exclusion constraint is generally the worst-performing – a result that also held in the previous experiment on colour spaces. The poor performance exists regardless of the number of histogram bins used. In contrast, the tracker combining background exclusion and multiple-part models does as well as, or better than, all of the others, depending on the colour space in which it operates.

Aside from robustness (the count of lost tracks), we also assessed the effect of the number of histogram bins on the accuracy of mean shift (the mean of its median dice coefficient over several sequences). The results for the RGB colour space are shown in the second diagram of figure 6.2. We can firstly see that trackers which incorporate the background exclusion constraint are significantly more accurate than those that do not, whether single- or multiple-part. Furthermore, higher accuracy is attained by using multiple-part trackers, regardless of the presence or absence of background exclusion. These results hold for each of the colour spaces tested (see figures A.1 and A.2 for RG and HSV), and for almost every choice of histogram size. In the case of the RGB tests (figure 6.2), increasing the number of bins



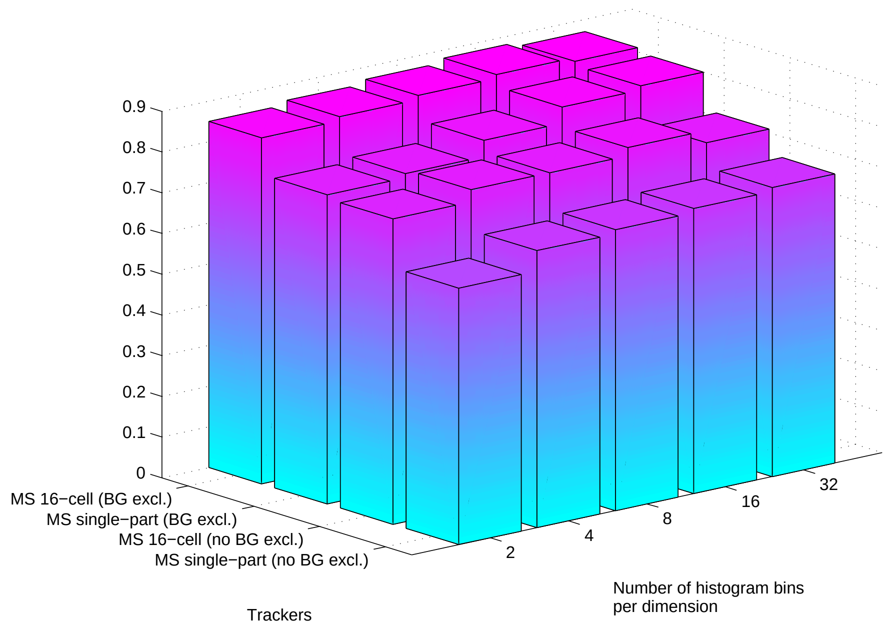


Figure 6.2: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various histogram sizes and trackers operating in RGB*

generally leads to an increase in accuracy for all of the trackers. However, this trend is not seen in the other colour spaces tested.

Taken together, the results of this section further support the claim that trackers which combine the background exclusion constraint and multiple-part models are the most robust and accurate of all of the mean-shift methods tested. In addition, provided we use one of the colour spaces tested, a small histogram (having, e.g., two or four bins per dimension) is likely to yield performance that is close to the best attainable for this type of tracker.

6.3 Convergence condition

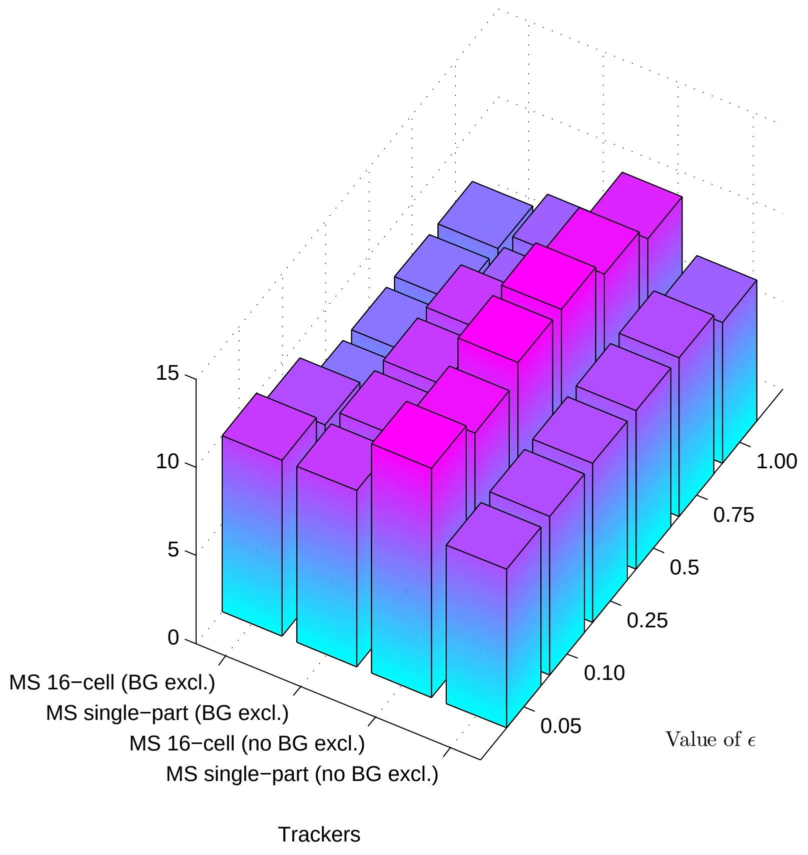
We now investigate the effect on tracking of changing the size of the parameter associated with the convergence condition of mean shift. The mean-shift algorithm is an iterative procedure; it terminates when the distance between the image locations returned by successive iterations is less than a specified threshold. It is this threshold, ϵ, whose size we will vary in the present experiment.

In mean-shift tracking it is common to set ϵ to a value of 1 pixel. However, using such a large threshold might result in the mean-shift iterations terminating (converging) before the tracker has reached a location that is sufficiently close to the local maximum of the Bhattacharyya surface. It is conceivable that the tracker would always tend to "lag" behind the correct location at convergence, reducing its accuracy. In the worst case, if such inaccuracy were compounded over several frames, mean shift might lose track of its target completely. The present experiment seeks to determine the extent to which such problems occur in practice.

Once again, we used the Epanechnikov kernel, and tested three of the colour spaces from section 6.1: RGB, RG and HSV. The number of bins per dimension was set to 16, and the single- and multiple-part mean-shift trackers of the previous section – both with and without background exclusion – were tested. Performance was assessed by using the same metrics as before – the count of lost tracks and the mean of the median dice coefficient (MMDC).

6.3.1 Results

The results of the experiment on the convergence condition for the RGB colour space are presented in figure 6.3. (Further results for the RG and HSV spaces can be found in appendix A, figures A.3 and A.4.) For most of the trackers, and for all of the three colour spaces tested, we find that a higher value of ϵ is usually associated with more robust tracking (fewer instances of lost tracks), although the trend is not very strong. The one exception is the multiple-part tracker without the background exclusion constraint: it does not display,



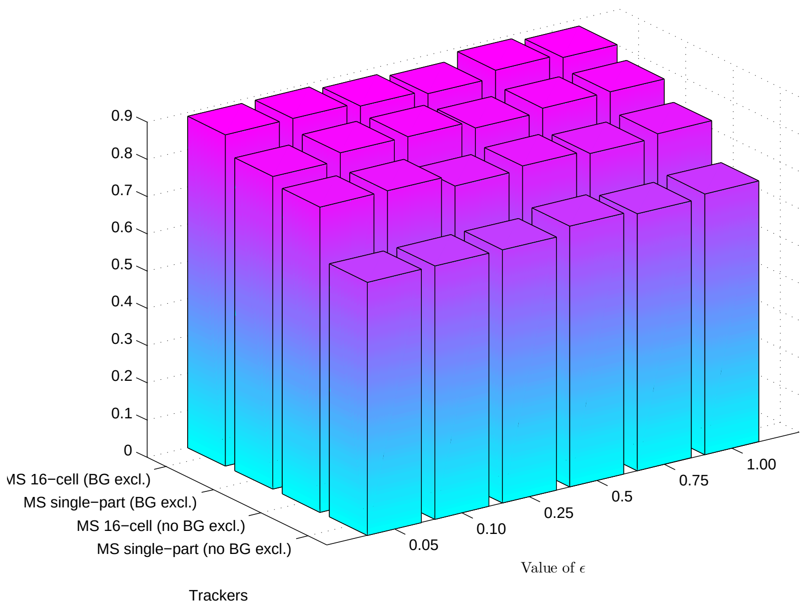


Figure 6.3: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various values of* ϵ *and trackers operating in RGB*

for any of the colour spaces tested, a consistent trend in the number of lost tracks as the value of ϵ is increased.

Again we see that the type of mean-shift tracker used has a significant influence on the quality of the results. The multiple-part tracker that exploits background exclusion always performs as well as, and usually better than, any of the other mean-shift methods tested, regardless of colour space or the value of ϵ. The worst-performing technique (in terms of lost tracks) is either the multiple-part method without the background exclusion constraint or the single-part tracker that relies on the constraint, depending on the colour space used for the experiment (see figures A.3 and A.4 for the RG and HSV results).

As before, we have assessed the accuracy of each tracker while changing the value of ϵ: the lower diagram of figure 6.3 displays the MMDC (accuracy) scores. The same trends as for the histogram size experiment (section 6.2.1) are seen: trackers which use background exclusion are more accurate than those that do not, whether single- or multiple-part; and higher accuracy can be achieved by using multiple-part trackers, whether or not we also employ background exclusion. Although these trends exist across the different types of trackers (and also across the colour spaces), the value of ϵ has no discernible impact on the accuracy of an individual tracker.

The present experiment has again shown that the tracker combining background exclusion and multiple-part models almost always outperforms all of the other mean-shift methods tested, both in terms of robustness and of accuracy. We should choose a large value of ϵ to maximise the performance of this tracker, a parameter setting which conveniently reduces the number of iterations needed for mean shift to reach convergence.

6.4 Kernel type

The final experiment in this chapter seeks to determine the effect of the type of spatial kernel used in mean-shift tracking. The standard version of the tracker uses the Epanechnikov kernel (see figure 6.4) because it simplifies the mean-shift formula:

$$\mathbf{y_1} = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\|\frac{\hat{\mathbf{y_0}}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{\mathbf{y_0}}-\mathbf{x}_i}{h}\right\|^2\right)} \tag{6.1}$$

Since the profile of the Epanechnikov kernel has a derivative g that is constant, it disappears from equation 6.1. However, there may be advantages to using alternative functions. Firstly, the choice of kernel will determine the relative influence of pixels closer to the centre of the target versus those nearer the boundary. Secondly, as explained by Hager *et al.* [62], a given kernel may be unable to detect certain types of translational or rotational motion that an object is undergoing. This idea of "kernel observability" [48] plays a crucial role in determining which parts of an image can be successfully tracked over time.

For the present experiment, we have used the Epanechnikov kernel and four others that are closely related to it. All five of the kernels have profiles that are defined by the equation[2]:

$$k(x) = \begin{cases} l\,(1 - x^n) & \text{if } x \leq 1 \\ 0 & \text{otherwise} \end{cases}, \quad n = \{0.33,\, 0.5,\, 1.0,\, 2.0,\, 3.0\} \tag{6.2}$$

The values of n in equation 6.2 lead us to refer to the kernels as *cube root* ($n = 0.33$), *square root* ($n = 0.5$), Epanechnikov ($n = 1$), *quadratic* ($n = 2$) and *cubic* ($n = 3$). The kernels and their profiles are shown in figure 6.4. (A kernel K is related to its profile k by the formula $K(\mathbf{x}) = k\left(\|\mathbf{x}\|^2\right)$) As before, we tested the RGB, RG and HSV colour spaces, used 16 bins per colour channel and set ϵ to 1.0. The same trackers and metrics as in the previous experiment were also employed.

6.4.1 Results

The results of the experiment on the kernel type for the RGB and RG colour spaces are presented in figures 6.5 and 6.6. (Further results for the HSV space can be found in appendix A, figure A.5.) The effect of the kernel type on the robustness of tracking depends strongly on the colour space and the particular tracker used. In the case of the RGB and RG colour spaces, the single-part tracker without the background exclusion constraint and the multiple-part tracker employing the constraint both perform better with kernels having smaller values of n (equation 6.2), e.g., the cube root and square root kernels. However, for the multiple-part tracker without background exclusion, the best performance in these colour spaces is achieved with a cubic kernel. The remaining tracker – single-part incorporating background exclusion – does not display a consistent trend in the number of lost tracks as the value of n (the kernel type) is varied. (For the HSV colour space, only one tracker – multiple-part without background exclusion – exhibits an obvious trend: in contrast to its behaviour in the other spaces, it performs best with a cube root kernel; see figure A.5.)

As with the results of the previous experiments, we find that the tracker combining multiple-part models and background exclusion almost always records the lowest number of lost tracks, regardless of the colour space or the kernel type used. If we omit background exclusion, however, we find that there are few choices of colour space and kernel type that

[2]As with mean-shift segmentation [30], we set the value of the normalising constant to $l = 15/(8\pi) \approx 0.59683$

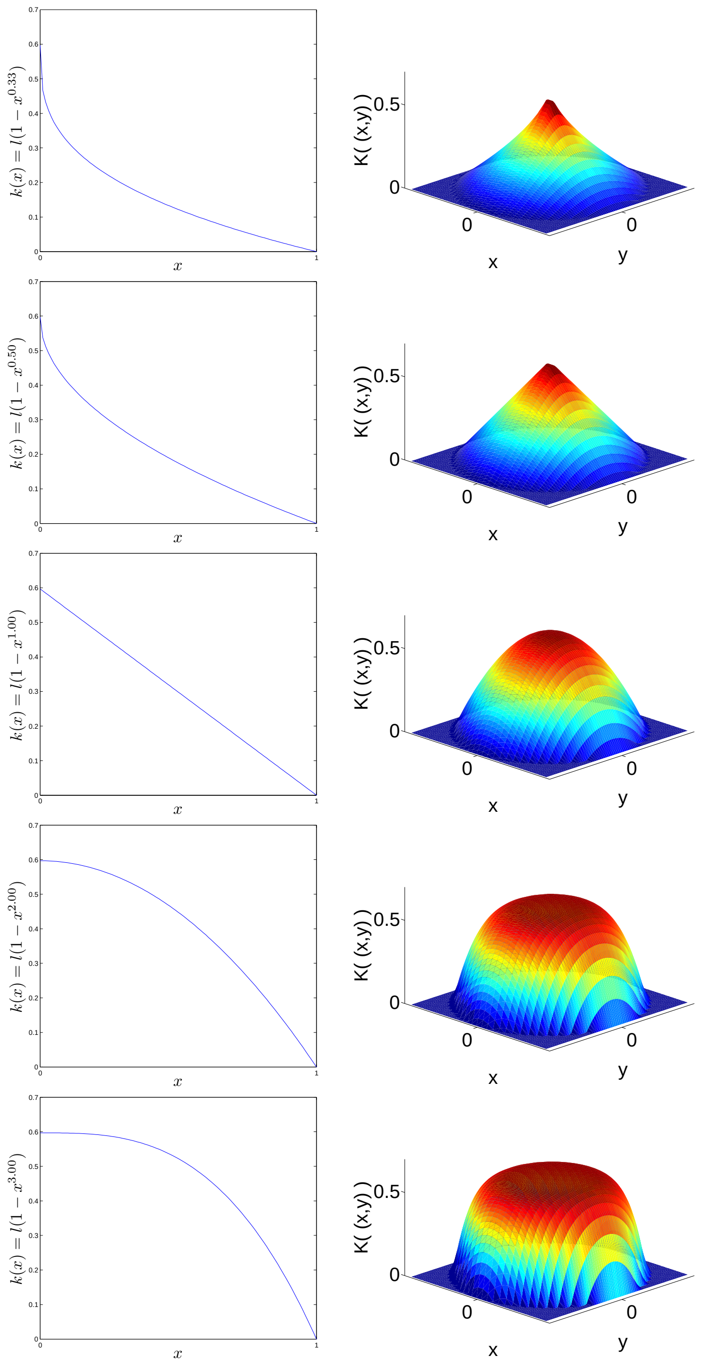


Figure 6.4: *Kernel profiles (left column) and their corresponding two-dimensional spatial structure for a circular image region (right column). Top to bottom: cube root, square root, Epanechnikov, quadratic, cubic*

can prevent the multiple-part tracker from recording the worst robustness scores of any of the mean-shift methods tested.

The lower diagrams of figures 6.5 and 6.6 display the MMDC (accuracy) scores of the four mean-shift trackers as the type of kernel is varied. We see the same inter-tracker trends as before: employing background exclusion improves accuracy, whether with single- or multiple-part models. Likewise, multiple-part trackers are always as accurate as, and often more accurate than, the single-part versions, regardless of the presence or absence of background exclusion. However, the type of kernel used has no discernible systematic impact on the accuracy scores of an individual tracker.

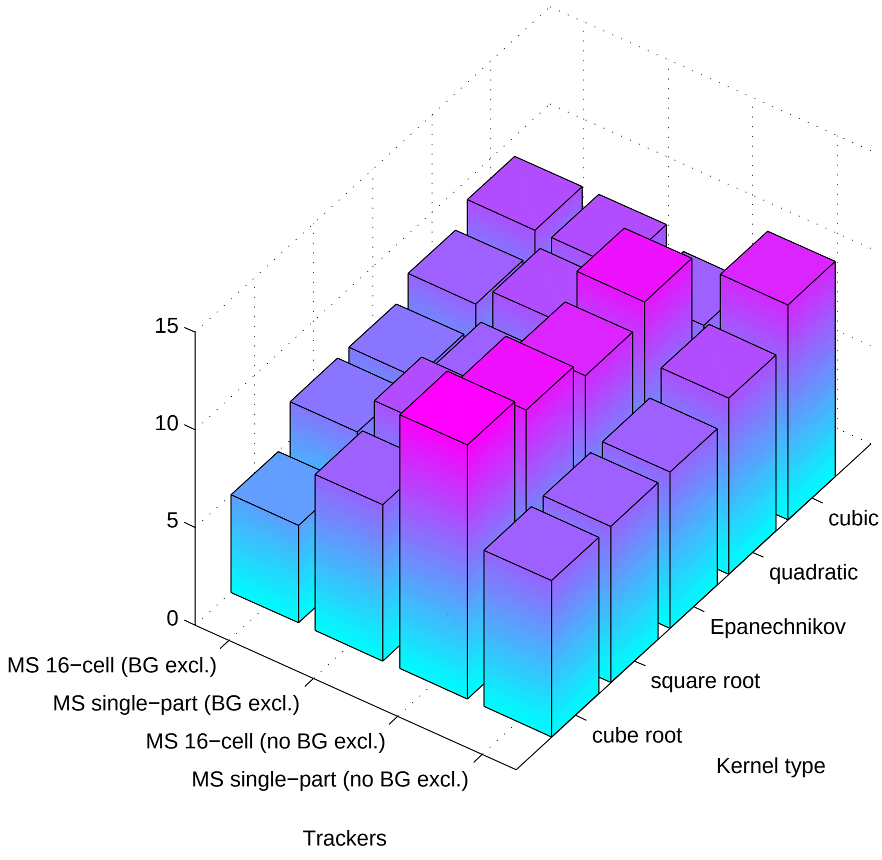
The experiments in this section again reveal that the tracker combining background exclusion and multiple-part models is amongst the best-performing, both in terms of robustness and of accuracy, of all of the mean-shift methods tested. By using the cube root kernel with this tracker, we may be able to improve its robustness further, depending on the colour space chosen.

6.5 Discussion

In this chapter we have investigated the effects of varying a number of parameters that are associated with the mean-shift tracking technique: the colour space, the size of the target and candidate histograms, the threshold used to decide when convergence has occurred and the type of spatial kernel employed by the tracker. Based on our experiments, we find that it is necessary to set the parameters very carefully in order to achieve optimal performance with mean shift. There are some discernible trends in the trackers' performance metrics, and these can be used to determine parameter settings that increase the chances of successfully tracking a target.

Looking at the results of the experiments in this chapter as a whole, it is clear that the single biggest influence on the performance of mean-shift tracking is the type of tracker used. The number of lost tracks decreases noticeably when we employ the background exclusion constraint in combination with multiple-part models, regardless of the settings of other parameters – reinforcing the results obtained in the previous chapter. In contrast, multiple-part trackers without background exclusion are usually the least robust of all the mean-shift methods and should be avoided. With regard to the accuracy of the trackers, the following trends are apparent: multiple-part models lead to higher accuracy than single-part ones, and trackers that incorporate background exclusion are more accurate than those that do not.

The question of which colour space the tracker should use has yielded some interesting data. Trackers based on cross-correlation perform very poorly in colour spaces that do not



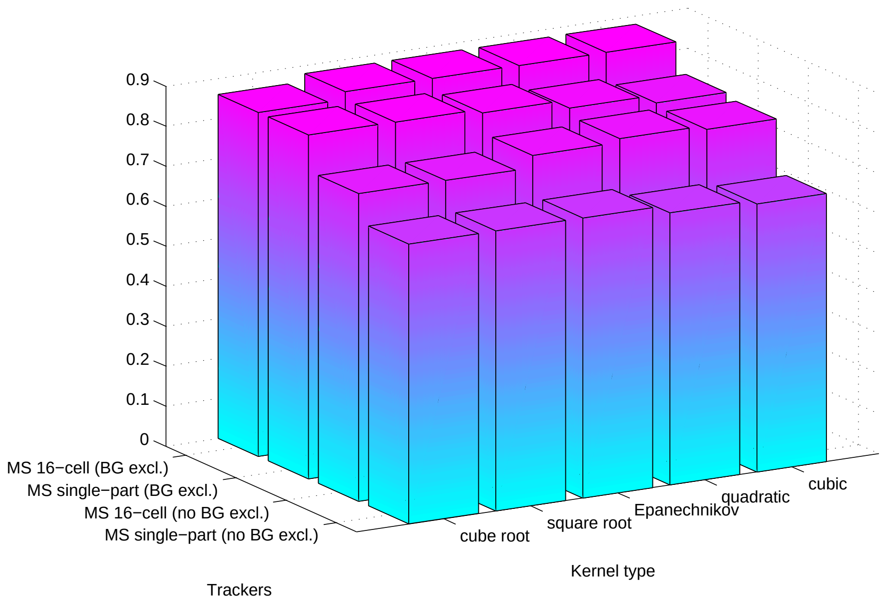
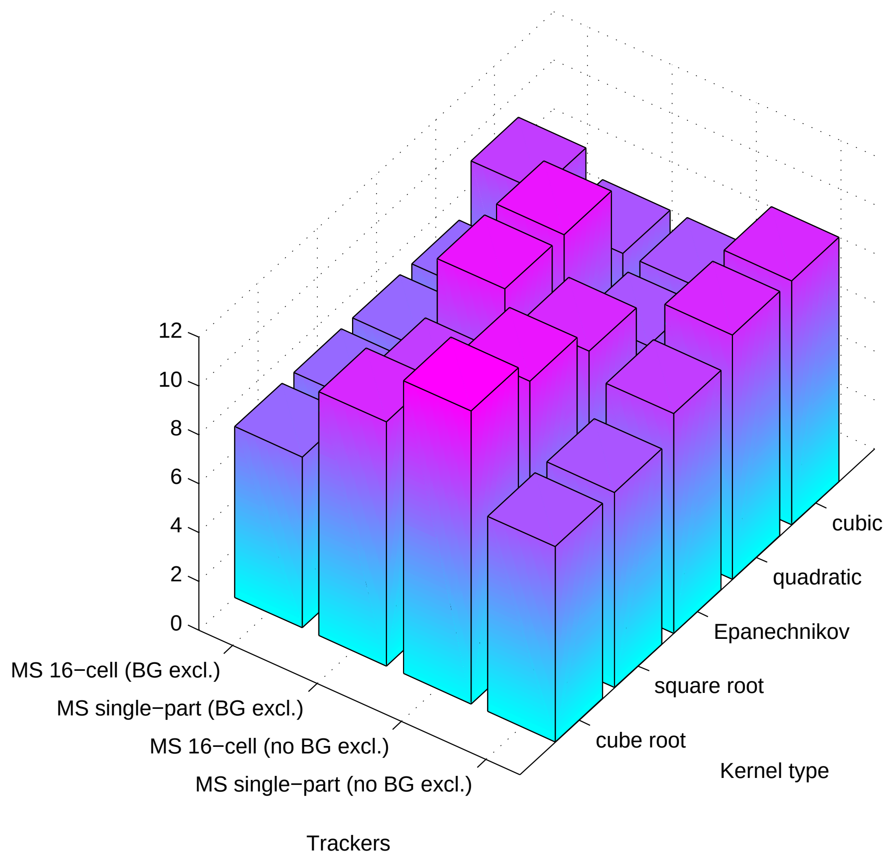


Figure 6.5: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various kernel types and trackers operating in RGB*



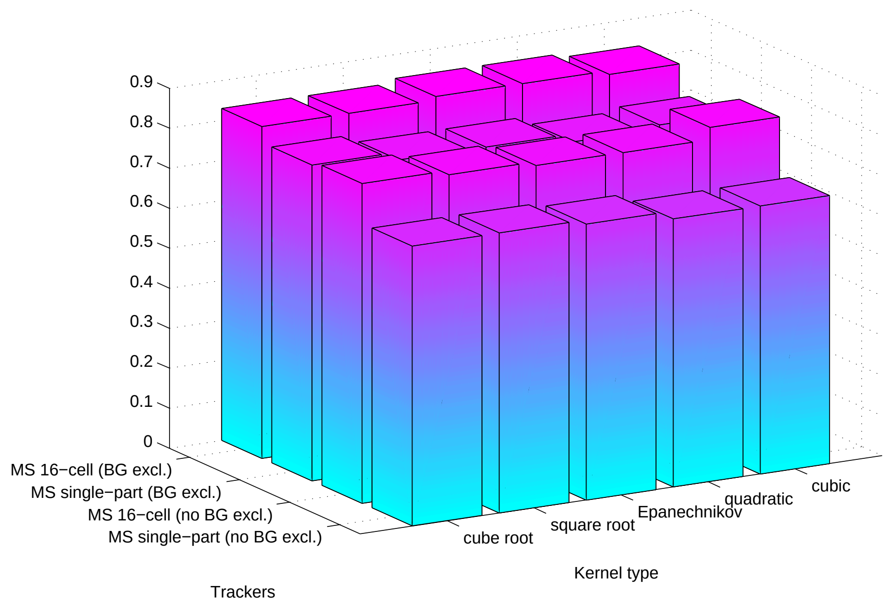


Figure 6.6: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various kernel types and trackers operating in RG*

encode image brightness in some form. For mean-shift methods, we should also avoid spaces that lack a brightness component. Beyond these, however, no great differences are seen with the other colour spaces, provided we disregard the poorly performing multiple-part tracker without background exclusion.

With regard to the size of target and candidate histograms, the results suggest that very small numbers of bins (e.g., two per dimension) should be avoided if we do not exploit background exclusion. However, if we make use of the background exclusion constraint, a low bin count can actually improve the robustness of tracking. In the case of the convergence parameter, higher values of ϵ are associated with greater robustness in the tracker combining multiple-part models and the background exclusion constraint (the best-performing tracker overall). The final experiment reveals that the effect of the type of spatial kernel used depends largely on the tracker and the colour space tested. For example, we can enhance the robustness of the best-performing tracker further by using the cube root kernel. We caution that the trends found in our results may be particular to the video sequences that we have used in our test set. However, this risk is mitigated somewhat by the large number of sequences used, and by the fact that they come from two separate datasets.

When the robustness of a particular mean-shift tracker is insensitive to the setting of some parameter, we can choose values that reduce the computational complexity of tracking. The use of four bins per colour channel in the target and candidate histograms, for example, requires less processing than the default model size of $16 \times 16 \times 16$ bins (in the case of a three-channel colour space). Similarly, we can set ϵ to a value of 1 pixel in order to reduce the number of iterations executed by mean shift before convergence without significantly affecting the performance of most trackers.

The experiments in this chapter suggest that the best mean-shift-based tracker combines multiple-part models with the background exclusion constraint, uses a two- or three-channel colour space encoding brightness information, and employs very small histograms for the target and candidate models. Very tight bounds on the convergence condition should be avoided, while a cube root kernel should be used with this tracker. However, even with these optimal parameter settings, the performance of mean shift is still some way behind cross-correlation-based tracking; the differences are even greater when the background exclusion constraint is not employed – as would be required for a fair comparison. We must therefore develop additional strategies to improve the robustness of object tracking in the surveillance domain. In the next chapter, we devise techniques for determining when tracking has failed, thereby allowing us to recover from a lost track situation and to follow the target successfully.

Chapter 7

Gradient-based NCC and track validation

The experiments of the previous chapter reveal that normalised cross-correlation (NCC) is significantly more robust than the mean-shift tracker, regardless of the parameter settings used for the latter method. However, NCC, when used with a brute-force search strategy, is computationally expensive. In this chapter we develop a gradient ascent version of NCC (section 7.1); it retains the robustness of its brute-force counterpart, but it executes much more quickly – previously the most notable advantage of mean shift. We also show that the similarity measure at the heart of the tracker – normalised cross-correlation itself – accounts for much of its robustness. Further experiments reveal that a popular, alternative measure used in both template matching and optical flow – *sum of squared differences* (SSD) – does not produce a robust tracker, regardless of whether a brute-force or a gradient-ascent search is used.

The new tracker also forms the basis of our *track validation* algorithm (section 7.2). By tracking a target forwards in time through the sequence, reinitialising the tracker with a new model taken from the end of the video, and following the target backwards in time, we can judge whether or not the tracking was successful: a large divergence between the forwards and backwards trajectories (*failed validation*) indicates that the object was lost by the tracker at some point. In such a situation the algorithm iteratively attempts to validate shorter subsequences of the video. A successful validation allows the method to switch to a new model of the target, which it uses in an effort to validate the object's trajectory in the remainder of the sequence. We apply the algorithm to all of the videos in our test set; the results show that it enables a target to be tracked even as it undergoes severe appearance changes, and it also draws attention to parts of the video and objects that are proving difficult to track.

7.1 Gradient-based normalised cross-correlation

The robustness of normalised cross-correlation (NCC) seen in previous chapters comes at the expense of relatively high computational complexity, especially when compared to mean-shift tracking. We therefore seek to develop a hybrid technique that combines the efficient gradient ascent search strategy of mean shift with the robustness and accuracy of NCC. In the following section, we present the derivation of such a tracker and describe the iterative tracking algorithm that it employs to converge to its target in each frame of the video sequence. Next, we show how the tracker relates to popular types of brute-force template matching, and also to the Lucas–Kanade optical flow method. We run versions of these trackers on our collection of video sequences, measuring their robustness and accuracy. Our method is seen to be significantly more robust than techniques based on the sum-of-squared-differences (SSD) similarity measure. It is also as robust as brute-force normalised cross-correlation, but it has much greater computational efficiency.

7.1.1 Derivation

We begin our derivation by defining certain entities, following the image sequence notation of Hager and Belhumeur [61]. Let $I(\mathbf{x}, t)$ be the image at time t, where $\mathbf{x} \triangleq (x, y)$ is a point in the image. Let $Q \triangleq I(\mathbf{x}, 0)$ be the target (image template) we wish to track, and let $I \triangleq I(\mathbf{x} + \mathbf{u}, t)$ be a candidate image region in the current frame. We can now rewrite the modified NCC similarity measure used in the previous chapter. The similarity $O(\mathbf{u})$ of a template Q and an image patch I displaced from the template by $\mathbf{u}$ is:

$$O(\mathbf{u}) \triangleq \frac{1}{n} \sum_{\mathbf{x} \in R} \left(I - \bar{I}\right) \left(Q - \bar{Q}\right)$$

where n is the number of pixels in the image patch and R is the set of pixel locations in the template. The above formula applies to single-channel images; for multiple-channel data we simply sum the contributions from the individual channels. As before, we do not divide by the standard deviations of the template and image patches. It can be shown that either $\bar{I}$ or $\bar{Q}$ – the means of the image patch and the template, respectively – can be removed from the formula without changing the result (see appendix B). We therefore adopt a simpler version of the similarity measure:

$$O\left(\mathbf{u}\right)=\frac{1}{n}\sum_{\mathbf{x}\in R}I\left(Q-\bar{Q}\right) \tag{7.1}$$

We can approximate I by the low-order terms of its Taylor series around $(\mathbf{x},t)$:

$$I=I\left(\mathbf{x}+\mathbf{u},t\right)\approx I\left(\mathbf{x},t\right)+u_1I_x\left(\mathbf{x},t\right)+u_2I_y\left(\mathbf{x},t\right)+\left(t-t\right)\times I_t\left(\mathbf{x},t\right) \tag{7.2}$$

where $\mathbf{u}\triangleq(u_1,u_2)$. The last term in the above equation contains the factor $(t-t)$. We have included this term, which evaluates to zero, to emphasise that the Taylor series expansion is indeed performed around the point $(\mathbf{x},t)$, even though the final approximation does not make use of any past or future frames. The expressions

$$I_x\left(\mathbf{x},t\right)\triangleq\frac{\partial I\left(\mathbf{x},t\right)}{\partial x};\qquad I_y\left(\mathbf{x},t\right)\triangleq\frac{\partial I\left(\mathbf{x},t\right)}{\partial y};\qquad I_t\left(\mathbf{x},t\right)\triangleq\frac{\partial I\left(\mathbf{x},t\right)}{\partial t}$$

are the spatial and (unneeded) temporal image derivatives, respectively. We use image differences to approximate the required derivatives:

$$I_x\left(\mathbf{x},t\right)\triangleq I\left(\mathbf{x},t\right)-I\Big(\left(x+1,y\right),t\Big)$$

$$I_y\left(\mathbf{x},t\right)\triangleq I\left(\mathbf{x},t\right)-I\Big(\left(x,y+1\right),t\Big)$$

Simplifying equation 7.2, we obtain:

$$I\left(\mathbf{x}+\mathbf{u},t\right)\approx I\left(\mathbf{x},t\right)+u_1I_x\left(\mathbf{x},t\right)+u_2I_y\left(\mathbf{x},t\right)$$

Our goal is to find a local maximum of the similarity measure O as we vary the displacement $\mathbf{u}$; therefore, we differentiate O with respect to $\mathbf{u}=(u_1,u_2)$:

$$\frac{\partial O}{\partial u_1}\approx\frac{1}{n}\sum_{\mathbf{x}\in R}I_x\left(\mathbf{x},t\right)\left(Q-\bar{Q}\right) \tag{7.3}$$

$$\frac{\partial O}{\partial u_2}\approx\frac{1}{n}\sum_{\mathbf{x}\in R}I_y\left(\mathbf{x},t\right)\left(Q-\bar{Q}\right) \tag{7.4}$$

At each iteration we move the tracker to the neighbouring (integer) pixel location "pointed to" by the gradient $\nabla(O)$:

$$\nabla\left(O\right)=\left(\frac{\partial O}{\partial u_1},\frac{\partial O}{\partial u_2}\right)$$

We terminate the iterations as soon as a move results in a decrease in the similarity measure. (The last tracker move, which led to the decrease, is also reversed.)

7.1.2 Comparison with other trackers

The tracker that we derived above can be seen as a gradient-based version of brute-force template matching; both approaches use normalised cross-correlation as the similarity measure. An analogous pair of tracking methods based on the sum-of-squared-differences (SSD) measure also exists: the Lucas–Kanade optical flow method [93] has its brute-force counterpart in SSD-based template matching. The relationship between the four methods is shown in table 7.1.

Similarity measure / **Search strategy**	**SSD**	**NCC**
Brute-force	Template matching	Template matching
Gradient-based	Lucas–Kanade optical flow	Gradient-based NCC

Table 7.1: *Classification of various trackers by similarity measure and search strategy*

The derivation of Lucas–Kanade optical flow begins with the SSD similarity measure P:

$$\begin{aligned} P(\mathbf{u}) &= \frac{1}{n}\sum_{\mathbf{x}\in R}(I-Q)^2 \\ &\approx \frac{1}{n}\sum_{\mathbf{x}\in R}\left[I(\mathbf{x},t)+u_1 I_x(\mathbf{x},t)+u_2 I_y(\mathbf{x},t)-Q\right]^2 \end{aligned} \tag{7.5}$$

The partial derivatives are then obtained:

$$\frac{\partial P}{\partial u_1} \approx \frac{2}{n}\sum_{\mathbf{x}\in R}\Big(I(\mathbf{x},t)+u_1 I_x(\mathbf{x},t)+u_2 I_y(\mathbf{x},t)-Q\Big)I_x(\mathbf{x},t)$$

$$\frac{\partial P}{\partial u_2} \approx \frac{2}{n}\sum_{\mathbf{x}\in R}\Big(I(\mathbf{x},t)+u_1 I_x(\mathbf{x},t)+u_2 I_y(\mathbf{x},t)-Q\Big)I_y(\mathbf{x},t)$$

and set to zero, so that a least-squares approach can be used to solve for $\mathbf{u} = (u_1, u_2)$, the unknown displacement vector. (Lucas–Kanade optical flow forms the basis for many other tracking techniques, including the *KLT tracker* [145] and Hager and Belhumeur's tracking of regions under geometry and illumination changes [61].)

Returning to equation 7.3 and 7.4, it is apparent that we were not able to take a similar least-squares approach in our derivation of the gradient-based NCC tracker: the partial derivatives are constant, and not a function of $\mathbf{u}$ (thus ruling out optimisation techniques such as the Newton–Raphson method [128, pp. 362–368]). It was for this reason that we employed the strategy of moving the tracker iteratively in the direction of the gradient until we found a local maximum.

In order to isolate the impact of a particular similarity measure on tracking performance, we have implemented a gradient-based SSD tracker that is analogous to our NCC technique developed in the previous section. We are therefore not testing the performance of Lucas–Kanade optical flow directly; instead, we are focusing on the SSD similarity measure on which it is based.

7.1.3 Results

We compare the performance of our gradient-based normalised cross-correlation technique against three others: the brute-force NCC and SSD trackers (equations 7.1 and 7.5, respectively) and the gradient-based SSD tracker of the previous section. Once again, we attempt to track designated targets in each of the 21 CAVIAR and PETS video test sequences (section 3.1). For each of the four techniques, operating in the RGB colour space, we record the number of times it lost track of the object it was following (a measure of its *robustness*), and the mean over all the sequences of its median dice coefficient (which reflects its *accuracy*). The results of the tests are presented in tables 7.2 and 7.3.

Similarity measure / Search strategy	SSD	NCC
Brute-force	14	4
Gradient-based	13	3

Table 7.2: *Count of lost tracks for each of the four trackers operating on the 21 video sequences in the dataset*

Similarity measure / Search strategy	SSD	NCC
Brute-force	0.91	0.90
Gradient-based	0.70	0.89

Table 7.3: *Mean of the median dice coefficient for each of the four trackers operating on the 21 video sequences in the dataset*

It is clear that the choice of similarity measure has a large impact on tracking performance. Both of the normalised cross-correlation techniques are significantly more robust than the SSD-based trackers. And although brute-force SSD is the most accurate of the trackers (table 7.3), it is the least robust, failing to track 14 of the 21 targets. (Trucco and Verri point out the close relationship between SSD and cross-correlation, but emphasise that the *normalised* version of the latter is needed to avoid biases caused by very bright or very dark image regions [161, p. 147].)

A breakdown of the occurrence of lost tracks by video sequence is shown in table 7.4. In the case of the two most successful methods (the NCC-based approaches), the lost tracks are *not* concentrated in the PETS sequences (scenarios 15 to 21). However, the brute-force SSD tracker is much less successful on these videos than on the CAVIAR sequences. Once again, we see that the PETS scenarios often present a much greater challenge to less-capable tracking approaches.

The results of tables 7.2 and 7.3 show that our gradient-based NCC tracker is slightly more robust than (and almost as accurate as) the brute-force version. A similar result was observed in section 4.4.1, where the brute-force Bhattacharyya technique was found to be less robust than its gradient-based counterpart (the mean-shift tracker). In both cases, it seems that a *local*, as opposed to a global, maximum of the similarity surface is more likely to correspond to the object's location. Such local maxima are reached by gradient ascent methods, whereas brute-force approaches will find global maxima, which sometimes result from nearby "distractors" – objects having an appearance similar to the target. If a distractor persists for several frames, it can cause a brute-force tracker to fail when a gradient-based method would otherwise succeed.

Figure 7.1 compares the execution time of the basic mean-shift technique and the gradient-based and brute-force normalised cross-correlation trackers on each of the 21 video sequences in the dataset. Our gradient-based NCC tracker is, on average, 4.8 times faster than the brute-force version, while being slightly more robust.[1] (The brute-force trackers must evaluate the similarity measure on a grid of 11×11 image locations.) The speed difference is more pronounced for the (lower-resolution) CAVIAR videos, where the gradient-based tracker generally requires fewer iterations to reach a local maximum. It is also seen that the basic mean-shift approach, which is much less robust than the other methods, is only 20% faster, on average, than our technique.

[1]The experiments were performed in MATLAB on a computer with a 3.2GHz Pentium 4 CPU and 3GB of RAM.

Tracker / **Scenario**	**Brute-force SSD**	**Gradient-based SSD**	**Brute-force NCC**	**Gradient-based NCC**
1	x		x	
2	x			
3	x	x		
4		x	x	
5				
6	x	x		
7				
8				x
9	x	x		x
10	x	x		
11	x	x		
12		x		
13	x	x	x	x
14		x		
15	x	x	x	
16	x	x		
17	x	x		
18	x	x		
19				
20	x			
21	x			
Num. trackers lost	14	13	4	3

Table 7.4: *The success/failure of the various SSD- and NCC-based trackers operating on each scenario. (Failures are indicated by an 'x'.)*

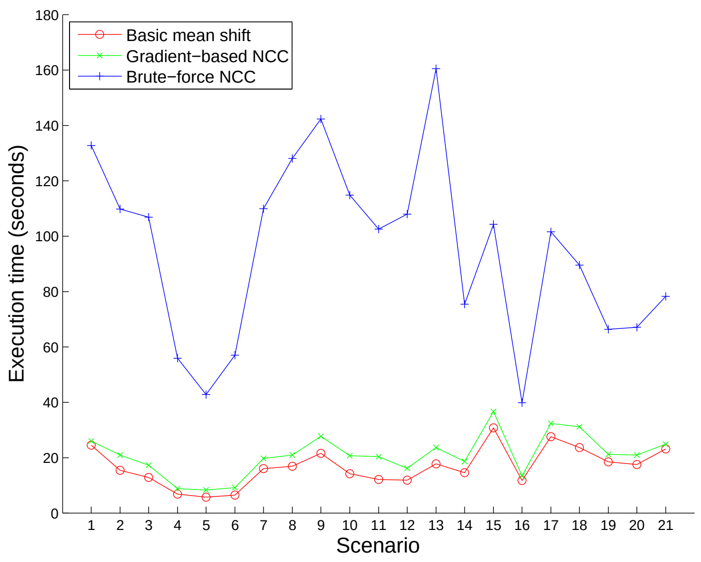


Figure 7.1: *Execution time of the basic mean-shift, gradient-based NCC and brute-force NCC trackers on each of the 21 videos in the dataset*

7.2 Track validation

The results of section 7.1 show that trackers based on normalised cross-correlation, whether using a brute-force or a gradient ascent search, can track most of the targets in our dataset of CAVIAR and PETS sequences. However, we only know that a target has been successfully followed by comparing the locations returned by the tracker to the ground-truth data for that target. Such data will not be available in a real-world system; while we may have certain expectations about the likelihood of the tracker successfully following a given target (based on how it has performed on our video test set), without a validation mechanism we cannot automatically distinguish between tracking successes and failures. This is the motivation behind the present section: it is apparent that tracking will sometimes fail; however, a mechanism that can detect such failures, which we refer to as *track validation*, would prove of great use. It would alert the system that the tracking of a given target did not succeed, which would in turn permit other tracking attempts to be made. For example, the system might use a different model of the target as the basis for further tracking, or it might alter the parameters associated with the tracking technique.

7.2.1 Overview and related work

Our implementation of track validation is based on a simple principle: tracking an object forwards in time through a sequence, stopping, reversing the sequence and tracking the object backwards in time should yield two very similar trajectories, provided the object has been tracked correctly. However, if the two trajectories do not match very closely, it is likely that either the forwards or the backwards tracking failed at some point (or at least became very inaccurate). This idea forms the basis of our track validation algorithm, presented in detail in the next section. Firstly, though, we review some related approaches.

Given the starting and ending locations (and models) for an object to be tracked, Sun *et al.* [156] determine the full trajectory of the object between the two points in spite of occlusions. They first extract short track segments that they hypothesise correspond to the object. A segment is only accepted as being part of the final trajectory if it can be joined with others to form a smooth path in space-time. Wu *et al.* [171] develop the "time-reversibility constraint" and apply it to the KLT tracker [145]. A tracked image point is retained by the method only if it moves in a similar direction when it is tracked forwards and backwards in time. The constraint is only applied to pairs of frames, and not to the entire video sequence.

The approach most closely related to our work – and developed at approximately the same time – is the "recurrent tracking" of Pan *et al.* [116] An object is first tracked forwards in time through the sequence. Tracking is then reinitialised with a new model taken from tracker's final location, and the object is followed backwards in time through the sequence. A significant difference between the forwards and backwards trajectories signifies that tracking has failed at some point.

7.2.2 Algorithm

We have implemented a track validation algorithm that is also based on forwards–backwards trajectory mismatches. However, it differs from the work of Pan *et al.* in the actions it takes when a tracking failure is detected: whereas Pan's approach is to shorten the video sequence one frame at a time until the forwards and backwards trajectories are sufficiently similar, we reduce the length of the sequence by *half* at each iteration. And in contrast to Pan, we then attempt to track the object through the remainder of the video, using a new model taken from the end of the validated portion. At all times our goal is to establish a trajectory for the object through the *entire* sequence, not merely an acceptably long part of it. Furthermore, we use a single type of tracker throughout the algorithm. Pan, on the other hand, switches from simpler mean-shift methods to more powerful particle filter trackers whenever the former have been unable to validate a sufficiently long trajectory for the target.

A pseudocode version of our approach is given in algorithm 1. The inner loop performs the forwards–backwards validation on a section of the sequence. The outer loop attempts to find a collection of these subsequences that, when concatenated, yield a trajectory for the object in the entire video clip. Two thresholds are required by the algorithm: `min_traj_length` is the minimum number of frames which each of the subsequences must contain, and `max_traj_diff` is the the maximum allowable average per-frame distance, in pixels, between forwards and backwards trajectories that are to be considered as matching (and hence validated). In our experiments we have set the thresholds to 25 frames and 5 pixels, respectively.

```
// We have validated tracking up to $pos1$ in the sequence
$pos1 = start$;
while $pos1 \neq end$ do
    $pos2 = end$ ; // Attempt to track up to $pos2$ in the sequence
    repeat
        track forwards from $pos1$ to $pos2$;
        reinitialise tracker at $pos2$;
        track backwards from $pos2$ to $pos1$;
        $diff$ = average per-frame distance
                between forwards and backwards trajectories;
        $pos2\_old = pos2$;
        $pos2 = (pos1 + pos2)/2$;
    until $(diff > max\_traj\_diff)$ and $(pos2\_old - pos2 \geq min\_traj\_length)$;
    if $diff > max\_traj\_diff$ then
        $print$('Unable to validate any further than ' + $pos1$);
        $break$;
    else
        $print$('Validated up to ' + $pos2\_old$);
        $pos1 = pos2\_old$;
    end
end
```

Algorithm 1: *Track validation and extension*

7.2.3 Results

We have compared the effectiveness of track validation when using two different underlying techniques: basic mean-shift tracking (chapter 4) and gradient-based normalised cross correlation (section 7.1). The proportion of each of the 21 video sequences in the test set that is successfully validated by our algorithm is shown in figure 7.2, for both of the tracker types.

It is clear from the results that gradient-based NCC significantly outperforms the mean-shift method when used for track validation. The former technique is able to validate three-quarters or more of the track's length in 17 of the 21 sequences in the dataset, whereas mean shift only reaches this level of validation on nine occasions. Indeed, with mean shift there are a further nine videos where none of the tracker's trajectory can be validated at all – something which never happens with gradient-based NCC. Figures 7.3 and 7.4 show examples of successful and failed track validation, respectively. These images only represent one iteration of the inner loop of algorithm 1: it is possible that, by halving the length of the sequence, a successful validation can be achieved, at which point the algorithm will attempt to validate the remainder of the sequence with a new model.

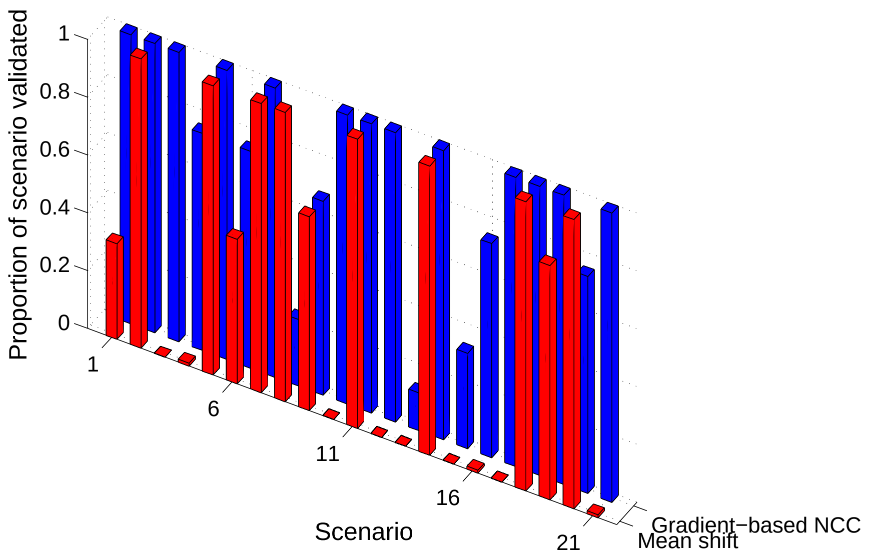


Figure 7.2: *Proportion of each video sequence validated by our algorithm for both mean-shift and gradient-based NCC trackers.*

We attribute the disparity in track validation performance between the two methods to the underlying difference in their robustness and accuracy. At any given frame, the gradient-based NCC tracker is more likely than mean shift to be positioned accurately on its target. This is a necessary condition for track validation to occur: at the moment of reinitialisation, the tracker's location must correspond well to the location of its target. Otherwise, the tracker will be reinitialised with a model that does not accurately represent the appearance of the object it is supposed to follow – for example, the new model may show half of the person being tracked, with the other half being occupied by the background of the scene. Table 7.5 shows, for each video sequence, the successive models used by the track validation algorithm as it attempts to construct the longest-possible trajectory for the target. (Gradient-based NCC was used as the tracking method.) Naturally, there are different numbers of models shown for each sequence; this is a consequence of the varying difficulty of the videos: some require more reinitialisations than others.

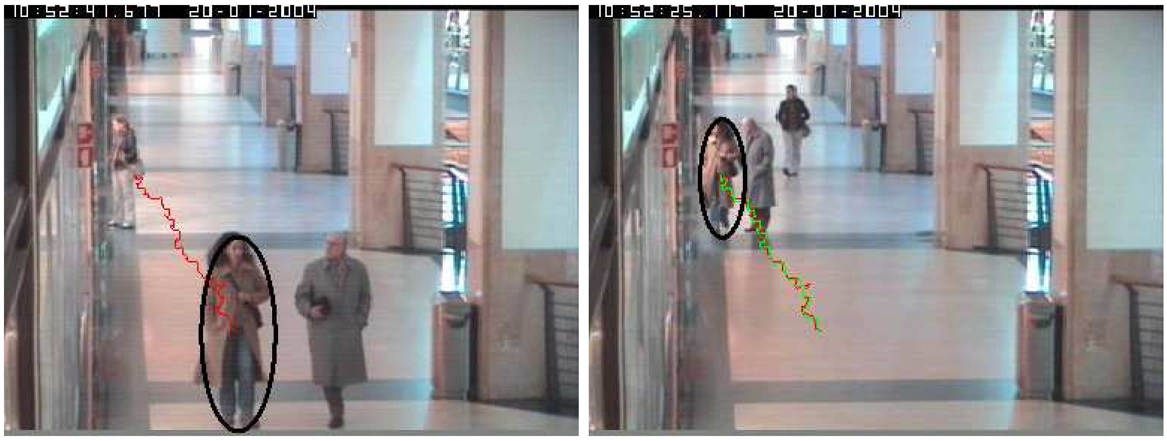


Figure 7.3: *An instance of successful track validation (using the gradient-based NCC tracker). The forwards trajectory (red, left) and the backwards trajectory (green, right) for scenario 2 match very closely.*

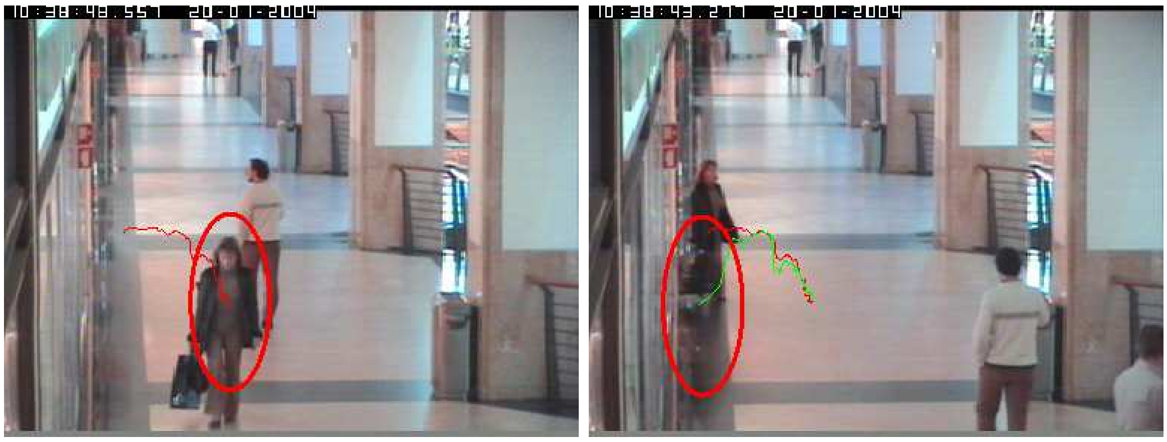


Figure 7.4: *An instance where tracking is not validated (using the mean-shift tracker). The forwards trajectory (red, left) and the backwards trajectory (green, right) for scenario 5 are very different.*

Table 7.5: *Various models found and used by the track validation algorithm for each of the 21 scenarios. The gradient-based NCC technique was employed to perform the tracking.*

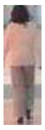
1					
2					
3					
4					
5					
6					
7					
8					

Table 7.5: *Various models found and used by the track validation algorithm for each of the 21 scenarios (continued)*

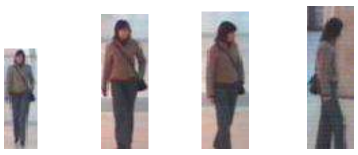
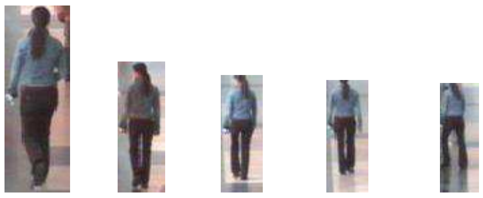
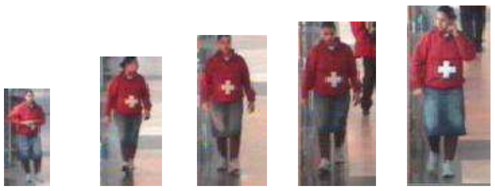
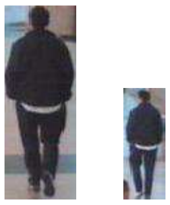
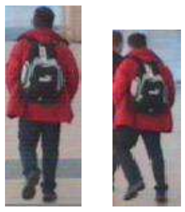
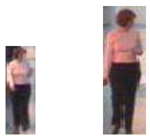
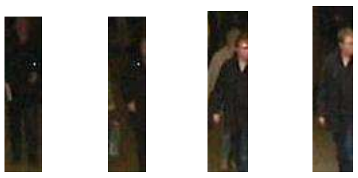
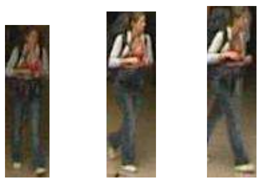
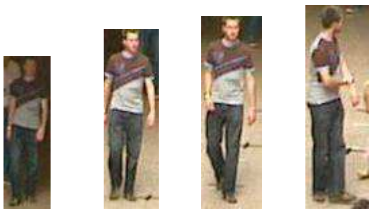
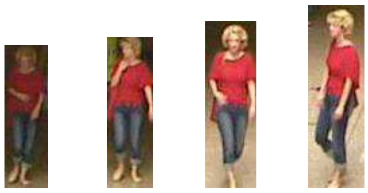
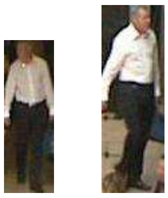
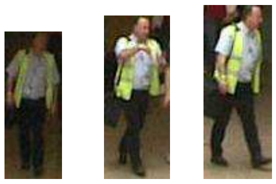
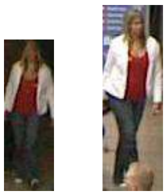
9	
10	
11	
12	
13	
14	
15	
16	

Table 7.5: *Various models found and used by the track validation algorithm for each of the 21 scenarios (continued)*

17	
18	
19	
20	
21	

Looking at the models in table 7.5 used by the track validation algorithm, it is apparent that some of them do not accurately represent the target: the tracker was badly placed at the moment of reinitialisation, but the forwards and backwards trajectories matched sufficiently well for track validation to occur. Scanario 5 demonstrates this effect, sometimes called the "template update problem": each successive model drifts further away from its target [99]. It can be seen that once such an inaccuracy has arisen, it is likely to remain present in the subsequent models. In one instance (scenario 3), the inaccuracy is compounded to the point where a portion of the track is validated using an entirely incorrect model (in this case, a model of the background).

It should be borne in mind that the inaccurate reinitialisations and false validations of the kinds described above occur only rarely in our video test set. The track validation algorithm provides significant benefits that outweigh these weaknesses. Firstly, it allows us to track objects whose appearance changes drastically over time: table 7.5 contains many examples of lighting changes, articulation and out-of-plane rotation, which make tracking a person through an entire sequence using only a single model very challenging. The approach can therefore be regarded as a form of "unsupervised model building" [99], where the model adapts to changes in the target's appearance without manual intervention. Secondly, even when our algorithm is unable to track an object for the full length of a video clip, it is providing valuable information to the higher-level processes that invoked it. Specifically, a failure of track validation indicates that tracking has become very difficult in that particular section of the video, and that other techniques and strategies should be considered. We note finally that there are ways of reducing the chance of a false validation occurring. As it stands, our algorithm only uses a certain selection of frames (shortening the sequence by half, stopping when the subsequence contains less than 25 frames) in its attempt to find forwards and backwards trajectories that match. A more exhaustive search through the sequence, or a comparison of the outcomes of runs that used different values for the thresholds, has the potential to reduce the number of false validations further.

7.3 Discussion

The present chapter has addressed two practical aspects of object tracking: execution speed and the trustworthiness of the tracker's output. Chapter 6 revealed that mean-shift tracking, although computationally inexpensive, is significantly less robust than normalised cross-correlation (NCC), seemingly due to the lack of discriminative power inherent to image histograms. We did not find any combination of mean-shift parameters that brought its performance up to the level of NCC. The latter technique, however, is much slower than mean shift. We have therefore developed a gradient-based version of NCC that is as robust as the standard, brute-force version but requires less computation and consequently runs much more quickly. Our experiments also show that the similarity measure at the heart of the tracking technique – normalised cross-correlation itself – has significant advantages over the popular sum-of-squared-differences measure, which is commonly used for template matching, in addition to forming the basis of the Lucas–Kanade optical flow technique.

The second half of this chapter is concerned with validating the object trajectories returned by a given tracking method, ensuring that they truly represent the object's path through the scene. We developed a technique that compares the trajectory obtained by tracking a target forwards in time to that obtained by reinitialising the tracker at the end of the sequence and attempting to follow the target backwards in time through the sequence. If the forwards and backwards trajectories are not sufficiently similar, the trajectories are not validated; it is assumed that in such a case the tracker has failed to track the object. Our algorithm builds on this basic principle, yielding a scheme that iteratively shortens the sequence until track validation is achieved. This allows us to switch to a new model of the target, which we use to attempt to track the object through the remainder of the video sequence. In effect, the algorithm serves as a conservative filter on the output of any tracking method: it is not always able to determine the entire trajectory of an object moving through the video sequence, but it almost never validates the paths returned by a tracker that has failed to follow its target correctly. The choice of tracking method used by our algorithm has an impact on its chances of validating an object's trajectory. Employing our gradient-based normalised cross-correlation tracker, owing to its better robustness and accuracy, leads to a greater number of validated trajectories than using the mean-shift method does.

Taken together, the two contributions of this chapter – a fast and robust normalised cross-correlation tracker and a technique for validating its output – can simplify the higher-level processing that would be performed in a complete tracking system. The track validation technique not only makes the output of the tracker more trustworthy, it also provides indications (whenever validation is not possible) of which objects and sections of a sequence are especially difficult to track. The system may respond to these signals by invoking more advanced techniques, possibly at the expense of increased computation, to track the challenging targets.

Chapter 8

Conclusions

This chapter summarises the research presented in the thesis and highlights the contributions that we have made in the area of gradient-based tracking. The results of our investigations into mean shift are also discussed; we describe how best to use the technique in the surveillance domain, with a particular focus on parameter settings. The performance of mean shift is placed in the context of other methods – we have found that it is inherently less reliable than older and simpler techniques, at least in the area of pedestrian tracking. We emphasise the importance of thorough evaluations on large video test sets, and the use of both ground truth data and automated methods; these elements allowed us to draw definitive conclusions about the performance of mean shift. We suggest that similar approaches should be used whenever investigations into tracking methods are being undertaken. Finally, a number of potential avenues of research relating to data-driven tracking are outlined.

8.1 Summary

In this thesis we have focused on the mean-shift tracking technique, applying it in the domain of surveillance. Below, we provide a summary of the research that we have conducted.

In chapter 4 we performed an assessment of the basic, unmodified mean-shift tracker. The method was used to track designated targets through each of the 21 videos in our test set of surveillance sequences. Since ground truth data was available specifying the location of the person to be tracked in every frame of the sequences, we were able to undertake a quantitative evaluation of mean shift. We recorded firstly the number of videos in which the technique lost track of its target (its *robustness*), and secondly the degree to which the position of the tracker matched that given by the ground truth data (its *accuracy*). For comparison, we used the same approach to evaluate three other data-driven trackers – based on the Bhattacharyya coefficient, the earth mover's distance and normalised cross-correlation (NCC), respectively. The last of the three proved significantly more robust than any of the others, including mean shift. Indeed, none of the modifications that we made to the mean-shift technique in subsequent chapters were able to make its performance match that of NCC. The inherent differences in how the two methods represent the image region to be tracked – mean shift by a histogram, NCC by a template – appear to be responsible for the large disparity in robustness; image templates carry more information and are simply more discriminative than histograms.

Chapter 5 presented our derivation of a modified mean-shift-based tracker. We fused two existing techniques from the literature – multiple-part models and background exclusion – into a single tracker that was still based on mean shift. A thorough assessment of the new method was undertaken, and it was found to outperform trackers that used only multiple-part models or background exclusion in isolation. However, its robustness remained somewhat lower than that of the NCC tracker.

A comprehensive evaluation of mean shift, both the original tracker and our modified version, formed the basis of the research in chapter 6. We studied the effect of changing the values of the various parameters that are associated with the method. The results demonstrate that care must be taken in setting the parameter values, and especially in deciding the particular multiple-part structure that the tracker should have. A choice that is not based on extensive testing can give rise to tracking performance that is far below that which the method is capable of achieving. (A more in-depth discussion of parameter settings can be found in section 8.3.1 below.)

In chapter 7 we developed a gradient-based tracker that uses templates as models of the image regions to be tracked, replacing the histograms employed by mean shift. It is essentially a hill climbing version of the NCC template matching tracker, whose performance has consistently beaten that of the mean-shift method. The gradient ascent nature of the new tracker makes it significantly faster than NCC, although still somewhat slower than mean shift. However, the robustness of the new technique compared to mean-shift tracking makes it a superior choice in almost every respect.

Chapter 7 also presented our *track validation* algorithm for assessing the trustworthiness of a given tracker's output. We follow a target forwards in time through the video sequence, reinitialise the tracker at the end, and run the sequence backwards while continuing to track the target. If the paths from the forwards and backwards tracking do not match sufficiently closely, it is likely that the tracker failed at some point. In such a case, we shorten the sequence and repeat the process until validation is achieved, i.e., until the forwards and backwards paths match. When part of a sequence is validated in this fashion, we use the new object model (taken from the point of reinitialisation) for further tracking. The approach allows us to update the model in a safe manner, even as the target undergoes significant appearance changes. The algorithm operates conservatively: although it is not always able to validate an entire sequence, it is very rare for incorrect tracking to go undetected. In this way, track validation serves as an effective, general-purpose filter on the output of any tracking method.

8.2 Contributions

Below, we highlight the elements of the thesis which represent contributions to the field of object tracking:

- A comprehensive, quantitative assessment of the performance of mean shift on a large video dataset. The technique is compared to other histogram-based trackers and to normalised cross-correlation.
- The development of a mean-shift method that unifies two elements – multiple-part models and the background exclusion constraint – in a single tracker. The new method is more robust than using either element on its own.
- An evaluation of the effect of several parameters on mean shift. We record the performance of the method as we vary their values and alter the multiple-part structure of the tracker. The results of the experiments reveal the optimal parameter settings for employing the mean-shift technique in the surveillance domain.
- The creation of a gradient-based normalised cross-correlation tracker. It combines computational efficiency (arguably the greatest advantage of mean shift) with the robustness of template matching.
- The development of a *track validation* algorithm for verifying that a tracker's output is trustworthy. As each portion of the video sequence is validated, we also update the object model, which allows the tracker to accommodate large changes in the appearance of the target.

8.3 Discussion

Our objectives in this thesis were to investigate the performance of mean-shift tracking, in both its basic and modified forms, and to determine how best to use the method in the domain of pedestrian tracking. Below, we discuss the specific elements and parameter settings that are needed to obtain optimal performance from the technique. Afterwards, we give some perspectives on mean shift in the context of similar data-driven tracking methods, and also on how tracking techniques in general should be assessed.

8.3.1 Improving the performance of mean-shift tracking

The evaluations performed in chapter 5, where our tracker combining multiple-part models and background exclusion was derived, suggested that both of those elements are required to achieve good performance with mean shift. However, it was not until chapter 6 that we were able to establish this result with greater confidence, and to determine optimal settings for the other parameters of the tracker. Our experiments show that a tracker which combines multiple-part models and background exclusion will usually outperform all other mean-shift variants, regardless of the choices we make for the colour space, the size of the histograms, the threshold on the convergence condition or the type of spatial kernel used. With regard to the precise multiple-part structure of the tracker, the performance seems to improve as the number of subregions (parts) in the model is increased, but only up to a point: robustness appears to reduce once we use 25 subregions or more. The results also strongly indicate that we should avoid multiple-part models if we do not also make use of the background exclusion constraint; such multiple-part-only trackers consistently demonstrate poor performance across all parameter settings.

If we settle on the combined tracker (since it is generally the most robust), we can choose values for the remaining parameters one at a time, with the objective of improving the performance further with each choice. The first of these parameters – the colour space – has a surprisingly small impact on mean-shift tracking in our experiments with surveillance videos. The dimensionality of the space, i.e., the number of channels it contains, may be the most important consideration. Of the colour spaces that we tested, those with two or three channels generally performed similarly. Only single-channel spaces seem to reduce robustness to any noticeable degree.

The ideal size for the target and candidate histograms strongly depends on the type of tracker we are considering. For the combined multiple-part–background exclusion tracker, a small number of bins – two or four per colour channel – gives the best performance. However, for the other mean-shift variants, such a choice will *reduce* the tracker's robustness. In the absence of a universal trend in performance as the number of bins is varied, we suggest that further experiments on the parameter's value be undertaken in the specific domain in which mean shift is intended to be used.

The threshold on the value of the convergence parameter, ϵ, should not be set to a very low value (0.05 or 0.10) when used with the combined tracker. A value of 1.0, which was used in the original mean-shift paper, is associated with the highest robustness. For the final parameter – the type of spatial kernel used by the tracker – the cube root profile (the most "peaked" of all of the kernels tested) gives the best results.

8.3.2 Perspectives on mean shift

Our extensive evaluations of mean shift have revealed the effect of the different parameters on its performance, and have allowed us to "tune" the method so that it becomes more robust. It is possible, in this manner, to make the tracker significantly more reliable than it would otherwise be if we used default settings. However, none of our suggested modifications or parameter values have made the method as robust as standard template matching. The mean-shift technique, as it exists today, is simply not as capable as normalised cross-correlation for pedestrian tracking, whether we use it in its basic form or augment it with multiple-part models and background exclusion. Indeed, for a fair comparison between mean shift and NCC to be made, we should not use background exclusion at all. Doing so reveals the gulf in performance between the methods. And with the development of our gradient-based NCC tracker (chapter 7) the primary advantage of mean shift – computational efficiency – is almost eliminated.

Mean shift should be regarded simply as an optimisation technique: it moves to the mode of a distribution in a small number of steps. Segmentation was the first widespread application of the technique in image processing [29]. Unfortunately, to make mean shift work in a tracking context it was necessary to represent image regions by their histograms, and to compare them using the Bhattacharyya coefficient. The resulting lack of discriminative power (versus comparing image templates using normalised cross-correlation) is at the heart of the method's lack of robustness. The histograms and/or the Bhattacharyya coefficient must be replaced with elements that are more discriminative if the reliability of mean-shift tracking is to be improved significantly.

8.3.3 Performance assessment and validation of results

We have been able to draw these stark conclusions about the different varieties of mean shift and NCC only by undertaking extensive, systematic and quantitative evaluations. Assessments of this kind for tracking methods require certain elements: a dataset of video sequences on which all of the techniques can reasonably be expected to operate; a collection of ground truth data; and the definition of metrics at the frame, sequence and dataset level. Together, these components allow the evaluations to be performed in an automated and repeatable fashion. While recognising that subjectivity can never be entirely eliminated from the process (e.g., in the definition of the ground truth data or in the selection of metrics), the quantitative nature of the experiments allows to have more faith in the results that are produced.

The research into *track validation* in chapter 7 has shown that providing ground truth data may not even be necessary for performing automated evaluations of tracking methods. The algorithm allows us to detect almost all occurrences of tracking failure; only a correct initialisation on each video sequence, possibly performed manually, is required. However they are accomplished, comprehensive, quantitative and comparative evaluations of tracking techniques are required if we are to accelerate progress towards more robust methods.

8.4 Future directions

It is apparent from the research in this thesis, and from the extensive literature on the subject, that tracking is far from a solved problem. There are many situations in video sequences that can cause a tracker to fail, whether it be the brightness and appearance changes that an object undergoes, the presence of "distractors" that draw the tracker away to an incorrect target, or the occurrence of partial or complete occlusions. The difficulties can be tackled from two sides: improvements in top-down control strategies can provide greater context to the tracker, allowing it to operate in a more sensible fashion; and the development of bottom-up (data-driven) tracking techniques that are better able to exploit the underlying image structure enables a greater proportion of targets to be followed. As we have focused in this thesis on data-driven methods, we will suggest some future avenues of research in that area.

Occlusions are a common occurrence in real-world video sequences, but few bottom-up methods attempt to tackle them directly, preferring to leave the task to the higher-level control strategies of the tracking system. However, our track validation algorithm presents an opportunity to overcome partial occlusions without relying on external context. The algorithm is conventionally used to verify the longest-possible portion of the object's path through the sequence. It will likely only be able to validate the trajectory up to the point where the target is obscured, but it is useful to know simply that an occlusion has occurred, even if we are unable to continue with the tracking. In certain circumstances, however, it may be possible to extend the trajectory through the obscuration. For example, we could hypothesise that the object is only partially occluded, and so use a corresponding partial model to track the portion of the target that remains visible. Track validation could be applied to the occlusion event in the same way as it is used in the case of normal tracking, i.e., to verify that the trajectory returned from around the time of the occlusion is trustworthy.

In section 8.3.2 we suggested that data-driven trackers built on histograms alone would struggle to operate robustly. However, template-based trackers have their own weaknesses: they can fail when the shape of the target changes (as is caused by the articulation of a person's limbs), and they are overly forgiving of brightness differences between adjacent candidate image regions (at least in the case of normalised cross-correlation). In recent years, a class of methods has been developed that seeks to forge a link between histogram-based and template-based models [44, 174]. And the work of Leung and Gong [83] shows that only a handful of pixels from an image region are required for these hybrid models to operate in a robust fashion, allowing them to be computationally efficient. We consider that the reliability could be increased further by combining the brightness normalisation elements of NCC with such a model in order to develop a tracker that is discriminative but not overly sensitive to changes in the shape of the target.

Much other interesting research into data-driven tracking has been undertaken recently. Hager *et al.* [62] have recast the equations that underlie mean shift in matrix form, and have shown that histogram-based tracking can be accomplished by a Newton-style optimisation procedure. It converges more quickly than mean shift, and it allows multiple kernels to be employed in a single tracker [48]. We believe that there are connections between such multiple-kernel methods and the multiple-part models that we have used with mean shift. A quantitative evaluation of the new techniques is needed to establish their robustness, and to determine which type of kernels are most useful in a given application domain. Some of the most recent work in this direction has shown how to determine the difficulty of tracking a particular image region [49]. Used together with our track validation approach, it would further increase our ability to discard incorrect results, yielding a data-driven tracker with very high reliability.

Appendix A

Tracker parameters: further results

In this appendix we present supplemental results for the experiments on mean-shift parameters performed in chapter 6. The following tables and graphs record the results of varying the colour space, histogram size, convergence condition and the type of spatial kernel used with the mean-shift method.

A.1 Colour spaces

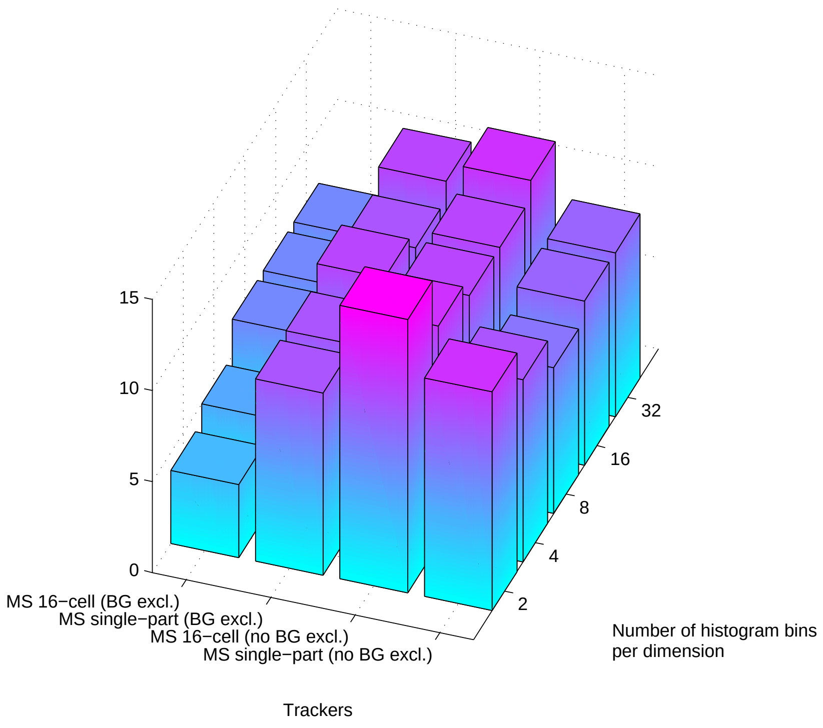
		Mean shift without BG excl.				**Mean shift with BG excl.**			
	Modified cross-corr.	**Single-part**	**Multiple-part (quadr.)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**	**Single-part**	**Multiple-part (quadr.)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**
HSV	3	10	9	11	12	8	7	5	6
RGB	4	8	8	10	11	8	7	8	7
YUV	4	10	7	11	12	8	6	8	6
SV	3	10	9	11	11	10	8	8	9
HV	5	12	7	9	12	9	7	8	8
RG	4	9	7	8	11	10	7	8	7
RB	4	9	8	10	14	10	7	8	8
GB	5	11	8	10	14	9	9	9	8
YV	4	10	7	10	11	10	7	7	6
YU	5	11	10	10	14	10	9	7	7
HS	15	8	12	13	14	8	5	5	6
UV	11	11	12	16	19	8	7	8	8
rg	12	9	12	14	18	8	7	6	8
rb	13	12	14	13	17	11	10	10	9
gb	11	7	11	14	18	9	7	9	7
R	6	11	9	11	14	10	10	8	6
G	5	10	9	11	14	9	8	9	8
B	6	11	12	12	16	10	10	8	10
Y	5	10	10	11	13	9	10	8	8
H	16	13	15	16	18	11	8	10	8
S	14	14	13	16	18	11	10	10	11

Table A.1: *Count of lost tracks on 21 video sequences for various colour spaces and trackers*

			Mean shift without BG excl.					Mean shift with BG excl.			
	Modified cross-correlation		**Single-part**	**Multiple-part (quadrants)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**		**Single-part**	**Multiple-part (quadrants)**	**Multiple-part (9 cells)**	**Multiple-part (16 cells)**
HSV	0.89		0.71	0.80	0.79	0.76		0.85	0.88	0.88	0.88
RGB	0.90		0.70	0.80	0.82	0.80		0.85	0.88	0.89	0.88
YUV	0.90		0.71	0.79	0.81	0.76		0.79	0.84	0.84	0.84
SV	0.91		0.67	0.81	0.83	0.79		0.82	0.88	0.89	0.88
HV	0.89		0.74	0.82	0.78	0.76		0.80	0.85	0.84	0.84
RG	0.90		0.69	0.81	0.81	0.79		0.78	0.83	0.84	0.83
RB	0.90		0.72	0.80	0.81	0.76		0.82	0.88	0.88	0.88
GB	0.87		0.67	0.73	0.79	0.76		0.79	0.84	0.86	0.84
YV	0.90		0.70	0.82	0.82	0.78		0.74	0.83	0.84	0.83
YU	0.90		0.72	0.75	0.78	0.73		0.74	0.80	0.82	0.83
HS	0.83		0.71	0.77	0.80	0.74		0.85	0.88	0.88	0.88
UV	0.76		0.64	0.69	0.68	0.62		0.75	0.83	0.86	0.84
rg	0.88		0.68	0.76	0.76	0.72		0.82	0.85	0.86	0.85
rb	0.85		0.64	0.70	0.70	0.69		0.76	0.79	0.81	0.82
gb	0.88		0.68	0.75	0.75	0.72		0.80	0.84	0.87	0.85
R	0.90		0.71	0.77	0.83	0.79		0.73	0.78	0.80	0.81
G	0.89		0.66	0.74	0.80	0.78		0.72	0.80	0.82	0.82
B	0.84		0.65	0.71	0.75	0.75		0.76	0.82	0.83	0.83
Y	0.90		0.72	0.72	0.80	0.80		0.73	0.79	0.82	0.83
H	0.78		0.65	0.73	0.67	0.68		0.78	0.85	0.85	0.85
S	0.82		0.70	0.76	0.73	0.71		0.75	0.82	0.85	0.85

Table A.2: *Mean of the median dice coefficient of 21 video sequences for various colour spaces and trackers*

A.2 Histogram size



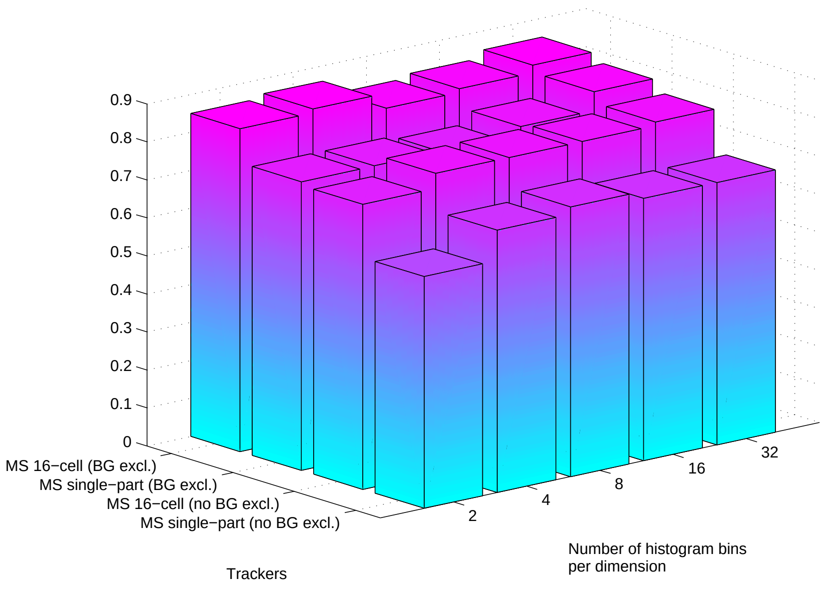
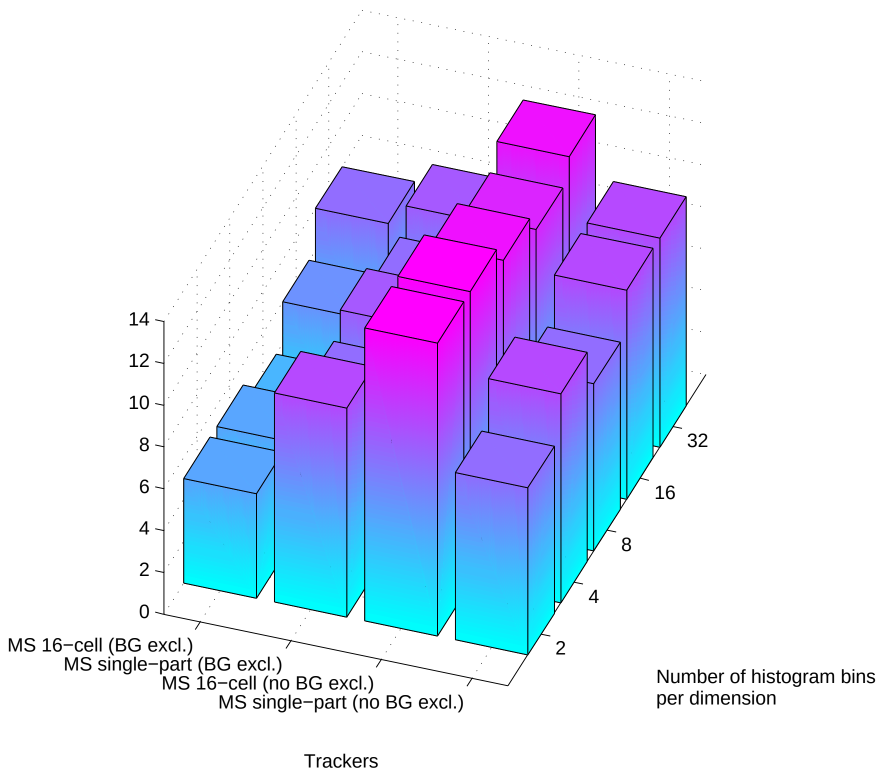


Figure A.1: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various histogram sizes and trackers operating in RG*



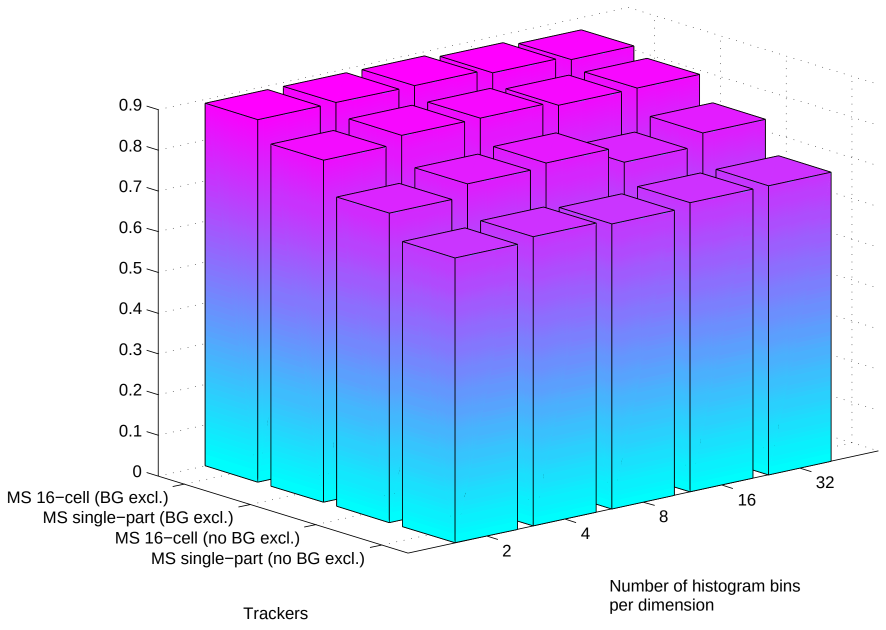
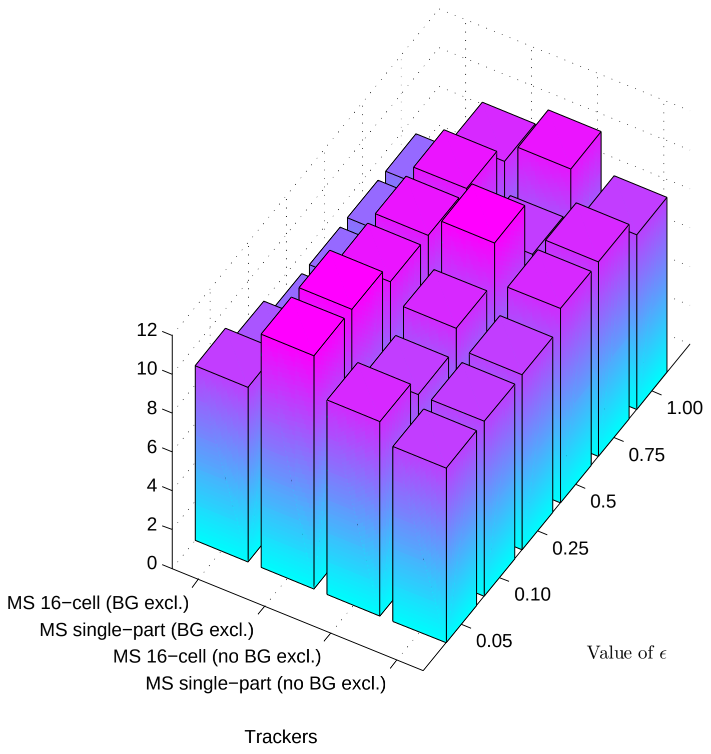


Figure A.2: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various histogram sizes and trackers operating in HSV*

A.3 Convergence condition



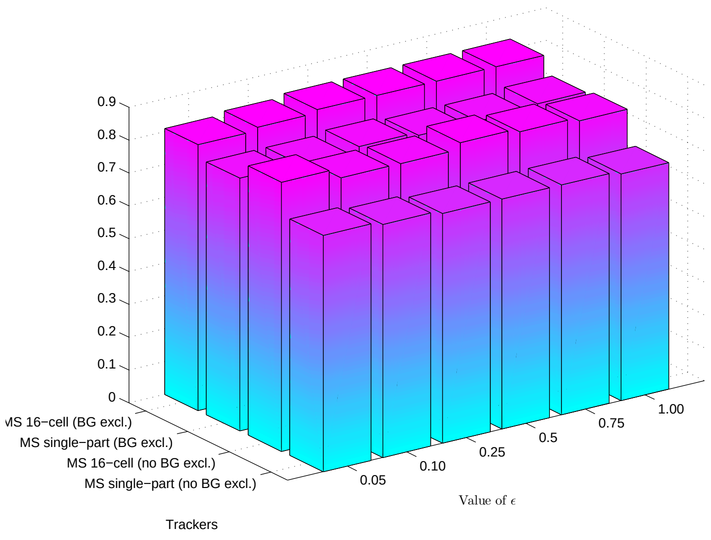
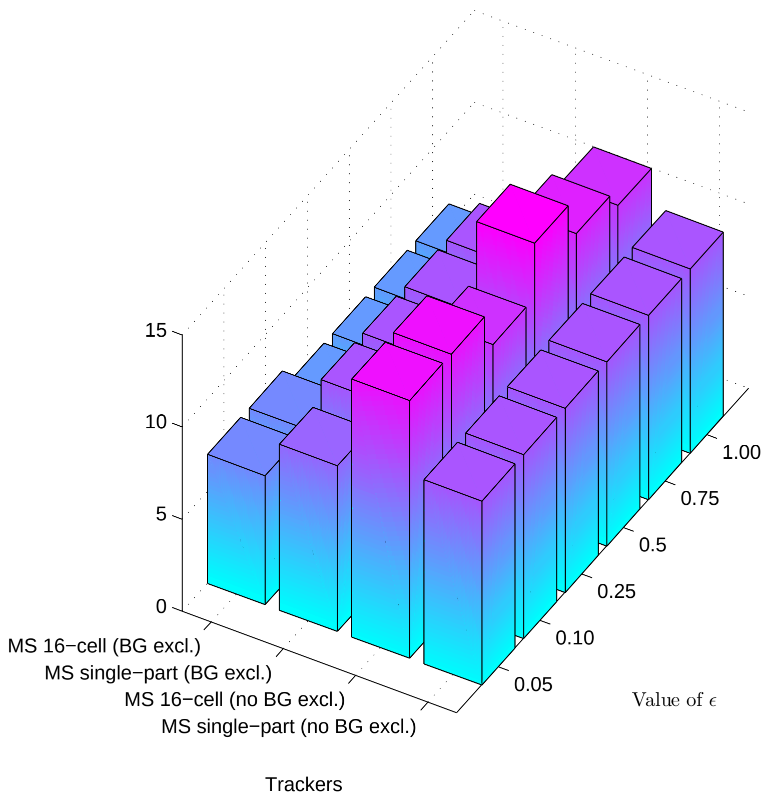


Figure A.3: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various values of ϵ and trackers operating in RG*



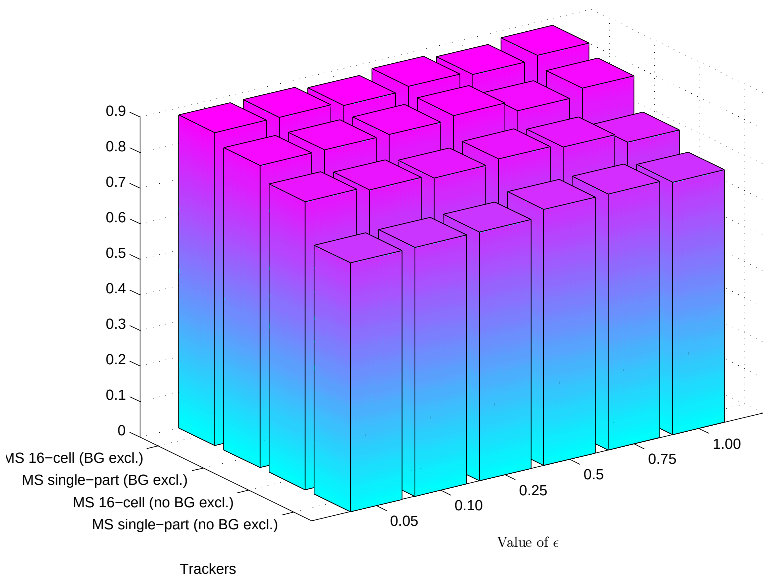
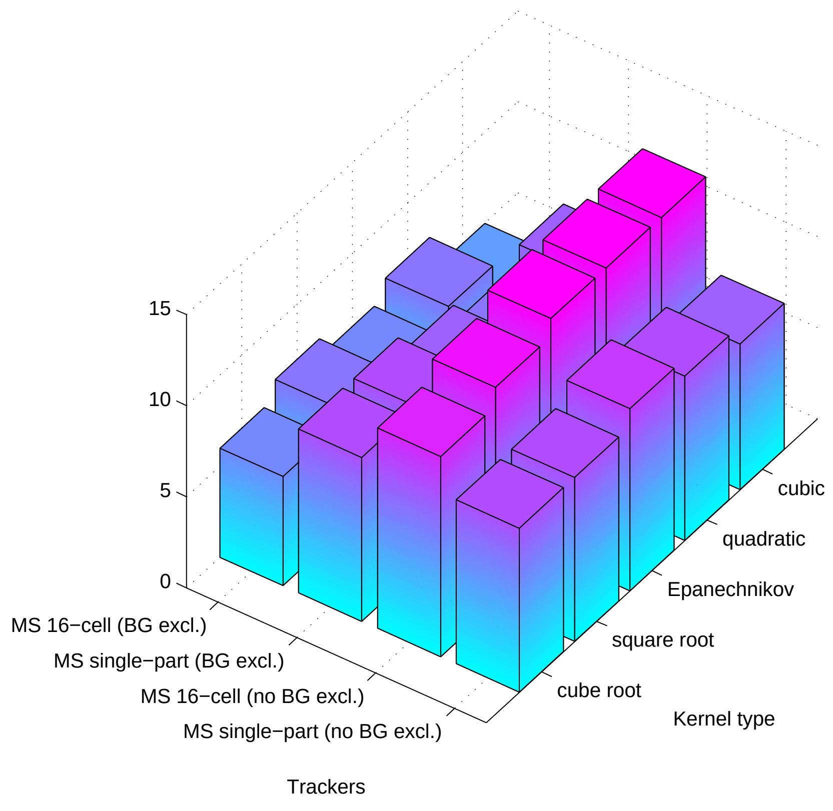


Figure A.4: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various values of ϵ and trackers operating in HSV*

A.4 Kernel type



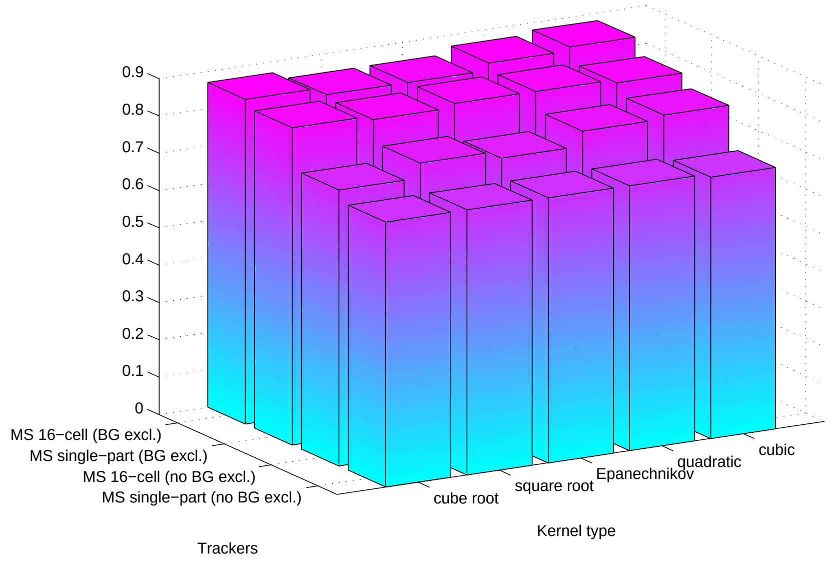


Figure A.5: *Number of lost tracks (top) and mean of the median dice coefficient (bottom) on 21 video sequences for various kernel types and trackers operating in HSV*

Appendix B

Simplifying normalised cross-correlation

In section 7.1.1 we used a modified version of the normalised cross-correlation (NCC) similarity measure as the basis for our gradient ascent NCC tracker. The similarity $O(\mathbf{u})$ of a template Q and an image patch I displaced from the template by $\mathbf{u}$ is:

$$O(\mathbf{u}) \triangleq \frac{1}{n}\sum_{\mathbf{x}\in R}\left(I-\bar{I}\right)\left(Q-\bar{Q}\right)$$

where n is the number of pixels in the image patch and R is the set of pixel locations in the template. Below, we show that either $\bar{I}$ or $\bar{Q}$ – the means of the image patch and the template, respectively – can be removed from the formula without changing the result.

$$\begin{aligned}
O(\mathbf{u}) &= \frac{1}{n}\sum_{\mathbf{x}\in R}\left(I-\bar{I}\right)\left(Q-\bar{Q}\right)\\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\sum_{\mathbf{x}\in R}\bar{I}\left(Q-\bar{Q}\right)\\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\times\bar{I}\sum_{\mathbf{x}\in R}\left(Q-\bar{Q}\right)\\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\times\bar{I}\left[\sum_{\mathbf{x}\in R}Q-\sum_{\mathbf{x}\in R}\bar{Q}\right]
\end{aligned}$$

(continued. . .)

$$\begin{aligned}
O\left(\mathbf{u}\right) &= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\times\bar{I}\left[\sum_{\mathbf{x}\in R} Q - \sum_{\mathbf{x}\in R}\left(\frac{1}{n}\sum_{\mathbf{x}\in R} Q\right)\right] \\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\times\bar{I}\left[\sum_{\mathbf{x}\in R} Q - \sum_{\mathbf{x}\in R} Q\right] \\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right) - \frac{1}{n}\times\bar{I}\times 0 \\
&= \frac{1}{n}\sum_{\mathbf{x}\in R} I\left(Q-\bar{Q}\right)
\end{aligned}$$

Bibliography

[1] Amit Adam, Ehud Rivlin, and Ilan Shimshoni. Robust fragments-based tracking using the integral histogram. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 798–805, 2006.

[2] M. Agrawal, K. Konolige, and M.R. Blas. Censure: Center surround extremas for realtime feature detection and matching. *Proceedings of the European Conference on Computer Vision*, 5305:102–115, 2008.

[3] Saad Ali and Mubarak Shah. Floor fields for tracking in high density crowd scenes. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 1–14, Berlin, Heidelberg, 2008. Springer-Verlag.

[4] D. Balcones, D. Llorca, M. Sotelo, M. Gaviln, S. lvarez, I. Parra, and M. Ocaa. Real-time vision-based vehicle detection for rear-end collision mitigation systems. In Roberto Moreno-Daz, Franz Pichler, and Alexis Quesada-Arencibia, editors, *Computer Aided Systems Theory – EUROCAST 2009*, volume 5717 of *Lecture Notes in Computer Science*, pages 320–325. Springer Berlin / Heidelberg, 2009.

[5] Y. Bar-Shalom. *Tracking and data association.* Academic Press Professional, Inc., San Diego, CA, USA, 1987.

[6] S. Barris and C. Button. A review of vision-based motion analysis in sport. *Sports Medicine*, 38(12):1025–1043, 2008.

[7] A.M. Baumberg. *Learning Deformable Models for Tracking Human Motion.* PhD thesis, School of Computer Studies, University of Leeds, 1995.

[8] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. *Proceedings of the 9th European Conference on Computer Vision*, 13:404–417, May 2006.

[9] SS Beauchemin and JL Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(3):433–466, 1995.

[10] M. Bertalmío, G. Sapiro, and G. Randall. Morphing active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):733–737, 2000.

[11] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 232–237, 1998.

[12] S. Birchfield and S. Rangarajan. Spatiograms versus histograms for region-based tracking. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, page 1158, 2005.

[13] N.D. Bird, O. Masoud, N.P. Papanikolopoulos, and A. Isaacs. Detection of loitering individuals in public transportation areas. *Intelligent Transportation Systems, IEEE Transactions on*, 6(2):167–177, 2005.

[14] M.J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[15] AF Bobick and JW Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, 2001.

[16] B. Bose and E. Grimson. Ground plane rectification by tracking moving objects. *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[17] G.R. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, 2(2):12–21, 1998.

[18] GJ Brostow and R. Cipolla. Unsupervised Bayesian Detection of Independent Motion in Crowds. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1, 2006.

[19] Kevin Cannons. A review of visual tracking. Technical Report CSE-2008-07, York University, Ontario, Canada, September 2008.

[20] D. Caulfield and K. Dawson-Howe. Direction of Camera Based on Shadows. *Proceedings of the Irish Machine Vision and Image Processing Conference*, pages 216–223, 2004.

[21] D. Caulfield and K. Dawson-Howe. Evaluation of multi-part models for mean-shift tracking. *International Machine Vision and Image Processing Conference*, pages 77–82, Sept. 2008.

[22] S.F. Chang, W. Chen, H.J. Meng, and H. Sundaram. A fully automated content-based video search engine supporting spatiotemporal queries. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):602–615, 2002.

[23] Y.L. Chang and JK Aggarwal. 3D structure reconstruction from an ego motion sequence usingstatistical estimation and detection theory. In *Visual Motion, 1991., Proceedings of the IEEE Workshop on*, pages 268–273, 1991.

[24] Y.Q.A. Chen, Y. Rui, and T.S. Huang. Jpdaf based hmm for real-time contour tracking. In *IEEE Computer Vision and Pattern Recognition*, pages I:543–550, 2001.

[25] S.C.S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. *Proceedings of SPIE*, 5308:881–892, 2004.

[26] W. J. Christmas, I. Kolonias, J. Kittler, and F. Yan. Improving the accuracy of automatic tennis video annotation by high level grammar. In *ICIAPW '07: Proceedings of the 14th International Conference of Image Analysis and Processing - Workshops*, pages 154–159, Washington, DC, USA, 2007. IEEE Computer Society.

[27] Robert Collins, Alan Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, and Osamu Hasegawa. A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2000.

[28] R.T. Collins. Mean-shift blob tracking through scale space. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 234–240, 2003.

[29] D. Comaniciu and P. Meer. Robust analysis of feature spaces: Color image segmentation. In *1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings.*, pages 750–755, 1997.

[30] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.

[31] 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*, volume 2, pages 142–149, 2000.

[32] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, 2003.

[33] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:681–685, 2001.

[34] I.J. Cox and S.L. Hingorani. An efficient implementation and evaluation of Reid's multiple hypothesis tracking algorithm for visual tracking. In *International Conference on Pattern Recognition*, pages 437–437, 1994.

[35] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1337–1342, 2003.

[36] R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani. Probabilistic people tracking for occlusion handling. *Proceedings of the 17th International Conference on Pattern Recognition*, 1, 2004.

[37] Xiangtian Dai. *Component-Based Tracking*. PhD thesis, Department of Computer Science, The Johns Hopkins University, October 2005. PhD Thesis, The Johns Hopkins University.

[38] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:886–893, 2005.

[39] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. *Proceedings of the European Conference on Computer Vision*, 1, 2006.

[40] E. R. Davies. Principles emerging from the design of visual search algorithms for practical inspection tasks. In *Proceedings of the 2008 International Machine Vision and Image Processing Conference*, pages 3–20. IEEE Computer Society, 2008.

[41] D.P. Dee and A.M. Da Silva. Data assimilation in the presence of forecast bias. *Quarterly Journal of the Royal Meteorological Society*, 124(545):269–296, 1998.

[42] D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. *Proceedings of the 15th International Conference on Pattern Recognition*, 4, 2000.

[43] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[44] A. Elgammal, R. Duraiswami, and LS Davis. Probabilistic tracking in joint feature-spatial spaces. *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1, 2003.

[45] A. Elgammal, R. Duraiswami, D. Harwood, and L.S. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163, 2002.

[46] H.L. Eng, K.A. Toh, A.H. Kam, J. Wang, and W.Y. Yau. An automatic drowning detection surveillance system for challenging outdoor pool environments. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, 2003.

[47] G. Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99, 1994.

[48] Z. Fan, Y. Wu, and M. Yang. Multiple collaborative kernel tracking. *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[49] Z. Fan, M. Yang, Y. Wu, G. Hua, and T. Yu. Efficient Optimal Kernel Placement for Reliable Visual Tracking. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:658–665, 2006.

[50] P. Fieguth and D. Terzopoulos. Color-based tracking of heads and other mobile objects at videoframe rates. In *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 21–27, 1997.

[51] P.J. Figueroa, N.J. Leite, and R.M.L. Barros. Background recovering in outdoor image sequences: An example of soccer players segmentation. *Image and Vision Computing*, 24(4):363–374, 2006.

[52] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[53] B. Friedland, S.G. Precision, and L.F. Incorporated. Treatment of bias in recursive filtering. *IEEE Transactions on Automatic Control*, 14(4):359–367, 1969.

[54] R. Frühwirth. Track fitting with non-Gaussian noise. *Computer Physics Communications*, 100(1-2):1–16, 1997.

[55] D. Gavrila. Pedestrian Detection from a Moving Vehicle. In *Proceedings of the 6th European Conference on Computer Vision*. Springer-Verlag, 2000.

[56] J. Goldbeck, B. Huertgen, S. Ernst, and L. Kelch. Lane following combining vision and dgps. *Image and Vision Computing*, 18(5):425–433, 2000.

[57] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE proceedings. Part F. Radar and signal processing*, 140(2):107–113, 1993.

[58] P.J. Green, N.L. Hjort, and S. Richardson. *Highly structured stochastic systems*. Oxford University Press, USA, 2003.

[59] I. Guskov. Kernel-based template alignment. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 610–617, June 2006.

[60] D. Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and AK Jain. A background model initialization algorithm for video surveillance. In *Proceedings of the Eighth IEEE International Conference on Computer Vision*, volume 1, 2001.

[61] G. D. Hager and P. N. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition*, pages 403–410. IEEE Computer Society, 1996.

[62] GD Hager, M. Dewan, and CV Stewart. Multiple kernel tracking with SSD. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:790–797, 2004.

[63] I. Haritaoglu, D. Beymer, and M. Flickner. Video-CRM: Detection and Tracking People in Stores. In *Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 141–148, 2003.

[64] I. Haritaoglu, D. Harwood, LS Davis, I.B.M.A.R. Center, and CA San Jose. W 4: real-time surveillance of people and their activities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):809–830, 2000.

[65] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[66] D. Hoiem, A.A. Efros, and M. Hebert. Putting Objects in Perspective. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

[67] BKP Horn and BG Schunck. Determination of optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[68] Jing Huang. *Color-spatial image indexing and applications.* PhD thesis, Cornell University, Ithaca, NY, USA, 1998. Adviser-Zabih, Ramin.

[69] S.S. Intille, J.W. Davis, and A.F. Bobick. Real-time closed-world tracking. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 697–703, 1997.

[70] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998.

[71] O. Javed, K. Shafique, and M. Shah. A hierarchical approach to robust background subtraction using color and gradient information. *Motion and Video Computing, 2002. Proceedings. Workshop on*, pages 22–27, 2002.

[72] A.D. Jepson, D.J. Fleet, and T.F. El-Maraghi. Robust online appearance models for visual tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1296–1311, Oct. 2003.

[73] J. Jeyakar, R.V. Babu, and K.R. Ramakrishnan. Robust object tracking with background-weighted local kernels. *Computer Vision and Image Understanding*, 112(3):296–309, December 2008.

[74] IT Jolliffe. *Principal component analysis.* Springer Series in Statistics, 2002.

[75] S.J. Julier and J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, volume 3, page 26, 1997.

[76] P. KaewTrakulPong and R. Bowden. A real time adaptive visual surveillance system for tracking low-resolution colour targets in dynamically changing scenes. *Image and Vision Computing*, 21(10):913–929, 2003.

[77] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[78] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and Jing Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):319–336, Feb. 2009.

[79] K. Kim, T.H. Chalidabhongse, D. Harwood, and L. Davis. Real-time foreground-background segmentation using codebook model. *Real-time imaging*, 11(3):172–185, 2005.

[80] H.W. Kuhn. The Hungarian Method for the assignment problem. *Naval Research Logistics*, 2:83–97, 1955.

[81] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, 2005.

[82] I. Leichter, M. Lindenbaum, and E. Rivlin. Tracking by Affine Kernel Transformations Using Color and Boundary Cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):164–171, 2009.

[83] A.P. Leung and S. Gong. Mean shift tracking with random sampling. In *Proceedings of the British Machine Vision Conference*, volume 2, pages 729–738, 2006.

[84] J.P. Lewis. Fast template matching. In *Vision Interface*, pages 120–123, 1995.

[85] Peihua Li. An adaptive binning color model for mean shift tracking. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(9):1293–1299, Sept. 2008.

[86] S.F. Lin, J.Y. Chen, and H.X. Chao. Estimation of number of people in crowded scenes using perspective transformation. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 31(6):645–654, 2001.

[87] Z. Lin, L. Davis, D. Doermann, and D. DeMenthon. Simultaneous Appearance Modeling and Segmentation for Matching People under Occlusion. *Asian Conference on Computer Vision*, pages 404–413, 2007.

[88] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:79–116, 1998.

[89] T. List and RB Fisher. CVML – an XML-based computer vision markup language. *Proceedings of the 17th International Conference on Pattern Recognition*, 1, 2004.

[90] T.L. Liu and H.T. Chen. Real-time tracking using trust-region methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):397–402, 2004.

[91] D.G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, volume 2, pages 1150–1157, 1999.

[92] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[93] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International joint conference on artificial intelligence*, volume 3, page 3, 1981.

[94] E. Maggio and A. Cavallaro. Multi-Part Target Representation for Color Tracking. *IEEE International Conference on Image Processing, 2005.*, 1:729–732, 2005.

[95] V.Y. Mariano, J. Min, J.H. Park, R. Kasturi, D. Mihalcik, H. Li, D. Doermann, and T. Drayer. Performance Evaluation of Object Detection Algorithms. *International Conference on Pattern Recognition*, pages 965–969, 2002.

[96] S. Maskell and N. Gordon. A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *Target Tracking: Algorithms and Applications (Ref. No. 2001/174), IEE*, 2001.

[97] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[98] Tom Mathes and Justus H. Piater. Robust non-rigid object tracking using point distribution models. In *Proceedings of the British Machine Vision Conference*, pages 849–858, 2005.

[99] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):810 –815, june 2004.

[100] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42–56, 2000.

[101] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *International Conference on Computer Vision*, volume 1, pages 525–531, 2001.

[102] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[103] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1615–1630, 2005.

[104] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *Proceedings of the European Conference on Computer Vision*, pages 128–142. Springer Verlag, 2002.

[105] T.B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2-3):90–126, 2006.

[106] A. Mohan, C. Papageorgiou, T. Poggio, K. Commun, and R. City. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, 2001.

[107] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background Modeling and Subtraction of Dynamic Scenes. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 2, page 1305. IEEE Computer Society, 2003.

[108] Andrew Moore. An introductory tutorial on kd-trees. Technical Report Technical Report No. 209, Computer Laboratory, University of Cambridge, University of Cambridge, Pittsburgh, PA, 1991.

[109] Hans Moravec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, page 584, August 1977.

[110] J.C. Nascimento and J.S. Marques. Performance evaluation of object detection algorithms for video surveillance. *Multimedia, IEEE Transactions on*, 8(4):761–774, Aug. 2006.

[111] Nassir Navab, Marco Feuerstein, and Christoph Bichlmeier. Laparoscopic virtual mirror – new interaction paradigm for monitor based augmented reality. In *Virtual Reality*, pages 43–50, Charlotte, North Carolina, USA, March 2007.

[112] Anh-Tuan Nghiem, Francois Bremond, Monique Thonnat, and Valery Valentin. ETISEO, performance evaluation for video surveillance systems. In *IEEE International Conference on Advanced Video and Signal based Surveillance*, Sept. 2007.

[113] Q.A. Nguyen, A. Robles-Kelly, and C. Shen. Kernel-based Tracking from a Probabilistic Viewpoint. *Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[114] K. Okuma, A. Taleghani, N. de Freitas, J.J. Little, and D.G. Lowe. A boosted particle filter: Multitarget detection and tracking. *Proceedings of the European Conference on Computer Vision*, 1:28–39, 2004.

[115] NM Oliver, B. Rosario, and AP Pentland. A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[116] Pan Pan, Fatih Porikli, and Dan Schonfeld. Recurrent tracking using multifold consistency. In *Proceedings of the Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.

[117] C.P. Papageorgiou, M. Oren, and T. Poggio. A General Framework for Object Detection. In *Proceedings of the Sixth International Conference on Computer Vision*. IEEE Computer Society, 1998.

[118] V. Parameswaran, V. Ramesh, and I. Zoghlami. Tunable Kernels for Tracking. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:2179–2186, 2006.

[119] V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):677–695, 2002.

[120] Ofir Pele and Michael Werman. A linear time histogram metric for improved sift matching. In *Proceedings of the 10th European Conference on Computer Vision*, pages 495–508, Berlin, Heidelberg, 2008. Springer-Verlag.

[121] Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *International Conference on Computer Vision*, 2009.

[122] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *Proceedings of the European Conference on Computer Vision*, 1:661–675, 2002.

[123] S. Perreault and P. Hebert. Median filtering in constant time. *IEEE Transactions on Image Processing*, 16(9):2389–2394, 2007.

[124] M. Piccardi. Background subtraction techniques: a review. *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, 4, 2004.

[125] M.K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, pages 590–599, 1999.

[126] F. Porikli. Integral histogram: A fast way to extract histograms in cartesian spaces. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 829, 2005.

[127] F. Porikli and O. Tuzel. Multi-Kernel Object Tracking. *IEEE International Conference on Multimedia and Expo*, pages 1234–1237, 2005.

[128] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press, New York, NY, USA, 1992.

[129] V. Rabaud and S. Belongie. Counting crowded moving objects. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 1:705–711, June 2006.

[130] RJ Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *Image Processing, IEEE Transactions on*, 14(3):294–307, 2005.

[131] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, 1979.

[132] J. Rittscher, PH Tu, and N. Krahnstoever. Simultaneous estimation of segmentation and shape. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2, 2005.

[133] M.D. Rodriguez and M. Shah. Detecting and segmenting humans in crowded scenes. *Proceedings of the 15th international conference on Multimedia*, pages 353–356, 2007.

[134] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.

[135] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proceedings of the European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[136] Y. Rubner, C. Tomasi, L.J. Guibas, et al. A metric for distributions with applications to image databases. In *Proceedings of the Sixth International Conference on Computer Vision*, page 59, 1998.

[137] M.S. Ryoo and J.K. Aggarwal. Observe-and-explain: A new approach for multiple hypotheses tracking of humans and objects. In *IEEE Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[138] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. *International Journal of Computer Vision*, 80(1):72–91, 2008.

[139] K. Schindler and H. Wang. Smooth Foreground-Background Segmentation for Video Processing. *Asian Conference on Computer Vision*, 3852:13–16, 2006.

[140] H. Schweitzer, JW Bell, and F. Wu. Very Fast Template Matching. In *Proceedings of the 7th European Conference on Computer Vision*, volume 4, page 372. Springer-Verlag, 2002.

[141] A. Senior et al. Tracking people with probabilistic appearance models. *Proceedings of the IEEE International Workshops on Performance Evaluation of Tracking and Surveillance*, pages 48–55, 2002.

[142] Y. Sheikh and M. Shah. Bayesian Modeling of Dynamic Scenes for Object Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1778–1792, 2005.

[143] C. Shen, M.J. Brooks, and A. van den Hengel. Fast Global Kernel Density Mode Seeking with Application to Localisation and Tracking. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, volume 2, page 1523. IEEE Computer Society, 2005.

[144] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[145] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[146] H. Sidenbladh. Detecting human motion with support vector machines. In *International Conference on Pattern Recognition, Cambridge, UK*, 2004.

[147] O. Sidla, Y. Lypetskyy, N. Brandle, and S. Seer. Pedestrian Detection and Tracking for Counting Applications in Crowded Situations. *Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*, 2006.

[148] N. Siebel and S. Maybank. Fusion of multiple tracking algorithms for robust people tracking. *Proceedings of the European Conference on Computer Vision*, pages 373–387, 2002.

[149] K. Smith, D. Gatica-Perez, and J.M. Odobez. Using particles to track varying numbers of interacting people. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, volume 1, 2005.

[150] C. Stauffer and WEL Grimson. Learning patterns of activity using real-time tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):747–757, 2000.

[151] C. Stauffer, K. Tieu, and L. Lee. Robust automated planar normalization of tracking data. *Proceedings of Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[152] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Filtering using a tree-based estimator. *Computer Vision, IEEE International Conference on*, 2:1063, 2003.

[153] M.S. Stephen and J.M. Brady. SUSAN – A new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–48, 1997.

[154] Moritz Störring. *Computer Vision and Human Skin Colour.* PhD thesis, Aalborg University, 2004.

[155] R.L. Streit and T.E. Luginbuhl. Maximum likelihood method for probabilistic multi-hypothesis tracking. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 2235, pages 394–405, 1994.

[156] Jian Sun, Weiwei Zhang, Xiaoou Tang, and Heung-Yeung Shum. Bidirectional tracking using trajectory segment analysis. In *Tenth IEEE International Conference on Computer Vision*, volume 1, pages 717–724 Vol. 1, Oct. 2005.

[157] Richard Szeliski and James Coughlan. Spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218, 1997.

[158] Y.L. Tian and A. Hampapur. Robust Salient Motion Detection with Complex Background for Real-time Video Surveillance. *Proceedings of the IEEE Computer Society Workshop on Motion and Video Computing*, Jan. 2005.

[159] Christopher Town. Multi-sensory and multi-modal fusion for sentient computing. *International Journal of Computer Vision*, 71(2):235–253, 2007.

[160] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. *Proceedings of the IEEE International Conference on Computer Vision*, pages 255–261, 1999.

[161] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[162] P.H. Tu, T. Sebastian, G. Doretto, N.O. Krahnstoever, J. Rittscher, and T. Yu. Unified crowd segmentation. In *European Conference on Computer Vision*, pages IV: 691–704, 2008.

[163] J. Vermaak, A. Doucet, and P. Perez. Maintaining multimodality through mixture tracking. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1110–1116, 2003.

[164] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:511–518, 2001.

[165] 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.

[166] H. Wang and D. Suter. Background initialization with a new robust statistical approach. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 153–159, 2005.

[167] H. Wang and D. Suter. A novel robust statistical method for background initialization and visual surveillance. In *Asian Conference on Computer Vision*, 2006.

[168] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[169] B. Wu and R. Nevatia. Tracking of multiple, partially occluded humans based on static body part detection. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, 2006.

[170] B. Wu and R. Nevatia. Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet based Part Detectors. *International Journal of Computer Vision*, 75(2):247–266, 2007.

[171] Hao Wu, Rama Chellappa, Aswin C. Sankaranarayanan, and Shaohua Kevin Zhou. Robust visual tracking using the time-reversibility constraint. *Proceedings of the Eleventh IEEE International Conference on Computer Vision*, 2007.

[172] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, pages 1101–1113, 1993.

[173] D. Xu, Y. Wang, and J. An. Applying a New Spatial Color Histogram in Mean-Shift Based Tracking Algorithm. *Image and Vision Computing New Zealand*, 2005.

[174] C. Yang, R. Duraiswami, and L. Davis. Efficient Mean-Shift Tracking via a New Similarity Measure. *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[175] C. Yang, R. Duraiswami, and L. Davis. Fast multiple object tracking via a hierarchical particle filter. In *Tenth IEEE International Conference on Computer Vision*, volume 1, 2005.

[176] Changjiang Yang, Ramani Duraiswami, Nail A. Gumerov, and Larry Davis. Improved fast gauss transform and efficient kernel density estimation. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 464, Washington, DC, USA, 2003. IEEE Computer Society.

[177] Mohammed Yeasin and Subhasis Chaudhuri. Development of an automated image processing system for kinematic analysis of human gait. *Real-Time Imaging*, 6(1):55 – 67, 2000.

[178] A. Yilmaz. Object Tracking by Asymmetric Kernel Mean Shift with Automatic Scale and Orientation Selection. *Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2007.

[179] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4), 2006.

[180] A. Yilmaz, X. Li, and M. Shah. Contour-Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1531–1536, 2004.

[181] L. Zhao and LS Davis. Closely coupled object detection and segmentation. In *Tenth IEEE International Conference on Computer Vision*, volume 1, 2005.

[182] Q. Zhao, S. Brennan, and H. Tao. Differential EMD Tracking. *Eleventh IEEE International Conference on Computer Vision*, 2007.

[183] Q. Zhao and H. Tao. Object tracking using color correlogram. In *Proceedings of the 14th International Conference on Computer Communications and Networks*, pages 263–270, 2005.

[184] T. Zhao and R. Nevatia. Tracking multiple humans in complex situations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1208–1221, 2004.

[185] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:406–413, 2004.

[186] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, 2003.

[187] Z. Zivkovic and B. Krose. An EM-like algorithm for color-histogram-based object tracking. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1, 2004.