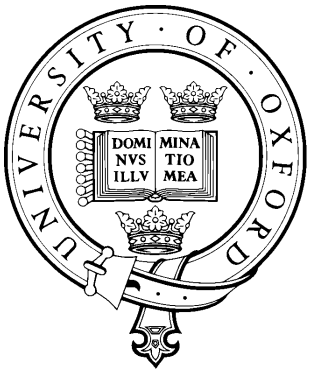
The Acquisition of Coarse Gaze Estimates in Visual Surveillance

Ben Benfold
Jesus College



D.Phil. Thesis
Supervised by Prof. Ian D. Reid

Robotics Research Group
Department of Engineering Science
University of Oxford

Trinity Term, 2011

This thesis is submitted to the Department of Engineering Science, University of Oxford, for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Ben Benfold
Jesus College
University of Oxford

Doctor of Philosophy
Trinity Term 2011

The Acquisition of Coarse Gaze Estimates in Visual Surveillance

Abstract

This thesis describes the development of methods for automatically obtaining coarse gaze direction estimates for pedestrians in surveillance video. Gaze direction estimates are beneficial in the context of surveillance as an indicator of an individual's intentions and their interest in their surroundings and other people. The overall task is broken down into two problems. The first is that of tracking large numbers of pedestrians in low resolution video, which is required to identify the head regions within video frames. The second problem is to process the extracted head regions and estimate the direction in which the person is facing as a coarse estimate of their gaze direction.

The first approach for head tracking combines image measurements from HOG head detections and KLT corner tracking using a Kalman filter, and can track the heads of many pedestrians simultaneously to output head regions with pixel-level accuracy. The second approach uses Markov-Chain Monte-Carlo Data Association (MCMCDA) within a temporal sliding window to provide similarly accurate head regions, but with improved speed and robustness. The improved system accurately tracks the heads of twenty pedestrians in 1920×1080 video in real-time and can track through total occlusions for short time periods.

The approaches for gaze direction estimation all make use of randomised decision tree classifiers. The first develops classifiers for low resolution head images that are invariant to hair and skin colours using branch decisions based on abstract labels rather than direct image measurements. The second approach addresses higher resolution images using HOG descriptors and novel Colour Triplet Comparison (CTC) based branches. The final approach infers custom appearance models for individual scenes using weakly supervised learning over large datasets of approximately 500,000 images. A Conditional Random Field (CRF) models interactions between appearance information and walking directions to estimate gaze directions for head image sequences.

Contents

List of Figures

List of Tables

Acronyms

CPU Central Processing Unit

CRF Conditional Random Field

CTC Colour Triplet Comparison

EM Expectation Maximisation

FP False Positive

FPS Frames Per Second

GPU Graphics Processing Unit

HMM Hidden Markov Model

HOG Histogram of Oriented Gradients

KLT Kanade-Lucas-Tomasi

LBP Loopy Belief Propagation

MAAE Mean Absolute Angular Error

MCMC Markov Chain Monte Carlo

MCMCDA Markov Chain Monte Carlo Data Association

MDL Minimal Description Length

MIG Maximal Information Gain

MOTP Multi Object Tracking Precision

MOTA Multi Object Tracking Accuracy

MRF Markov Random Field

RMSE Root Mean Square Error

SLAM Simultaneous Localisation And Mapping

SVM Support Vector Machine

TAN Tree Augmented Naive Bayes

Acknowledgements

First of all I would like to thank my supervisor Ian for the all of the guidance and feedback he has given me throughout my time as a DPhil student. The Active Vision Group has been the ideal place to develop and share ideas and it has been a pleasure to interact with so many enthusiastic and creative people. Two members of the group who I would like to thank in particular are Victor Prisacariu, for creating the fastHOG library without which much of the research in this thesis would not have been possible, and Eric Sommerlade, from whom I learned a lot about computer vision through frequent discussions and collaborations on various projects.

I have also benefitted greatly from collaborations with researchers from outside the group, both as part of the EU Hermes project and the Oxford Risk project. In particular I would like to thank Joe Hale and Terry Thomson for their part in the filming of the Town Centre and Transport Terminal datasets and producing the model data used in chapter 6.

I would also like to thank my examiners, Professors Andrew Zisserman and Richard Bowden, for giving up their valuable time to read this thesis and for their helpful and constructive feedback.

Finally I cannot thank my wife Kathryn enough for putting up with my absence during the evenings and weekends before deadlines and for the continual support and encouragement she has given me over the last four years.

CHAPTER 1

Introduction

1.1 Problem Definition

Video surveillance is an expanding industry sector with cameras being introduced in increasingly diverse locations. Although there are no official statistics, a recent study [36] estimated that there are approximately 1.85 million surveillance cameras in the UK, of which approximately 30,000 monitor public spaces. Unfortunately, as the number of cameras increases, so does the need for monitoring them.

Currently almost all surveillance monitoring is performed by humans and each individual is expected to monitor approximately 20-100 cameras simultaneously [56], with the probability of crimes being missed increasing with the number of cameras being monitored [38]. Although the video is usually recorded, this is typically at a low frame rate ($\approx 1fps$) and highly compressed due to storage limitations. The retrospective examination of surveillance footage also removes the possibility

of intervention. One of the primary uses of surveillance is to provide the evidence required to convict criminals, however prosecutions are often impossible because of the poor video quality [37]. When an observer notices a crime taking place, the majority of surveillance systems will allow them to manually increase the quality of the recording, either for a single chosen camera or for a limited period of time. Observers may also control active cameras to obtain higher resolution imagery.

In addition to crime detection, visual surveillance allows the detection of various other types of behaviour requiring intervention. An observer can call for help if a person appears to be in need of medical assistance and can locate a lost child or animal. There is also the potential to detect events involving many people, for example in a crowded place the behaviour of multiple people could indicate a risk of people getting crushed, or a large number of people rushing out of a building could suggest the possibility of a fire.

In the foreseeable future it is unlikely that the role of people in a surveillance system could be entirely replaced by automated systems, however there are two key areas where computer vision systems can be of assistance. The first involves increasing the frequency with which a human observes the correct camera when an event requiring intervention is taking place. If a computer vision system were able to obtain even a small amount of information on the likelihood of anomalous events taking place, the frequency with which each camera is viewed could be adjusted accordingly to increase the likelihood of detection.

The second area where computer vision could be of assistance is to improve the quality of recordings where either no human observer is present or the likelihood of a crime or other event taking place is not large enough to justify immediate notification. Existing recording systems have the ability to increase the recording quality based on motion, however storage space could be used more efficiently by only increasing the recording quality for particular areas of a scene. If enough information were available, it would also be possible to automatically direct active

cameras to obtain high resolution images of the people involved.

To date, most approaches for the detection of anomalous events have been based on trajectories [54, 79, 80], however trajectories provide only a limited amount of information and are likely to be insufficient for detecting subtle differences in behaviour. The video quality and resolution, coupled with real-time requirements, make detailed analysis of each individual infeasible, however one aspect that is expected to be relatively simple to measure is the direction in which a person is facing. The pose of a pedestrian's head has the potential to yield a significant amount of useful information because the direction in which people look is a strong indicator of their focus of attention. The focus of an individual's attention often indicates their desired destination whereas mutual attention between people indicates familiarity, and any single object or person receiving attention from a large number of people is likely to be worthy of further investigation.

There are many types of behaviour for which we expect patterns of gaze behaviour to be different from that of ordinary pedestrians, whose gaze direction tends to be towards their direction of motion. A person in distress is likely to look in many different directions in the hope of obtaining assistance, and a criminal is likely to frequently look at security guards, shop assistants and security cameras. In addition to these specific patterns of behaviour, we expect that any event of sufficient interest to attract the attention of security staff is also likely to attract the attention of other people in the vicinity. This is particularly useful because pedestrians perform the difficult task of detecting any anomalous events and provide output in the form of their gaze direction, which we expect to be significantly easier to measure than the anomalous event directly.

The purpose of the work described in this thesis is to develop methods for obtaining coarse gaze estimates from surveillance video, with the aim of providing information that can benefit either a human observer or a fully automated system. Although methods for gaze estimation have already received significant attention

Figure 1.1: Sample frames from the video datasets that are used as a source of head images. Videos exhibit a wide variety of viewpoints, lighting conditions and crowd densities.

(see section 2.1), very little past work has dealt with gaze estimation in video which is representative of what would be obtained in a genuine surveillance installation. On the contrary, the work presented in this thesis is intended to address the full difficulty of the problem by working with the most realistic data that could be reasonably obtained. Some examples from the datasets that will be used are shown in figure 1.1.

1.2 Practical Considerations

In the intended application domain of visual surveillance little can be assumed about the people being monitored or their surroundings. In particular for a gaze estimation

system to be of practical value it must be able to estimate the gaze directions for previously unobserved people. This is significantly more difficult than working with only known people in known locations because there are many factors which can cause the appearance of individuals to differ:

- **Pedestrian Appearance:** People have different skin colours and textures, different facial structure, and variations in hair colour and style.
- **Scene Lighting:** Scenes can be illuminated from different directions, with variations in the level of diffusion and the occurrence of shadows.
- **Colour Appearance:** Colours have different appearances under sunlight and different forms of artificial light
- **Camera Intrinsics and Extrinsics:** The height of the camera results in people being observed from more or less acute angles and there may be distortion in the images.
- **Scene Contents:** Scenes often contain cluttered backgrounds and people can be occluded by other people or foreground objects.
- **Clothing:** Pedestrian clothing has a wide variety of shapes and colours which can make people difficult to locate.
- **Headwear:** Hats, glasses and scarves can obscure parts of the head
- **Pedestrian Behaviour:** Pedestrians behaving naturally will talk on mobile telephones, eat, drink, and talk, which all result in parts of the head temporarily changing in appearance.
- **Image Sensor:** Incorrectly focussed cameras can contain blur, low light levels will produce images with high frequency noise and/or motion blur, and image compression can result in quantisation artefacts being introduced.

Some of the variation in appearance due to these factors will not be present between training and testing datasets if the same people or scenes appear in both, so it is likely that any experiments using them would have better performance than with independent training and testing datasets. Unless otherwise specified, the experiments in this thesis result from testing on images from people who do not appear in any training data, so the results reflect what would be expected in a genuine installation.

A second consideration which has influenced the choice of research direction is the need for a practical system to be able to process video in real-time so that the output can be used to alert the relevant person during or shortly after an event occurs. Any system which cannot operate in real-time is unlikely to provide any significant benefit over simply recording the video.

Since surveillance cameras are generally required to monitor reasonably large spaces, people tend to occupy only a small portion of the image area. The result is that we must be able to estimate the gaze direction for head images that are typically between ten and thirty pixels in diameter.

The last consideration results from our inability to constrain the movement of pedestrians. In many cases pedestrians will be facing away from the camera so we must be able to estimate their gaze direction through a full 360° range of rotations about a vertical axis, which will be referred to as the head *pan* angle. The heads of pedestrians in all datasets exhibit rotations other than pan but no attempt will be made to measure these rotations. The head pan angle is considered to be the most important because in most indoor and outdoor scenes, the ground plane location is sufficient to identify people and objects.

1.3 Thesis Outline

In this first chapter the problem that is the subject of this thesis has been introduced at a high level. Chapter 2 provides a detailed review of the state-of-the-art in the estimation of coarse gaze directions, an overview of applications, and a review of methods for obtaining head images. Chapter 3, the first technical chapter, begins by describing attempts at gaze estimation in very low resolution images. To validate this approach, in chapter 4 an automatic head tracker is developed for obtaining head images and improvements to the gaze estimation methods are made as a consequence. A complete working system is demonstrated, allowing the key issues affecting the accuracy to be identified. These issues are addressed in the following two chapters.

Chapter 5 improves the accuracy of the automatic head location estimates by developing a sliding window based tracking system that fully exploits all of the information that is available. In chapter 6 the problem of insufficiently representative training data is addressed by automatically learning gaze estimators from hundreds of thousands of unlabelled images. Finally chapter 7 concludes the thesis by summarising the key points and suggesting directions for future research.

1.4 Contributions

The research described in this thesis makes contributions in the areas of coarse gaze direction estimation, pedestrian tracking, and weakly supervised learning. The specific contributions made in each of the four technical chapters are the following:

Chapter 3:

- A randomised fern based image classifier with predicate based branches is proposed along with a corresponding algorithm for inference.

- A method for learning colour distributions corresponding to abstract hair, skin and background labels to improve estimation accuracy is developed.
- Modifications are proposed to the standard methods for combining decision tree estimates to make best use of the predicate ferns.

Chapter 4:

- A multi-target tracking system is developed for the specific purpose of obtaining stable head image sequences
- Robust randomised fern based gaze classifiers are developed, making use of proposed Colour Triplet Comparison (CTC) image measurements to provide invariance to lighting effects.
- A complete system for measuring attention in large scale scenes is demonstrated. This is believed to be the first system to measure attention in scenes where pedestrians are free to walk around and behave naturally.

Chapter 5:

- A principled objective function is developed to allow both accurate location estimates and robust data associations to be made, including the ability to track through total occlusions.
- A new move type for Markov-Chain Monte-Carlo Data Association (MCMCDA) is introduced to allow the removal of false positives, with the potential for generalisation to also identify different object types.
- An efficient and scalable multi-threaded architecture is proposed to allow large crowds of pedestrians to be tracked in real-time.

Chapter 6:

- A Conditional Random Field (CRF) model is developed to accurately represent the interactions between gaze directions and walking directions.
- A complete system for inferring gaze directions and training a forest of randomised trees is demonstrated.
- A hybrid decision tree is proposed to allow efficient inference, with the potential for additional benefits in a distributed system.

CHAPTER 2

Related Work and Datasets

Having introduced the problem, we now examine existing work with either similar objectives or potentially useful methods. The chapter establishes a foundation upon which the research described in later chapters will be built.

The contents of this chapter are intended to give an overview of existing work that is related to the use of coarse gaze estimation in visual surveillance. The review is divided into five sections; section 2.1 covers existing work that is relevant to head pose estimation, with the gaze estimation performance of humans briefly covered in section 2.2 and a discussion of applications of gaze estimation in section 2.3. Section 2.4 describes methods which have been or could be used to locate heads in surveillance video. The majority of algorithms are only briefly described here, since details of relevant algorithms have been included in the corresponding chapters, however section 2.5 covers randomised trees and ferns in detail since they will be

referred to extensively throughout this thesis. A description of the datasets used for the training and evaluation of algorithms throughout the remaining chapters is covered in section 2.6.

2.1 Methods for Coarse Gaze Estimation

Many different methods have been applied to the problem of coarse gaze estimation, however the low resolution of the head images that are typically found in surveillance video prevents the application of techniques which require detail such as those which track feature points or detect facial features [86, 52, 24, 92]. The majority of past research into head pose measurement in low resolution video has involved the use of labelled training examples which are used to train various types of classifiers.

The following sections provide an overview of existing approaches to gaze direction estimation, categorised by the main algorithm used. Publications that have used high resolution images have been included only if they could potentially be applied to lower resolution imagery. A table comparing the quantitative results from all of the related work can be found at the end in section 2.1.7, along with a discussion of some of the general problems with existing approaches.

2.1.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are used to map continuous inputs to one or more outputs, so can be used for both direct regression and classification. ANNs consist of one or more layers of connected nodes which output values based on their inputs from the previous layer. The interested reader is referred to Bishop [13] for a more detailed explanation.

Zhao et al. [137] trained a multi-layer network using greyscale images which had been normalised so that the distribution of intensities were similar. Nineteen output nodes provided the relative probabilities of different pan angles spaced ten degrees

apart. Brown and Tian [19] use a similar method and subsequent work by Tian et al. [122] improved the accuracy by combining results from two cameras. Gourier et al. [42] used similar input data, but trained individual single-layer networks for each of 22 different pan classes and estimated the angle using the maximum response. These approaches resulted in small errors of approximately 10° for a frontal 180° range of movement, but the datasets were somewhat contrived in that images were collected in controlled environments with fixed backgrounds and no variation in lighting, scale or rotations about other axes.

A similar line of research began simultaneously with the work of Stiefelhagen et al. [120, 119]. A three layer network was trained using normalised greyscale images but images filtered with both horizontal and vertical edge detectors were also included, with the best results being obtained from a combination of all three. Voit et al. [125, 126] continued with the same approach, but classified images over a full 360° range. Although the classification rate was almost random at 15.8% for a single camera, the accuracy was improved to 39.4% by combining the results from four cameras, each having a different view of the subject.

Rae and Ritter [100] used a combination of colour and texture information. One neural network was trained to recognise skin-coloured regions of the image and another was trained to fit an ellipse around the skin region corresponding to the face using responses from Gabor wavelet filters. Finally a third network mapped more Gabor wavelet filter responses to estimations of the head pan and tilt angles. Although accurate with only 9° RMSE (Root Mean Square Error), a problem identified by the authors was a significant degradation in performance when the network was tested on images from a different person to the one used for training.

Another variation was made by Seeman et al. [108], who used stereo cameras to obtain a depth map which was used in addition to the normalised greyscale head image to train a three layer network.

One of the main problems of ANNs is that they have a tendency to over-fit to

their training data. It seems likely that this is a problem in practice, since none of the reviewed approaches show evidence of having different people in the testing and training datasets. The appearance of people across different scenes varies enormously and it would only be feasible to collect training data representing a small subset of appearances. In recent years, ANNs have been largely superseded by Support Vector Machines (SVMs) which tend to generalise better to unseen data.

2.1.2 Dimensionality Reduction

Dimensionality reduction methods map images to points in a lower dimensional space, with the aim of increasing the ratio of useful data to redundant data and in many cases to allow direct pose estimation from the shape of the manifold in the low dimensional space.

The first approaches were those of Gong at al. [39, 78] which were based on Principal Components Analysis (PCA). PCA identifies a set of principal components that can be used to represent the original data, where the first components represent the directions of most variation and the last components represent the directions of least variation. For high dimensional data such as that from images, many of the principal components have little or no variation so can be omitted. The result is a compact representation of the original data which improves the efficiency of subsequent analysis using standard classification and regression algorithms.

The approach taken by Morency et al. [82] also used PCA but made use of both intensity and depth image priors to customise the pose estimator for individual people. Srinivasan and Boyer's approach [116] used PCA to define eigenspaces for discrete classes so that the gaze direction for a test image could be determined by the best matching eigenspace. Wei et al. [129] also found separate eigenspaces for discrete direction classes, but in this case the input images were first filtered using Gabor wavelet filters. Guo et al. [43] tried two variations on PCA (2DPCA and BMPCA) which were intended to take advantage of the 2D structure of the images.

PCA has been superseded in more recent times by manifold learning methods [34, 8, 6, 23, 71, 49] which also map images into a lower dimensional space, but with different objectives. Where PCA aims to reduce the redundancy in the original image data to allow a more compact representation, manifold learning methods do not attempt to represent all of the original data. Manifold learning methods attempt to learn a one-way mapping from the high dimensional space into a low dimensional space, with the aim of making images that have similar parameters close together and those with very different parameters far apart.

An advantage of these methods over PCA is that the parameter (i.e. the gaze direction) has some geometric significance in the low dimensional space. After mapping the test image to the lower dimensional space in which the manifold is embedded, various different methods have been used to estimate the gaze direction such as K-Nearest Neighbours [34], linear regression [6], a relevance vector machine [128] or support vector regression/cubic splines [8].

Although these methods report superior accuracy over other methods, they have all been applied to particularly contrived datasets where great care has been taken to constrain the environment and head motion to ensure that the only differences between images result from changes to the head pan angle and in some cases a one dimensional rotation in the lighting direction. Also, none of the reviewed work included a comparison of the method to the equivalent method (e.g. Nearest Neighbour) in the original high dimensional space, so the benefit of the dimensionality reduction is unproven. Two of the reviewed papers [34, 128] performed experiments in which the number of dimensions used to embed the manifold were varied and in both cases the accuracy increased with the number of dimensions, which suggests that any performance benefits might be due the nonlinear transformation from the original high dimensional space to the embedding space rather than the removal of redundant information.

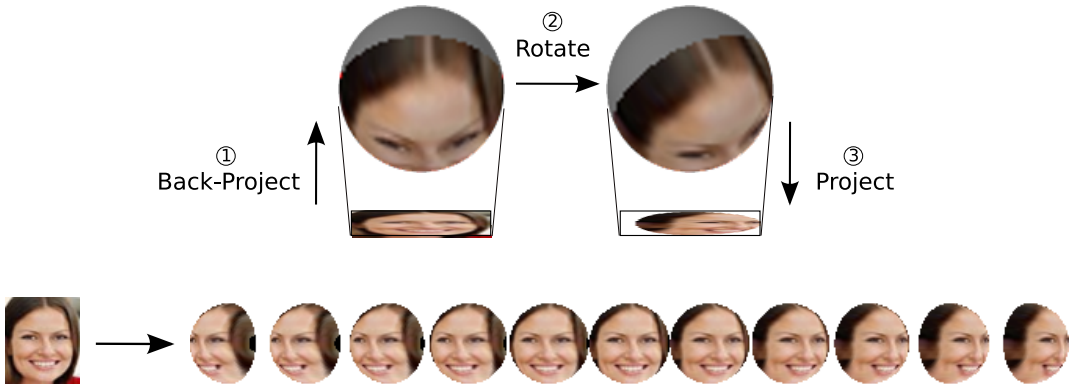


Figure 2.1: Reprojection of a head image using an ellipsoid model. The sample images show the reprojections in steps of ten degrees.

2.1.3 Geometric Models

One of the earliest and simplest attempts at head pose estimation was by Brunelli [20], whose model was based on the assumption that front facing heads are more symmetrical than non-frontally facing heads. By measuring the level of asymmetry in the image gradients around the eye locations, clockwise rotations of up to 50° were measured. It is unclear how the system would perform if a person capable of rotating their head both clockwise and anticlockwise were encountered.

Chen et al. [25] used a slightly more sophisticated model in which the hair and skin regions were identified using colour models. A model was learned which linked the head pose with the region centres and axes of least inertia. Sherrah and Gong [109, 110] learned the correlation between the head pose and the size and location of the face relative to the rest of the head. Faces were detected using an SVM and the head was tracked using a particle filter based skin colour tracking algorithm.

Pappu and Beardsley [95] started a line of research where the head was modelled as an ellipsoid. A single reference image was used to generate synthetic rotated head views which are matched to head images in the video to determine the relative rotation. The reference image is simply back-projected onto the ellipsoid and various rotations are applied before projecting back onto the initial image to give the synthetic views, as shown in figure 2.1. The head rotation is then estimated by

taking the synthetic image that is closest to the observed image.

A similar approach was taken by Wu and Toyama [134], who also modelled the head as an ellipsoid but projected the locations of wavelet-like features rather than a plain texture. This had the advantage of providing a certain amount of invariance to lighting and colour information which allowed a generic head model to be learnt, though the head images had to be detailed enough to allow the calculation of wavelet responses.

Canton-Ferrer et al. [21] approached the problem by first using colour histograms to estimate the probability that pixels were skin. The projections from four different camera views onto an ellipsoid allowed a skin probability texture to be learnt which was fitted to subsequent views using a particle filter to determine the relative pose.

Most recently, Zabulis et al. [136] modelled the head using a sphere, but instead of learning an appearance model, a generic face detection algorithm [124] was used. The appearances from multiple camera views were mapped onto a sphere and the texture was unwrapped before the detector was applied. An advantage of this approach is that a standard face detection module that is known to work well for many people under large variations in lighting could be used. Unfortunately eight cameras were required to ensure that the face was always visible from at least one view, making it impractical for most purposes.

2.1.4 Nearest Neighbour and Decision Tree Classifiers

Nearest neighbour classifiers compare test images with training examples to find the closest match, with the resulting classification being that of the matching training example. Decision tree classifiers perform a similar function but are able to locate close training data more efficiently. Since decision trees are frequently used throughout this thesis, a detailed description is deferred to section 2.5.

Niyogi and Freeman [87] classified head orientations by comparing the test image to a number of examples for which the correct orientation was known and taking

the orientation of the most similar example. A decision tree was built using the training examples to allow the closest match to be efficiently found. Robertson et al. [106, 105] used a similar method but performed the additional step of using background subtraction and a skin colour histogram to identify the probabilities of pixels belonging to the foreground and skin regions respectively before classification. To allow nearest neighbours to be identified more efficiently, Robertson also constructed decision trees from the examples. A significant drawback of Robertson's approach is that the skin colour depends on the individual person and the lighting conditions of the scene, so the skin colour histogram has to be manually defined for each video sequence.

2.1.5 Probabilistic Appearance Models

Ba and Odobez [4] used the product of a texture term and a colour term to infer head poses. The texture term resulted from first dividing example head images into classes according to their orientations and building feature vectors from the convolutions of 2D Gabor wavelets. These feature vectors were used to learn Gaussian mixture models for each orientation, which were then used to test the relative probability of a test image belonging to each direction class. The colour based term was obtained by classifying each image pixel as skin or non-skin based on a fixed colour model and comparing the test image to the mean pixel classification from each group of training data. Later work by Smith et al. [114] used a similar set of features for which class probabilities were estimated using simpler diagonal Gaussian models.

Aghajanian and Prince [1] divided each head image into a number of smaller patches and estimated the probability of angle classes at 10° intervals by comparing the observed patches with those from a library of patches. The probabilities of each patch belonging to each direction class were learned using approximately 10000 training images.

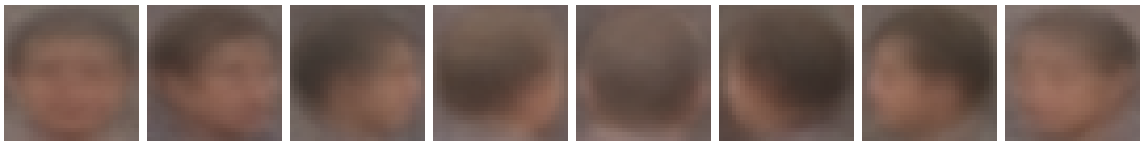


Figure 2.2: Examples of mean templates for eight direction classes, similar to those used by Orozco. These were the result of averaging over approximately 200 images per class.

2.1.6 Support Vector Machines

A standard Support Vector Machine (SVM) classifies data points into one of two classes by partitioning the space of corresponding feature vectors using a hyperplane. During training, the hyperplane is optimised to maximise the distance between the plane and the closest points from the two classes, which helps the classifier to generalise to unobserved data. Multi-class SVMs work by training a binary SVM either for each pair of classes (one-against-one) or between each class and all other classes combined (one-against-rest). A one-against-one SVM uses voting to combine the estimates from each of the binary classifiers whereas a one-against-rest SVM selects the class for which the data is furthest from the dividing hyperplane.

In the most recent work by Launila and Sullivan [63] a number of different image measurements were made. These included the normalised greyscale image, a gradient filtered image, edge detections, Gabor filter responses, Gaussian filter responses, and the Luv colourspace representation of the image. Since using all of this information would encourage overfitting, a subset of the features were selected for SVM training. Three methods were tested; the first was a greedy selection method that chose the features providing the greatest performance increase. The second chose all features that had sufficient mutual information with the true class. The third method involved training individual classifiers for each feature type and using boosting to combine the outputs, which performed the best out of the three methods.

Another recent system described by Orozco et al. [91] generates feature vectors

from head images by comparing them to *mean templates* for each class, which were generated by taking the mean pixel intensities across all of the training data in the corresponding class. Some example mean templates are shown in figure 2.2. The Kullback-Liebler divergence was used to compare the pixel intensities of a test image with those from each of the eight class templates, the result of which was classified with a one-against-rest SVM. The reported results were good with up to 81% correctly assigned to one of eight classes, however it seems likely that this was due to the training and testing data being taken from the same two video sequences.

Ng and Gong's system [85] trained five binary SVM classifiers to distinguish head images within different ranges of gaze directions from the same set of non-head images. Within a low dimensional eigenspace, the support vectors from the individual classifiers were all close to those from adjacent pose ranges. This property allowed the gaze directions for head images to be estimated by finding the closest support vectors and averaging over the corresponding gaze directions from the data used to train the SVM. The method appears to make the assumption that similar support vectors will be found for adjacent pose classes, which might not be the case in general.

Huang et al. [51] trained a 3-class one-against-rest SVM using normalised greyscale images, and were able to achieve 100% accuracy on their dataset. There were however only three direction classes representing frontal gazes and directions of 30° to the left and to the right, and the test images were captured under controlled conditions. Ma et al. [77] used Gabor filter responses as input to a one-against-one multi class SVM with seven classes. Another recent paper by Siriteerakul et al. [113] also used one-against-one SVMs, in this case to classify images into eight pose classes. The paper postdates the majority of work in this thesis and the features used as input to the SVMs appear to have been inspired by the features that are used in chapters 3 and 4. The results show Siriteerakul's SVM to have an accuracy of 58.2% on a realistic low resolution dataset, and provide a comparison with Orozco's mean

template method [91] which only classified 31.3% of images correctly.

An unusual geometric approach was taken by Wang and Brandstein [127], who detected the hairline and sampled the location in six places to construct feature vectors. Sixty training samples of known head pose were used to train a support vector machine, which could then be used to identify the most likely head pose. The method is heavily dependent on the hairline being easy to extract and has little room for generalisation so would be unlikely to work with anybody not included in the training data.

SVMs have been used in many recent approaches, but again none have shown results for datasets containing the large numbers of different appearances of people that would have to be processed for practical applications.

2.1.7 Summary of Existing Methods

A comparison of publications reporting experimental results for coarse gaze direction estimation is provided in table 2.1. Results were reported either as a Mean Absolute Angular Error (MAAE), Root Mean Square Error (RMSE), or as a percentage of test images that were correctly classified into a number of discrete classes. A validity rating from A-E has been assigned to each of the publications in the table to indicate the level of separation between the datasets used for training and testing; the meanings of the validity ratings are explained in table 2.2.

Direct comparison between methods is difficult, not only because of differences in the way in which accuracy is measured but also due to differences in training data. Most data appears far from that which might be encountered in a security monitoring situation due to a lack of variation in lighting conditions, appearances of the test subjects and range of head rotation. Backgrounds were often plain to simplify foreground separation and in many cases the distribution of gaze directions in the test data was highly non-uniform, a property which was exploited through priors or motion models that were learned from the same situation [4].

In addition to the papers reviewed here, the interested reader is referred to a recent review of both fine and coarse gaze estimation methods by Murphy-Chutorian and Trivedi [83], which includes a comparison of results using popular datasets.

Table 2.1: A comparison of research into coarse gaze direction estimation. In all cases where the range of directions is less than 360°, the range was centred around frontal face views. In some datasets test images were not evenly distributed over all directions, U and NU are used to indicate uniform and non-uniform distributions. The validity column indicates how realistic the testing and training scenario was, see table 2.2 for more details.

Author/Year	Method	Image Size	Pose Range	Validity	Accuracy
Aghajanian'09 [1]	Probabilistic	60 sq	180∘ (NU)	E	13.2∘ MAAE
Ba'05 [4]	Probabilistic	64 sq	120∘ (NU)	E	8.7° MAAE (1 cam)
Balasubramanian'08[6]	Dim. Reduction	$32sq$	180∘ (U)	D	2° MAAE
BenAbdelkader'10 [8]	Dim. Reduction	$25sq$	180∘ (U)	B-D	8.6∘ MAAE
Brown'02 [19]	Neural network	$8-64$ sq	180° (NU)	D	9 classes, 87-91%
Canton-Ferrer'07 [21]	Ellipsoid Model	-	360°	D	20.5° MAAE (4 cams)
Chen'03 [23]	Dim. Reduction	-	20° (U)	C	4° MAAE
Fu'06 [34]	Dim. Reduction	25×30	180∘ (U)	B	1.8° MAAE (1 cam)
Gourier'06 [42]	Neural network	23×30	180°(U)	D	10.3° MAAE
Guo'06 [43]	Dim. Reduction	32×32	120°(U)	D	9 classes, 95.5%
Hu'05 [49]	Dim Reduction	24×16	360°	D-E	None provided
Huang'98 [51]	SVM	-	67.55°(U)	D	3 classes, 100%
Launila'10 [63]	SVM	$10-20$	360°	D	8 classes, 0.53 mean class error
Li'07 [71]	Dim. Reduction	-	180°	C	19.1° MAAE
Ma'06 [77]	SVM	64 sq	90° (U)	D	7 classes, 97.1%
Morency'03 [82]	Dim. Reduction	32 sq	60° (NU)	D	5° RMSE
Ng'99 [85]	SVM+Dim. Reduction	-	180° (NU)	A-D	$8.7° - 13.6°$ MAAE
Niyogi'96 [87]	Nearest neighbour	16 sq	100°(U)	D	15 classes, 48%
Orozco'09 [91]	SVM	$5-40$ sq	360°	B	8 classes, 16-81%

(continued on next page)

Author/Year	Method	Image Size	Pose Range	Validity	Accuracy
Pappu'98 [95]	Model	32 sq	112° (NU)	C	None provided
Rae'98 [100]	Neural Network	$80sq$	150°	C	9° RMSE
Robertson'06 [105]	Nearest neighbour	$20-40$ high	360°	A-E	None provided
Seeman'04 [108]	Neural Network	24×32	180° (NU)	D	$7.5° - 9.7°$ MAAE
Siriteerakul'10 [113]	SVM	20 sq	360°	B/E	8 classes, 78.6%/58.2%
Srinivasan'02 [116]	Dim. Reduction	60×48	180°	D	12.9° RMSE
Stiefelhagen'99 [120]	Neural Network	20×30	180° (NU)	D	9° MAAE
Tian'03 [122]	Neural network	16/32 sq	180°	A-E	9 classes, 79%/98% (2 cams)
Voit'06(1) [125]	Neural network	20×25	360° (NU)	A-D	8 classes, 15.8/39.4% (1/4 cams)
Voit'06(2) [126]	Neural Network	32 sq	360° (NU)	C	11.2° MAAE (4 cams)
C.Wang'00 [127]	SVM	-	360° (NU)	A-C	None provided
X.Wang'08 [128]	Dim. Reduction	32 sq	180° (U)	D	$2-4°$
Wei'02 [129]	Dim. Reduction	-	120° (U)	E	91% < 10° error
Wu'00 [134]	Ellipsoid Model	32 sq	$90° - 360°$(NU)	D	$19.2° - 47.5°$ MAAE with prior
Zabulis'09 [136]	Ellipsoid Model	-	360°	D	1.9-5.4% (8 cams)
Zhao'02 [137]	Neural Network	48 sq	180°	C	9° MAAE

Validity	Description
A	The testing and training datasets contain the same images
B	Same videos used, but no images appear in both datasets
C	Same people and scenes but separate videos for each dataset
D	Same people but different scenes for each dataset
E	Different people and different scenes in each dataset

Table 2.2: The five levels used to rate the validity of experiments due to correlations between training and testing datasets for the publications in table 2.1. A range is specified in some cases where insufficient details on datasets were provided. Note that only E represents a realistic scenario.

2.2 Human Performance

As with most vision problems, it is interesting to compare performance with the ability of a human. Some existing work has included studies on the ability of humans to classify head images according to the gaze direction. Gourier et al. [42] performed tests which gave a MAAE of 11.8° measured across 72 different people. This result was consistent with the 10.9° MAAE that was measured by Aghajanian and Prince using two human test subjects.

2.3 Applications of Coarse Gaze Estimates

Once the gaze direction of an individual has been determined, it must be used to provide information that is useful to humans or to an automated reasoning system. This section covers research which attempts to derive useful information from the gaze direction of one or more people.

A large amount of research into head pose estimation has been intended for the purpose of tracking attention in meetings to allow cameras to be automatically directed at the speaker for video conferencing. Stiefelhagen [118] measured both head pose and eye gaze simultaneously for such a purpose and deduced that with four speakers, the focus of attention could be identified from head pose alone 87% of the time. The research demonstrated that the gaze direction is highly correlated with

the direction in which a person is facing, a promising result for surveillance applications. Sherrah et al. [111] use both head pose and action recognition as inputs to an attention tracking system with the aim of identifying the speaker in a simple meeting scenario involving three people. Gaze directions and recognised actions were used as input to a time-delay radial basis function neural network to identify the current speaker. Ba and Odobez [89, 5] showed that gaze estimation accuracy in meeting scenarios could be improved by modelling the interactions between contextual information such as the locations of people and whether or not people were currently participating in the conversation.

Another common application is in the context of driver monitoring [84, 95, 26, 43]. The two main motivations are to detect driver fatigue and to monitor where the driver is looking so that a warning can be given if potential hazards develop outside of their field of view. An individual's gaze direction also has potential for use in more general control applications; Wei's [129] intended application was to control automated wheelchairs.

Some more relevant applications are those where the gaze is estimated for freely moving people. Both Farenzena et al. [76] and Robertson et al. [106] used gaze estimations to automatically identify interactions between people. Individuals were considered to be interacting if they were looking at one another and were in close proximity.

Robertson et al. [104] also developed a surveillance system based on a combination of gaze direction, recognised actions, and the person's location and direction of motion. A database of this information was collected from a training sequence and used to classify behaviour in subsequent video by identifying primitive behaviour types. The behaviour types were learnt from manually labelled sequences in which scene locations and different types of behaviour had been identified. Specific sequences of low level behaviour which constituted high level behaviour were detected using a Hidden Markov Model, for example the primitive actions of *walking on a*

pavement followed by *walking on a road* and then *walking on a pavement* again would cause the detection of the high level action *crossing a road*.

A system for identifying the path of a customer's gaze across shelves in a shop was developed by Liu et al. [74]. In this work a head model was fitted to high resolution video of a single customer. A related application was proposed by Haritaoglu and Flickner [44], who used measurements such as the gaze direction of observers to customise the advertisements that were displayed on video screens. The intended application of Smith et al. [114] was also in the domain of advertising, however in this case the purpose was to automatically calculate the number of times that advertisements were viewed.

A recent application by Patron et al. [97] used the head orientation in combination with local motion cues to recognise specific interactions (e.g. handshakes) across a wide variety of video sequences. Including the gaze direction improved the classification accuracy because people usually look at one another when they interact.

2.4 Head Tracking

Having reviewed the different methods that have been used for estimating gaze directions from head images, we now consider methods that could be used to locate the head images within surveillance video.

General multi-target tracking is an old problem in computer vision and the relevant literature is vast. In this section only methods that have been used for head tracking in the past and recent methods that are of particular relevance will be reviewed.

2.4.1 Existing Methods for Head Tracking

In existing literature on coarse gaze estimation, in most cases the head locations were hand labelled [34, 8, 6, 116, 134, 126, 125]. In other cases the head was tracked but it was assumed that the head moved very little throughout the video sequence [100, 95]. Although these methods appear to give reasonable results, they were all only applied to very small datasets and it is either stated or seems likely that images of the test subjects were used to train them.

Traditional methods for pedestrian tracking rely on background subtraction. Some gaze estimation systems used background subtraction followed by silhouette analysis to locate pedestrian's heads. Robertson [105] simply assumed that the top one-seventh of the silhouette represented the head, whereas Tian [122] used a model of the silhouette outline to locate the extremity corresponding to the head. Zabulis [136] found the head by fitting a sphere to silhouettes from eight cameras simultaneously.

Another popular tracking algorithm is mean-shift [28], which was used in other work by Robertson et al. [106], however this required manual initialisation for each pedestrian so would have to be combined with a detection algorithm for initialisation.

A method which combined background subtraction with Histogram of Oriented Gradient (HOG) based detections was used by Launila [62]. Moving regions were identified by running an unspecified tracker in a corresponding overhead view, from which location estimates were converted to the side view using a homography. A HOG [29] detector was then used to localise the head[1]. A Kalman filter was used to smooth location estimates, which were then further refined by fitting ellipses to the corresponding images with backgrounds subtracted. Finally, the colours in each image were clustered and any clusters that were identified as background were removed.

[1]This is similar to the approach developed in chapter 4, however it should be noted that the work in chapter 4 pre-dates Launila's

Lee [64] used background subtraction to separate people from a cluttered background and located the heads using a simple model of a head and shoulders which searched for a specific configuration of oriented edges around the silhouettes. Unfortunately a number of assumptions are made and there is no evidence of substantial evaluation which indicates that the method is unreliable.

The most sophisticated tracking systems to be used for coarse gaze estimation are based around particle filtering. Lanz's multibody tracker [61], which was used by Farenzena et al. [76] models the interactions between the movements of individuals using a Markov Random Field (MRF) which allowed predictions to take into account the possibility of occlusions. Smith et al. [114] modelled the interactions between people in a similar way using a Dynamic Bayesian Network, but modelled both head and body locations jointly to provide more accurate head locations. A particle filter was also used by Ozturk et al. [92]. One recent paper of particular relevance, which again postdates the work in chapters 3 and 4, is Ali and Dailey's [2] system for tracking heads in dense crowds. Heads were detected using Viola and Jones' algorithm [124] and tracked using a particle filter.

Li et al. [69] combined the outputs from multiple detection and tracking algorithms using a particle filter to track the head location. The first detector was a tree structured detector that was trained using an algorithm derived from AdaBoost and used the same Haar-like features as Viola and Jones for branch tests. The second detector was a *colour spatiogram*, which is similar to a standard colour histogram but also includes spatial information. Each bin in a spatiogram stores not only the quantity of pixels in the corresponding colour range but also the mean and variance of their location, allowing regions containing the same colours as the object but in a different arrangement to be rejected. The third detector was a simple contour detector which modelled the head as an ellipse. Results from three data sets demonstrated a detection rate of 80%-90%.

Orozco et al. [91] used Dalal and Trigg's Histogram of Oriented Gradients (HOG)

detector [29] to find full and then upper body detections, followed by connected components analysis on the result of background subtraction to identify the head region. Gong et al. [40] improved on the method by learning a part-based model which was able to improve the accuracy of the head location estimates by learning separate models for pedestrians when viewed from different directions.

Seeman et al. [108] used Viola and Jones' face detector [124] to detect faces and initialise a skin colour model. In subsequent frames, skin coloured regions were identified and validated by ensuring that the size in pixels was consistent with the depth estimations from stereo images using a known camera calibration.

2.4.2 Potential Methods for Head Tracking

In order for a head pose estimation to be made, the head must first be located in every frame of video. This is a difficult task because although frontal views of faces contain a large amount of information and are easily identifiable in high resolution images, in low resolution images there is much less detail and the face is not visible at all when pedestrians are looking away from the camera. There has been little research into the specific task of identifying non-frontal views of heads, however the more general task of locating people has received much more attention.

There are two main groups of algorithms that can be used; those which detect objects in single frames and those which track either just the head or the entire person over multiple frames. Both groups cover broad areas of computer vision, so only the research that is most likely to be of relevance has been reviewed here. A broad review of methods for finding and tracking people with the aim of identifying their body pose can be found in the survey undertaken by Moeslund et al. [81].

Detection Algorithms

One of the most commonly used detection algorithms, in particular for face recognition, is the detector developed by Viola and Jones [124] and later improved by

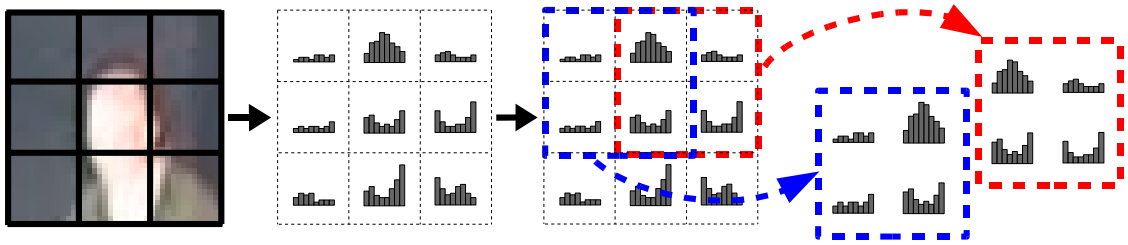


Figure 2.3: Illustration of how HOG descriptors are calculated. Images are divided into cells and the gradient direction and magnitude at every pixel is used to make a weighted vote into the corresponding orientation bin for the cell's histogram. Dalal and Triggs used nine orientation bins per histogram and considered the sign of the gradient to be irrelevant, so each bin represents a 20° range within the 180°. The orientation histograms are then separately normalised across every possible 2×2 block of cells and combined to give the complete descriptor. The result is that most cells (those not at the edges of the image) have their histograms included four times under different normalisations.

Lienhart and Maydt [72]. The classifier uses AdaBoost to find the features which are most useful for identifying a given object and combines their responses to form classifiers, which are then combined to make a cascade where each classifier can choose to either reject an image region or pass on the decision to the next classifier. A sliding window search is performed over the entire image, however this is fast enough for real-time applications because the features, which resemble Haar wavelets, can be calculated efficiently using an integral image. Krupper et al. [57] used a second classifier to detect upper body images which improved the accuracy of face detection by providing context-based priors.

More recently, Dalal and Triggs' Histogram of Oriented Gradients (HOG) based detector has proven to be extremely effective for pedestrian detection and it has also proven valuable for detecting other objects such as upper body regions [31]. The detection algorithm uses normalised local gradient histograms (see figure 2.3) as input to a binary SVM classifier. The local gradient histograms are calculated by dividing images into cells and generating gradient orientation histograms for each. The histograms are all normalised across blocks of four cells before being included in the descriptor. Objects are detected using a sliding window search over all possible

scales and image locations, which takes up to 30 seconds depending on the image size and scale range. Implementations for Graphics Processing Units (GPUs) have however recently become available and can reduce the processing time to less than one second.

Liebe et al. [65] developed the *Implicit Shape Model* (ISM). The method recognised objects using a codebook to map local features to corresponding offsets from the object centre. To detect objects, local features in the image were detected and the codebook was used to generate votes for the object location. Although the approach appears robust, it is not clear how dependent it is on the high resolution images that were used. The method also requires time in the order of minutes to process a single image, so is unlikely to be fast enough for real-time tracking.

Tracking Algorithms

Most recent work on multiple target tracking has focused on appearance based methods, which can be divided into two broad categories.

The first group covers feed-forward systems which use only current and past observations to estimate the current state. A significant recent publication in this category is the work of Breitenstein et al. [18]. The method involved applying particle filter tracking to the continuous likelihood output from a densely applied object detector such as HOG or ISM.

The second group covers data association based methods which also use future information to estimate the current state, allowing ambiguities to be more easily resolved at the cost of increased latency. This involves analysing observations either by processing an entire video sequence as a batch or by operating on a sliding window covering a short period of data (e.g. the most recent ten seconds). Most approaches involve grouping observations into tracks corresponding to individual people by using an affinity function to measure the likelihood of observations being from the same track.

Wu and Nevatia [133] used a method which performed data association on detections but could also use mean-shift tracking when detections were missed. The correct data associations were based on the similarity of location, size and appearance, and detected trajectories could be grown both forwards and backwards in time.

Huang and Nevatia [50] performed data association using different methods first at a low level using information such as the target size and appearance, then at a mid level where motion information was included, and finally at a high level using knowledge of the entire scene such as entry and exit points. The approach was extended by Li et al. [70] by learning a boosted affinity classifier for identifying related observations. The learning algorithm interleaved the identification of likely tracks with the optimisation of the affinity classifier to obtain iterative improvements.

Since tracks and genuine detections cannot exist without one another, Leibe et al. [66] developed a method for identifying both simultaneously by optimising over pedestrian detections and track likelihoods jointly. This allowed weak detections to be accepted if there was sufficient support at the trajectory level.

Some alternative approaches work by identifying paths rather than linking up observations. The approach of Berclaz et al. [9] reasoned about gaps in observations by modelling the movements of observations as flow in a graph, with constraints to prevent multiple objects occupying the same space and to ensure that objects could only enter or exit the scene from particular locations. Stalder et al. [117] started with a dense space-time volume of detector confidence scores similar to Breitenstein's. Various filters were applied to enforce requirements such as geometric consistency and trajectory smoothness before the final step generated trajectories using a particle filter.

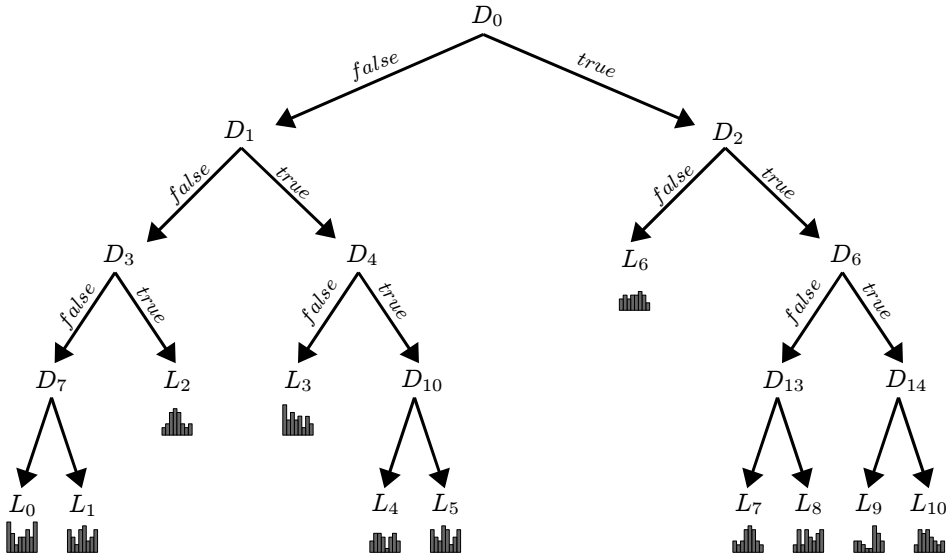


Figure 2.4: A randomised tree with depth 4. The chosen branches depend on the outcomes of different branch tests (D_i) and determine the leaf node (L_j) that is reached. Each leaf node has a histogram representing the amount of data from each class that reached the leaf.

2.5 Randomised Trees and Ferns

Randomised trees and ferns are used extensively throughout this thesis to classify images into discrete classes corresponding to different gaze directions. Randomised fern classifiers are a specialisation of randomised tree classifiers and are used in chapters 3 and 4, and a variation on standard decision trees is used in chapter 6.

Although randomised trees and ferns are relatively simple, they have a number of advantages over other popular types of classifier. One of the key advantages is that randomised trees and ferns easily generalise to work with any number of classes. Other classifiers such as SVMs require a number of binary classifiers to be trained to achieve multi-class classification.

A second advantage is that trees and ferns can output a probability distribution over classes, whereas SVMs and ANNs provide output for each class that is only expected to be a monotonic function of the probability. Probability distributions are useful because they are easily integrated into temporal models, which can be used to overcome classification errors.

2.5.1 Randomised Trees

Decision tree classifiers are based on tree data structures, which consist of a hierarchy of branch nodes and leaf nodes, as illustrated in figure 2.4. Branch nodes perform a test on the input data and use the result to determine which of the child nodes below it in the hierarchy should be examined next. Eventually a leaf node will be reached, which contains a histogram representing the probability estimates of the input data being an instance of each class given the outcomes of the branch tests. Trees are usually implemented with each branch node having two child nodes, since this allows a simple and efficient implementation.

To train a tree, a large quantity of labelled training examples are passed down the tree until they reach a leaf node corresponding to the outcome of the encountered branches, where they are added to the histogram bin corresponding to their class.

Decision tree classifiers originate from sets of hand-selected rules that were manually interpreted for purposes such as medical diagnosis. Early work such as that of Breiman et al. [17] developed methods for automating the process. Randomised trees are a specific type of decision tree where the training process has a random element to ensure that there is some variation in the structure of the tree each time it is trained.

2.5.2 Branch Selection

In order to achieve fast lookups with decision trees, it is desirable to make each branch as informative as possible so that fewer branches are required. To obtain the minimal mean number of evaluated branches requires branch tests where both outcomes are equally likely.

An approach that is commonly used for training randomised trees is to choose the branch tests which result in the greatest information gain for the examples reaching the branch [14, 68, 3]. Many other approaches have also been tried such as training an SVM at each branch or using clustering to find a separating hyperplane, however

it has been shown that the optimal method is problem specific [47].

2.5.3 Forests of Trees

Classification can be performed with a single randomised tree, but greater accuracy is usually achieved by combining the output from a number of trees, which are collectively called a *forest*. A single tree cannot be grown indefinitely because eventually there will be no data left to populate the leaves. Different combinations of branch tests are more informative for some examples than others, so having multiple trees increases the chance of informative branch tests being applied to every example. Forests were simultaneously proposed for randomised trees by both Ho [46] and by Breiman [15, 16].

Both Ho and Breiman suggested averaging the output distributions across all of the trees in a forest to provide an improved estimate. If we represent the class using the random variable c and the information from tree f of n as D^f then this amounts to the following:

$$P(c|D^1 \dots D^n) = \frac{1}{n} \sum_{f=1}^{n} P(c|D^f) \tag{2.1}$$

The original training methods for decision trees were deterministic, which would make all of the trees identical and yield no benefit over a single tree. To reduce the correlation between the estimates from individual trees within a forest, Ho proposed using a randomly chosen subset or *subspace* [47] of branch tests to train each tree and Breiman proposed *bootstrap aggregation* (shortened to just *bagging*) in which each tree is trained with a randomly chosen subset of the training data. A comparison of both methods [48] has shown that the two approaches result in similar performance.

2.5.4 Randomised Ferns

In their work on randomised trees, Lepetit and Fua observed that the training time for randomised trees could be significantly reduced if the branch tests were chosen

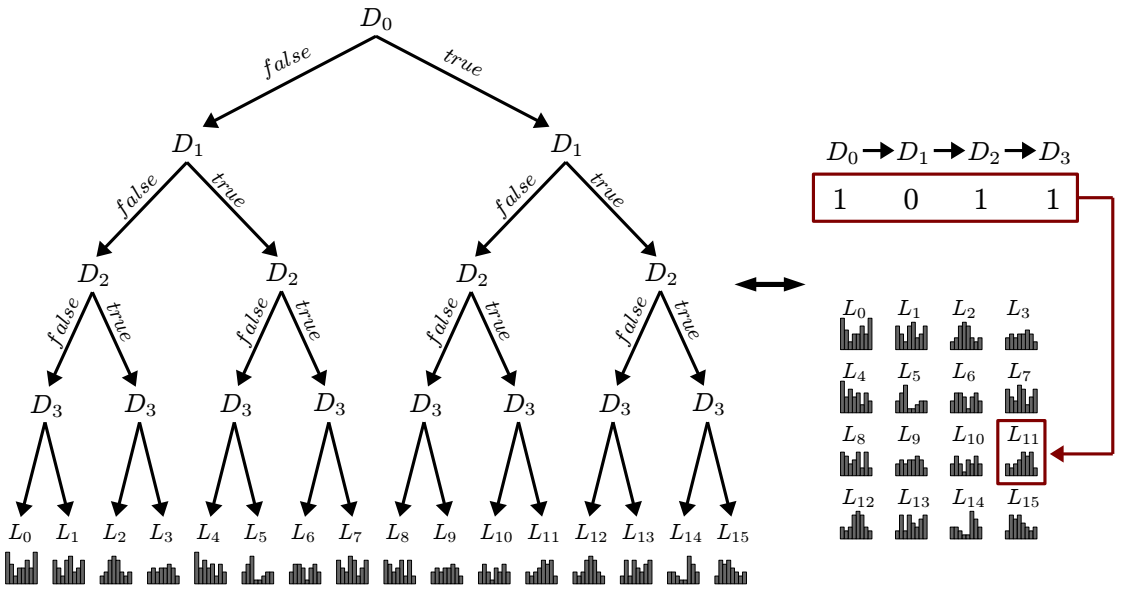


Figure 2.5: On the left is a representation of a fern as a type of decision tree where the same branch tests are made at each level regardless of previous branch outcomes and where all leaf nodes are at the same depth. In practise there is no need for the tree structure, since the same tests are always applied. This allows an efficient implementation, as shown on the right, where the branch tests are all applied immediately. The results are then combined to form an index allowing the leaf data to be found.

entirely at random, with the only negative effect being a small drop in classification accuracy [68]. Soon after it was noted that there was nothing to be lost by making branch tests identical at every level in a randomised tree if they were chosen at random, which resulted in the development of randomised ferns [94, 131].

Randomised ferns are equivalent to decision trees where the tests for branches at equal depths are the same. The suggested method for constructing ferns is to select branch tests at random and then populate the leaf histograms using examples for which the correct class is already known in the same way as for randomised trees. The constrained structure of a randomised fern allows the tree structure to be replaced with a set of tests for which the outcomes specify an index into the leaf data, as illustrated in figure 2.5. When the leaf index is represented in binary, each bit corresponds to the result of evaluating one of the individual tests.

The main advantage of randomised ferns over randomised trees comes from the reduced computational requirements. In a randomised tree, each branch test must be evaluated before the next branch test can be located in the computer's memory, so to reach a leaf at depth d requires d unpredictable memory accesses. Unpredictable memory accesses usually miss the processor cache so introduce a delay typically in the order of 100 clock cycles, making them the bottleneck for performance. In a randomised fern, the same d tests are always applied, so only one unpredictable memory access is required for the final leaf data. Although the structure and training process generally result in ferns being less accurate than trees, such an efficient implementation allows significantly faster training and classification. An additional advantage of ferns is exploited in chapter 3.

A randomised fern requires only d branch tests to be stored, compared to $2^d - 1$ for a tree, however the space required for the leaf data is generally larger for a fern since all 2^d possibilities must be stored. This limits the maximum depth due to the required memory capacity. To overcome this problem, an ensemble of ferns can be used, with the output of all the ferns being combined to provide an

improved estimation. An ensemble of n ferns with depth d allows a total of nd branch tests to be used, but with the need to store only $2^d n$ possible outcomes rather than the 2^{dn} outcomes that would be required if the tests were all in a single fern. The disadvantage of using an ensemble of ferns is that the outcomes each depend on a subset of the branch tests rather than all of them jointly, but this is usually outweighed by the benefit of being able to use more branch tests than would otherwise be possible.

Individual randomised ferns can be considered as a type of nearest neighbour classifier, since the histograms at each leaf consist of all the images that are considered to be identical given the branch outcomes required to reach that leaf. When multiple ferns are used, the behaviour differs depending on how the individual estimations are combined.

2.5.5 Forests of Ferns

As with trees, the combined estimates from a forest of ferns is generally used rather than that of just a single fern, however the approach that was suggested by Ozuysal differs to that for randomised trees. Ozuysal takes the product of output distributions, which assumes independence between the estimates from each fern:

$$P(c|D^1 \dots D^n) = \frac{\prod_{f=1}^{n} P(c|D^f)}{\sum_{c_i} \prod_{f=1}^{n} P(c_i|D^f)} \tag{2.2}$$

Some recent performance tests [93] in the domain of corner matching demonstrate significantly improved performance of this naive Bayes combination over averaging.

2.6 Datasets

The next four chapters examine algorithms both for estimating the gaze direction from cropped head regions and for tracking heads in video sequences to provide the cropped regions. The following sections provide details on the datasets used to test

the performance of these algorithms.

2.6.1 Video Datasets

A variety of video datasets are used both for evaluating the accuracy of tracking algorithms and to test the combined tracking and gaze estimation system. The most important properties of the video datasets are summarised in table 2.3.

Town Centre Video The Town Centre video covers a pedestrianised town centre street where approximately 10-30 people are visible at a time, all of whom are unaware that they are being filmed. Pedestrians frequently occlude one another and often walk in groups of two or more people. The video was filmed from a first storey window close to a genuine CCTV installation and includes events such as pigeons flying in front of the camera and transitions from direct sunlight illumination with shadows to diffused sunlight. The video was filmed at 1920×1080 resolution, with two short clips extracted for evaluation purposes. The first is a short clip lasting five minutes that was used to test the tracking performance in chapters 4 and 5. All of the head bounding boxes in the first three minutes of the clip were hand-labelled to provide a total of 71473 ground truth head regions covering 231 different people. The five minute video clip and the ground truth data have been made publicly available to be used by others for comparing performance. The second clip is an extended 22 minute version without ground truth that is used for the attention measurement system in section 4.4 and for extracting head image dataset C (see section 2.6.2). A full calibration relative to the ground plane was found using measurements of markers on the pavement.

i-Lids AB Easy Video The second dataset with ground truth is the i-Lids AB Easy video, which is a section of genuine security footage that has been released by the UK Home Office. The video covers a train station platform with relatively low quality 720×576 video that has been digitised from an analogue camera. The

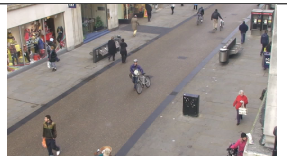
Video	Duration (hh:mm:ss)	Resolution	Frames per second	Ground Truth		Sample Frame
				People	Total Regions	
Town Centre	00:22:27	1920×1080	25	231	71473	
i-Lids AB Easy	00:03:39	720 × 576	25	16	9718	
PETS 2007	00:03:00	720 × 576	30	-	-	
Atrium	02:18:09	1024 × 768	15	-	-	
Hermes Crosswalk	00:01:47	1392×1040	15	-	-	
Transport Terminal	00:08:46	1920×1080	25	-	-	

Table 2.3: Datasets that were used for automatic tracking. Some video sequences include ground truth, others are used to provide qualitative results for the attention measurement system in chapter 4 or for automatically acquiring head image datasets.

ground truth annotations were full body bounding boxes that were made available by Stalder [117] and Breitenstein [18], who had previously used them for evaluating tracking systems, and covered 16 people with a total of 9718 bounding boxes. The dataset is used for evaluating the tracking accuracy in section 5.3.2. Ground plane measurements were not available, so an approximate calibration was found by projecting a ground plane grid with 1.7 metre long height markers and manually adjusting the calibration parameters until the ground plane and height markers were aligned with the floor tiles and heights of the pedestrians respectively.

PETS 2007 Video Although the PETS 2007 Video does not have ground truth, it is used in chapter 5 because it contains a higher density of people than the other datasets. The video is 720×576 resolution and has approximately thirty people visible at a time, many standing in groups where they are heavily occluded. The dataset was supplied with the full camera calibration relative to the ground plane.

Atrium Video The Atrium Video was filmed for the specific purpose of evaluating the attention measurement system in section 4.4. The video covers an indoor atrium where people frequently walk between doorways but rarely stand still within the visible area. The dataset is approximately two hours long and is divided into two halves. In the first half of the video the scene is observed normally, but in the second half an artificial stimulus was introduced as an attempt to attract the attention of the people in the scene. The stimulus consisted of a light attached to the wall at eye level with a short message in print large enough to be read by passing people without diverting their trajectories. Measurements of ground plane locations were used to calibrate the camera relative to the ground plane.

Hermes Crosswalk Video The Hermes Crosswalk video sequence is a staged clip involving three actors which was chosen as a third clip for testing the attention measurement system in section 4.4. During the clip, two of the actors watch a car

as it passes, providing a well defined focus of attention. An approximate calibration was found in the same way as for the i-Lids video by manually fitting a ground plane grid with height markers. The video has a resolution of 1392×1040 and covers a small area of ground, so provides high resolution images of the actors.

Transport Terminal Video The final video was filmed from a first storey balcony in an open space at a busy transport terminal. Most of the pedestrians are standing still, with the others crossing the scene from one side to the other. The video is 1920×1080 resolution, has no ground truth, and was calibrated using measurements of markers on the ground. The automatic tracking algorithm from chapter 5 is used in chapter 6 to extract head image dataset E from the video sequence.

2.6.2 Head Image Datasets

This section describes the head image datasets that were used to test the performance of the gaze direction estimation algorithms. Three of datasets were extracted from still images and video sequences using manually annotated head bounding boxes and the other three were extracted from the video datasets using automatic tracking. All of the datasets consist of cropped images which are stored in sequences corresponding to each of the people in a scene, where those from still images are simply sequences of length 1. The frame timestamps are stored with the images, as well additional information for some datasets. Ground truth labels were assigned to some or all of the images in the datasets by manually aligning an approximate 3D head model with the image and recording the resulting pan angle. The head image datasets have been given a label from A-F and the most important details for all six are summarised in table 2.4.

Dataset A The first dataset was sourced from a combination of images from the INRIA Person Dataset [29] and still photos which were obtained through web searches for crowd images. The head bounding boxes were hand labelled and used to

Dataset	Source	Acqusition Method	Image Count	Labelled Image Count	Sequence Count	Mean Images Per Sequence	Head Width (pixels)		
							Mean	Min	Max
A	Assorted web images and INRIA dataset	Hand Cropped	1475	1475	1475	1	37.9	9	157
B	Assorted video sequences	Hand Cropped	9260	9260	40	231	28.8	4	220
C	Town Centre video	Tracking (chapter 4)	473412	4347	2258	210	24.2	14	56
D	iLIDS ABTRA104a01 video	Tracking (chapter 4)	38514	1161	208	185	33.0	14	90
E	Transport Terminal video	Tracking (chapter 5)	639581	1866	3861	166	24.1	15	38
F	Assorted still web images	Hand Cropped	1244	1244	1244	1	28.4	10	156

Table 2.4: Datasets that were used for training and testing gaze direction estimation algorithms with their corresponding identifiers. A sequence represents the set of images from the same tracked target. The number of individual people is only and indication for the datasets obtained through tracking, since targets are sometimes lost and re-acquired and in other cases the tracker moves between people.

extract the individual head images. Unless otherwise stated, the extracted images include a border around the annotated head region so that they cover a region 1.2 times the width and height of the head bounding box. The original images were retained to allow head images to be extracted with larger or smaller borders, or with scale or location perturbations. Dataset A covers a wide range of appearances, so it used as the main dataset for the evaluation of gaze direction classifiers in chapter 4 and also for a baseline comparison in chapter 6.

Dataset B The head images in dataset B were hand labelled as with dataset A, but in a variety of surveillance dataset videos rather than still photos. Seven short video sequences from the Hermes [41] and Terrascope [53] datasets were selected to cover different lighting conditions and actors in both indoor and outdoor scenes. The resulting head image sequences are used to test the gaze direction classifiers in chapter 3 and in chapter 4 to compare the performance of classifiers without the inaccuracies of automatic tracking.

Dataset C Dataset C was obtained by applying the Kalman filter based tracking algorithm described in chapter 4 to the Town Centre video (see section 2.6.1). The resulting dataset consists of almost half a million images covering more than two thousand people who were each observed for an average of approximately ten seconds. In contrast to datasets A and B, the extracted images include cases where the tracker performs suboptimally, resulting in sequences of misaligned heads and some without heads at all. Every one hundredth image was hand labelled with a ground truth gaze direction, but images where less than half of the head was within the extracted region were marked as invalid and not used in the evaluations. The dataset is used for the evaluation of gaze direction classifiers in chapter 4 and in chapter 6 in combination with the tracker output for both learning and evaluation.

Dataset D The Kalman filter based tracking algorithm of chapter 4 was also applied to the i-Lids ABTRA104a01 video sequence to extract the head image sequences for dataset D. The video is from the same train station scene as the i-Lids AB Easy video (see section 2.6.1), but contains more people. The dataset contains a poor distribution of gaze directions, but the dataset was included to allow a comparison in chapter 4 with results from Orozco's system [91], which provided results from this video sequence.

Dataset E A second large dataset of more than six hundred thousand head images was collected from the Transport Terminal video (see section 2.6.1). The majority of images in this dataset were from people who were standing still, in contrast to dataset C which consists of images mostly from walking people. The images were extracted using the tracking system described in chapter 5, which is able to track a higher proportion of the stationary people than the one from chapter 4. The dataset includes the ground plane locations and velocities and is used for evaluating the gaze estimation system in chapter 6.

Dataset F The final dataset consists of still images that were manually cropped from web image searches for surveillance footage. The resulting images were mostly frames extracted from surveillance video, so were smaller and of lower clarity than those from dataset A. The dataset was primarily intended for training the gaze direction estimation algorithm in chapter 3, so the images had their pixel colours clustered into six groups, ignoring position information. The image regions covered by the six colour groups were manually assigned a label of either *hair*, *skin* or *background* depending the image region that they mostly covered. In chapter 3, the objective is to work with low resolution images so the images in dataset F were resized to 10 pixels square before use. The images were used at their full resolution for the manual annotation of ground truth and for the experiments in section 4.3.5 which use the dataset to compare different algorithms.

2.7 Conclusion

A considerable amount of past research has been directed at the problem of estimating head poses in low resolution video, however in almost all cases the conditions under which the systems have been developed are unrealistic in the context of visual surveillance. Of the 35 publications listed in Table 2.1, only eleven used images smaller than 30 pixels square, just ten attempted to measure the head around the full 360° range, and only four had evidence of proper separation between their testing and training datasets. This is in agreement with the conclusions of an independent review [83], where it was observed that no existing gaze estimation systems are suitable for practical use.

Systems for coarse gaze estimation have been developed for many different applications, however in most cases few people appear at a time. This is reflected in the methods used for tracking, which often rely on background subtraction, a technique which becomes increasingly unreliable with more people present. The most promising methods for head tracking use appearance-based detection.

CHAPTER 3

Low Resolution Colour Invariant Gaze Classification

We begin by attempting to estimate gaze directions for particularly low resolution images where the heads are approximately ten pixels in diameter. A novel approach based on randomised ferns is used to classify head images into classes according to their gaze direction without making any assumptions about skin, hair and background colours. Standard branch tests are replaced by predicates testing for the presence of abstract labels, which are inferred as part of the classification process. The research presented in this chapter was published at BMVC 2008.

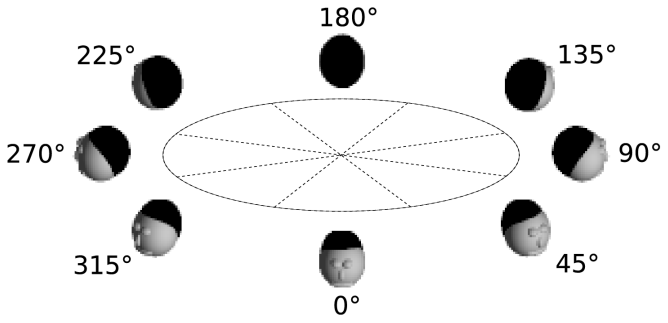


Figure 3.1: The eight head pose classes into which head images were classified based on the head pan angle. The angle was measured about the y-axis of the image and for heads with rotations about other axes pan was considered the first applied.

3.1 Introduction

It is often desirable for an automatic surveillance system to work with images where people appear as small as possible, since this allows each camera to monitor the largest area. This preference is the motivation behind the research described in this chapter.

Problem Statement: The problem addressed in this chapter is that of taking a single head image or sequence of head images as input and processing the images to produce estimates of the direction in which the person is facing. The source images are assumed to be already cropped to a region containing only the head with no border, and we would like the system to be capable of operating on images where the heads are as little as ten pixels in diameter.

The size of ten pixels was chosen because experimental evidence [123] exists showing that humans are capable of distinguishing faces from non-faces in images this small without seeing any of the surrounding image for context. In this chapter it is assumed that the bounding boxes around the heads in the video sequences are already known so the focus is on the problem of estimating the coarse gaze direction, however methods for automatically finding the bounding boxes are developed in chapters 4 and 5. The approach is based around a classifier which attempts to allocated each sample head image to one of eight discrete classes, each representing

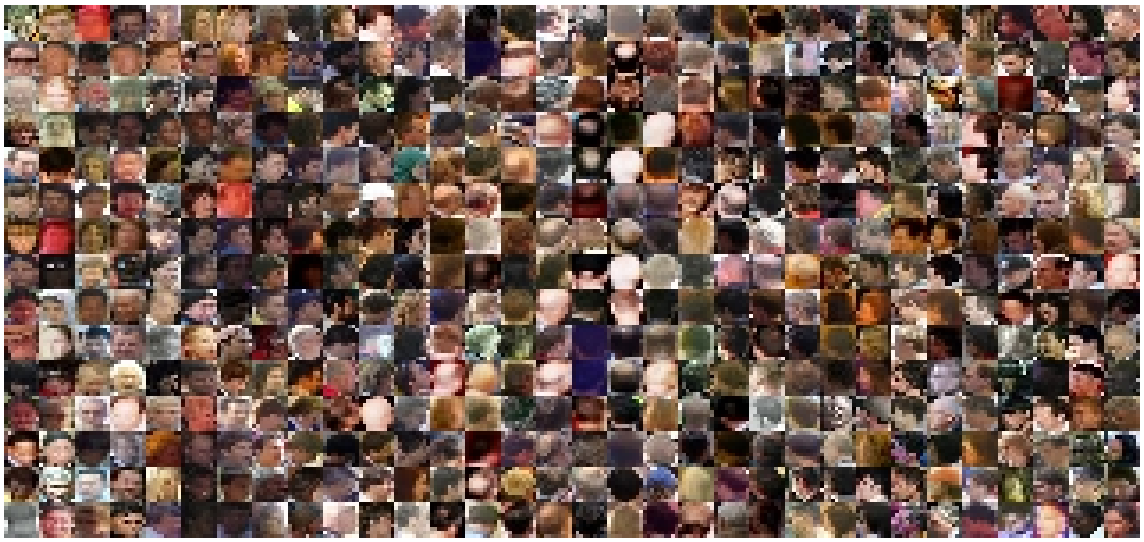


Figure 3.2: Samples of head images that were used to test head pose estimations. The hair colours of some individuals are the same as the skin colours of others which demonstrates the ineffectiveness of skin separation based on a fixed colour model.

a 45° range of gaze directions, which are illustrated in figure 3.1. The classification is based on the pan angle of the head, which is considered the be the rotation about the vertical axis in the image, however the classes are assumed to include images with rotations about the other axes. A particular side effect of the approach is to label every pixel in a sample head image as either *background*, *hair* or *skin*.

Some examples of cropped head images from the dataset that is used for training the classifier in this chapter (dataset F in section 2.6.2) are shown in figure 3.2. In these images the most important cues for the gaze direction come from the regions of the image that can be identified as hair and skin, since facial features such as the mouth and eyes usually cannot be seen. Many previous approaches to gaze estimation have relied on the identification of hair and skin using prior knowledge of what the colour distributions will be. In some systems such as Robertson and Reid's [105], the colour histograms were manually defined for each video sequence, which would be impractical for a genuine installation. Other approaches have attempted to build generic models of skin colours using one or more other images or videos that do not include the test dataset [100, 4, 120, 25]. These rely on the assumption

that the distribution of skin colours across all images is separable from that of the hair or background distributions, which figure 3.2 shows to be incorrect. There is an overlap between potential skin and hair colours so identification based on colour alone will often fail. The approach that we take attempts to resolve this issue by taking inspiration from the work of Ramanan et al. [101], in which a generic person model bootstrapped the online learning of a person-specific colour model for tracking.

This chapter describes a method for gaze estimation which is based on the identification of hair and skin regions, but works without making any prior assumptions about what the colour distributions should be. The approach makes use of randomised fern classifiers, but with a layer of abstraction introduced so that the ferns are used to estimate the probability distribution over the eight direction classes for an image where every pixel has an abstract label, rather than from the observed image directly. During the inference process, the gaze classification and abstract labels are both inferred simultaneously. This abstraction allows the system to cope with a wide variety of different skin and hair colours and also makes it largely invariant to different lighting conditions.

The output of the classifier is a probability distribution over the eight discrete direction classes, but as a side effect each pixel is also assigned one of the *hair*, *skin* or *background* labels. In addition to gaze direction estimates, the system is also able to learn the colour distributions for the three label types online for each individual person that is observed. These colour models are used to improve the accuracy of subsequent gaze estimates for sequences of head images that have been extracted from video.

3.2 Randomised Fern Structure

In a standard decision tree or fern, the outcomes of the tests at all of the branches are conditionally independent given the data upon which the tests are based. The proposed classifier is motivated by the idea that the branch tests could also depend on latent variables which are not known in advance, but which can be inferred during the classification process. In this case, the latent variable is a hypothesis which specifies which of the three logical labels *background*, *hair* or *skin* should be applied to different image regions.

To keep the labelling hypothesis reasonably simple, the pixels in each image were divided into six *segments* by using k-means clustering to find groups of similarly coloured pixels. The term *segment* will be used to refer to the set of image pixels that were assigned to the same colour cluster, which may consist of multiple disconnected components. Six segments were used to ensure that the hair and skin were separated in cases where lighting conditions caused their colour distributions to be multimodal. In the remainder of this chapter, the term *hypothesis* will be used to describe a set of constraints on which of the three labels could be correct for each of the six colour segments. A hypothesis can be uninformative, such as the empty hypothesis which allows any segment to have any label, or it could contain constraints on some segments but not others. A hypothesis that allows only one possible label for each segment will be referred to as a *complete hypothesis*, and a hypothesis that does not allow any label for a segment is considered impossible because it represents contradictory assumptions.

The branch tests in the randomised ferns each propose that the segment corresponding to a randomly chosen image location has a particular label. The outcome of the branch test is determined by truth of the proposition, which is tested using the image and the current hypothesis. These tests are equivalent to the predicates that are used in first order logic, and in this context the hypothesis can be seen as providing a mapping which unifies the segmented image with one of the examples

represented by the fern.

Unlike ordinary decision trees and ferns, a single image can reach a number of different leaf nodes, albeit under different hypotheses. There are two different ways in which these leaf nodes can be found; both produce the same results and each can be more efficient than the other depending on the size of the ferns and the complexity of the hypothesis.

The first method that was tried involves working with a set of hypotheses, initially containing just one empty hypothesis, and carrying out the predicate tests required by the fern one by one. When each test is applied, every hypothesis in the set is processed with two possible outcomes depending on how the segment corresponding to the tested pixel is constrained by the hypothesis. The first possibility is that the hypothesis doesn't constrain the segment enough to determine whether the outcome of the test should be true or false. Since both are possible, the hypothesis is replaced by two new hypotheses, one of which assumes that the test outcome should be true and the other assuming that the outcome should be false. Both hypotheses are equivalent to the old hypothesis with the constraints updated to reflect the implication of the corresponding assumption. The second possibility is that the hypothesis already includes enough constraints to determines the outcome of the predicate test, in which case it remains in the set unchanged. This process is illustrated in figure 3.3 for a simple fern containing four tests and with just three segments. The hierarchy shows how the set of possible hypotheses is updated after each test, with the constraints on the labels for each segment represented by the colour as indicated by the Venn diagram. Note that the diagram shows how the set of hypotheses is expanded and not a decision tree; the same sequence of tests are still applied as with a standard fern. When the first test is applied, both outcomes are possible so the hypothesis is split into two new hypotheses. The second test is applied to the same segment as the first, so for the hypothesis on the right would represent a contradiction if the outcome were true because the same segment cannot

be both skin and hair. The result is that the outcome is already determined, and the portion of the decision space represented by the red crosses cannot be reached by the example image.

The second approach is based in the observation that with six segments and three labels a maximum of 729 hypotheses can exist, which occurs when all hypotheses are complete. By simply evaluating every possibility in turn, the branch outcomes and leaf nodes corresponding to possible hypotheses can be identified more efficiently. For the particular problem in this chapter this second approach of exhaustive searching was used because it allows a more efficient implementation. The hypothesis splitting approach is likely to be more efficient for problems where the domain of the hypotheses is considerably larger, where it can be expected that few hypotheses will be complete after all of the tests in the fern have been applied.

Performing classification based on a logical representation has a number of advantages over a classifying based on directly measurable properties such as pixel brightness, the biggest of which is colour invariance. There is enough variation of hair and skin colours between people and under different lighting conditions to prevent them from being accurately distinguished by a simple comparison of pixel colours, so the branch tests in a standard fern do not allow as much of the important structural information to be tested as those in the abstracted equivalent. A second advantage is that labelled images provide a higher ratio of useful information to noise than the observed images, which helps to prevent over-fitting.

3.2.1 Training

The randomised fern classifiers in this chapter were trained using dataset F (see section 2.6.2), which consists of approximately 1000 head images that were collected from a wide variety of digital photos obtained from web searches. The images were cropped to include only the head region, resized to 10 pixels square, and manually assigned to one of the eight classes based on the head orientation. Each of the six

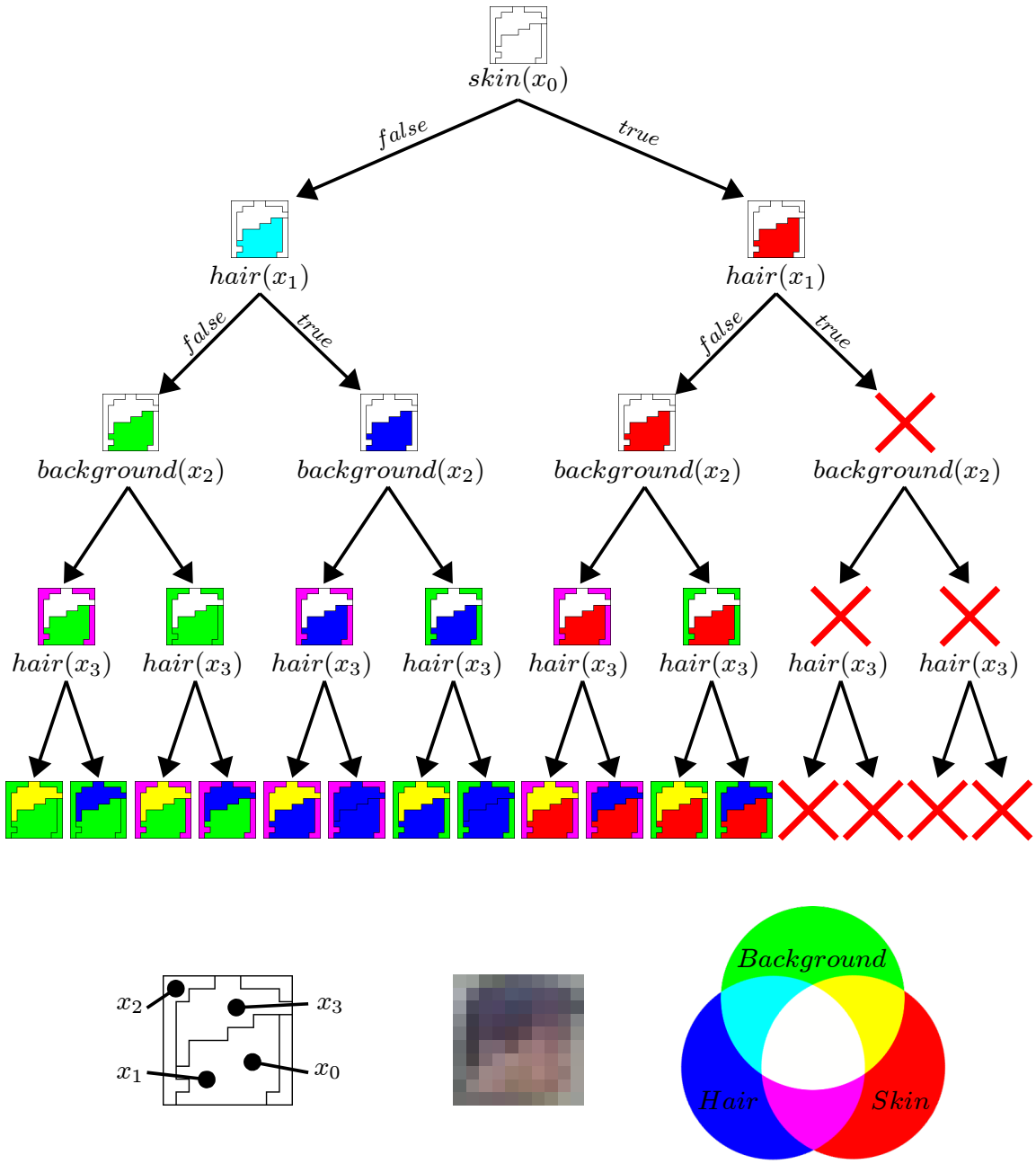


Figure 3.3: An example showing a simplified case where just three segments are labelled by a small predicate fern. The hierarchy shows how the set of hypotheses grows as each test is applied. Note that the same sequence of tests are still applied to every image as with standard ferns. Here the hypotheses are represented as the segmented image with the segments coloured according to the set of possible labels that the segment could have, as indicated by the Venn diagram. Large red crosses show branches that represent contradictions for this particular image. For larger ferns, a much smaller fraction of the leaf nodes would be reached because many more contradictions would occur.

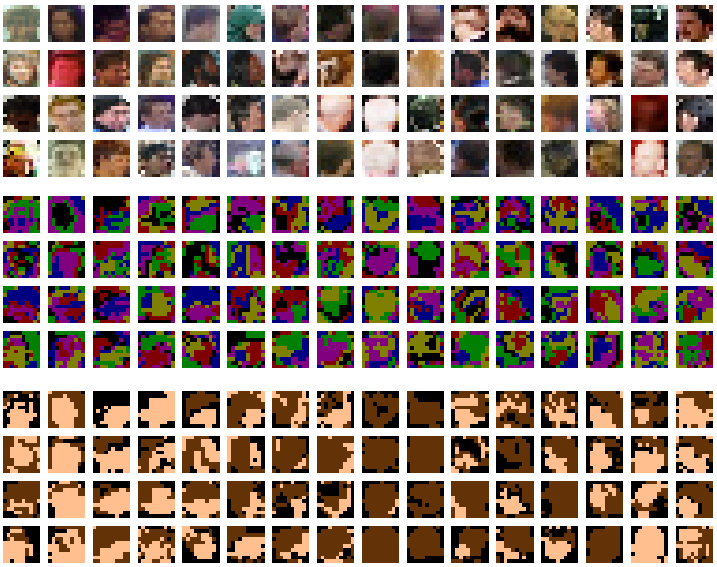


Figure 3.4: Sample images from the training data (top) with the 6 colour groups identified using k-means clustering (middle) and the hand labelled images (bottom). In the labelled images, dark brown represents hair, beige represents skin and black represents background.

segments obtained using k-means clustering were manually assigned one of the *hair*, *skin* or *background* labels. Examples from the training data and the corresponding segmentations and manually assigned labels are shown in figure 3.4.

In addition to a histogram of correctly labelled images, a second set of histograms were populated at each leaf using the training images under every possible incorrect labelling hypothesis. This second set of histograms represents the probability that an image reaching the leaf would be allocated to each class given that it has been labelled with an incorrect hypothesis.

3.2.2 Classification

Having identified the leaf nodes in the fern that can be reached by an image under all valid hypotheses, the next stage is to estimate the joint probability distribution $P(h, c)$ over hypotheses h and direction classes c. The sets of all complete hypotheses and all direction classes will be represented by H and C, so $h \in H$ and $c \in C$, and the outcomes of the branches in fern f from a total of n when the segmented image

is labelled using hypothesis h will be represented by d_h^f.

Each leaf node has a corresponding histogram representing the fraction of the total training images in each class that reached the leaf, which is used to estimate the probability $P(d_h^f|h)$ that any correctly labelled image will reach that leaf. Unfortunately, incorrectly labelled images are more likely to reach some leaf nodes than others due to structural information being lost when multiple segments are given the same label. Images with fewer boundaries between differently labelled regions contain less distinctive information, so are more likely to reach the same leaf node. An extreme example would be a hypothesis which labels all segments as background; under this hypothesis all images would reach the same leaf. To prevent bias towards these degenerate hypotheses, it is important to also take into account the probability $P(d_h^f|\overline{h})$ of a leaf being reached by incorrectly labelled images.

The joint probability given the branch outcomes for a single fern can be calculated by expanding Bayes' formula with the assumption that $P(h)$ and $P(c)$ are independent:

$$P(h, c|d_h^f) = \frac{P(d_h^f|h, c)P(c)P(h)}{\sum_{c_i \in C} P(d_h^f|h, c_i)P(c_i)P(h) + P(d_h^f|\overline{h}, c_i)P(c_i)P(\overline{h})} \tag{3.1}$$

The values of the conditional probabilities $P(d_h^f|h, c_i)$ and $P(d_h^f|\overline{h}, c_i)$ are represented in the histograms from the leaf corresponding to d_h^f.

As mentioned in section 2.5, the probability distributions from randomised ferns are usually multiplied to provide the overall combined distribution under the assumption that each fern will provide an independent estimate. In our case a single labelled image contains a maximum of $\log_2 3^{100}$ (≈ 158) bits of information whereas the ensemble of ferns makes a total of 320 binary tests, so there is likely to be a large amount of mutual information between the estimations from the ferns. This mutual information has the potential to cause the product of the estimations from the ferns to be severely biased. Combining the estimations by taking the mean over

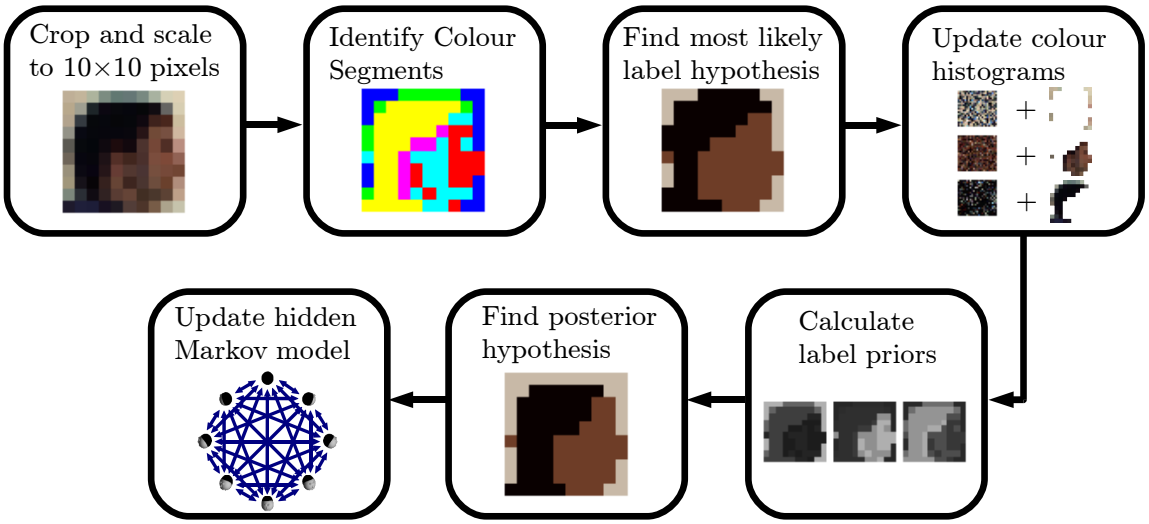


Figure 3.5: An overview of the classification process for a head image in a video sequence. The example image is incorrectly classified using the maximum likelihood hypothesis, but correctly classified when the learned colour distributions are used to find the best posterior hypothesis.

all ferns in the same way that one normally would for a forest of randomised trees avoids this problem. Marginalising and taking the mean over all n ferns provides an estimate of the probability that each hypothesis and class is correct for the image:

$$P(h|d_h^1 \dots d_h^n) = \frac{1}{n} \sum_{f=1}^{n} \sum_{c_i \in C} P(h, c_i | d_h^f) \tag{3.2}$$

$$P(c|D^1 \dots D^n) = \frac{1}{n} \sum_{f=1}^{n} \sum_{h \in H} P(h, c | d_h^f) \tag{3.3}$$

In these equations D^f represents the set all possible d_h^f for fern f.

This method for combining estimations is a direct transfer of the method for standard randomised forests to our predicate ferns and is the method that was used when the work was first published. Many other ways for combining the fern estimates were subsequently tested; these are described in section 3.5.

3.3 Classification in Video

When classifying a series of head images of the same person in video, the information that is learned from one frame can be combined with information that has been learnt from previous frames. An overview of the resulting classification process for each head image in a video sequence is shown in figure 3.5. The image is first processed by the ferns to obtain a maximum likelihood hypothesis, and the pixels colours covered by each of the labels in the hypotheses are added to cumulative histograms for those labels. The cumulative histograms are then used to calculate prior probabilities for each segment having each label based on the similarity to the previously observed colours for the labels. A new posterior hypothesis is then found, from which the distribution over classes is smoothed using a Hidden Markov Model (HMM) to provide the final class probabilities. The following two sections describe the colour model and HMM in more detail.

3.3.1 Learning Colour Distributions

The segment structure alone provides enough information to classify the head image successfully in most cases, however the classification accuracy can be improved by learning the colours represented by the labelled image segments. Although no assumptions are made about the hair and skin colours of the surveillance subject, it is assumed that they will stay reasonably constant while they are being observed. Small changes in the colour distributions might result from changes in illumination but are acceptable provided that the colour distributions do not overlap significantly.

The colour distributions for the labels are learnt by taking the hypothesis which maximises the probability in equation 3.2:

$$h_{max} = \underset{h}{\operatorname{argmax}}\, P(h|d_h^1 \ldots d_h^n) \tag{3.4}$$

In every frame, the pixel colours from each of the segments in the head image are

added to the histograms corresponding to their labels in h_{max}. The resulting colour histograms $Y = \{Y_{hair}, Y_{skin}, Y_{background}\}$ that are built up over all frames preceding the current one are used to generate a prior $P(h|Y)$ over hypotheses for the current frame. The prior is calculated from the probability of each pixel colour q in each segment s occurring given the accumulated colour histograms for the label that the hypothesis assigns to it. The occurrence probabilities are averaged over the pixels in each segment before the product is taken over all s in the set S of all six segments:

$$P(h|Y) \propto \prod_{s \in S} \sum_{q \in s} P(q|Y_{h(s)}) \tag{3.5}$$

In the above equation, the notation $Y_{h(s)}$ is used to represent the colour histogram corresponding to the label that h provides for segment s. The pixel colour probabilities $P(q|Y_{h(s)})$ are calculated using the distribution represented by the histogram (i.e. as a single density estimate from a multinomial distribution). Equation 3.1 is then adjusted to provide an improved estimation by taking into account the colour histograms Y:

$$P(h, c|d_h^f, Y) = \frac{P(d_h^f|h, c)P(c)P(h|Y)}{\sum_{c_i \in C} P(d_h^f|h, c_i)P(c_i)P(h|Y) + P(d_h^f|\overline{h}, c_i)P(c_i)P(\overline{h}|Y)} \tag{3.6}$$

Equations 3.2 and 3.3 are updated similarly to give improved estimates of the class and hypothesis probabilities.

The use of colour as a temporal constraint allows ambiguities to be resolved in situations where the segment structure alone does not provide enough information. An example of this is shown in figure 3.5 in which the most likely hypothesis for the segmented image gives an entirely plausible initial labelling which causes the image to be misclassified. When the colour priors are used, the improved posterior hypothesis provides more accurate labels which result in correct classification.

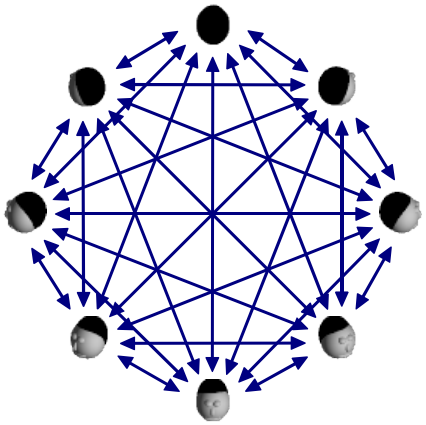


Figure 3.6: The HMM used to model gaze directions has eight states corresponding to the eight head direction classes. The state can change to any other state in one update, though large changes are considerably less likely.

3.3.2 Hidden Markov Model Filtering

The pose of a head at any time provides information relating to the pose shortly after due to the physical constraints of human head motion. This information is exploited through the use of a Hidden Markov Model (HMM) [107] to filter the head pose estimations.

Hidden Markov Models are used to model processes which transition between discrete states at regular time intervals, but for which the state cannot be directly measured. The probability distribution over states is stored and updated at regular time intervals when observations are made. Although the fern estimates for individual frames are often sufficient to estimate the current state of the system, the use of a HMM allows them to be filtered by using past observations to make estimations more consistent. HMMs can also be used to smooth data by taking into account both past and future observations when they are available, but due to the real-time requirements only filtering is used in this work.

Each of the eight head pose classes is modelled as an individual state in a HMM where any transition between states is possible, as shown in figure 3.6. Let c_t represent the distribution over direction classes and $\mathcal{D}_t$ represent the information $D^1 \dots D^n$ from the ferns at time t. The probability of transitioning from state i to state j is

estimated as a Gaussian function of the smallest positive angle a_{ij} between them:

$$P(c_{t+1}=j|c_t=i) \propto e^{-\frac{a_{ij}^2}{2\sigma^2}} \tag{3.7}$$

The smoothed distribution over classes at time $t+1$ is then estimated recursively as the product of the predicted and observed distributions:

$$P(c_{t+1}=j|\mathcal{D}_0 \ldots \mathcal{D}_{t+1}) \propto P(c_{t+1}|\mathcal{D}_{t+1}) \sum_{i \in C} P(c_{t+1}=j|c_t=i) P(c_t=i|\mathcal{D}_0 \ldots \mathcal{D}_t) \tag{3.8}$$

This can be written more compactly using matrix notation if the vector s_t represents the probability distribution of c_t over the discrete class states and the observation matrix O_t is a diagonal matrix with $P(c_t|\mathcal{D}_t)$ as the ith diagonal element:

$$s_{t+1} \propto O_{t+1} T^{\mathrm{T}} s_t \tag{3.9}$$

where

$$T_{ij} = P(c_t=j|c_{t-1}=i) \tag{3.10}$$

This model is however based on the assumption that the observations are all made at equal intervals which is often insufficient. Cameras operate at different frame rates and one cannot guarantee that every frame will be processed in a real-time system, so to cope with unequal observation intervals equation 3.9 was modified to allow an arbitrary observation interval Δt:

$$s_{t+\Delta t} \propto O_{t+\Delta t} (T^{\mathrm{T}})^{\Delta t} s_t \tag{3.11}$$

Since the transition probabilities are modelled using a Gaussian distribution, $T^{\Delta t}$ can be calculated simply by scaling the variance σ^2 in equation 3.7 according to the time interval:

$$\sigma^2 = k\Delta t \tag{3.12}$$

where k is a constant representing the standard deviation of the angular velocity over one second. The use of this model allows filtering to be performed when frames are missed and ensures that the constant k need not be adjusted for cameras with different frame rates.

In the absence of any empirical measurement, the value of 0.15° per millisecond was used for the head motion standard deviation.

3.4 Branch Decision Selection

The branch tests that are used in the randomised ferns determine how images are grouped together, so the method by which they are chosen is important. As mentioned in section 2.5.2, there are many ways to select branches, but many do not work well with more than two classes. One particular method, which involves maximising the information gained from each branch, is frequently used and has a natural generalisation to any number of classes. If a branch splits the training examples with distribution S at that node into K subsets with class distributions S_k, then the expected information gain from knowing the outcome of the branch is defined by the difference in entropy:

$$\Delta I = H(S) - \sum_{k=1}^{K} \frac{|S_k|}{|S|} H(S_k) \tag{3.13}$$

The entropy $H(S_k)$ is calculated from the class probabilities $p(c)$ defined by S_k:

$$H(S_k) = -\sum_{c=1}^{C} P(c) log_2 P(c) \tag{3.14}$$

Although this method for branch selection was intended for training trees, we apply the same principle to ferns by using the entropy over the distributions at all nodes for which the branch is chosen.

To encourage variation between different ferns, the branch tests were chosen

using the random subspace method by forcing each branch to choose between a small number of tests.

3.5 Combining Estimations

So far only one simple method for combining the estimations from each fern and hypothesis has been proposed. A number of alternative approaches were also tested; these were motivated by two different observations. The first was that the mutual information between the ferns was sufficiently large to introduce bias when estimations were combined using a naive Bayes approach, but small enough so that averaging did not take advantage of the available information. The application of Chow-Liu trees was tested in an attempt to address this issue. The second observation was that the consistency of the labelling hypothesis across all ferns was not enforced by equation 3.3.

3.5.1 Chow-Liu Trees

In section 3.2.2 the combination of fern outputs by taking the mean was justified by the high probability of mutual information between ferns. Each fern uses a subset of the information in the head image, but often the same branch test is used in more than one fern. Even if there is no overlap between the pixels tested by two ferns it is still likely that they will be correlated because close pixels tend to have similar colours. Ideally, one would learn the full joint distribution and calculate the combined estimation using a series of conditional probabilities to prevent the combined estimation from being biased by the mutual information. This would require equation 3.3 to be replaced by a conjunction of conditional probabilities:

$$P(c|D^1 \dots D^n) \propto P(c)P(D^1|c)P(D^2|D^1, c) \dots P(D^n|D^1 \dots D^{n-1}, c) \qquad (3.15)$$

The problem is that to use this method would require an unfeasibly large amount of training data to learn the values for the higher order conditional probabilities. Chow-Liu trees provide a mid-point between the Naive Bayes estimation and the estimation using the full joint distribution by making a second order approximation:

$$P(c|D^1 \dots D^n) \propto P(c)P(D^1|c)P(D^2|D^1, c) \dots P(D^n|D^{n-1}, c) \tag{3.16}$$

This type of classifier is known as a Tree Augmented Naive Bayes (TAN) classifier and details of how the terms are calculated can be found in the original paper by Friedman et al. [33]. The approximation considers each random variable to be dependent on just one other random variable, however the choice of the other random variable has a significant effect on the performance. In their paper Chow and Liu [27] propose the optimal solution to be that which chooses dependent variables to be those with the largest amount of mutual information. Mutual information is a measure of how well the values of two random variables can be predicted from one another:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{3.17}$$

Chow and Liu suggested a method for maximising the mutual information which involves constructing a fully connected graph where vertices represent random variables and edges represent mutual information between variables. An example of the mutual information matrix and spanning tree for an forest of ten ferns is shown in figure 3.7. The optimal second order approximation is the spanning tree which maximises the edge weights. The maximal spanning tree was found using Kruskal's algorithm, which simply involves repeatedly adding the maximal edge that would not form a cycle to the tree, repeating until all vertices are connected.

Due to the large domain of the branch outcomes d_h^f, the mutual information was approximated by that of the corresponding leaf histograms to minimise the amount of training data that was required.

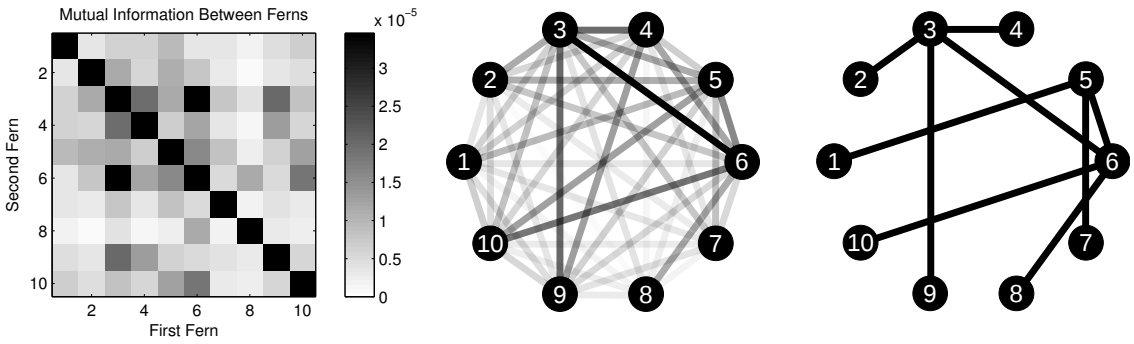


Figure 3.7: The mutual information matrix (left) for an example with ten ferns. The two graphs represent the mutual information between ferns (middle) and the corresponding Chow-Liu tree (right). The mutual information matrix and graph share the same scale, which is measured in bits.

3.5.2 Combination Methods

A total of seven different methods for combining the probabilities across ferns and hypotheses were tested.

Method 1: Simple Averaging A minor modification of the standard method for aggregating results from randomised trees:

$$P(c|D^1 \dots D^n) = \frac{1}{n} \sum_{f=1}^{n} \sum_{h \in H} P(h, c|d_h^f) \tag{3.18}$$

Method 2: Voting In other classifiers such as multi-class SVMs, voting is often used to combine multiple estimations. Although it is not directly based on any probabilistic model, voting is useful as a comparison because it is robust to monotonic transformations of the class distributions.

$$P(c|D^1 \dots D^n) \propto \sum_{f=1}^{n} \delta_{c,c_{max}} \tag{3.19}$$

where

$$c_{max} = \underset{c}{\operatorname{argmax}} \sum_{h \in H} P(h, c|d_h^f)$$

Here $\delta_{i,j}$ represents the Kronecker delta, which is 0 except for when $i = j$ in which case it has the value 1.

Method 3: Naive Bayes Another simple method is the Naive Bayes classifier, which has been previously used to combine estimations from randomised ferns. This approach assumes independence between the estimates from each tree:

$$P(c|D^1 \dots D^n) \propto \prod_{f=1}^{n} \sum_{h \in H} P(h, c|d_h^f) \tag{3.20}$$

Method 4: Chow-Liu Tree Combination The next method uses Chow-Liu trees to model single dependencies between observations as a TAN classifier:

$$P(c|D^1 \dots D^n) \propto \left(\sum_{h \in H} P(h, c) \right) \left(\sum_{h \in H} P(d_h^1|h, c) \right) \prod_{i=2}^{n} \sum_{h \in H} P(d_h^i|d_h^{\Phi(i)}, h, c) \tag{3.21}$$

In this equation $\Phi(i)$ represents the parent of observation i in the learned Chow-Liu tree and fern 1 is assumed to be at the top of the tree.

These first four methods marginalise over hypotheses before combining across ferns. Although this is analogous to the method by which estimates are combined across standard trees, our predicate trees can take advantage of the labelling hypothesis to ensure that the ferns produce a consistent labelling of the image across all of the ferns.

Method 5: Consistent Averaging The standard averaging method in equation 3.18 results in the class distributions from each fern being weighted according to the individual hypothesis probability from that fern. An alternative is to assume independence between the ability of a fern to estimate the class and the hypothesis probabilities for an image. Under this assumption, both are separated before being

aggregated across ferns and then combined again:

$$P(c|D^1 \dots D^n) \propto \sum_{h \in H} \left(\sum_{f=1}^{n} P(c|h, d_h^f) \right) \left(\sum_{f=1}^{n} P(h|d_h^f) \right) \tag{3.22}$$

where

$$P(h|d_h^f) = \sum_{c \in C} P(h, c|d_h^f)$$

and

$$P(c|h, d_h^f) = \frac{P(h, c|d_h^f)}{P(h|d_h^f)}$$

Method 6: Consistent Naive Bayes For the Naive Bayes classifier, reversing the sum and product enforces the consistency of the hypothesis across ferns:

$$P(c|D^1 \dots D^n) \propto \sum_{h \in H} \prod_{f=1}^{n} P(h, c|d_h^f) \tag{3.23}$$

Method 7: Consistent Chow-Liu The hypothesis consistency is also enforced for the Chow-Liu tree approach by swapping the sum and product:

$$P(c|D^1 \dots D^n) \propto \sum_{h \in H} P(h, c) P(d_h^1|h, c) \prod_{i=2}^{n} P(d_h^i|d_h^{\Phi(i)}, h, c) \tag{3.24}$$

3.6 Evaluation

The method was evaluated by training classifiers using head image dataset F and measuring the estimated gaze direction accuracy on head image dataset B, which consists of a total of 9260 head images from a set of videos including sequences from the Hermes and Terrascope datasets [41, 53]. The test videos covered a wide variety of different lighting conditions and included images from sixteen actors with different hair and skin colours, none of whom appeared in any of the training data. Before evaluation, the head images from the dataset were scaled to 10 pixels square to simulate low resolution video. Sample frames from the videos are shown in figure

Figure 3.8: Sample frames from three video sequences with estimated head pose angles annotated. Although these frames are shown in high resolution, the head images were scaled to ten pixels square before classification. In the bottom sequence, the actor on the right has only hair visible for the majority of the video which prevents the colour distributions from being learnt correctly and results in large errors.

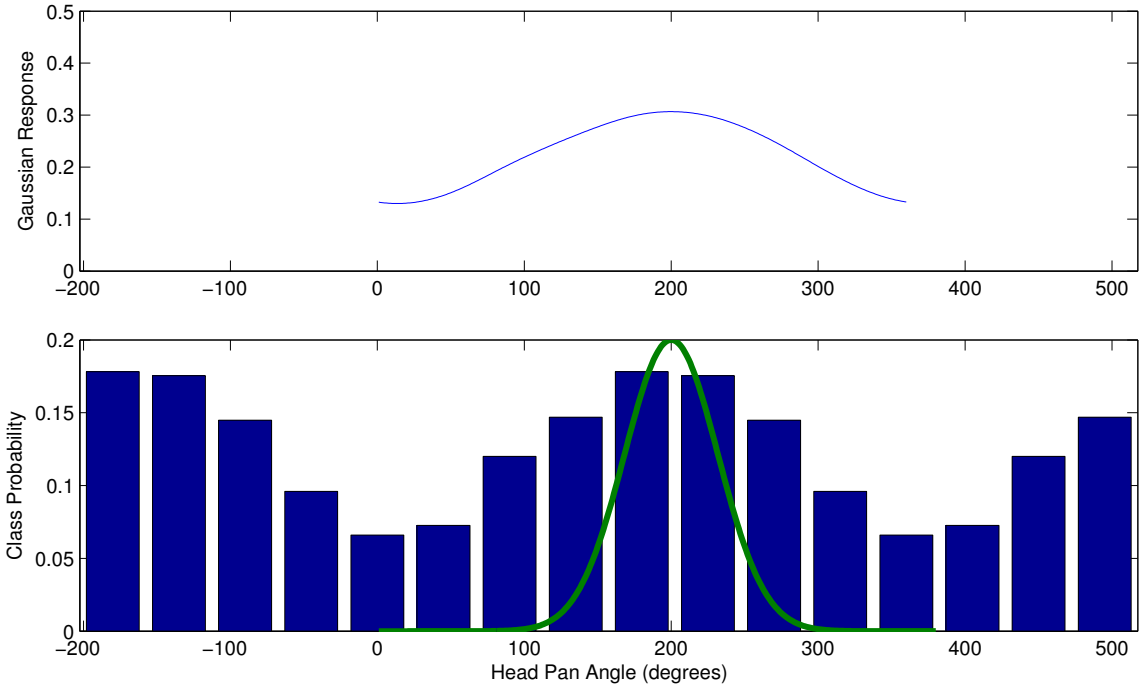


Figure 3.9: A graph demonstrating the calculation of a probability density estimate through the application of a Gaussian Parzen window to the discrete class probabilities (top) and the corresponding class probability histogram (bottom) with the Gaussian located at the point where the convolution gave the largest response.

3.8. Ground truth for each head image was hand labelled using high resolution frames, so the angular errors calculated using it are likely to be overestimates due to human error.

In past work on gaze direction classification, the two most popular performance measures have been the percentage of images for which the correct class was estimated and the Mean Absolute Angular Error (MAAE). The MAAE was chosen for use because it does not depend on the number of classes and also represents a preference for incorrect classes to be close to the correct class, which is particularly important if test images represent directions that are close to a class boundary. The MAAE results in this section are the mean of ten repetitions for each experiment.

The use of only eight classes results in the pan angle estimations being severely quantised, so a more accurate pan angle estimation was found by taking a Parzen window density estimate across the eight classes, as illustrated in figure 3.9. The

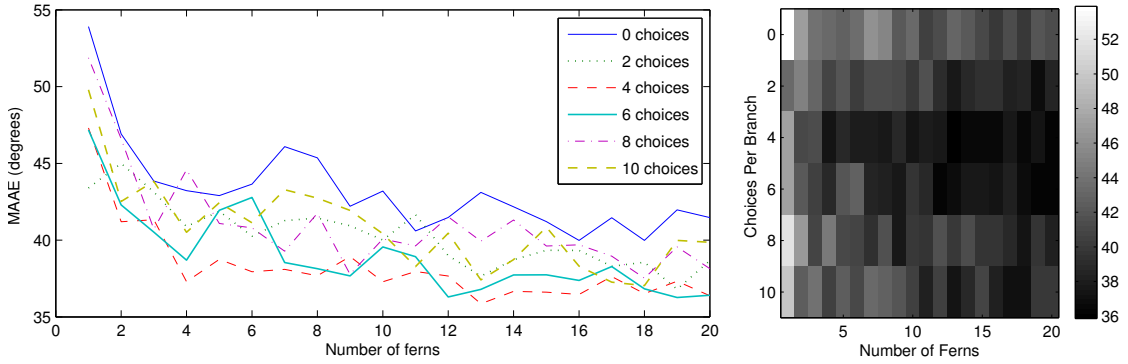


Figure 3.10: Graphs showing the effect of varying the number of choices per branch on the estimation accuracy. The values plotted are the mean of ten repetitions of each experiment. The graph on the right suggests that the MAAE (measured in degrees) is lowest when a choice of between four and six different tests is allowed for each branch.

Parzen window kernel was a normal distribution with a 45° standard deviation and the window was wrapped around the estimated distribution for 180° on either side of the centre, by which point the density was negligible.

A number of different experiments were performed to test the effect of different parameters on the classification accuracy. Unless otherwise stated, experiments were performed using twenty ferns with sixteen branches each, branch tests were selected entirely at random, and fern estimates were combined by taking the mean (method 1). Temporal information from the colour histogram learning and the HMM were also used.

Maximal Information Gain Branch Selection

Some experiments were performed to compare the performance of the ferns with branches selected randomly and using the MIG (Maximal Information Gain) method. A choice of between zero (no choice) and ten different branch tests was allowed for each branch during training. The results plotted in figure 3.10 show that the accuracy is very sensitive to the number of choices that were allowed, with the optimal being between four and six per branch. The reduction in performance with more

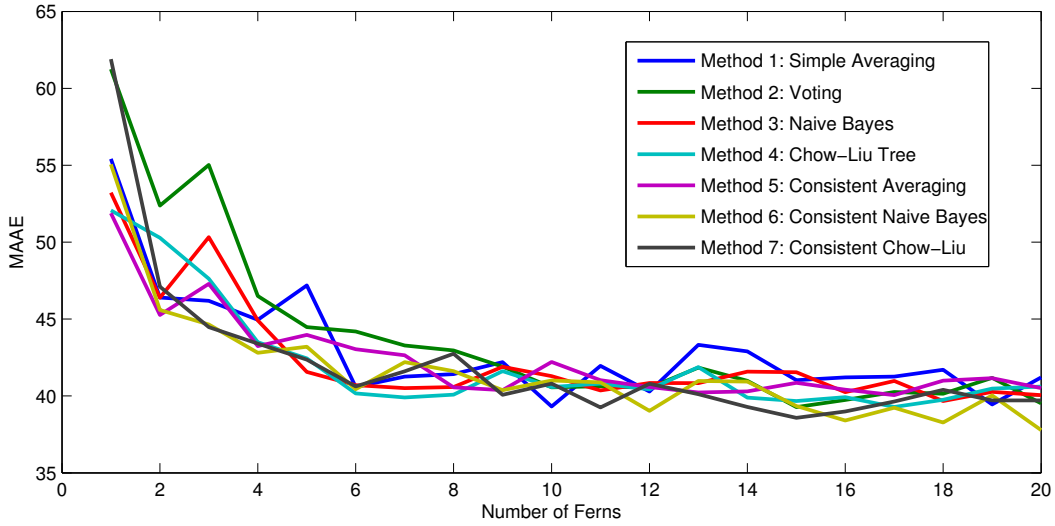


Figure 3.11: Classification accuracy resulting from each of the seven methods for combining fern estimates. Values plotted are the mean of ten repetitions. The standard deviation of the mean in most cases is less than one degree.

choices suggests that the branch tests in the ferns become too similar.

Combination Methods

The performance from each of the seven different methods for combining fern estimates was tested for varying numbers of ferns, which were trained separately for every experiment. The results from repeating each experiment ten times are shown in figure 3.11. The methods all perform similarly and the random noise in the estimates do not allow any conclusive judgement on the best method, however the consistent Naive Bayes approach appears to perform slightly better.

A surprising result is that although the simple voting approach performs the worst with fewer than ten ferns, there does not appear to be a significant difference to the other combination methods with more than ten ferns. This suggests that the error models used by the other approaches become less important with more ferns.

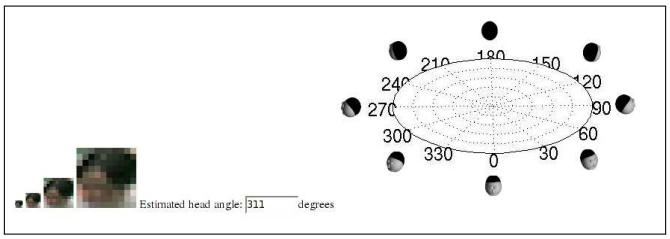
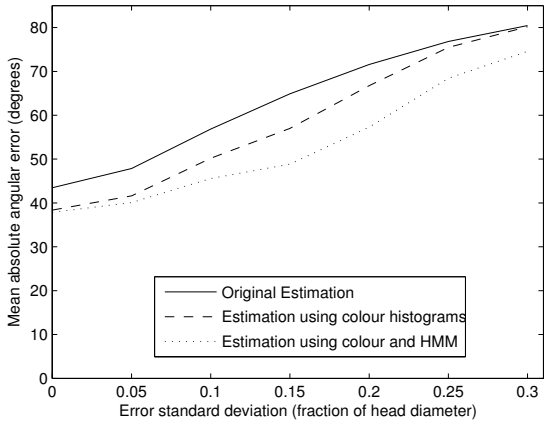


Figure 3.12: A sample from the test used to measure human performance.

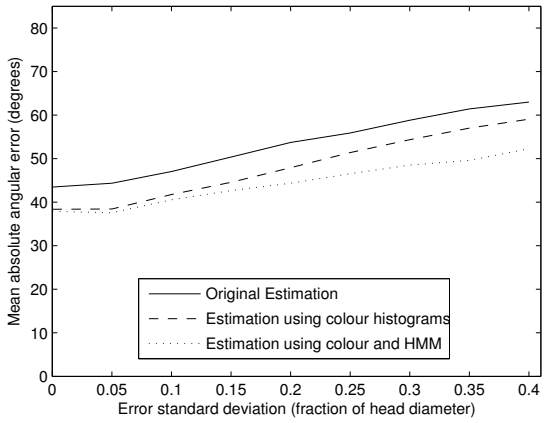
Human Performance Comparison

People are capable of identifying frontal face views in very low resolution images but non-frontal views tend to be less distinctive and in many cases people find the identification of the head pose challenging. To provide a comparison with human performance, twelve people were asked to estimate the pan angle of every one-hundredth head image in the same video sequences that were used to test the classifier. To ensure a fair comparison, the head images from the same actors were grouped to allow the test subjects to infer the hair and skin colours. The head images were shown at four different sizes since some images were easier to interpret when enlarged and others when shrunk due to the false edges introduced by pixellation. A diagram was provided to allow angles to be entered by clicking at the desired angle to prevent errors due to estimated angles being incorrectly converted to numerical form (figure 3.12).

The combined results showed that people are capable of estimating the head pose of the images in the test sequences with a mean absolute angular error of 26.6 degrees. Although the ability of people to identify head pose is fairly uniform, performance is dependent on the amount of useful information in the images, so the mean human error provides a useful measure for comparing the difficulty of different data sets. The level of human error represents the amount of ambiguity in the test data as well as the error in ground truth labelling.



(a) Head pose error resulting from the head region being incorrectly positioned



(b) Head pose error resulting from the head region being too large or too small

Figure 3.13: Graphs showing the effects of random translational and scale errors on the accuracy of the head pose estimation. Errors were introduced with a Gaussian distribution and resulted in the head region being translated or scaled along both axes. The classifier performs well when the size of the head region is incorrect but performance rapidly diminishes as translational errors are introduced.

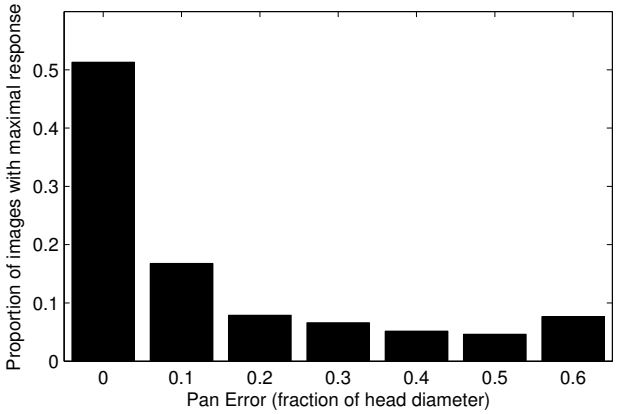


Figure 3.14: The fraction of head images for which the randomised ferns estimated the greatest probability at each pan location.

Translation and Scale Errors

The experiments in this chapter test performance using a dataset with hand labelled head regions (dataset B), which in most cases are accurate bounding boxes, however in a real system this is unlikely to be the case. Some experiments were performed to test how both translation and scale errors in the head regions from the dataset would affect the classification accuracy, the results of which are shown in figure 3.13. The experiments were conducted by introducing Gaussian error into each of the bounding box position and the scale of the bounding box, with varying magnitudes of error. Both cause a reduction in the classifier performance, with the translation error having the larger impact.

Since both translation and scale errors result in a significant performance reduction, a further experiment was performed to determine whether the ferns could also be used to distinguish between head and non-head images and therefore be used to help with localising the precise head region. The hand labelled head regions were translated in the horizontal direction by a distance between 0 and 0.6 times the head width. At each location, the probability of the location being correct was estimated by summing over equation 3.2 for every hypothesis h, the result of which is the probability that any hypothesis is true for the image.

The head likelihood was calculated at each of the different translated positions across the entire dataset. Figure 3.14 shows a histogram representing the fraction of the dataset that gave the largest probability at each of the translated locations. Although the ferns recognise the correct alignment most frequently, the performance is not good enough to justify the extra processing time that would be required to search across combinations of different scales and both horizontal and vertical translations.

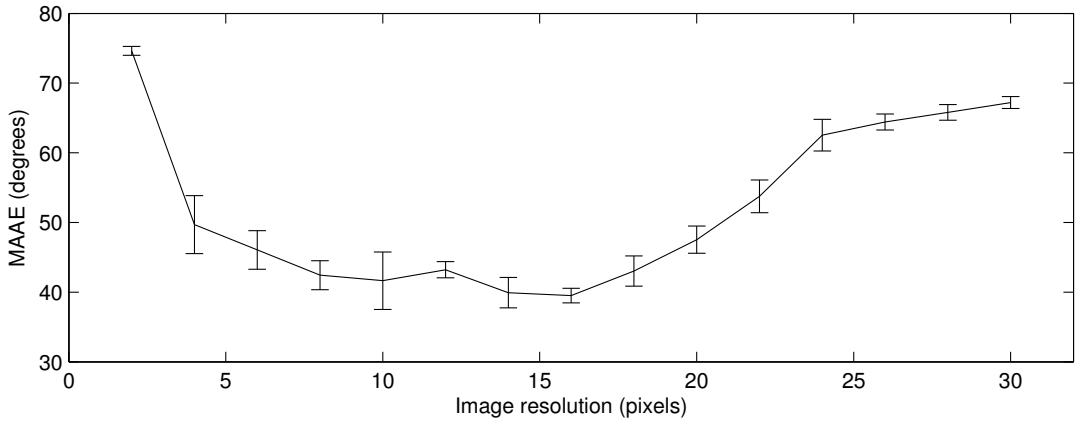


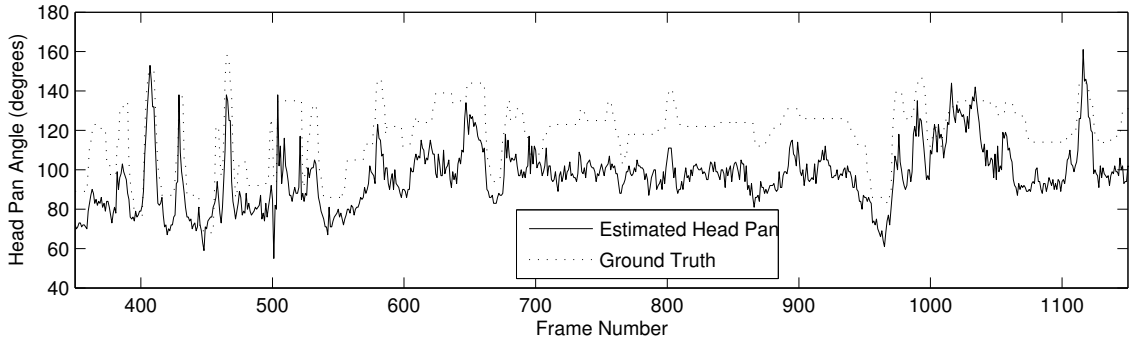
Figure 3.15: A graph showing how the classifier performance is affected by the size of the head images. Each experiment was repeated ten times and the error bars represent 99% confidence intervals for the mean.

Image Size

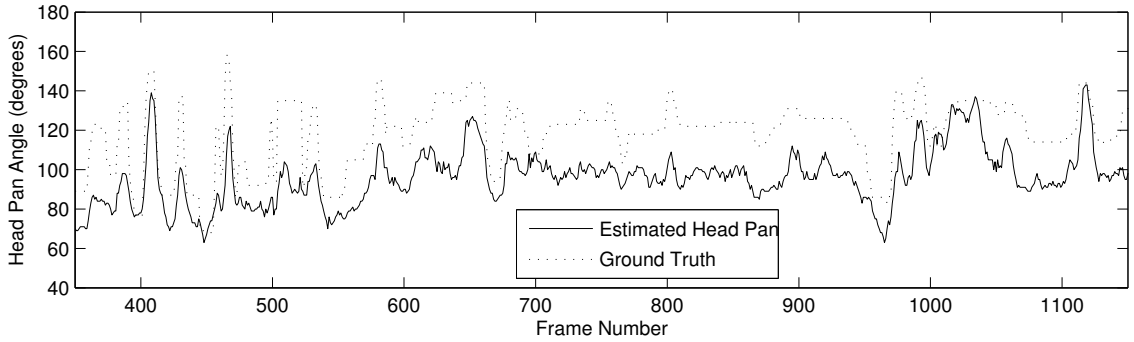
Although the main objective of this chapter has been to develop a method for estimating gaze directions in low resolution video, experiments were performed to measure the performance of the classifier on higher and lower resolution head images. As before dataset F was used for training and dataset B for testing, but this time the images were scaled to a variety of sizes rather than just 10 pixels square to simulate different video resolutions. The somewhat surprising result, shown in figure 3.15, is that the performance drops when the image dimensions are increased above sixteen pixels square. A likely reason for this is that the high resolution images contain a wider variety of colours, which reduces the ability of the k-means clustering to separate the background, hair and skin regions. This could be resolved by using a more sophisticated segmentation algorithm.

Overall Results

The previous experiments have shown that the best branch selection method was to maximise the information gain using a choice of four tests per branch, and that the consistent naive Bayes combination method appears to work the best. Using the



(a) Head Pose estimation without filtering



(b) Head Pose estimation filtered using a HMM

Figure 3.16: Pose estimation graphs for a single head compared with ground truth both with and without filtering through the use of a Hidden Markov Model. The filtering reduces the amount of noise and makes small changes in the head pose more visible.

Figure 3.17: Five different head images and samples from the colour histograms that have been learnt up to and including the current frame. From top to bottom, the colour blocks show samples from the background, skin and hair histograms respectively.

combination of the two improves the performance further to give a MAAE of 34.8°.

The same experiment was repeated without the HMM, and in this case the MAAE was 36.6°. In addition to giving a small performance improvement, the HMM also removes some of the high frequency noise in the direction estimates to make them smoother, which allows more accurate estimates of the instantaneous angular velocity. The difference between the sequence of angle estimates both with and without the HMM for one person are shown in figure 3.16. These plots also give some insight into a possible cause of errors, since there is a consistent difference of approximately 30° between the estimated and ground truth angles. This could be caused because the hair or skin region for the individual is larger or smaller than the closest matching training data.

A similar experiment was also performed to demonstrate the benefits of learning the colour distributions for individual people. With the colour learning disabled, the performance was reduced to 36.5°. With no temporal information at all (no colour or HMM) the MAAE was 41.7°. Some examples of colour histograms that were learned for individual people are shown in figure 3.17. These histograms could potentially be used to localise the head region in subsequent frames or to help identify the same person in situations where there are multiple cameras.

Finally figure 3.18 shows a histogram of the error magnitude for one experi-

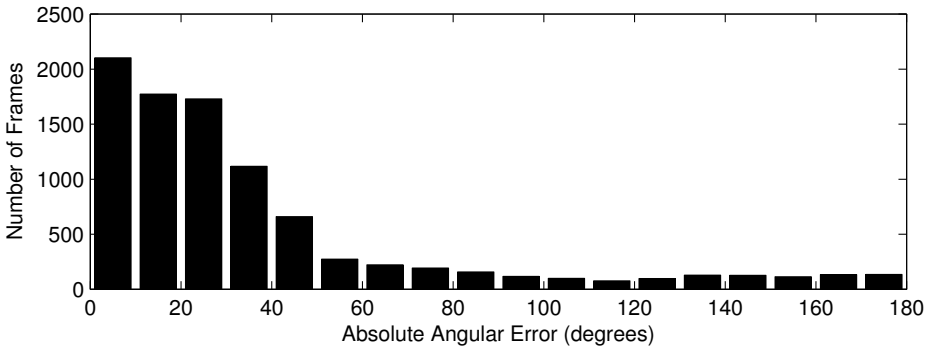


Figure 3.18: Angular error histogram from tests using 9260 head images. Small angular errors are common due to the similarity of adjacent poses

ment. This demonstrates the advantages of the MAAE performance measure over the percentage correctly classified because there is no clear divide between correct and incorrect estimates.

3.7 Conclusion

In this chapter a method for estimating gaze directions in low resolution video has been proposed. The approach addresses the problem of variations in appearance by allowing classification without making any prior assumptions about the distributions of hair and skin colours. In particular, the following contributions have been made:

- A randomised fern based image classifier with predicate based branches was proposed along with a corresponding algorithm for inference.
- A method for learning colour distributions corresponding to the abstract labels to improve estimation accuracy was developed.
- Modifications were proposed to the standard methods for combining decision tree estimates to make best use of the predicate ferns.

The implementation is suitable for real-time use, since each estimation takes only ten milliseconds on a 2.4GHz CPU (Central Processing Unit). The coarse gaze

classifier presented in this chapter has also been reimplemented independently since publication [62], which verifies that the approach is sound.

The results in this chapter show good performance from particularly low resolution video where the head regions have been manually labelled, however the experiments to show the effects of scale and translation errors suggest that performance will be reduced if the head regions are less accurate, as one might expect from automatic tracking. In the next chapter, a completely automatic system will be developed to allow the evaluation of the gaze estimator in a more realistic context.

CHAPTER 4

Automatic Gaze Estimates for Attention Measurement

In this chapter a fully automatic system for tracking, coarse gaze estimation and attention map generation is developed. A Kalman filter based head tracking system combines HOG detections with Kanade-Lucas-Tomasi (KLT) tracking to provide head image sequences. The head images are then classified and the resulting gaze direction estimates used to identify the regions of a scene where people frequently look. The result is believed to be the first large scale attention measurement system for pedestrians who are able to walk freely and behave naturally. A paper based on the work was published at BMVC 2009.

4.1 Introduction

Chapter 3 developed a method for estimating gaze directions without making any assumptions about the appearance of the observed person. This was shown to be effective when tested using a variety of videos, however the scenario was unrealistic because the regions occupied by heads in the video sequences used for testing were hand-labelled. An automatic tracking system would output bounding boxes that are less accurate than those from a human labeller, and we have shown that less accurate head regions reduce the accuracy of the gaze direction estimates.

Problem Statement: The objective of the work described in this chapter is to construct a completely automatic gaze estimation system that can take a video sequence as input and produce useful gaze direction estimates as output without any manual intervention. A complete system is required to allow the evaluation of gaze estimation methods in a realistic context, and to determine whether or not the levels of pose accuracy obtained are useful in practice.

The problem can be broken down into specific requirements. The first is to obtain bounding boxes around the heads in the video sequence, which is addressed by the multi-target tracker described in section 4.2. The second requirement is to process the regions that are output by the tracking system to produce gaze direction estimates for all of the people in the scene. This is covered in section 4.3, which describes a variety of approaches to coarse gaze estimation that were tested using the tracker output. The final requirement is to show that the fully automatic system is able to produce output that is useful in a surveillance context. Section 4.4, describes the attempted method for using the automatic gaze direction estimates with an approximate model of gaze behaviour to infer the subject of interest in various surveillance scenarios. The tracking, gaze estimation and attention measurement components are all evaluated individually in the corresponding sections.

4.2 Multi-Target Tracking

The first step of processing requires the pedestrians in a scene to be tracked, with the purpose of providing stable head images for the following pose estimation step.

In surveillance video, existing approaches track whole pedestrians and then locate the heads using a further stage of processing. Our proposed approach makes the unusual choice of tracking the heads of pedestrians directly, rather than their entire bodies. The first reason for this choice is that security cameras are generally positioned sufficiently high to allow pedestrian's faces to be seen, so their heads are obscured much less frequently than other body parts. The second is that the offset between the centre of a pedestrian's body and their head changes as they walk, so tracking the head directly provides more accurately positioned head images.

Of the methods for pedestrian tracking that were reviewed in section 2.4, the appearance based methods are more capable of coping with crowded scenes where pedestrians are frequently occluded. The approach that we develop is most similar to Wu and Nevatia's [133], which found pedestrians using an edge based body part detector and filled in the gaps between detections with mean-shift tracking [28]. Our approach combines absolute location estimates from a head detector based on Histograms of Oriented Gradients (HOGs) [29] with velocity estimates from feature-based tracking.

Instead of using the feature tracking only to fill in gaps, our approach combines the measurements probabilistically using a Kalman filter (see Appendix 8.3.1). We replace the system evolution model, which usually predicts the next state based on a physical model, with the velocity estimations from feature tracking. The resulting behaviour has different characteristics when compared to that of a Kalman Filter under normal usage because the uncertainty in the predictions is usually much smaller than that of the observations.

Since the pedestrians in the video sequences have a wide range of sizes, the sequences were all fully calibrated relative to a known ground plane. Using calibrated

Figure 4.1: Image measurements are made by following corner features using KLT tracking (left) and by finding heads with a HOG detector (right).

videos allowed the locations of people's feet on the ground plane to be estimated from their head locations by assuming an average human height of 1.7 metres. The calibrations also allowed the approximate head size to be calculated to limit the scale range of the HOG detector, which significantly reduces the processing requirements.

4.2.1 Kalman Filter Formulation

The tracking system uses head location estimates from the HOG detector and motion estimates from KLT feature tracking (figure 4.1). A Kalman Filter normally combines absolute observation measurements with predictions from a state evolution model. The KLT motion estimates that we obtain are relative rather than absolute, so they are used in the place of the standard state evolution model to predict the state at the next time step. Table 4.1 shows how the observations correspond to the parts of the standard Kalman filter model using the notation described in appendix 8.3.1.

The accuracy of the KLT motion estimate u is particularly important because the HOG detector is unreliable and will often miss detections for seconds at a time. During these gaps the KLT motion estimates must be relied upon to maintain the track.

Term	Representation in our tracking system
x	A 2D vector representing an image location, measured in pixels
P	The covariance of x, measured in pixels
F	Identity Matrix
B	Identity Matrix
Q	Covariance of KLT motion estimates
R	Covariance of HOG detection
u	Motion estimate from KLT tracking
z	Location estimate from HOG detection
H	Identity Matrix

Table 4.1: Terms used in the Kalman filter with their corresponding representations in our tracking model.

4.2.2 KLT Motion Estimates

This section describes the method used to obtain robust motion estimations for the heads by combining the motion estimates from multiple tracked KLT corner features [75, 22]. The KLT algorithm tracks corners between frames by following the image intensity gradient. The tracking algorithm is fast, but sometimes fails when the algorithm converges on an incorrect corner feature in the background or another object. To prevent these individual failures from affecting the tracker, approximately ten features were tracked for each object (head) and their performance was accurately modelled so that incorrectly or inaccurately tracked corner features could be identified.

The KLT corner features were initially found by processing the initial head region using the Shi-Tomasi [112] corner detection algorithm, which was run with a low acceptance threshold to ensure that corners were almost always found. Whenever corner features were lost, the current best estimate of the head region was processed to replace the lost corners.

When corner features provide good motion estimates, we would like to retain them to track in subsequent frames. This is because some corners are more distinctive and so inherently easier to track than others, so a corner feature that has been tracked reliably in the past is likely to be tracked reliably in the future. Let v_k^i be

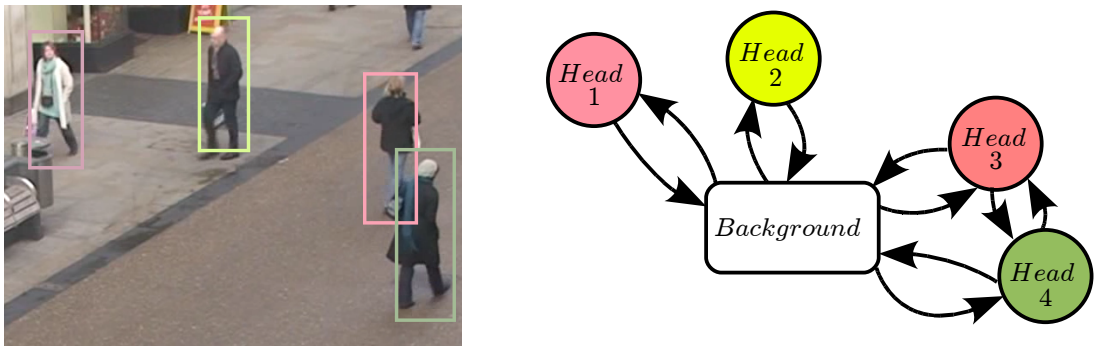


Figure 4.2: An example illustrating how $P(o_k^i = j | o_{k-1}^i = m)$ is estimated. When the bounding boxes of pedestrians overlap, it is possible that corners could be incorrectly tracked between them. The possible state transitions for the frame on the left are shown by the state machine on the right, where arrows represent possible transitions between the objects that corner features follow. Although the tracked corners are restricted to heads, full body bounding boxes are used here because corners could be incorrectly tracked to any body part.

the motion estimate for corner feature i between frames k and $k+1$, where i is an identifier rather than an index. The feature tracking model aims to maintain estimates of $P(o_k^i = j)$, the probability that corner feature i provides a motion estimate representative of object j in frame k. Let V_k represent all of the motion estimates from frame k. The probability estimates are updated recursively in three steps.

Step 1: Prediction In the first step, the object probabilities for the features are predicted from those in the previous time step:

$$P(o_k^i = j | V_0 \dots V_{k-1}) \propto \sum_{m \in J} P(o_k^i = j | o_{k-1}^i = m) P(o_{k-1}^i = m | V_0 \dots V_{k-1}) \qquad (4.1)$$

In this equation, J represents the set of all objects. The first term represents the likelihood of corner features being incorrectly tracked in a way that causes them to jump to a corner on a different object, and the second is the result of the update step from the previous frame.

Each corner feature is assumed to be incorrectly tracked with probability τ, so when $j = m$ the first term has the value $1 - \tau$. When a corner feature is incorrectly

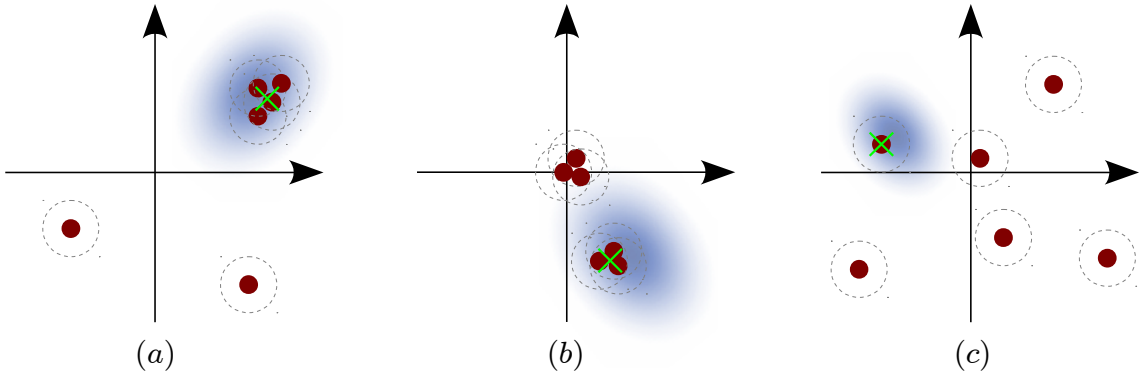


Figure 4.3: Diagrams illustrating the combination of the two factors in equation 4.2 to obtain an estimate of the object velocity u_k^j, which is shown by the green cross. The 2D plane represents the domain of u_k^j, which is the movement of the object between frames $k-1$ and k. The blue background is proportional to the density estimate from the constant velocity prior and the red circles represent the individual motion estimates from the tracked KLT corner features, with the range of the Parzen window shown by the dotted circle. Situations like the one shown in (a) are the most frequent, with a well defined mode and only one or two outliers. The situations in (b) and (c) are ambiguous, but the constant velocity component increases the chance of the correct mode being found. In (b), half of the corners have been incorrectly tracked to the background and in (c) another object has occluded all but one of the corners.

tracked, the KLT algorithm is much more likely to converge on a nearby object in the image than one that is far away. The probability of a corner from object m being incorrectly tracked to one on object j was approximated as being proportional to the area of intersection between the bounding boxes of the two objects. The background was treated as an object with an infinite bounding box because it is close to all of the objects in the scene. The state machine corresponding to an example frame is shown in figure 4.2. In this example the two people on the right are close in the image so the state machine models the possibility that KLT corners will be incorrectly tracked from each to the other. Although these probability estimates are not particularly accurate, they are important for ruling out the possibility of KLT corner features being incorrectly tracked between objects with similar velocities that are not close to one another.

Step 2: Object Velocity Estimate The next step is to estimate the object velocity u_k^j using the predictions and the motion estimates from individual corners. First the most likely candidate velocity is found:

$$u_{max}^j = \underset{u^j}{\operatorname{argmax}}\, P(u^j|u_{k-1}^j)P(u^j|V_0 \dots V_k) \tag{4.2}$$

The first term, $P(u|u_{k-1}^j)$, is based on a constant velocity model with large Gaussian error. This component is usually of little significance, but in circumstances where the distribution of motion estimates becomes multimodal it increases the chance of the correct mode being identified. These ambiguous multimodal distributions occur in situations such as when half of the corners are incorrectly tracked and end up on the background (see figure 4.3). The second term is estimated using a Parzen window density estimate over the individual motion estimates:

$$P(u^j|V_0 \dots V_k) \propto \sum_{|u-v_k^i|<r} P(o_k^i = j|V_{k-1}) \tag{4.3}$$

The window radius r is large enough to include most of the correctly tracked corners but removes outliers so that they do not bias the velocity estimate.

For efficiency, an approximate maximum to equation 4.2 is found by evaluating it at the locations corresponding to each of the v_k^i motion estimates from the tracked corners. If more time were available for processing, an iterative mode seeking algorithm such as mean-shift could be used, though it would have to be modified to cope with multiple modes.

Finally the object velocity is calculated as a weighted mean of the inlier motion estimates:

$$u_k^j = \frac{\sum_{|u_{max}^j - v_k^i|<r} v_k^i P(o_k^i = j|V_{k-1})}{\sum_{|u_{max}^j - v_k^i|<r} P(o_k^i = j|V_{k-1})} \tag{4.4}$$

Here it is assumed that the object motion is only a 2D translation because the small

head images do not provide enough information to reliably constrain a model with more degrees of freedom. For larger images of more complex objects, methods such as affine transfer [103] which take object structure into account would potentially be more appropriate.

Step 3: Update The final step is to calculate the posterior probabilities using the object velocity estimate:

$$P(o_k^i = j|V_0 \dots V_k) \propto P(v_k|o_k^i = j, u_k^j)P(o_k^i = j|V_{k-1}) \tag{4.5}$$

The value of $P(v_k|o_k^i = j, u_k^j)$ is estimated by assuming that the error in the KLT motion estimates has a Gaussian distribution. When an object moves slowly, the distributions for the object and the background overlap, so for a newly found corner many frames are required to reach a high probability of the corner being on the object. When objects move quickly there is better separation between the distributions so only one or two frames are required. The parameters for this and the other distributions that are used in this section were estimated empirically by comparing the motion estimates with hand labelled data in a short section of video.

4.2.3 Implementation Details

There are a number of details that are not of theoretical interest but which would be required for an accurate reimplementation.

HOG Detection Gating The tracking algorithm was originally developed using a CPU implementation of the HOG detector, which required approximately 22 seconds to process a 1920x1080 image. This is far too slow for real-time usage, so instead it was only applied to small regions that were defined by three standard deviations of the uncertainty in the head location. The expected size of the head in each of the corners of the detection region was calculated and used to constrain the scale range of

the detector as well. These constraints reduced the detection time to approximately 200ms per frame.

2D and 3D Conversions Aspects of the tracking system rely on the camera being calibrated with a known ground plane. The calibration is used to calculate bounding boxes, to convert head locations to ground plane locations, and to estimate the correct size for a head at any image location. Humans were modelled as cylinders $1.7m$ in height and $0.5m$ in diameter, with the heads represented as ellipsoids $0.22m$ in height and $0.20m$ wide.

Adding Targets The HOG head detector was too slow to run over full video frames so new targets were added by detecting motion consistent with that of a pedestrian. Two hundred KLT features were continuously tracked, with their locations reassigned every ten frames to a random image location. If any features were found to move in the same direction (i.e. dot product of 2D velocity is positive) for three frames or more then a tracker was initialised. Since the movement could be on any part of the person, the trackers were initialised with a covariance that was large enough to cover all possible head locations.

Removing Targets Successfully identified targets still have a significant chance of being lost. It is important to detect lost targets both to reduce the number of false positives and to prevent processing time from being wasted. Targets were considered to be lost if any of the following conditions were broken:

- The target speed must be less than 3× mean human walking speed (1.26m/s) on at least 90% of measurements
- The target speed is more than 2cm/s on at least 50% of measurements
- The direction of the target must have changed by less than 90° on at least 90% of measurements

- The Kalman filter standard deviation is less than twice the object size
- At least half of the head area must be within the bounds of the current video frame.

These conditions were evaluated over the most recent three seconds of data. They allow a large margin for noise in the measurements but are able to quickly remove targets when the tracking fails or people leave the scene. The only case where these conditions frequently fail is when people stand still for long periods of time, in which case they are very difficult to distinguish from false positives in the background.

4.2.4 Tracking Evaluation

Two experiments were carried out to test the performance of the tracking algorithm, both using a the Town Centre video which is three minutes long and has all 71473 ground truth head regions hand labelled (see section 2.6.1). The first experiment examined the accuracy gained from modelling the tracking errors of the KLT corner features using the state machine method described in section 4.2.2. The aim was to justify the benefit derived from the proposed method, so two baseline methods were implemented to allow a comparison. The first baseline method estimated the object velocity using the mean of the KLT motion estimates and the second found the mode using the Parzen window as in equation 4.3, but with feature velocities having equal weights instead of the learned probabilities. The heads of pedestrians were initialised to the correct location and tracked for twenty second intervals without any measurements from the HOG detector and compared with the ground truth data to measure the rate of drift.

The rate of drift was measured by taking the mean overlap between the ground truth head regions and the regions that were predicted by the KLT tracking. This measure is plotted over the twenty second interval in figure 4.4 (left) for the proposed method and the two baseline methods. The plot shows that the method we use for

modelling the KLT tracking errors results in the estimated regions having an average overlap of approximately 40% after twenty seconds compared to approximately 20% for the two baseline methods. The results include estimates with no ground truth overlap, so if these were excluded the mean overlap would be considerably higher. The drift rate decreases with time because some heads are easier to track than others, so those that have been tracked with little drift in the past are likely to have little drift in the future.

The second experiment tested the ability of the tracking algorithm to locate the heads in each frame of the video sequence compared with HOG detections alone. Performance was measured using the standard precision and recall measures, the former being the fraction of estimated regions that were genuine, and the latter being the fraction of ground truth regions that were found. Ground truth regions were considered to be matched with estimated regions if the area of their intersection was more than 25% of the area of the ground truth region. Both the tracking system and the HOG detector could be tuned to provide a different balance between precision and recall, so figure 4.4 (right) plots the two measures against each other. The balance between the precision and recall was adjusted for the HOG detector by changing the threshold on the required distance from the dividing hyperplane, and was adjusted for the tracker by applying a threshold to the Kalman filter covariance. The plot shows that the tracking algorithm is able to locate approximately twice as many heads as the HOG detector alone before experiencing a significant drop in precision. In addition to providing more accurate head locations, the tracking allows the detector to be applied to only a small subset of each frame, reducing the processing time by a factor of approximately 100.

A comparison of the general tracking performance of this system with others can be found in section 5.3.2.

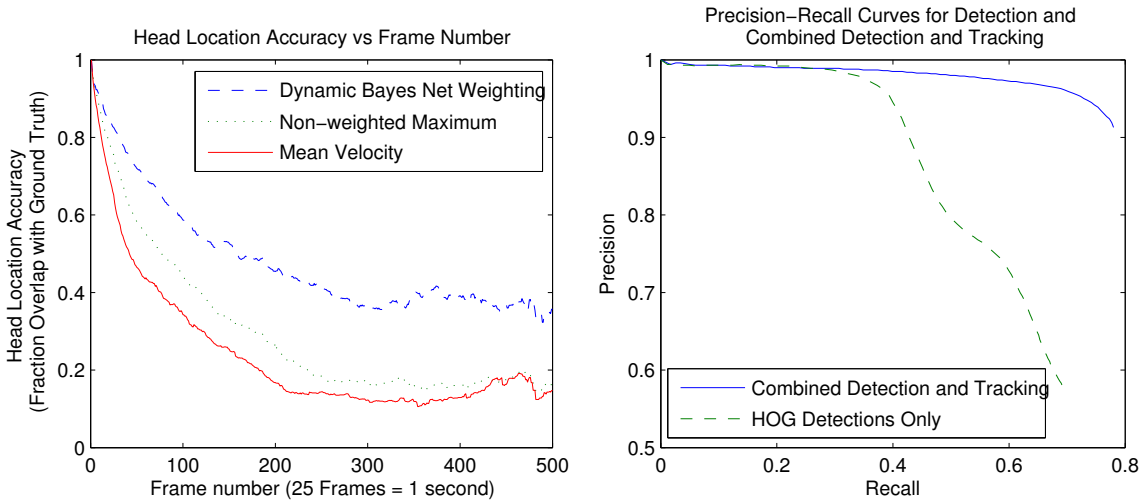


Figure 4.4: Left: Drift from cumulative errors in the motion estimation without guidance from HOG detections over twenty seconds of tracking (500 frames) Right: Comparison of the combined tracking algorithm to HOG detections alone. The tracking algorithm locates approximately twice as many heads before having a significant drop in precision.

4.3 Head Pose Estimation

This section describes the approaches to coarse gaze direction estimation that were attempted. There are four methods in total. The first is the method from chapter 3 which was based on predicate ferns. The next approach was also based on ferns, but with different branch tests based on two new feature types. These two feature types were also used in the third method, but with a SVM rather than ferns. The fourth method was an implementation of Orozco's method [91] which was based on *Mean Template* comparisons with an SVM classifier. All four methods are based around classification into eight direction classes, as described in chapter 3.

4.3.1 Predicate Ferns

The first method to be considered was the low resolution classifier that was described in chapter 3. Provisional testing showed that the predicate ferns did not perform as well on the tracker output as for still images, which provided motivation for trying other approaches. The lower performance was probably due to a combination of

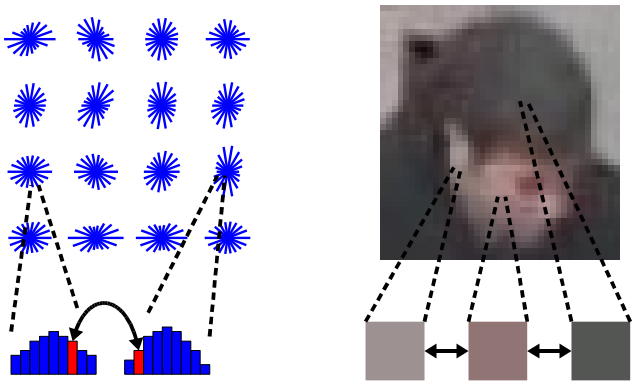


Figure 4.5: The two branch test types that were tested for use in the randomised fern classifiers. The first test type (left) is determined by comparing two bins from normalised HOG descriptors and the second test type (right) compares three different colour samples

head location errors and compression artifacts which were not as significant in the datasets that were previously used.

4.3.2 HOG/CTC Ferns

The choice of branch tests for a gaze direction classifier is critical; branch tests must be able to recognise general properties of each direction class irrespective of the large variations in appearance between people and any errors in the head location estimate. Two different branch test types were tried, both compare values from different image locations against one another rather than against a fixed threshold which makes them robust to brightness variations and colour tints. The branch test types are illustrated in figure 4.5.

HOG Descriptor Decisions

The first branch test type is based on the HOG descriptors that Dalal and Triggs [29] used as input for their pedestrian detector. Dalal and Triggs showed that the coarse spatial binning of their descriptors was beneficial for object detection because it made the descriptor robust to small differences in the locations of edges. This is a desirable property for making gaze classifiers invariant to translation errors.

The descriptors are calculated by first measuring the image intensity gradient at every pixel in nine different directions. The image is then divided into a square of sixteen cells and for each cell a histogram over the nine orientations is generated from all of the pixels within that cell. The histograms for each cell are then normalised across blocks, where each block consists of 2×2 cells, of which there are 9 possibilities for the 4×4 arrangement of cells that are used. The final descriptor has 324 elements, consisting of each of the four histograms in every block, where each of the histogram bins is normalised using the same angle bin in the other three histograms. Having generated the descriptors, the HOG fern branch tests simply compare two randomly chosen elements in the normalised descriptor and the test outcome depends on whether the first or second is larger.

Colour Triplet Comparison Decisions

A new descriptor, the Colour Triplet Comparison (CTC), was developed with the intention of providing some of the benefits of the predicate ferns but without the cost of the hypothesis evaluation. Each CTC branch test samples colours from pixels at three different locations within the tracked head region and makes a binary decision based on whether the first and second colours are more similar than the second and third colours. Similarity is measured as the sum of the differences in each of the RGB components (i.e. the L1 Norm of the vector difference).

The reason for including three colour samples is to make the comparison more robust; in particular the branch test is invariant to any change in the brightness or contrast of an image.

4.3.3 HOG/CTC SVM

Many of the other recent approaches to coarse gaze direction estimation have been based around SVMs. As a comparison, the HOG and CTC branch tests were also tested with a multi-class SVM classifier using a polynomial kernel function. An

eight-class classifier was trained using the one-against-one method, which was found to perform better than a one-against-rest classifier. Since many different HOG and CTC branch tests are possible, 10000 randomly chosen branch tests were used to train the SVM. The branch tests were used rather than the raw image measurements because of the invariance they provide to natural image variations, and because using them allows a more direct comparison of the performance of the ferns and the SVM.

4.3.4 Mean Templates

A recent method for gaze estimation that was applied to similar video was the mean templates method of Orozco et al.[91]. The method was based around a one-against-rest SVM classifier with a polynomial kernel function, where the input vector consists of the difference between the test image and a series of eight *mean templates*. Each mean template was the mean of the training data images belonging to the 45° range of the class, where the background was labelled as black either through manual labelling or background subtraction. Orozco's method was implemented so that a comparison could be made with our various approaches.

4.3.5 Gaze Estimation Evaluation

The acquisition of head images using automatic tracking allowed the different estimation methods to be compared in a realistic setting. The gaze estimation algorithms were trained using the two still image datasets; the first (dataset A) was higher resolution than the second so was more appropriate for testing on the video datasets, but the second training dataset (dataset F) had labelled segmentations, so was required for comparing with the predicate ferns. For testing, large datasets of head images from automatic tracking in the Town Centre and i-LIDS video sequences were used (datasets C and D), with the hand labelled video dataset from chapter 3 (dataset B) used as a comparison. Details of all datasets can be found in section 2.6.2.

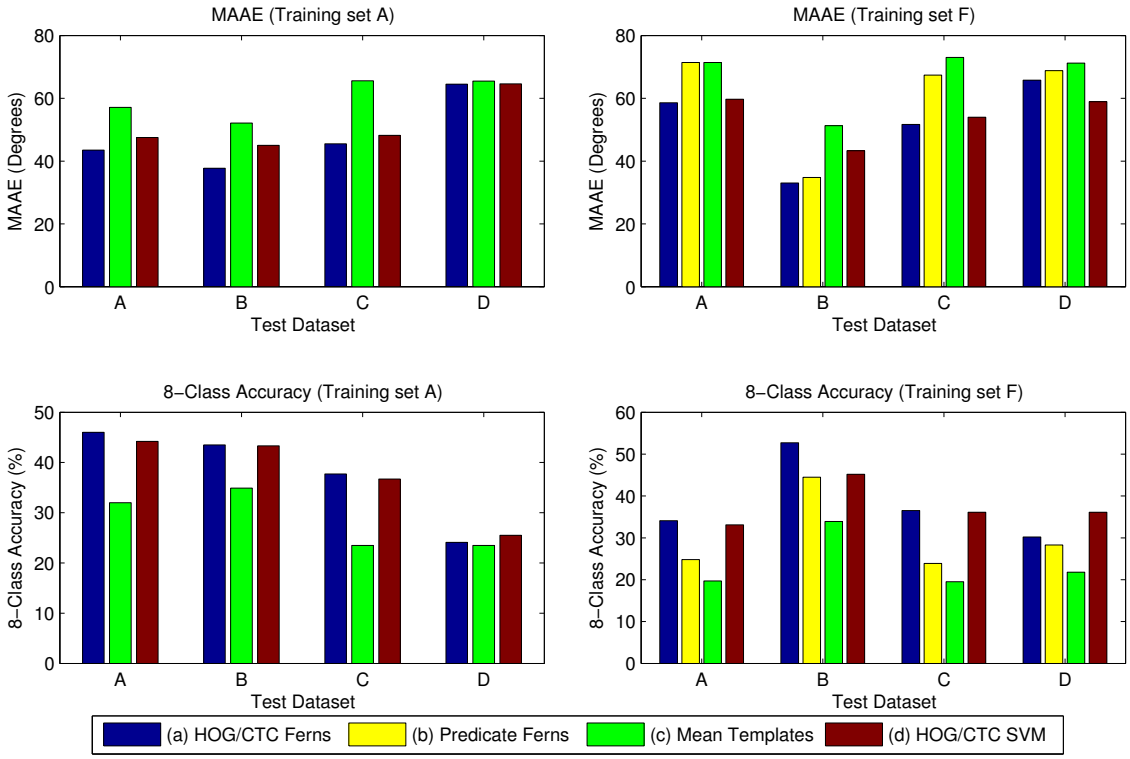


Figure 4.6: Results from testing the gaze estimation algorithms on four different datasets. Each result is the mean of ten repetitions. The top two charts measure performance using the MAAE, for which lower values are preferable, and the bottom two charts use the percentage correctly classified with eight classes, for which larger values indicate better results. The left two charts show results from training on dataset A and the right two show results from training on dataset F.

Comparison of Gaze Estimation Accuracy

Figure 4.6 shows the results of experiments where each of the four algorithms were trained using datasets A and F and tested on datasets A, B, C and D, which are the most realistic conditions. A table showing the result of comprehensively training and testing every algorithm on every dataset can be found in appendix 8.2. For this experiment, the results using both the MAAE and the 8-class accuracy are shown to demonstrate that the quantisation from a class-based measure can mask the difference in performance between algorithms. An example of this can be seen by comparing the two measures for the HOG/CTC ferns and SVM when training on dataset A and testing on dataset B. The 8-class accuracy suggests that the two methods perform equally well, but the more accurate MAAE shows that the ferns

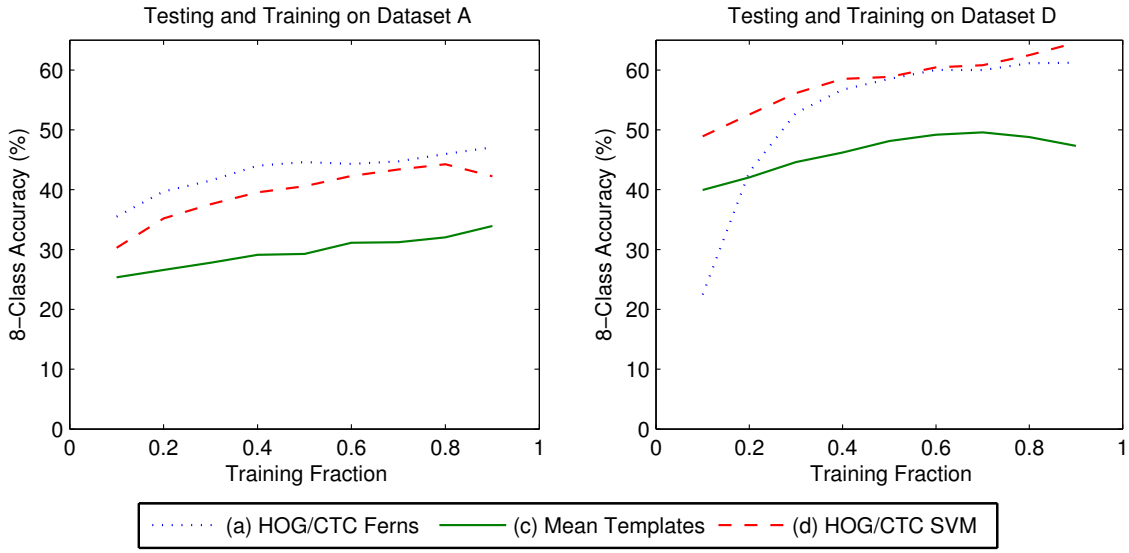


Figure 4.7: The 8-class accuracy resulting from testing and training on the same dataset. The left plot shows results from dataset A, where images are uncorrelated so the experiments are fair. The right plot shows the result of testing on dataset D, a video dataset where many of the head images are similar. The results are significantly higher than those obtained on the dataset with correct separation of the training and testing datasets.

outperform the SVM.

These results show conclusively that the HOG/CTC ferns perform the best across all four datasets. The predicate ferns perform better than the HOG/CTC SVM on dataset B, where head regions were hand labelled, but perform worse on the others. This suggests that the HOG and CTC decisions provide better invariance to location errors.

The mean templates algorithm had the worst performance on all datasets, however the 8-class accuracy of 23% using dataset D is inconsistent with the 77%-80% presented by Orozco et al. [91] using a dataset acquired from the same video sequence. It seems likely that the discrepancy was caused by the training data, which in our case was assorted still photos, but in Orozco's experiments is likely to have come from the same dataset. Some experiments were used to test this theory, the results of which are shown in figure 4.7. The plot on the left shows the result of

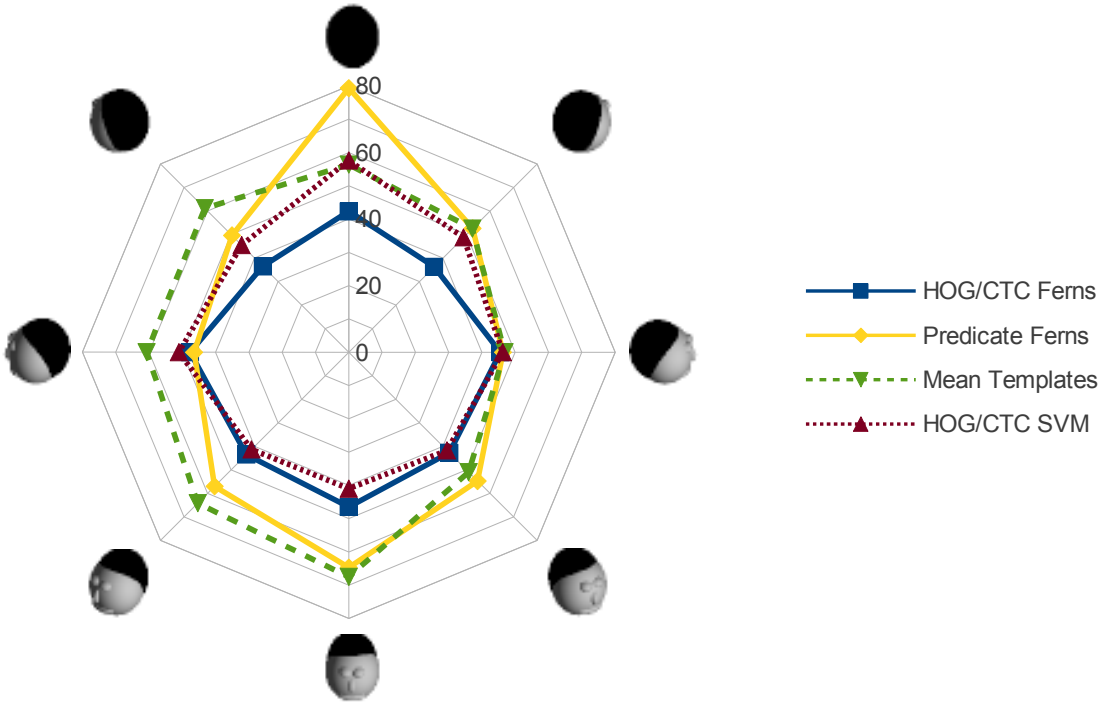


Figure 4.8: A graph plotting the MAAE for each of the eight individual direction classes using the four gaze estimation algorithms. The experiments use half of dataset F for training and the remainder for testing. The largest errors occur for the images facing directly towards or away from the camera.

testing and training each algorithm on dataset A, which consists of individually acquired still images so provides realistic results. The plot on the right shows how testing and training on the head images from dataset D, which was extracted from a video, results in apparently better performance. Images that are collected from different frames in the same video dataset tend to be very similar, so any testing and training on the same video dataset results in classifiers that are highly customised for the particular video but which do not generalise well so are not useful for most applications. Independent experiments using the mean templates on other datasets [113] demonstrated similar performance levels to those obtained in our experiments.

Some additional insight into the failure cases of the four methods can be found in figure 4.8, which plots the error for each of the eight individual direction classes. The algorithms were trained using 50% of dataset F (equally distributed among the classes) and tested using the remaining 50%, and the mean was calculated using

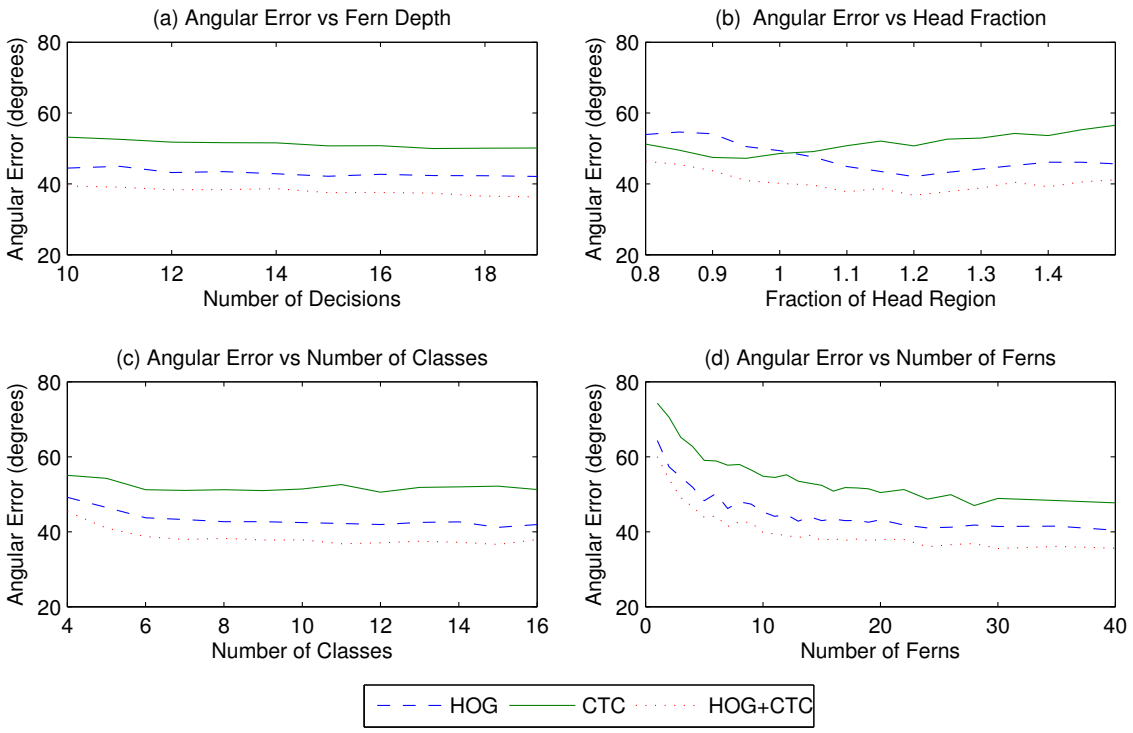


Figure 4.9: Effects of different training parameters on the estimation accuracy of randomised ferns using the two feature types both individually and combined. Unless otherwise specified, 20 ferns with 16 branch tests and 8 classes were trained using a region 1.2 times the size of the head. The errors specified are the mean absolute difference between the angles interpolated from the ferns and the ground truth.

twenty repetitions of each experiment. The peaks in the error can be found at the classes representing head images that face directly towards or away from the camera. The peak in the error for heads facing away from the camera is largest for three out of the four algorithms and is likely to occur because neither the hairline or facial features are visible. The smaller peak in the error for heads facing towards the camera could be due to more variation in the images for this class than the others.

Analysis of HOG/CTC Ferns

The HOG/CTC ferns performed the best out of the tested algorithms, so some further experiments were carried out to find out which factors were important for their performance.

The effects of altering the basic parameters when training the HOG/CTC ferns using both feature types were tested using the hand labelled head images (dataset A). Ferns were trained using 80% of the images (approximately 1200) and were tested using the remaining 20% (300 images). The two subsets were randomly chosen but it was ensured that both subsets were evenly distributed around the 360 degree range. The results in figure 4.9 show that the HOG features performed better than the CTC features alone, but a combination of the two gave the best performance. An interesting observation is that the CTC descriptors are more accurate than the HOG descriptors when the head images are cropped without a border. This suggests that the outline of the head provides important edge information, but the colour information from the background is not useful.

There is inevitably some amount of translation error in the location estimations from the head tracker which introduces error in the gaze direction. This was observed from the experiments in section 3.6 of the previous chapter, but the following methods were not tested with the predicate ferns due to the expense of labelling the training data and the time required for classification. Since location errors affected the accuracy the most, three different approaches to correct for them were tried:

Training with artificial errors: Classifiers were trained with example images to which translational error with a standard deviation of up to half the head diameter was introduced.

Local Maximum: An additional classifier was trained using head/non-head examples and used to search for the most likely head location in a small region around the given position. The most likely region was then classified as usual to estimate the gaze direction.

Weighted Mean: Similar to the local search, except that classification was performed at every location within the search region. The resulting gaze estimate was the mean of the classifications weighted by the likeliness from the head/non-head classifier.

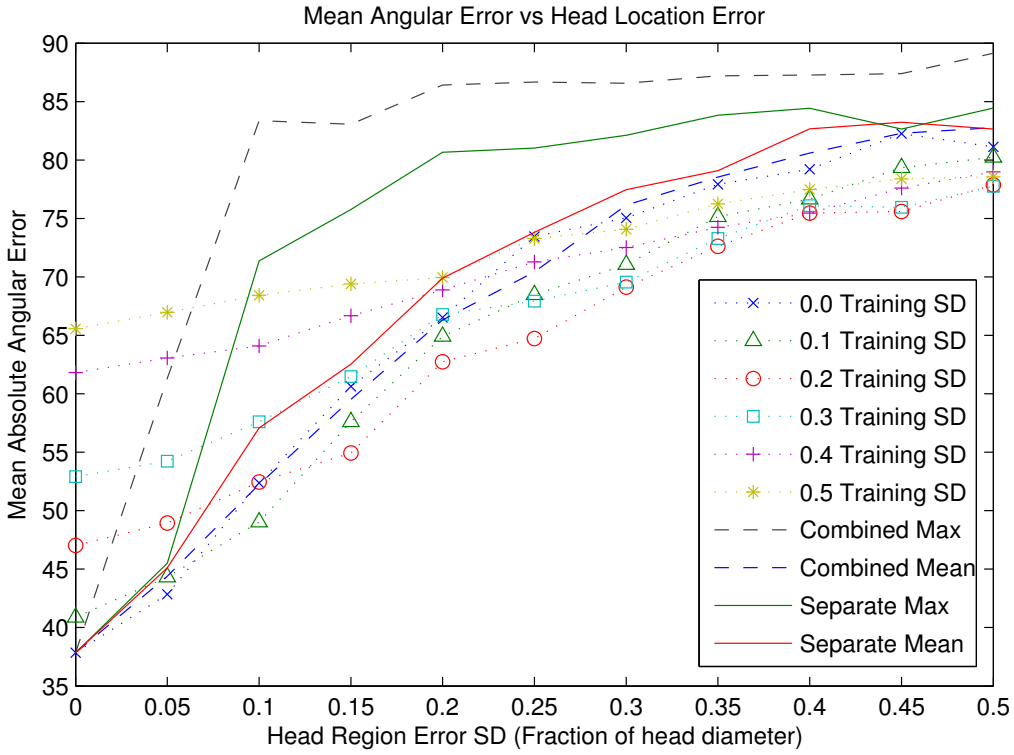


Figure 4.10: Results from attempts to improve classification accuracy in the presence of random translational error of varying magnitude. The 'Combined' estimates use a nine-class classifier with the extra class used to represent non-head images, whereas the 'Separate' estimates have a separate binary classifier to distinguish head/non-head images.

The second two approaches were tested both using separate ferns for detection and classification and also using a combined detector and classifier which had an additional class to represent non-heads. As before, 80% of dataset A was used for training and the remaining 20% for testing, but normally distributed random error was introduced into the test images with a standard deviation of up to half the head diameter. The results of the experiments are shown in figure 4.10. Neither the local search or weighted mean improved the accuracy but training with small artificial errors improved the performance when the test region error was larger or approximately equal to the training error.

4.4 Measuring Attention

In ordinary day-to-day behaviour humans identify interesting objects in their surroundings by drawing on knowledge of the world that they have accumulated throughout their lifetime. In contrast, for an automatic reasoning system world knowledge is very limited, so making such inferences is extremely difficult. This section describes our attempts to measure the subject of interest of the people present in a scene automatically and unobtrusively from remote security camera footage. The resulting information can be used to direct the attention of an observer towards locations that might be of interest.

4.4.1 Attention Model

The accuracy of the tracking and gaze direction estimation can be measured objectively, but the level of accuracy that would be required for the system to be of practical value is difficult to define and likely to be application dependent. To demonstrate that the system is able to produce accurate enough results for at least some applications, an ad-hoc model of attention was developed and used to build *attention maps* representing the amount of interest received by different areas in a

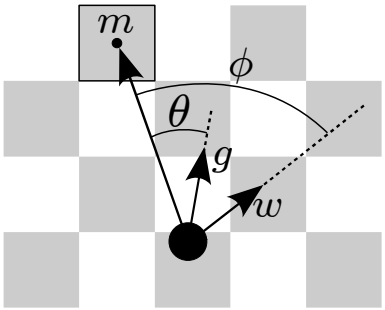


Figure 4.11: A diagram representing how θ, and ϕ are calculated from the walking direction w, the estimated gaze direction g and the attention map location m.

scene.

These attention maps work on a similar principle to *Gaze Fixation Maps* [132] which are commonly used with high resolution gaze tracking systems to represent the most frequently viewed areas of an image in psychological studies. Our attention maps are different because the map is used to represent attention on the ground plane where the line of sight is approximately parallel to the plane, whereas fixation maps represent attention on a plane that is approximately perpendicular to the line of sight. Fixation maps are used to measure observation times over small regions in the order of $1m^2$, in contrast to our attention maps which typically represent large areas of approximately $100m^2$. A final difference is that our attention maps estimate the amount of attention that is received by a scene location rather than just cumulative gaze time.

The attention map is stored as a 2D grid where each square represents the level of attention received by one square metre of the ground. Let m be a vector from the pedestrian's 2D ground plane location to the centre of a square on the attention map. We would like to determine the probability that m is the subject of the pedestrian's attention, which is calculated from unit vectors g and w representing the gaze and walking directions respectively, as illustrated in figure 4.11. The attention received

is then calculated using the following:

$$P(a, l(m)|\theta, \phi) = P(a|l(m), \phi)P(l(m)|\theta) \tag{4.6}$$

When a pedestrian walks it is likely that they will be looking forwards but in most cases this will be to avoid collisions with objects rather than to look at a specific scene location. This is modelled using the random variable a, which is true if the location where the pedestrian looks actually has the attention of the pedestrian and false if the pedestrian is looking in the direction but not at anything specific. The function $l(m)$ is true when the pedestrian is looking at location m and false otherwise.

The value of $P(l(m)|\theta)$ was modelled as the product of two components. The first was based on the assumption that the probability of a scene location being viewed depends on the amount of space that the scene location occupies in the pedestrian's field of view and the second modelled the error in the gaze direction estimate as having a Gaussian distribution:

$$P(l(m)|\theta) \propto \frac{1}{|m|} e^{-\frac{\theta^2}{2\sigma_g^2}} \tag{4.7}$$

The value of $P(a|l(m), \phi)$ was estimated as $1 - P(\overline{a}|l(m), \phi)$, with $P(\overline{a}|l(m), \phi)$ depending only on the value ϕ which was modelled using a normal distribution with variance σ_a^2. In the absence of any empirical measurements, σ_g and σ_a were assigned the values 45° and 30° respectively. The result of this model is that pedestrians glancing away from their direction of motion are considered to be much more likely to be showing interest in a scene location than those who look in their direction of motion.

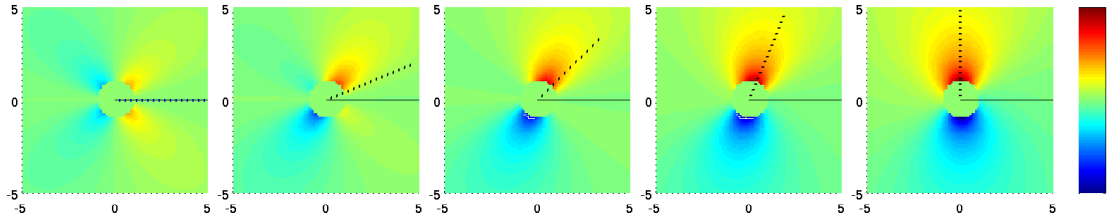


Figure 4.12: Examples showing the contribution to the attention map from a single frame where a pedestrian faces in different directions relative to their walking direction. The solid line shows the direction of motion and the dotted line represents the estimated gaze direction. The scale is shared between examples; red indicates large positive values and blue indicates large negative values. Gaze directions facing away from the direction of motion are more significant, so have a greater influence on the attention map.

4.4.2 Measuring Static and Transient Attention

Different approaches were used for measuring the focus of attention depending on whether stationary or moving objects were of interest.

Transient Attention To identify a transient subject of interest requires information from multiple people to resolve depth ambiguities. The attention map estimates from equation 4.6 were summed for each person over the most recent three frames before the product was taken over the people in the scene to identify intersections of their attention.

Static Attention If the focus of attention is assumed to be static, the locations of interest are found by accumulating the attention estimates over an extended period of time. There is however a problem with summing equation 4.6 because the attention model assumes that the focus of attention is more likely to be close to the pedestrian than far away. The result is that erroneous attention estimates accumulate along paths frequently travelled by pedestrians. To prevent this from occurring, the following function was accumulated instead:

$$\mathcal{A}(m) = P(a, l(m)|\theta, \phi) - P(a, l(m)|\theta + \pi, \phi + \pi) \tag{4.8}$$

This function sums to zero across the attention map so does not bias the attention map towards frequently travelled paths. The motivation for subtracting the probability of the subject of interest being in the opposite direction is that a pedestrian will only move their head away from a scene location if there is nothing of interest there. The function $\mathcal{A}(m)$ is plotted in figure 4.12 for different relative gaze directions.

4.4.3 Attention Measurement Evaluation

The tracking and head pose estimation were combined to make a fully automatic system which could be used to measure the amount of attention received by different areas of a scene. When applied to video sequences, the direction estimates from the randomised ferns were smoothed using a hidden Markov model to enforce temporal constraints. The gaze direction estimates were also limited to a 180 ° field of view around the direction of motion when people were moving at more than $0.63ms^{-1}$ (half the mean human walking speed). Using a GPU implementation of the HOG head detector [99], the complete system runs at 15 frames per second (fps) on 640×480 video or approximately 5fps on 1920×1080 video.

Three different video sequences were used to test different applications of the attention maps; details of all three can be found in section 2.6.1. The first experiment involved the analysis of the Town Centre video, which covers a busy town centre street with up to thirty pedestrians visible at a time. The aim was to identify areas receiving attention by accumulating gaze estimates over twenty-two minutes of video. The results from tracking approximately 2200 people are shown in figure 4.13. The frequently viewed areas that were identified by the gaze map highlight the shop window in the scene as a popular subject of attention.

In the second experiment, we attempted to artificially draw the attention of people to a particular location in the scene by attaching a light to the wall at eye level. For this experiment attention maps were generated both with and without

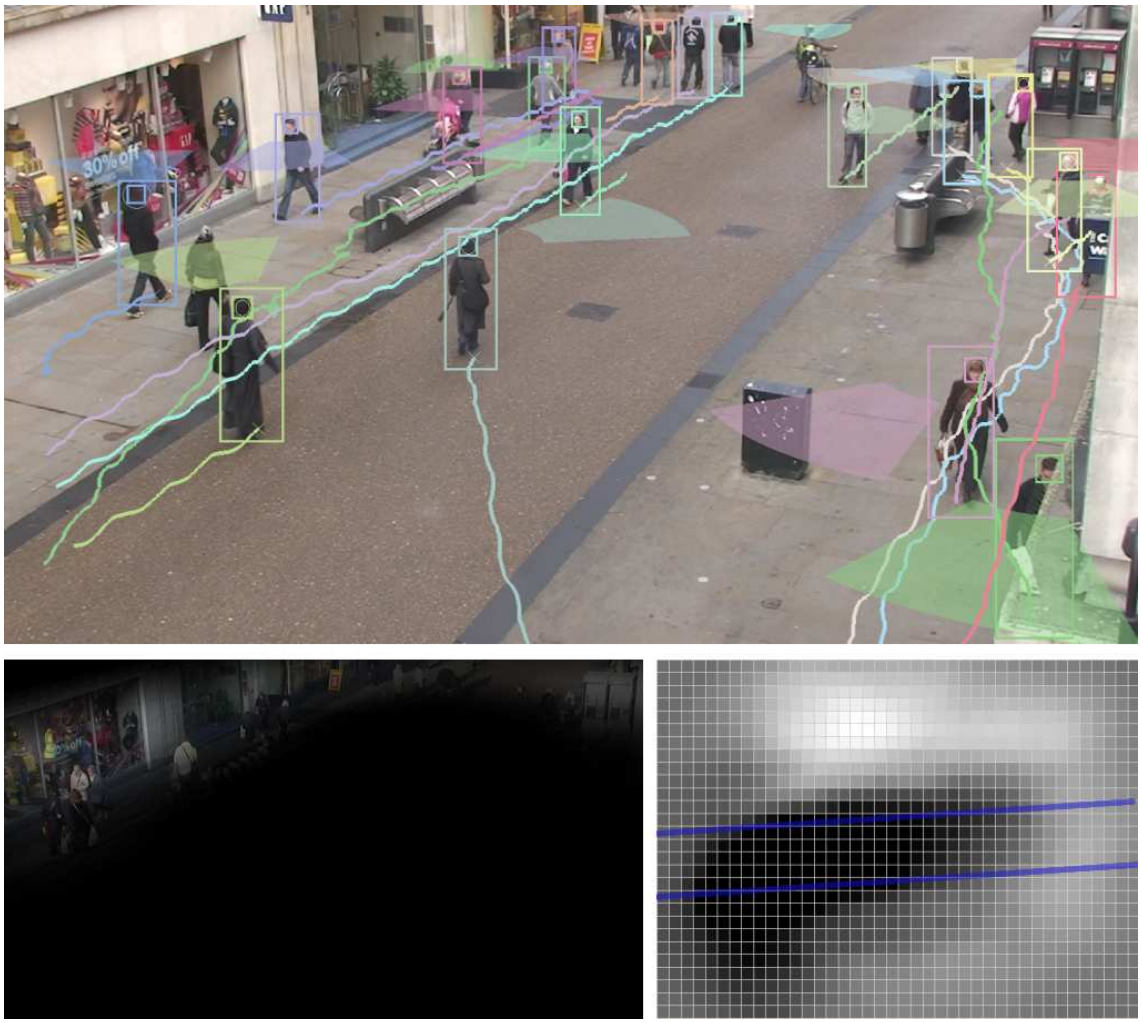


Figure 4.13: A frame showing the gaze direction estimates and the paths along which pedestrians were tracked. The lower images show the accumulated attention map and the result of projecting it onto a video frame, identifying the shop window as a popular subject of attention. The blue lines on the attention map show the edges of the road.

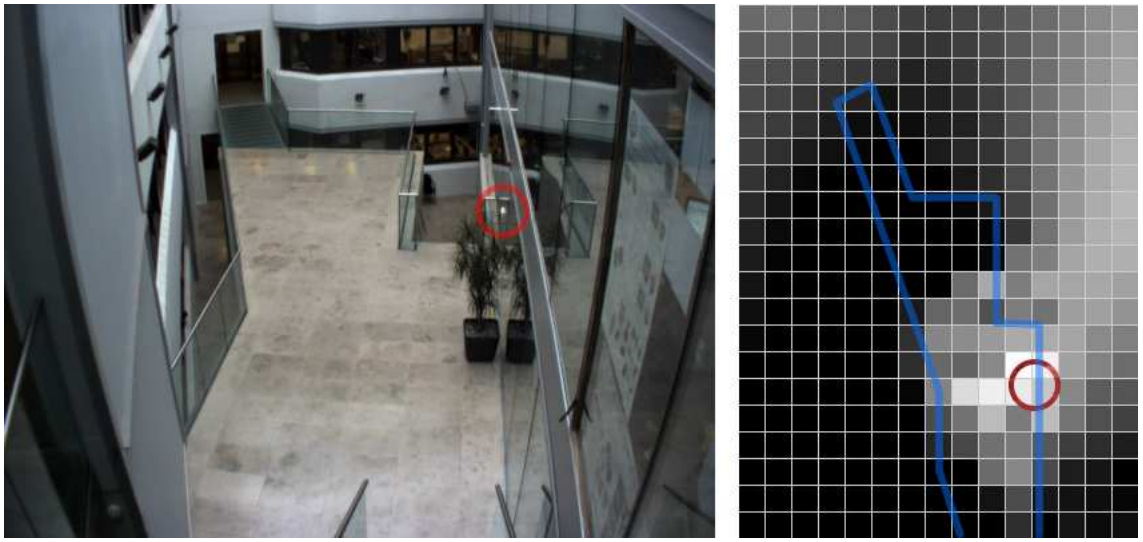


Figure 4.14: Attention map (right) resulting from attempts to artificially attract attention using a light mounted at eye level, the location of which has been marked with a red circle. Blue lines show the approximate floor outline. The extension of the highlighted region to the left is probably due to most observers walking along the left side of the walkway and the depth ambiguity.

the light stimulus. The accumulated map without the light was subtracted from the map that was built with the light present to correct for the stimuli normally present in the scene. The resulting attention map from tracking a total of 477 people over two hours of video is shown in figure 4.14. The location where the attention map measured the largest increase in attention is around the red circle, which marks the location of the light stimulus.

Where the purpose of the first two experiments was to measure the attention received by static objects, in the third the aim was to identify a transient source of interest. To resolve the ambiguities caused by not knowing the distance between the pedestrians and the subject of their attention, the gaze estimates from the two people in the scene were multiplied and combined over a sliding window of three frames. The resulting intersection, shown in figure 4.15 identifies the passing car as the subject of attention.

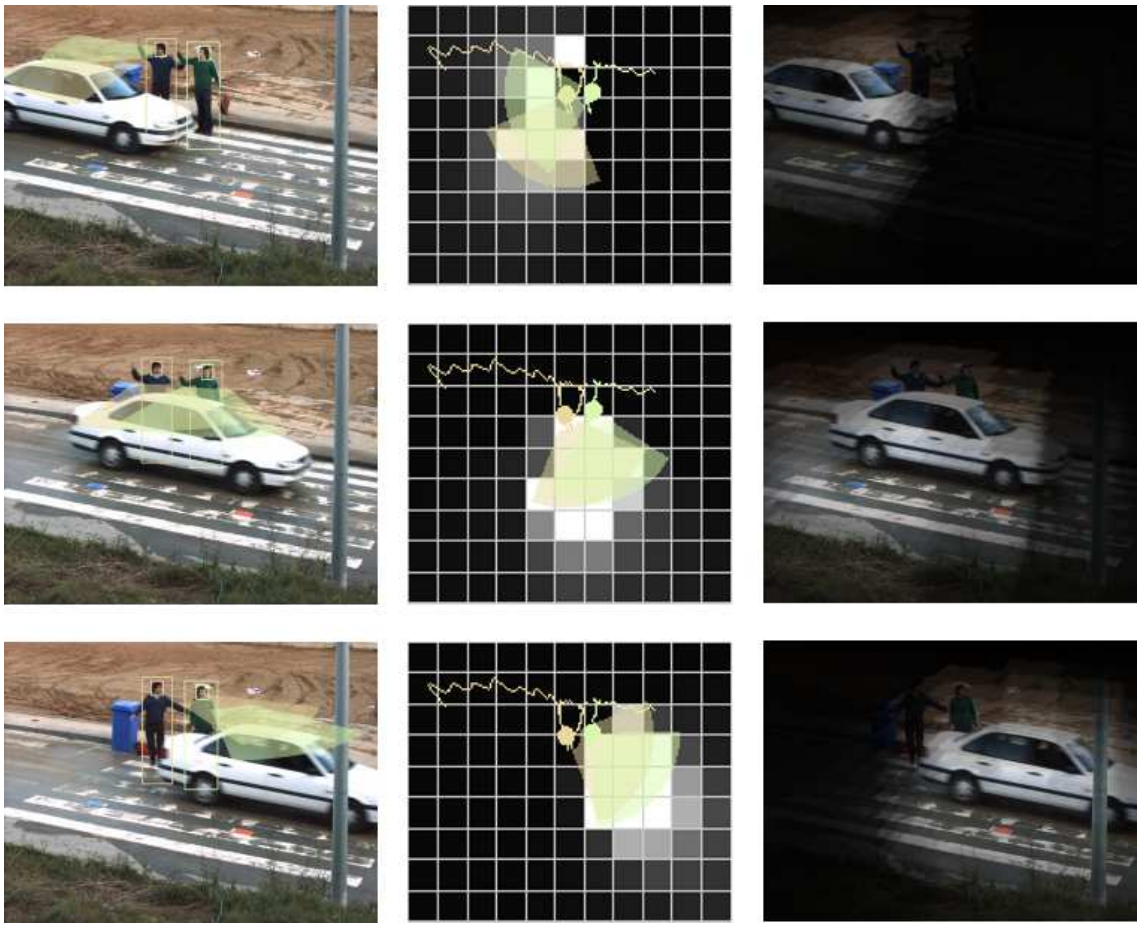


Figure 4.15: Sequence showing how the attention map can be used to highlight transient areas of interest. The left column shows video frames with annotated gaze directions, the middle column shows the corresponding attention maps and the third column shows the video frame modulated with the projected attention map, under the assumption that the subject of interest is between 0 and 2 metres above the ground

4.5 Conclusion

We have demonstrated a system that is capable of both automatically tracking a number of pedestrians in the presence of occlusions and which can estimate the amount of attention that the pedestrians give to different areas of the scene. In a simple scenario we demonstrated the measurement of transient interest, which could be used to guide a dynamic camera to observe the most interesting areas.

The following specific contributions were made:

- A multi-target tracking system was developed for the specific purpose of obtaining stable image sequences
- Robust randomised fern based gaze classifiers were developed, which included the new CTC image measurement to provide invariance to lighting effects.
- A complete system for measuring attention in large scale scenes was demonstrated. This is believed to be the first system to measure attention in general unconstrained scenarios.

The tracking system is robust and capable of tracking multiple people in low resolution video in real-time. The method for modelling the accuracy of motion estimates from corner features was shown to improve performance over naive alternatives.

The comparison of gaze estimation methods showed that the proposed HOG/CTC fern classifier outperforms the other methods that were tested. The extra experiments that were carried out determined the optimal training parameters for the gaze direction classifier. The error in head localisation was identified as a weak point in the system, with none of the attempted methods for improving invariance resulting in a worthwhile performance increase.

The attention maps from the three experiments demonstrate the potential of the system to provide useful information which could be used for higher level reasoning

or camera control, but there is still significant room for improvement. A limitation of the basic attention model used in this chapter is that it only allows us to determine the relative levels of interest between different scene locations. Although this allows the most observed locations to be identified, there is no way to determine whether or not the level of attention that was received is significant or not. This limits the applicability to fairly constrained application domains where, for example, one would like to compare the level interest received by two objects. Less well defined scenarios such as anomaly detection would require a more principled model to determine the absolute level of interest that is significant.

The method for determining transient subjects of interest is limited by the assumption that any overlap in the attention maps from different people is significant. The demonstration scenario involved only two people, but the number of overlaps will increase quadratically with the number of people. In more crowded situations, it would be necessary to consider all intersections jointly and use that fact that people are unlikely to observe more than one location simultaneously to develop a more sophisticated probabilistic model.

The prototype system that was developed in this chapter has provided a proof of concept, but also verifies the problem of misaligned head images that was predicted in chapter 3. The next chapter addresses this problem by improving the tracking algorithm to provide more accurate and stable head regions, as well as improved real-time performance and robustness. The method for learning classifiers that is presented in chapter 6 also attempts to improve performance with misaligned head images by allowing vast quantities of training data to be used.

A second issue that became apparent when the tracking system was tested on a variety of video sequences was that the tracker would often lose people who were standing still. The problem resulted partly from stationary people being difficult to distinguish from repeated false-positive detections in the same background location, but also because the use of KLT motion to detect people prevented stationary people

from being re-acquired after being lost by the tracker. The three test sequences used in this chapter contained mostly moving people, but in practice this is often not the case. The tracking algorithm that is developed in the next chapter also attempts to resolve this issue.

CHAPTER 5

Stable Multi-Target Tracking with Markov-Chain Monte-Carlo Data Association

This chapter describes the development of an improved tracking system with the specific aim of acquiring accurate head location estimates. The system tracks crowded scenes where pedestrians frequently occlude one another in real-time using Markov-Chain Monte-Carlo Data Association (MCMCDA) within a temporal sliding window. The accuracy levels are comparable to those previously only achievable through offline processing. A paper based on the work was published at CVPR 2011.

5.1 Introduction

Chapter 4 described the development of a working system for automatically tracking pedestrians, measuring their coarse gaze directions, and then using the results to infer properties of the scene. Having proven the value of the general concept, we now attempt to improve upon it. A significant issue affecting the accuracy of the system was shown to be that of incorrectly aligned head regions. Although the system typically tracks 80-90% of the visible pedestrians, in order to obtain good results from the classifiers, it was necessary to only make use of head regions from people for which the tracker covariance was very small, so only approximately 50% were used to generate the gaze maps. The high tracker covariances resulted from the HOG head detector firing infrequently on some people.

A second significant issue occurs when pedestrians stand still, making them difficult to distinguish from the background. The tracker from chapter 4 uses a series of rules based on motion to remove false positives, however this approach often results in genuine pedestrian tracks being deleted if the pedestrian stands still. The statistics of the head detections and motion estimates for a stationary pedestrian are very similar, so the information from one or two frames is not enough to reliably distinguish between them.

There are also various other failure cases that identify areas where the system could be improved. When pedestrians walk close to one another, the data association often becomes unclear when people are not detected for a few seconds at a time, the result being an *identity switch* where the tracker moves from one person to another. Also when a pedestrian's head is obscured, the KLT tracking can fail completely, causing the tracker to lose them. A final issue to be addressed is that of speed, since to be of value the system must be able to run in real-time.

Problem Statement: The work described in this chapter aims to develop a head tracking system that is more accurate than the Kalman filter based system of chapter 4. The two specific primary objectives are to generate more precisely

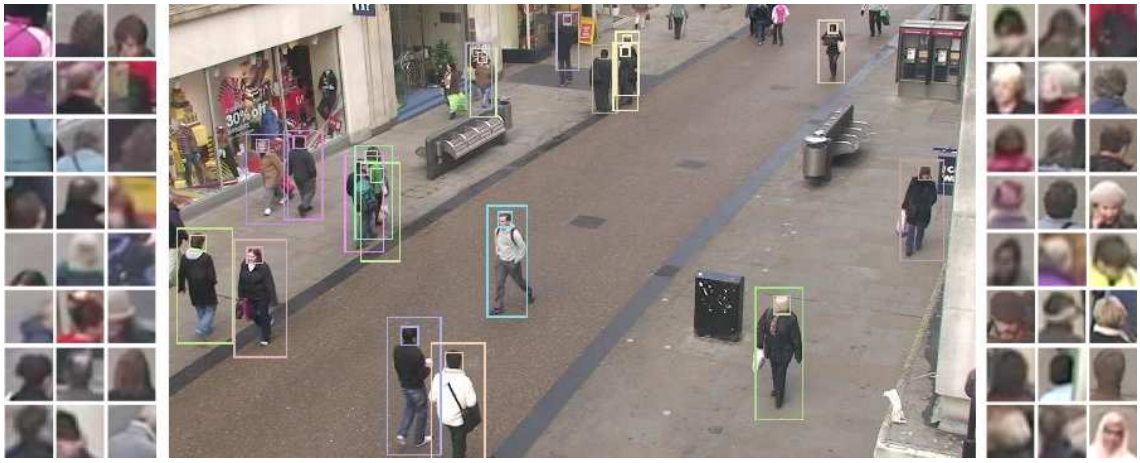


Figure 5.1: An example of a frame in which we would like to track (centre) and sample output from our system (right). The images on the left show the result of a naive approach which applies a fixed offset to a pedestrian detection to estimate the head location, the result of which is badly centred and drifts around as the pedestrian walks.

aligned bounding boxes around the head images and to increase the fraction of heads that can be reliably tracked. There are also two secondary objectives, which are a reduction in the number of identity switches and improved efficiency to allow video containing large crowds to be processed in real-time.

The remainder of this chapter covers the development of a tracking system that was designed specifically to address these issues, with the knowledge gained from the failure modes of the Kalman filter based tracker of chapter 4 guiding the design. The resulting tracker is able to achieve high levels of accuracy whilst maintaining real-time (25 fps) performance when tracking multiple pedestrians in high definition surveillance video. As before, we track the heads of pedestrians rather than their full bodies. The benefit in terms of tracker stability of tracking heads directly rather than taking an offset from a full body tracker are demonstrated in figure 5.1. To generate the images on the left, the mean offset between full pedestrian bounding boxes and the centre of their heads was calculated. This mean offset was used to crop the head images from the mean location within the full body bounding boxes. The resulting head images are poorly centred and drift around because the genuine

offset to the head changes as a pedestrian walks. The example head images on the right are much more accurate and are the result of tracking heads directly.

To cope with the complex data association problems that occur when tracking large crowds of people, we make use of Markov-Chain Monte-Carlo Data Association (MCMCDA), which has only recently been applied to surveillance video. Data association is the problem of matching image measurements (observations) to the target (pedestrian) that caused them to be made. We will use the term *track* to refer to the sequence of observations that have been associated with an individual target. MCMCDA optimises the data association hypothesis by considering the joint probability of all the observations from a short time period, allowing ambiguities to be easily resolved, and with the additional benefit of having no fixed requirement for processing time.

The system makes use of a temporal sliding-window so that the MCMCDA can use future as well as past information, which is of particular benefit when detections are missed. Traditional feed-forward methods for data association (see section 5.1.1) only use past information, which is often insufficient. Using a sliding window rather than an optimisation over the entire video sequence allows the tracker to be run continuously without limitations on the length of the video that can be processed, and also allows output for a video frame to be generated shortly after it is received.

A tracking model is developed to accurately represent the error accumulation and failure modes of the observations, which consist of both HOG head detections and short KLT motion estimates. This allows the frame-to-frame motion to be estimated with pixel-level accuracy and also provides inherent robustness to brief occlusions. The resulting location estimates are accurate enough for the generation of stabilised sequences of images from most of the pedestrians in a scene.

Next we consider the problem of false-positives. These frequently occur in the background when the detector fires on a fixed object and are often difficult to distinguish from people standing still. Most previous approaches have assumed

that false positives occur as individual uncorrelated detections, however in practice appearance-based detectors often generate repeated false-positive detections in similar locations. Nonetheless we observed that sequences of false-positives exhibit subtly different motion than sequences resulting from genuine people, who tend to move at least a small amount. Bibby and Reid [12] recently showed how multiple motion models could be used to distinguish moving objects from stationary ones in a SLAM (Simultaneous Localisation And Mapping) context using a temporal sliding window. We apply a similar approach to the domain of pedestrian tracking by treating the identification of false positives as a model selection process. Separate models are created for the stationary false-positives and moving pedestrians, and the identification of the appropriate model is combined with the data association.

Finally, the speed of the system is improved through the development of an efficient asynchronous multi-threaded architecture which ensures that the system is able to operate robustly under real-time constraints. In particular it is shown that the system is able to produce good results even if only a small latency is allowed. A comprehensive evaluation of the real-time tracker on two different datasets is performed using the standard CLEAR MOT (Multi Object Tracking) [11] evaluation criteria.

5.1.1 Data Association

To track an object, observations at regular intervals are required. The data association problem is that of determining which observations are from each target. The presence of false-positives, missed observations and multiple targets in similar locations can make the true data association difficult to determine. Since the first tracking systems were developed, various methods for solving the data association problem have been developed.

Nearest Neighbour (NN) is the most basic method for data association, and simply involves assuming that the closest observation to the predicted target location

is the correct one. An extension to situations with multiple targets is Global Nearest Neighbour (GNN). GNN prevents any observation from being assigned to more than one target by finding the globally optimal pairing of observations to closest targets, typically using the Hungarian algorithm [59].

One failure mode of NN and GNN occurs when the observations are equally likely, so there is a high probability that the wrong observation will be associated. Bar-Shalom and Tse's Probabilistic Data Association Filter (PDAF) [7] attempts to deal with this. Instead of accepting a single observation, an average of observations is taken, each weighted by their association probability. Although PDAF produces reasonable results for single targets, issues can arise when there are multiple targets present because individual observations can contribute to the updates of multiple targets. This problem was addressed by Fortmann et al.'s Joint Probabilistic Data Association Filter (JPDAF) [32] in which assignments of observations to targets are considered jointly, allowing multiple assignments to be prevented. The marginal assignment probabilities for each target are calculated and used as for PDAF.

A more robust solution is Multiple Hypothesis Tracking (MHT) [102], which maintains a set of hypotheses that are expanded at each time step using every possible combination of data associations. The hypotheses each maintain their own estimate of the target state. When the set of hypotheses grows too large, the set is pruned by removing the least likely hypotheses. Another robust solution is Reversible Data Association (RDA) [12], which maintains just a single hypothesis but allows the contributions of incorrect associations to be removed if it later becomes apparent that they were incorrect. This has the advantage of only requiring a single hypothesis to be stored.

5.1.2 MCMCDA

The tracking algorithm described in this chapter is based around MCMCDA, which involves the use of stochastic sampling to explore the space of possible data asso-

ciations. This has a number of advantages over other data association methods. Although MHT considers many hypotheses, the number of possibilities grows exponentially with time. For situations where there are many targets present, this could result in hypotheses being pruned before enough information has been gathered for their validity to be accurately determined. MCMCDA improves on RDA because it not only allows past data associations to be removed but also allows them to be added or reinstated.

The first systems to make use of MCMCDA were developed for tracking a single [10] or fixed number [96] of targets. Oh et al. [90] developed the approach for general multi-target tracking problems to associate sequences of absolute observations into an unknown number of tracks.

Later work developed MCMCDA tracking systems specifically for visual tracking by associating object detections resulting from background subtraction [135] and a boosted Haar classifier cascade [73]. The most recent work [35, 115] further specialises the approach for visual tracking by using not only object detections but also motion estimations or *tracklets* by applying a standard tracking algorithm for a short period of time after each detection. Our method also uses a combination of detections and motion estimates and bears closest resemblance to the work of Ge and Collins [35], however we make a number of improvements.

5.2 Sliding Window Tracking

Achieving real-time performance remains beyond the reach of most existing tracking-by-detection systems. We note that the detection stage is a bottleneck and most algorithms cannot run at full framerate, especially in high definition video. Dalal and Triggs' Histogram of Oriented Gradients (HOG) [29] based detector, one of the most popular and successful detectors, requires approximately 200-1200 milliseconds to run even when using GPU acceleration [99]. Our multi-threaded architecture

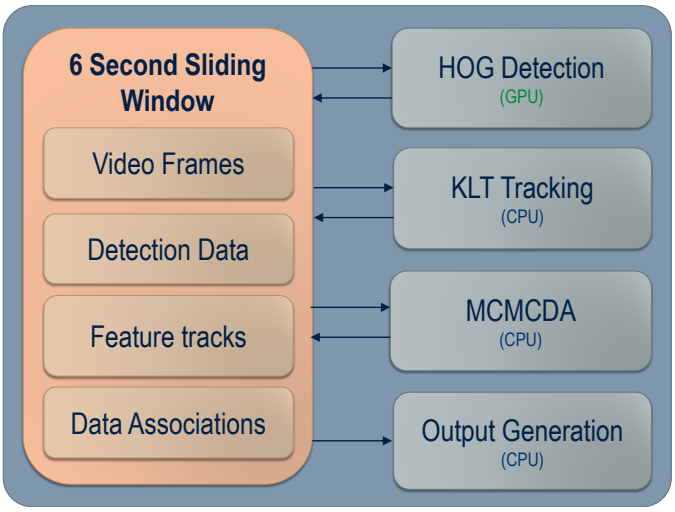


Figure 5.2: A block diagram for the MCMCDA tracking system. The boxes on the left show the information that is stored for the frames in the sliding window and the boxes on the right show the individual processes which all operate asynchronously in separate threads.

addresses this by having one thread produce asynchronous detections, while another performs local feature based tracking (KLT), a third performs data association and a fourth generates and optimises the output. The individual processes and the data they operate on are represented as a block diagram in figure 5.2. This approach ensures that the system is able to take full advantage of the processing time that is available whilst providing guaranteed real-time performance.

A key feature of our work is the use of MCMC data association within a temporal sliding window. The advantage of this approach is that at any instant in time the system can report its current best estimate of all target trajectories, but these may change either with more video evidence, or with further iterations. In particular this gives the system the ability to cope with full occlusions for short periods of time.

5.2.1 Observations

To make the tracking algorithm robust to false detections, the data association and location estimates are performed by considering all of the data within a sliding window representing the most recent six seconds of video that has been received.

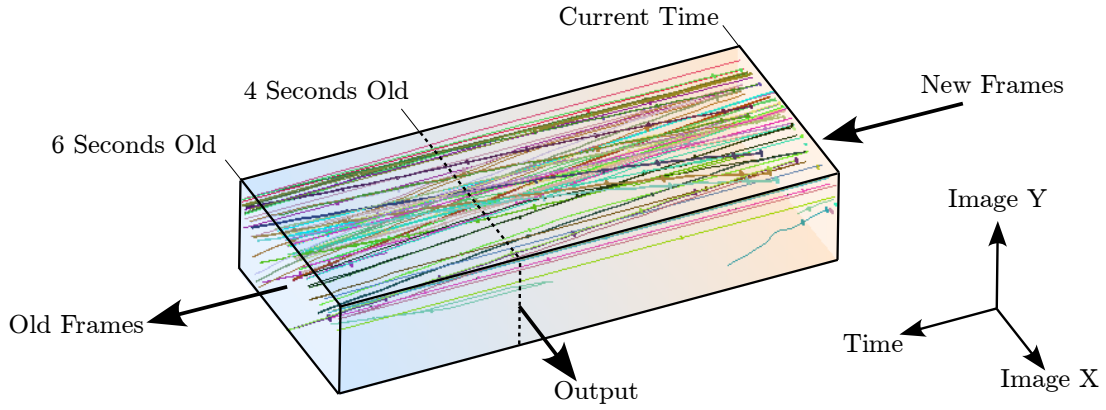


Figure 5.3: A diagram showing the passage of frames through the sliding window. Small rectangles represent head detections and lines represent KLT motion estimates. Captured frames enter the sliding window at one end and and spend a total of six seconds in the sliding window before they pass out of the other end and are deleted. Output for the frames is generated after they have been in the sliding window for four seconds.

We obtain object detections using Dalal and Triggs' HOG [29] detection algorithm for which we trained a detector using head images rather than full body images. Using a GPU implementation [99] of the HOG detector, detections are received at intervals from approximately 200 milliseconds for 640×480 resolution video to 1200 milliseconds for 1920×1080 video.

Since detections are received infrequently, motion estimates are necessary to ensure that data associations can be made correctly. As in chapter 4, we make motion estimates by following corner features with pyramidal Kanade-Lucas-Tomasi (KLT) tracking [75, 22]. To provide robustness against KLT tracking failure, up to four corner features are tracked both forwards and backwards in time from each detection for up to s seconds, so between any sequential pair of detections there will be relative location estimates in both directions. In practice we use a value of four seconds for s to limit the processing requirements and because associations are unlikely to be made for larger gaps. Figure 5.3 shows a 3D plot of the 2D image observation coordinates against the frame time within the sliding window to show how they can be linked together to form tracks. KLT tracking was chosen because

it is more precise than alternatives such as mean-shift and because tracking multiple corner features provides redundancy against tracking failures. For the purposes of of data association, the motion estimates will be considered part of the detection from which they originated so must be associated with the same target as a combined unit.

5.2.2 Data Association

The purpose of the data association stage is to select a hypothesis H_i which divides the set of detections D into disjoint subsets $T_1, T_2 \ldots T_J$ where each subset T_j contains all of the detections corresponding to a single person. Figure 5.4 shows an example frame with the corresponding image observations coloured according to the track that the data association hypothesis assigned them to. Since not every detection that occurs is a true positive, for each T_j we also attempt to infer the type c_j of the corresponding track. We use $c_j = c_{ped}$ to represent the property of T_j being a genuine pedestrian track or $c_j = c_{fp}$ if we believe T_j is a track of false positives, which will be abbreviated to just c_{ped} and c_{fp}. For more general situations, this variable could be extended to represent a number of different moving object types such as cars, bicycles, and trees, each of which would have an individual motion model to facilitate classification.

Exhaustively evaluating the space of hypotheses is too slow even for small sets of detections, so we use MCMC sampling to efficiently explore the space of data associations by generating a sequence $H_0, H_1, H_2, \ldots$ of sampled hypotheses. These sampled hypotheses will be distributed according to their relative probabilities which are defined by our likelihood function. The stochastic nature of MCMC helps to prevent the search from becoming stuck at local maxima of the likelihood function.

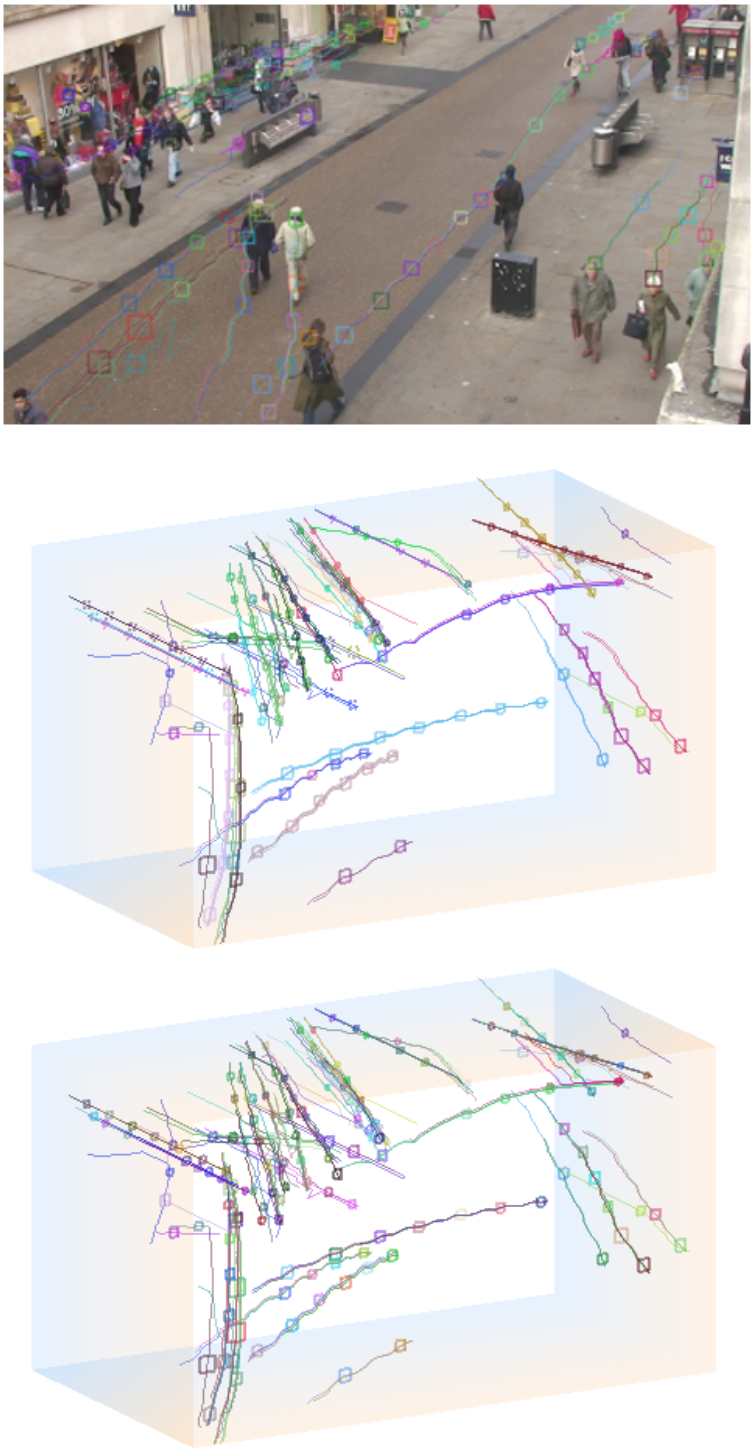


Figure 5.4: (requires colour) Views showing all of the head detections (small rectangles) and the corresponding KLT motion estimates (thin lines) within the sliding window. The colours represent the tracks to which the observations have been assigned. The top image shows the observations projected onto the current frame, the middle plot shows the data association hypothesis that has been made and the bottom image shows the result without data association.

Likelihood Function

Our likelihood function $p(H_i)$ is proportional to the probability of H_i representing the correct data associations and track types. In previous approaches, the likelihood function has been estimated as a product of a number of terms based on specific properties such as the length of tracks, velocity coherence, spatial overlap and the number of detections considered to be false alarms. We take an approach based on the Minimal Description Length (MDL) principle by attempting to find the hypothesis which allows the most compact representation of the observed detections.

MDL was used by Leibe et al. [66] to arbitrate between different combinations of tracks where the track likelihood functions included heuristics to encourage desirable properties such as long tracks. Our objective function does not use any heuristics and instead uses MDL to define every aspect of the objective function. The overall objective function used by Leibe also consisted of components with arbitrary weights applied to adjust their influence, so it did not correspond directly to the description length of the observed data. We avoid the practice of using arbitrary weights or costs for the encoding of values and instead assume that measurements are to be encoded to the same level of accuracy regardless of the hypothesis, which depends only on the probability density of the distribution used to encode them.

Being able to accurately estimate the likelihood of the data associations for a target is particularly important when there are multiple targets being tracked. For a single target, optimising the data associations using an objective function that is only a monotonic function of the true data association likelihood will often lead to the correct sequence being identified. With multiple targets, the likelihood of different combinations of competing tracks must be compared. We can arbitrate between two different combinations of tracks by taking the product of the individual track likelihoods for each and choosing the largest. If we have only a monotonic function of the individual track likelihoods, the product of the objective function across the two sets of tracks will not necessarily result in the same order of preference, so could

lead to the wrong combination being chosen.

The code length L required to encode both the data D and our hypothesis H_i to a given accuracy is dependent on a corresponding likelihood function:

$$L(D|H_i) + L(H_i) = -log(p(D|H_i)p(H_i)) \tag{5.1}$$

Although finding the hypothesis which minimises the description length is equivalent to maximising the joint likelihood of the data and the hypothesis, the principles of MDL guide the choice of likelihood function to one which allows observed variables to be encoded efficiently.

First we consider the encoding of the hypothesis H_i, which requires each detection d to be assigned to a track and each track to be given a type label. The track membership is most efficiently encoded when the prior over track membership has a distribution where probabilities are proportional to the track lengths, resulting in the following prior for H_i:

$$p(H_i) = J! \prod_{T_j \in H_i} \left(\frac{|T_j|}{|D|} \right)^{|T_j|} p(c_j) \tag{5.2}$$

where $p(c_j)$ is a prior over the different track types and the notation $|D|$ is used to denote the cardinality of the set D. The factor of $J!$ arises because the ordering of the subsets is not important, so the first detection in any track may be encoded with any of up to J identifiers which have not already been used.

Detections genuinely from the same track are expected to be highly correlated so can be efficiently encoded in terms of one another once divided into tracks. If a detection genuinely belongs to a track, we expect the information saved by encoding the detection as part of the track to be greater than the information required to encode the track membership, so the overall description length will be reduced.

Next we break down the likelihood function into components representing each track. Let d_n^j be the nth detection in a track T_j, where the index n indicates only

the order within the track:

$$p(D|H_i) = \prod_{T_j \in H_i} \left[p(d_1^j|c_j) \prod_{d_n^j \in T_j \setminus d_1^j} p(d_n^j|d_{n-1}^j, c_j) \right] \tag{5.3}$$

For each detection, we would like to encode the scale of the detection s_n, the location x_n within the frame, and an approximation to the KLT motion m_n. To ensure equivalent behaviour over different scales, the location accuracy is measured relative to the size of the object, so the units of x_n are multiples of s_n rather than pixels. Ideally we would consider the coding of the KLT motion estimates too, but due to the quantity of data this would have a negative impact on performance. Since the magnitude of the KLT motion is important for distinguishing between true positive and false positive detections, we instead approximate the motion by building a histogram m_n from the magnitude of every frame-to-frame motion estimate originating from detection n. The likelihood functions for individual detections are then broken down into components representing these observed properties:

$$p(d_1^j|c_j) = p(s_1)p(x_1)p(m_1|c_j) \tag{5.4}$$

$$p(d_n^j|d_{n-1}^j, c_j) = p(s_n|s_{n-1}, c_j)p(x_n|x_{n-1}, c_j)p(m_n|c_j) \tag{5.5}$$

Variables upon which the probabilities are conditional have been omitted where independence is assumed.

Detection Scales The scale of the first detection in each track cannot be encoded in terms of any preceding detection, so a global prior log-normal distribution with mean μ_p and variance σ_p^2 is assumed:

$$\ln s_1 \sim N(\mu_p, \sigma_p^2) \tag{5.6}$$

The scales for the following detections in the track can then be encoded more efficiently in terms of the previous scale:

$$\ln \frac{s_n}{s_{n-1}} | c_{ped} \sim N(0, \delta_t \sigma_{sp}^2) \tag{5.7}$$

$$\ln \frac{s_n}{s_{n-1}} | c_{fp} \sim N(0, \delta_t \sigma_{sf}^2) \tag{5.8}$$

where δ_t is the time difference between the frames in which the detections were made.

Image Location A similar approach is used when considering the optimal method for encoding the image location. It is assumed that the locations of both pedestrians and false-positives are uniformly distributed around the image, so the probability density of x_n depends on the image area a relative to the object size in pixels:

$$p(x_1) = \frac{s_k^2}{a} \tag{5.9}$$

For subsequent detections, the locations can be better explained in terms of the preceding detections, however the way in which we do this depends on the track type c_j. For genuine pedestrians, we first make a prediction based on a constant velocity model:

$$x_p = x_{n-1} + \delta_t v_p \tag{5.10}$$

$$\Sigma_p = \delta_t \Sigma_v \tag{5.11}$$

the velocity estimate v_p comes from the result of the KLT tracking in the frames immediately before and after the detection, by which point it is unlikely to have failed. The error in the velocity due to unknown accelerations is modelled by the parameter Σ_v. While the constant velocity model is an improvement on the uniform prior, the error Σ_p is still large partly due to the cyclic gait motion and also because

detections are infrequent and humans often change direction when in crowds. The full KLT motion estimates generally provide much more accurate predictions, so for each KLT motion estimate y we calculate a posterior distribution over locations using a calculation equivalent to the update step of a Kalman filter:

$$x_y = x_{n-1} + \Sigma_p(\Sigma_p + \delta_t\Sigma_{klt})^{-1}(x_{n-1} + y - x_p) \tag{5.12}$$

$$\Sigma_y = (I - \Sigma_p(\Sigma_p + \delta_t\Sigma_{klt})^{-1})\Sigma_p \tag{5.13}$$

The parameter Σ_{klt} represents the rate at which KLT feature tracking accumulates random error and δ_t is the time difference between detections d_n and d_{n-1}. The possibility that a tracked KLT feature fails completely is modelled using the parameter α, where $1 - \alpha^{\delta_t}$ is the probability of failure after tracking for δ_t seconds. The detection location is then encoded using a mixture of the prior and posterior distributions:

$$x_n|x_{n-1}, \mathbf{y}, c_{ped} \sim \alpha^{\delta_t}\frac{1}{|\mathbf{y}|}\sum_{y\in\mathbf{y}} N(x_y, \Sigma_y + 2\Sigma_d) + (1 - \alpha^{\delta_t})N(x_p, \Sigma_p + 2\Sigma_d) \tag{5.14}$$

Some plots of the probability density that this equation predicts for the vertical image location of a target over the following three seconds are shown in figure 5.5 to provide some insight into the properties of the individual components.

In the event that $\mathbf{y}$ is empty, α is set to 0 and the first term is omitted. The additional uncertainty of $2\Sigma_d$ is included to model the error in the two detection locations. Tracks consisting of repeated false-positives are usually caused by static objects in the background so are assumed to be stationary:

$$x_n|x_{n-1}, \mathbf{y}, c_{fp} \sim N(x_{n-1}, 2\Sigma_d) \tag{5.15}$$

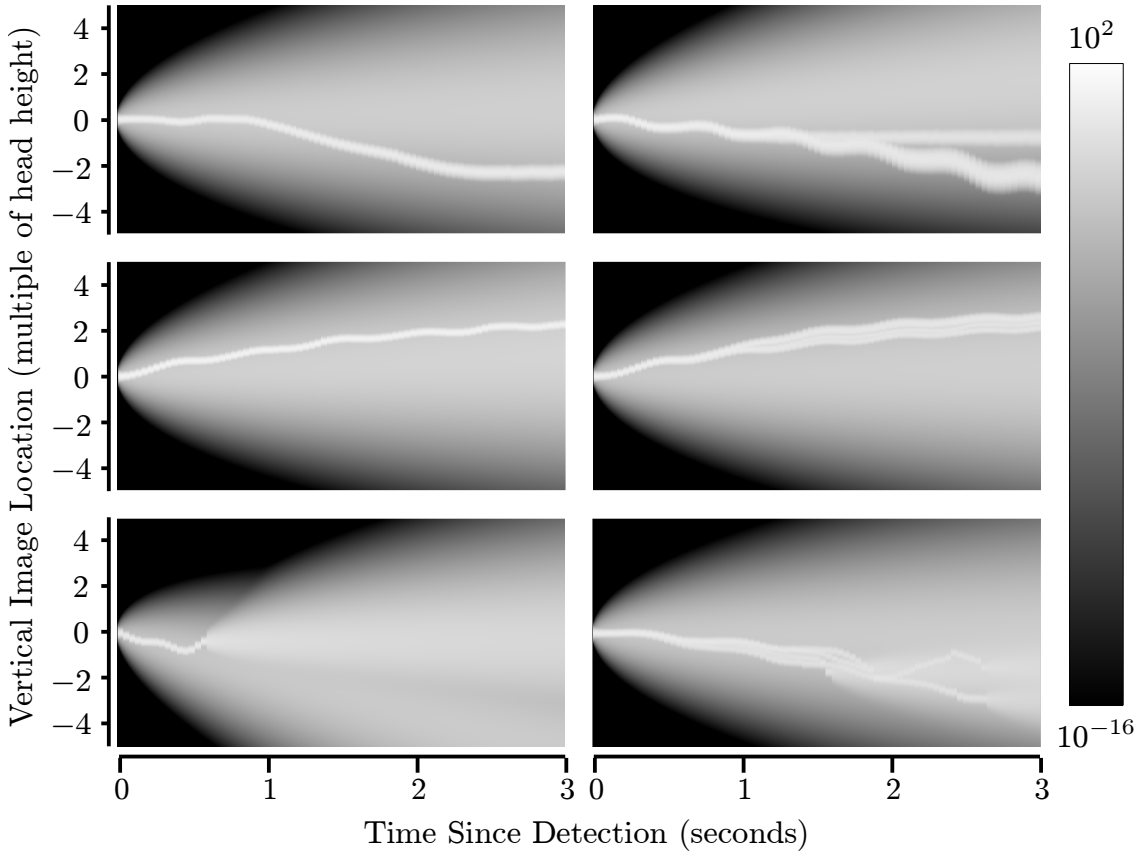


Figure 5.5: Plots of the log probability density for the location of one detection relative to the previous one. The broad component corresponds to the constant velocity prior, which represents approximately 5% of the total probability mass after one second. The narrow bands come from KLT motion estimates which accumulate error much more slowly. The left three plots only use one KLT feature and the right three plots use four features. The bottom two plots show examples where the KLT features have either been lost or not yet tracked for the full three second duration.

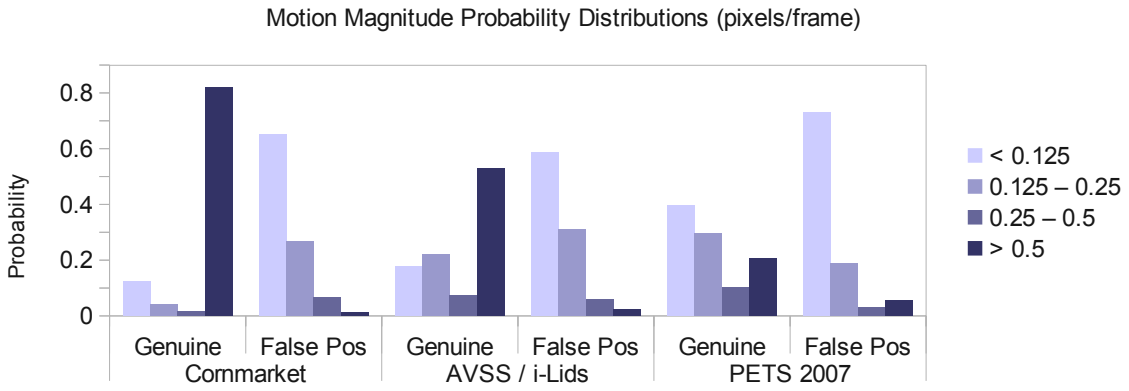


Figure 5.6: Motion magnitude probability distributions for false positives (m_{fp}) and genuine pedestrians (m_{ped}) in the three scenes used for evaluation. The distributions are the parameters to the multinomial distributions and were learned automatically.

Motion Magnitude The last observation considered is the motion magnitude histogram. This is included only to help distinguish between false positives which are expected to have no movement and true positives which are expected to move at least a small amount, so the histogram has just four bins with boundaries representing movement of $\frac{1}{8}$,$\frac{1}{4}$ and $\frac{1}{2}$ pixels per frame. The histograms are expected to conform to one of two multinomial distributions depending on the type of track:

$$m_n|c_{ped} \sim Mult(m_{ped}) \tag{5.16}$$

$$m_n|c_{fp} \sim Mult(m_{fp}) \tag{5.17}$$

The probability distributions that were learned for the three scenes used in the evaluation are shown in figure 5.6 to demonstrate the effectiveness of the motion magnitude for discriminating between false positives and genuine pedestrians.

Throughout this section the probability of each detection being in a track has been calculated by considering only the immediately preceding and immediately following detections. This approximation was made to ensure that the MCMC data association can be performed efficiently, with each proposal requiring only a minimal calculation. The optimisation of the joint probabilities is deferred to section 5.2.3.

Sampling

There are three types of move which can be made during the sampling process; the first two moves affect the state of the data association and the third has the potential to change the type of a track. The first type of move involves randomly selecting one detection and one track, and proposing that the chosen detection be moved to the chosen track, as illustrated in figure 5.7. In the event that the track already contains a detection from the same frame, both are swapped in the proposal. In the second type of move we choose two tracks and a random time within the sliding window and propose that all of the detections occurring after the time in both of the tracks are swapped with those in the other.

Calculating the likelihood $p(H^*)$ of the first proposal requires at most four evaluations of equation 5.5 and the second requires no more than two. The third move type, in which a change of track type is proposed, requires the probability of every detection in a single randomly chosen track to be re-evaluated. Fortunately this third move type depends on just one track so does not need to be attempted as frequently.

The Metropolis-Hastings acceptance function defines the likelihood with which the proposal should be accepted:

$$p(H_{i+1} \Leftarrow H^*) = \min\left(\frac{p(H^*)q(H_i|H^*)}{p(H_i)q(H^*|H_i)}, 1\right) \tag{5.18}$$

In most cases the proposal density q will be the same for both the forward and the reverse move, however there are some cases where it is not. Random tracks for proposals are drawn from the set of existing tracks plus one additional empty track. This empty track allows a new track to be created, and similarly either of the two moves could leave one of the tracks empty, in which case it is destroyed. Since only one empty track is retained, the creation or destruction of a track affects the probability of the move being made.

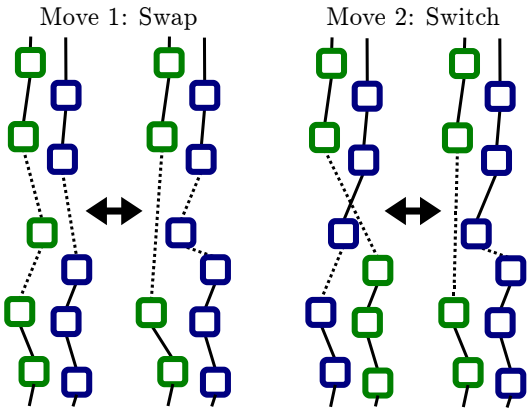


Figure 5.7: Examples of the first two types of move used for MCMCDA. Only the probabilities for pairwise associations with dotted lines need to be recalculated when each move is proposed.

Although Metropolis-Hastings is good at overcoming local maxima of the likelihood function, we prefer stable output rather than samples. To obtain stable output we keep track of the most likely hypothesis since observations were last received and output the local maximum, which is found by only accepting proposals that are more likely than the current hypothesis for a short period of time.

Parameter Estimation

Some of the model parameters such as the detector covariance are likely to depend on characteristics of the particular video sequence such as the level of image noise and blur. In our system these are learned automatically using an approach based on that of Ge and Collins [35] by interleaving the MCMCDA sampling with additional Metropolis-Hastings updates of the parameters. Provided the parameters are initialised to values allowing some tracks to be correctly associated, both the parameter samples and the data association converge to a good maximum of the likelihood function. Parameter updates take considerably longer than data association updates because the log-likelihood must be recalculated for all of the data. In a live system the parameters could be learned online over an hour or two. However, since most datasets are too short for this, we slow down the video used for training

so that there is enough time to learn the parameters.

5.2.3 Output Generation

The final stage is to generate estimates for the object location in each frame. First we estimate the true image locations $\hat{x}_n$ for all of the detections in each track:

$$p(\hat{T}_j) = p(x_1|\hat{x}_1)p(\hat{x}_1) \prod_{1<n\leq N} p(x_n|\hat{x}_n)p(\hat{x}_n|\hat{x}_{n-1}, \mathbf{y}, c_{ped}) \tag{5.19}$$

$$x_n|\hat{x}_n \sim N(\hat{x}_n, \Sigma_d) \tag{5.20}$$

The term $p(\hat{x}_n|\hat{x}_{n-1}, \mathbf{y}, c_{ped})$ is equivalent to equation 5.14 but without the $2\Sigma_d$ terms, and $p(\hat{x}_1)$ is equivalent to equation 5.9. Since multiple KLT estimates result in multimodal probability distributions, we again optimise using Metropolis-Hastings sampling.

Since detections do not occur in every frame, the location estimates for the other frames are made by interpolating between detections. Interpolations are made by averaging each of the relevant KLT motion estimates, weighted by the corresponding contribution to the mixture in equation 5.14.

5.3 Evaluation

The purpose of our system is to provide stable head location estimates in surveillance video, however there are no standard datasets for evaluating this so we use our own video dataset of a busy town centre street. The video is high definition (1920×1080 at 25fps) and has ground truth consisting of 71500 hand labelled head locations, with an average of sixteen people visible at any time. This video sequence with the ground truth data and our tracking output has been made publicly available to encourage future comparisons. For more information on this and other video datasets, see section 2.6.1.

5.3.1 Evaluation Criteria

Four measurements are used to evaluate the tracking performance. These use counts of number of true positive n_{TP} and false positive n_{FP} estimations made by the algorithm, the number of false negative regions n_{FN}, and the number of identity switches n_{ID}. An estimation is a true positive if it can be matched with a ground truth region and a false positive if there is not corresponding ground truth region. False negatives are ground truth regions that were not matches with any estimated regions, and identity switches occur when either the estimated identity of a ground truth track changes or when the tracker estimate assigns the same identity to two different ground truth tracks.

The first two measurement types were proposed by the CLEAR evaluation project [11]. Multiple Object Tracking Precision (MOTP) measures the precision with which objects are located using the intersection of the ith detected region R_i^D with the ground truth region R_i^{GT}:

$$m_{MOTP} = \frac{1}{n_{TP}} \sum_{i=1}^{n_{TP}} \frac{|R_i^D \cap R_i^{GT}|}{|R_i^D \cup R_i^{GT}|} \tag{5.21}$$

The modulus symbols are used here to denote the area of the union $\cup$ and intersection $\cap$. Since the CLEAR MOT metrics do not precisely define how to determine whether two rectangles match, the requirement for body regions was $|R_i^D \cap R_i^{GT}| > \frac{1}{2}|R_i^D \cup R_i^{GT}|$ as used by Stalder et al. [117]. Since head regions are significantly smaller, the threshold ratio was reduced to $\frac{1}{4}$ for a match to avoid errors in the ground truth from skewing the results. Note that the effect of changing the threshold is fairly neutral, since higher values result in more false negatives and lower values result in more false positives.

Multiple Object Tracking Accuracy (MOTA) is a combined measure which takes

into account false positives, false negatives and identity switches:

$$m_{MOTA} = 1 - \frac{n_{FP} + n_{FN} + n_{ID}}{n_{TP} + n_{FN}} \tag{5.22}$$

The second two performance measures are well established for object detection and are included to enable comparisons with other work. The detection precision measures the ability of a system to return a high ratio of true positives to false positives:

$$m_P = \frac{n_{TP}}{n_{TP} + n_{FP}} \tag{5.23}$$

Finally the recall measures the proportion of ground truth regions that were found:

$$m_R = \frac{n_{TP}}{n_{TP} + n_{FN}} \tag{5.24}$$

5.3.2 Experiments

Since head regions are considerably smaller than full body boxes, any error in the location estimates has a much more significant impact on the performance measures than for the full body regions. For this reason the two should not be directly compared, however to allow some comparison to be made with full-body trackers, we also calculate the performance measures using full-body regions that are estimated from the head locations using the camera calibration parameters and a known ground plane. All experiments were performed on a desktop computer with a 2.4GHz quad-core CPU with GPU [1] accelerated HOG detections and in real-time unless otherwise stated. Although the model allows for multiple motion estimates per detection, for the experiments only one was used because with the processing resources that were available the cost of calculating multiple motion estimates outweighed the benefit.

The results for the town centre video are shown in table 5.1. Since the dataset was only recently released, the tracker could only be compared with the Kalman

[1] An Nvidia GeForce GTX 295 card was used

filter tracker from the previous chapter and baseline results from raw HOG head and full body detections. The sliding window tracker improves the accuracy of the head location estimates and has approximately double the MOTA of the Kalman filter tracker. To demonstrate the benefits of two of the contributions made in this chapter, experiments were performed with the KLT failure model and the separate false-positive model each disabled. Both resulted in a significant reduction in the MOTA and either the precision or recall.

We also examine the effect of adjusting the latency between when frames are received and when output is generated for them (figure 5.8) because this is relevant for many practical applications. Our results show that the drop in performance when the latency is reduced to 1.5 seconds is small and below that the performance degrades gracefully.

Some insight into the cause of most tracking failures can also be gained from figure 5.9, which shows how tracking at lower speeds affects the performance. The result is that although more frequent detections increase the recall, the precision drops because there are more false positives. These false positives are the result of incorrect head detections on other body parts such as shoulders or on bags and often occur repeatedly in the same place as the pedestrian moves. A potential solution to this problem would be to model the distribution of false positives relative to true positives.

Although the system we describe was intended for the purpose of obtaining stable image streams, we also demonstrate the general tracking performance by performing a quantitative analysis on a standard test video from the i-Lids AVSS 2007 dataset. The video is of a train station platform, has a resolution of 720 x 576 pixels at 25 fps and has 9718 labelled ground truth regions. Since the video includes distant people that are too small for the head detector to detect, head detections were interleaved with the standard full body HOG detector with a fixed offset to estimate the head location. The covariance of the full body detector, which was also learned

	Method	MOTP	MOTA	Prec	Rec
Head Regions	Our tracking	50.8%	45.4%	73.8%	71.0%
	Our tracking without KLT model	50.9%	38.0%	72.9%	61.1%
	Our tracking without FP model	51.2%	22.4%	59.5%	71.5 %
	Kalman Filter	44.8%	22.5%	65.3%	48.1%
	HOG head detections	45.8%	-	35.0%	52.7%
	HOG body detections	44.3%	-	44.7%	39.3%
Body Regions	Our tracking	80.3%	61.3%	82.0%	79.0%
	Our tracking without KLT model	80.2%	53.5%	82.2%	68.9%
	Our tracking without FP model	80.2%	41.2%	67.4%	81.0%
	Kalman Filter	76.2%	58.6%	90.1%	66.2%
	HOG head detections	76.1%	-	49.4%	74.5%
	HOG body detections	72.7%	-	82.4%	72.3%

Table 5.1: Tracking performance on the town centre sequence compared to the Kalman filter based tracker with baseline estimates using raw HOG head and full body detections, and results for our tracking system with the KLT failure model or false-positive (FP) model disabled. Body regions from the HOG detector were converted to head regions using a fixed offset and head regions were converted to body regions using the camera calibration. MOTA takes into account identity switches so cannot be calculated without data association. The HOG detectors were applied to every video frame, which is approximately ten times too slow to run in real-time.

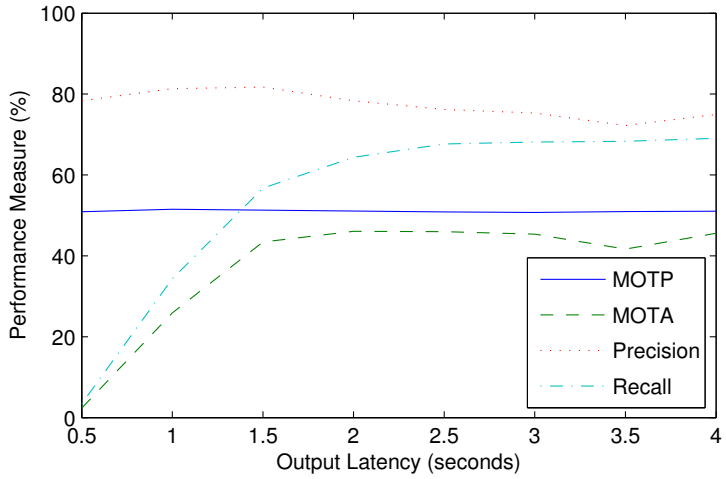


Figure 5.8: Reducing the latency introduced between frames arriving and their output being generated causes the recall to decrease. Performance measures are for head regions.

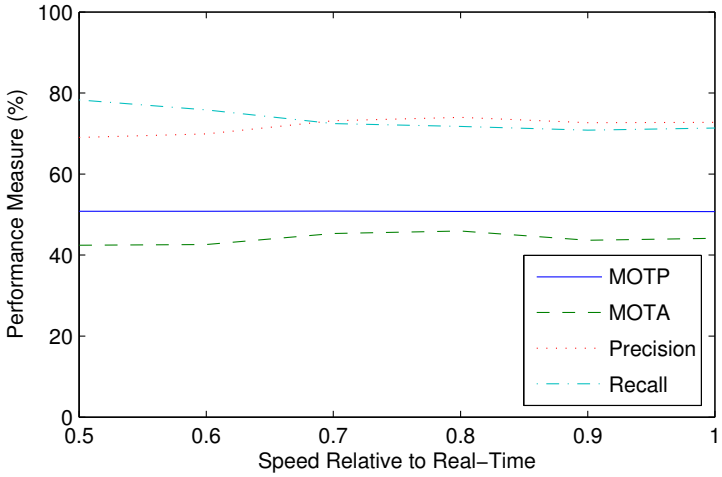


Figure 5.9: A graph showing the effect of slowing down the system so that more time is available for processing the video. Performance measures are for head regions.

automatically, is large enough to have little influence when the head detector worked but could maintain the track when it did not. Table 5.2 compares our results with those of two recent state-of-the-art systems, both of which are unable to process video in real-time.

Our implementation continually optimises the data association hypothesis, but to measure the efficiency an experiment was conducted where the data association was disabled until the six second sliding window had been filled with data. The data association was then enabled and the result used to analyse the performance in a controlled situation, the results of which are shown in figure 5.10. The optimisation converges after approximately 200 milliseconds, which is only 3% of the time taken to accumulate the data. Although our system continually optimises the data association, the purpose is to immediately incorporate new observations, and these results show that considerably less processing time is actually required.

An unexpected benefit of using MDL to model the hypothesis likelihoods is that the system can cope with total failures of the KLT tracking and HOG detections for short periods of time, which often occurs when an individual is fully occluded. Some examples from the Town Centre and i-Lids datasets are shown in figure 5.11. Despite no explicit planning for these circumstances, the tracks continue through the

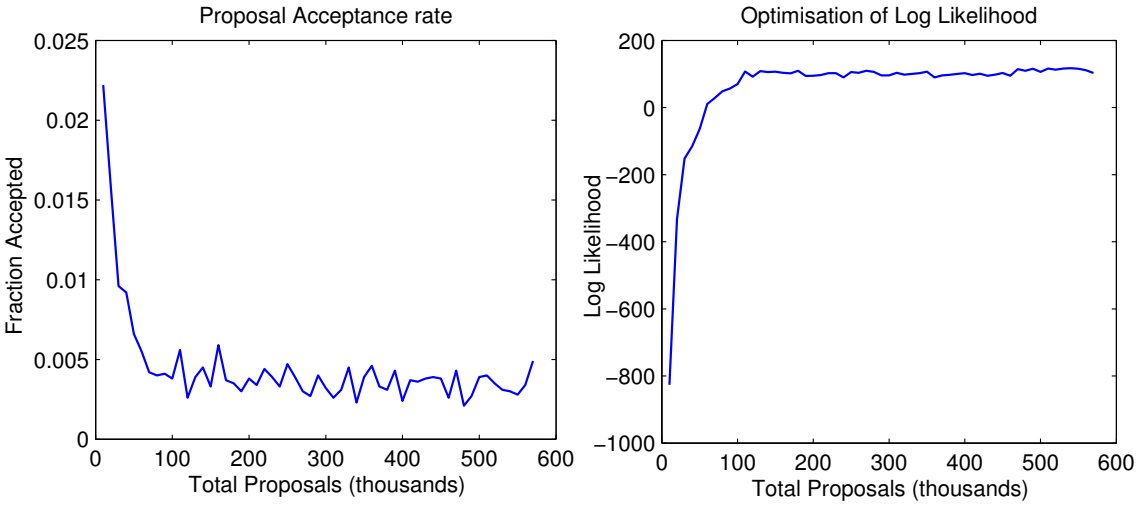


Figure 5.10: Graphs showing properties of the MCMC optimisation when applied to unassociated data. The left graph shows the acceptance rate for proposals and the right graph shows the total log-likelihood. The total of 570000 proposals took one second to evaluate, of which approximately 200ms was required for the mode to be reached.

Method	MOTP	MOTA	Prec	Rec
Ours	73.6%	59.9%	80.3%	82.0%
Breitenstein et al.	67.0%	78.1%	-	83.6%
Stalder et al.	-	-	89.4%	53.3%

Table 5.2: Tracking performance of our system compared to that of Breitenstein et al. [18] and Stalder et al. [117]. Results are shown for full body regions on the i-Lids AB Easy sequence using both the CLEAR MOT metrics and standard detection precision and recall.

occlusions with paths interpolated according to the motion priors from the closest observations before and after.

In addition to these datasets, figure 5.12 also shows qualitative results on the PETS 2007 videos, for which we do not have ground truth data, to demonstrate the ability of the system to cope with dense crowds of people. Some sample sequences of cropped head images are shown in figure 5.13 to demonstrate the stability of the tracking.

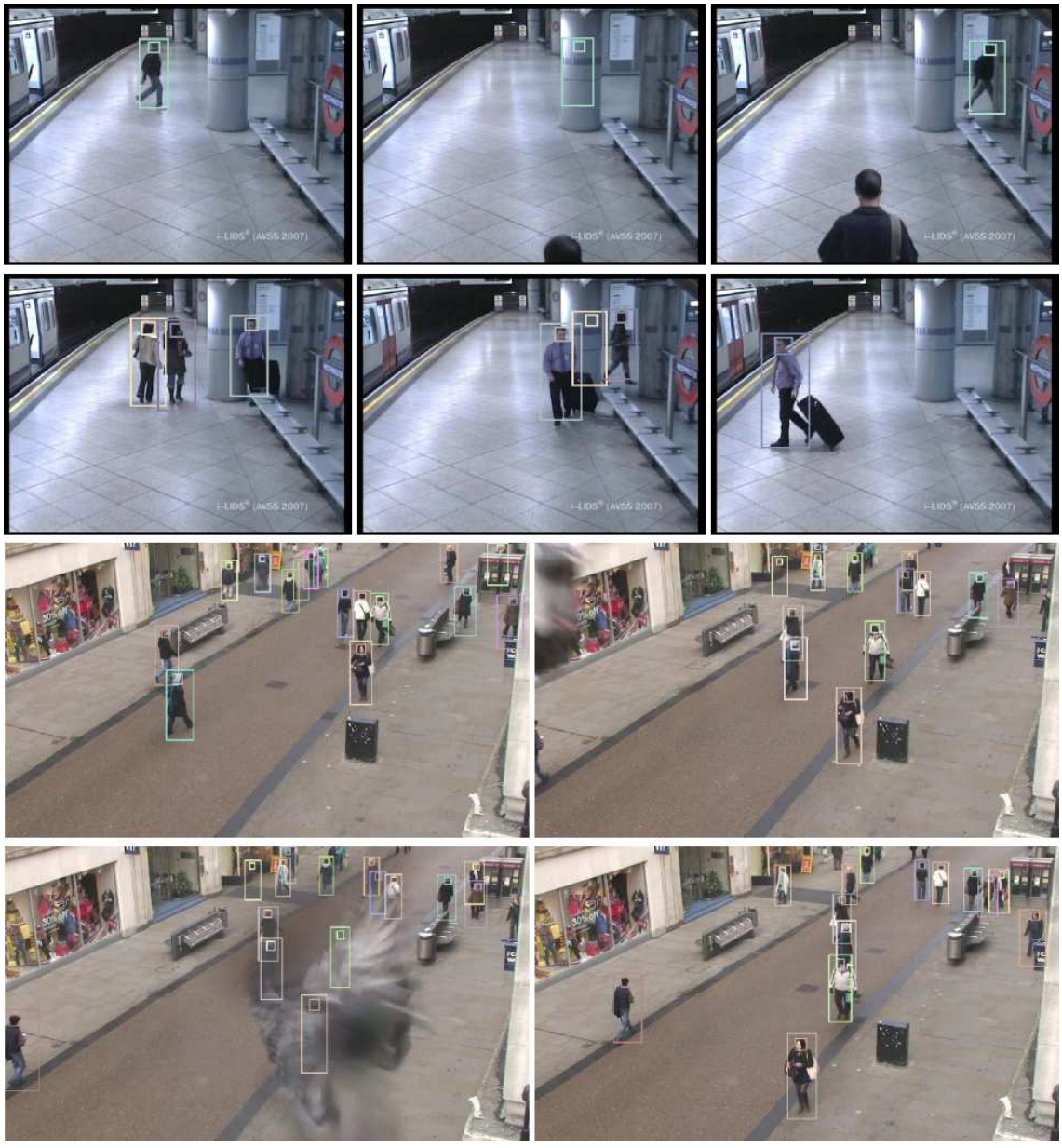


Figure 5.11: Sample video frames from the i-Lids dataset (top), and the town centre dataset (bottom). The two sequences demonstrate the ability of the system to deal with temporary occlusions, caused by the pillar in the i-Lids video and by a pigeon in the town centre video. The bottom rows show examples of the stable head sequences that result from the tracking.

Figure 5.12: Tracking results from the PETS 2007 dataset. Using the head as the point of reference allows most of the people in the crowd to be tracked even though they occlude one another. The false negatives at the top are caused by the heads being too small; this could easily be resolved by using higher resolution video.

Figure 5.13: Examples of the stable head sequences that result from the tracking. The head remains well centred in the image which is critical for good classification performance.

5.4 Conclusion

This chapter has described the development of a scalable real-time system for obtaining stable head images from pedestrians in high definition surveillance video. A number of contributions were made, primarily to address the issues that were encountered with the tracking system of chapter 4:

- A principled objective function was developed, allowing accurate location estimates and robust data associations to be made, including the ability to track through total occlusions.
- A new move type for MCMCDA was introduced to allow the removal of false positives, which could be also extended to identify different object types.
- An efficient and scalable multi-threaded architecture was proposed to allow large crowds of pedestrians to be tracked in real-time.

The use of MCMCDA makes the system robust to occlusions and allows false positive detections in the background to be identified and removed. The system has many potential applications in activity recognition and remote biometric analysis or could be used to provide close-up head images for a human observer. The experiments performed show that the efficient approach provides general tracking performance comparable to that of state-of-the-art offline tracking systems, whilst being advantageous in terms of speed and tracker stability.

The MCMCDA based tracker is not only able to deliver more accurate head regions than the Kalman filter based tracker, but is able to provide them for 71% of targets rather than 48% while achieving real-time performance. Although the error in the head locations has been reduced, it has not been removed completely. The next chapter addresses the problem by making a gaze classifier that is more robust to the remaining error.

CHAPTER 6

Unsupervised Coarse Gaze Direction Estimation

In this chapter a method is presented for learning a coarse gaze direction classifier using the output from an automatic tracking system without any hand labelled image data. A Conditional Random Field is used to model the interactions between the head motion, walking direction, and appearance to recover the gaze directions and simultaneously train randomised decision trees. The resulting direction estimations outperform conventionally trained classifiers on two large surveillance datasets. A paper based on the research described in this chapter was published at ICCV 2011.

6.1 Introduction

The previous chapter addressed the problem of misaligned head regions, with the resulting sliding window based tracking algorithm giving a significant improvement over the previous feed-forward approach. The research presented in this chapter was motivated by the need to provide invariance to the remaining head location error, but also by some other observations that were made in chapter 4.

One of the experiments from section 4.3.5 examined the possibility of making classifiers more robust to location error by introducing random noise into the head locations of the training data. The result was that each classifier performed best when the noise in the test data was approximately the same as that introduced during training, but when the test data had less noise than the training data the performance was reduced. In particular, with no noise in the test data, training with noise in the head locations considerably lowered the performance. Unless we could accurately determine the amount of error in the head location for an unknown video sequence, the method would be of little value.

A second observation comes from the experiments in section 4.3.5 where classifiers were tested and trained on the same datasets (more extensive results can be found in appendix 8.2). In all of the video datasets, every classifier performed considerably better if the training and testing data were taken from the same dataset. Again, these were unrealistic scenarios because we cannot expect to have a custom hand-labelled training dataset for every surveillance installation.

In both cases, the potential benefit could not be realised in any genuine installation because the cost of the human input required to customise the system for the particular environment would be prohibitively expensive. Most of the existing approaches that were reviewed in chapter 2 do not recognise this, and include either the same people or the same scenes in both training and test datasets. There is however a way to avoid the problem altogether, which is to develop a method for estimating gaze directions that does not require any hand labelled training data at

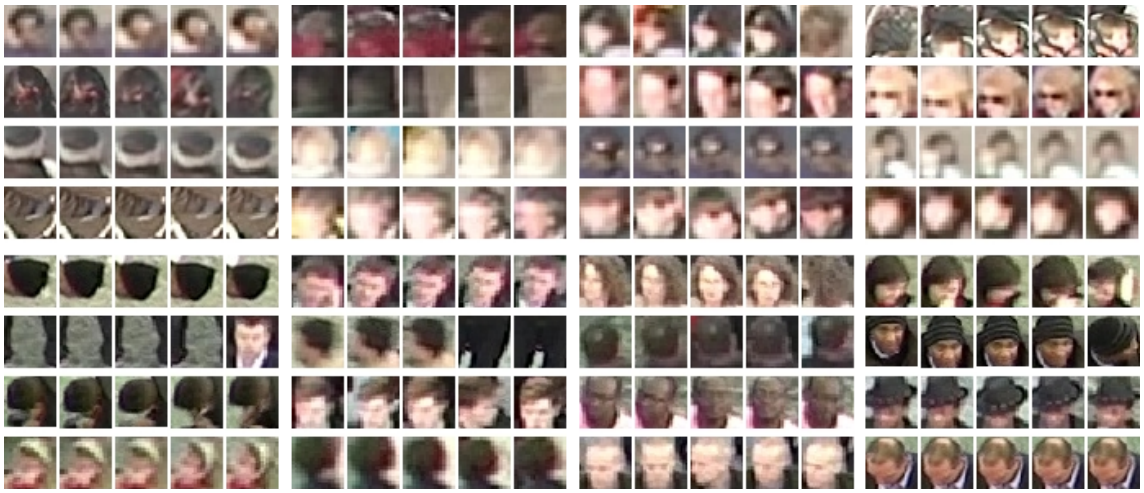


Figure 6.1: Randomly chosen sequences from the two video datasets. The sequences demonstrate the problems of image blur, tracking failures, incorrect detections and unusual appearances caused by clothing such as hats.

all.

Problem Statement: The objective of the research presented in this chapter is to develop a system that can take as input a video sequence and output gaze direction estimates for all of the pedestrians in every frame of the video. In contrast to the system in chapter 4, the aim here is to avoid any manual labelling of training images and instead learn an appearance model for gaze directions automatically from the input video sequence without any manual intervention.

This chapter examines the possibility of learning to estimate gaze directions in an unsupervised manner using the output of the automatic tracking systems such as those described in chapters 4 and 5. By tracking pedestrians in a scene for an extended period of time, we can have some confidence that we will have acquired head images representing the full 360° range of gaze directions. Furthermore we also note that people tend to look in their direction of motion. The combination of these two factors yields the potential to automatically acquire a very large set of weakly labelled data without human intervention.

Estimating gaze directions in unconstrained environments is difficult because of the many variables affecting the appearance of an individual, which are illustrated in

figure 6.1. Facial structure, hair style and colour, skin colour, blurry images, lighting conditions and accessories such as hats and sunglasses contribute to large variations in the appearance of different people looking in the same direction compared with potentially subtle differences between the appearance of the same person looking in different directions. Automatically acquired training sets have the potential to be many orders of magnitude larger than manually labelled ones, so can provide examples covering far more combinations of these variables.

Such training sets also have the potential to yield classifiers customised to the specific conditions of a particular installation, such as the viewpoint, focus and lighting conditions which would be difficult to train for explicitly. Provided the same tracking algorithm is used to acquire all of the head images, we can expect the head location error to be the same during training and testing which we know to be beneficial.

The drawback of using automatically acquired training data is that we do not know the true gaze direction for any of the images, so they must be inferred. The inference method described in this chapter is based around a Conditional Random Field (CRF) which models different aspects of gaze behaviour, such as the tendency for pedestrians to look in their direction of travel. When applied to a dataset, the CRF automatically infers the gaze directions for the images and as a side effect also trains a forest of randomised tree classifiers. The randomised forest models the interaction between the head images and the gaze direction variables within the CRF, but once trained can also be used as a standalone gaze classifier without the CRF.

Datasets acquired from two different scenarios were used to evaluate the idea. The first was the Town Centre video which was processed using the tracker from chapter 4 to yield a dataset comprising 473412 images from 2258 people. The second dataset was acquired by applying the sliding window tracker from chapter 5 to a different video covering a busy transport terminal, and consists of 639581 images

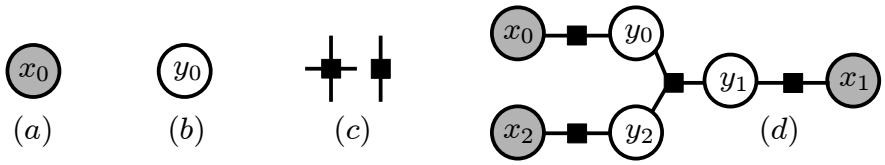


Figure 6.2: An explanation of factor graph notation for representing CRFs. Observed variables are represented by circular nodes with shaded backgrounds (a) and latent variables are unshaded circles such as that shown in (b). Factors are plain black squares (c) and are connected with edges to two or more variable nodes with edges, where the edges show which variables are used by the factor function. An example factor graph is shown in (d).

from 3861 people. The different tracking systems were used because the first dataset was acquired and labelled with ground truth before the development of the improved tracking system. The datasets consist of image windows averaging 24×26 pixels in size representing putative head regions, and for each image the tracker also provides an instantaneous ground plane velocity estimate for the corresponding pedestrian.

The method for learning a classifier that we describe is weakly supervised when considered in isolation, however the weak supervision consists of the direction of motion from an unsupervised tracking system. When considered in combination with the tracking system, our approach is fully unsupervised. In the context of learning, our system has some similarities to that of Leistner et al [67], who used multiple instance learning with randomised trees to classify object images into categories.

As a result of participation in a study of human gaze behaviour, a large amount of hand labelled ground-plane velocities and gaze directions were available. We used this data, which we will refer to as the *model data*, to infer some of the general parameters for our gaze behaviour model. The data did not include any image data so the parameters are expected to generalise to any video sequence.

6.1.1 Conditional Random Fields

The inference of gaze directions in this chapter is based around a CRF model. CRFs were originally proposed by Lafferty et al. [60] and are a type of undirected graphical

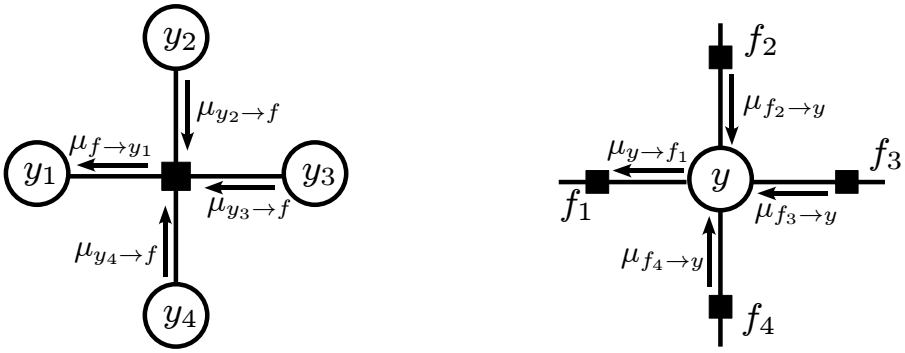


Figure 6.3: Diagrams illustrating the calculation of messages to and from variables and factors. In the left diagram, the factor node combines messages from three variable nodes and sends the result to the fourth. Similarly, in the right diagram the variable node combines messages from three factor nodes and sends the result to the fourth.

model. In this chapter the CRF is represented as a factor graph, which is bipartite with one set of vertices corresponding to variables and the other set of vertices corresponding to factors. Edges in the graph represent the variables that affect the outcome of each factor function. The notation is briefly explained in figure 6.2.

The main difference between CRFs and Markov Random Fields (MRFs) is the function that is modelled. A MRF typically models the joint distribution $P(X, Y)$ of the observations X and the latent variables Y, whereas a CRF models the conditional distribution $P(X|Y)$. Modelling the conditional distribution is preferable because there is no need to model the prior distribution over the observations.

6.1.2 CRF Inference

To infer the latent variables in the CRF, a combination of Expectation Maximisation (EM) and Loopy Belief Propagation (LBP) was used. EM was originally formalised by Dempster et al. [30] and is well established in computer vision. The reader is referred to Bishop [13] for more details.

Loopy Belief Propagation Belief propagation [98], also known as the sum-product algorithm, is a method for maximising the overall likelihood of the variable

values in a graphical model. When applied to a tree or chain structured graph, the method is guaranteed to result in the optimal solution. The algorithm is known as Loopy Belief Propagation (LBP) when applied to graphs with cycles and in this case the solution is not guaranteed to be optimal, however in practice it usually yields a good solution. The following overview is loosely based on the description by Kschischang et al. [58], to which the reader is referred for more details.

LBP works by calculating messages to be sent along each of the edges to or from a variable, where each message represents the current probability estimate for the variable from the part of the graph where the message originates. Messages can be categorised into two types depending on whether they are sent from variable nodes to factor nodes or from factor nodes to variable nodes, as illustrated in figure 6.3. Let y represent a variable and f_i be one of n_f factors connected to y. The message from y to f_1 is the product of the messages to y from the other factors:

$$\mu_{y \to f_1}(y) = \prod_{i=2}^{n_f} \mu_{f_i \to y}(y) \tag{6.1}$$

Here the messages both to and from y are distributions over the random variable y, so the calculation is fairly simple. The messages from factors to variables are more complex because they require the factor function ψ to be evaluated. The messages to and from factors are also distributions, but typically over different random variables. If y_i is one of the n_y variables connected to factor f, the message to y_1 is the result of marginalising over the other variables:

$$\mu_{f \to y_1}(y_1) = \sum_{y_2} \cdots \sum_{y_{n_y}} \left(\psi(y_1 \dots y_{n_y}) \prod_{i=2}^{n_y} \mu_{y_i \to f}(y_i) \right) \tag{6.2}$$

For chain structured graphs, it is usually only necessary for each message to be sent once, provided they are sent in the correct order. When loops are present, messages are usually evaluated multiple times until the distributions converge.

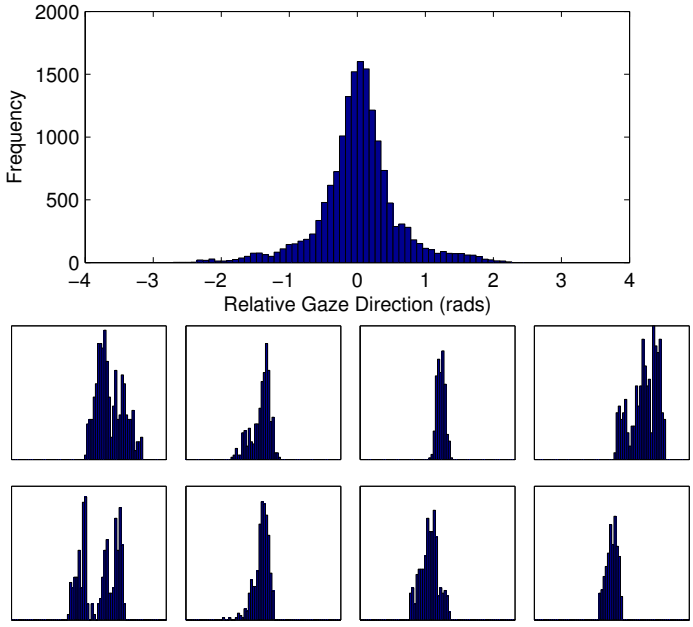


Figure 6.4: The distribution of gaze direction relative to walking direction over all people (top) and eight individual people (bottom) taken from the model data.

6.2 Model Formulation

The observations used as input to our system consist of a set of head image sequences $I = \{\boldsymbol{i}_x\}$ where every head image has a corresponding movement direction and magnitude v^t representing the individual's ground plane velocity. There are three key properties of the tracked images that we harness to infer the gaze directions. The first is that people tend to look most frequently in their direction of travel, an

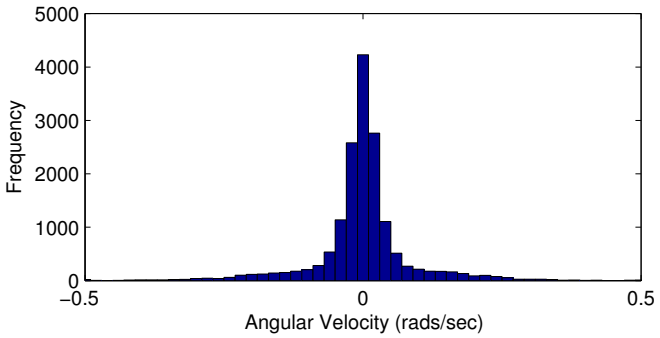


Figure 6.5: The distribution of head angular velocity relative to the walking direction over all of the people in the model data.

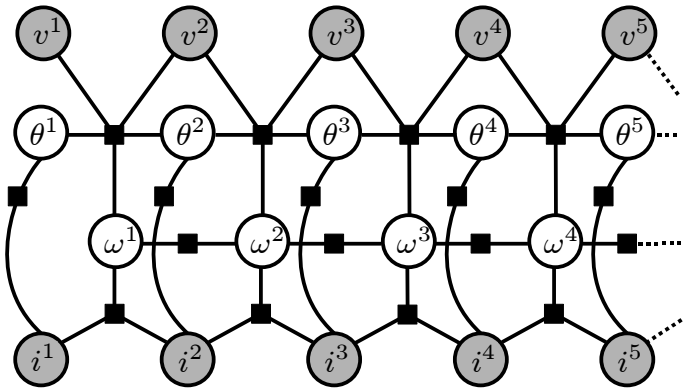


Figure 6.6: A factor graph showing how cliques and variables interact in the CRF. The angle estimate at time t is represented by θ^t, the angular velocity between times t and $t+1$ is represented by ω^t, and the observed image information and walking velocity are represented by i^t and v^t respectively. Variable nodes are shaded if they are observed and unshaded if they are latent.

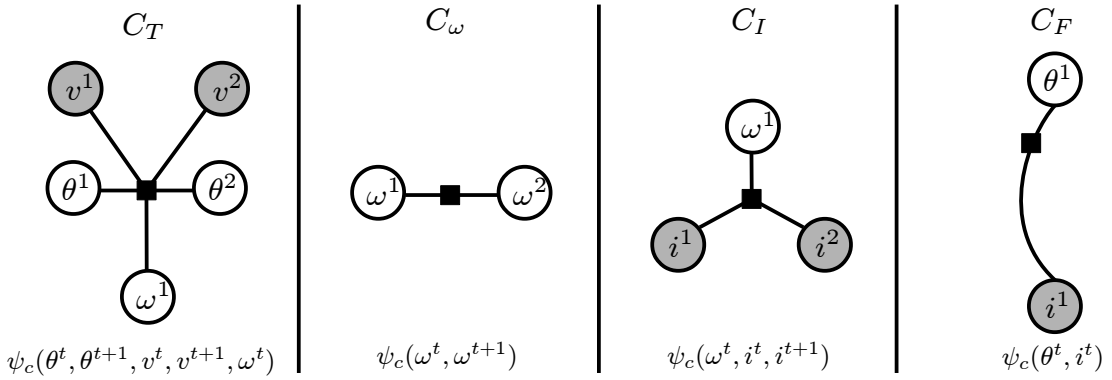


Figure 6.7: The four factor types that are used in the CRF, each shown with the variables they interact with and the corresponding factor function.

intuition which is confirmed by the histogram in figure 6.4 which shows that the distribution of gaze directions relative to walking directions in the model data has an approximately normal distribution with the mean at zero. The second property is that people usually move their heads slowly, so we expect sequential gaze directions to be reasonably similar. Again, this is confirmed by the model data in figure 6.5, which shows that the distribution of angular velocities has a strong peak at zero. Lastly, we expect the appearance of people to be more similar when they are looking in the same direction compared to when they are not.

The overall estimation is based around the optimisation of a CRF, the structure

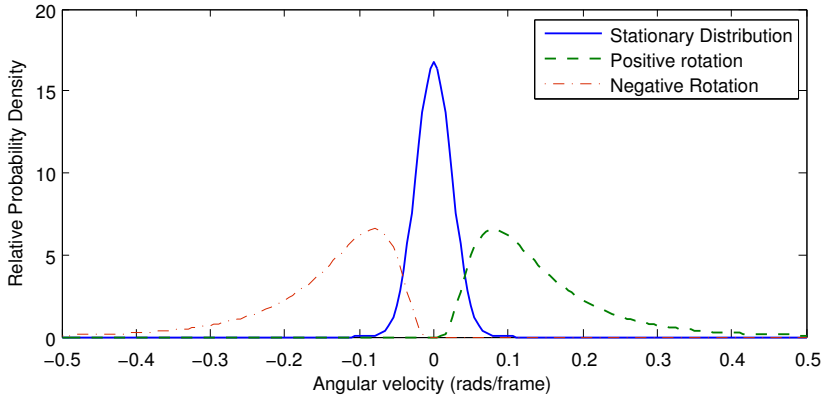


Figure 6.8: The three components of the mixture model used to represent the angular velocity. The probability distribution of an angular velocity variable ω^t is represented as a mixture $\mathbf{w}^t$ of these three components.

of which is represented as a factor graph in figure 6.6. Using the clique template representation of Sutton and McCallum [121], we divide the factors ψ_c into four sets $\mathcal{C} = \{C_T, C_F, C_\omega, C_I\}$ depending on which random variables they combine, as shown in figure 6.7. The overall conditional probability is represented as the product of the individual factor functions:

$$p(\boldsymbol{\theta}, \boldsymbol{\omega} | \boldsymbol{i}, \boldsymbol{v}) = \frac{1}{Z(x)} \prod_{C_p \in \mathcal{C}} \prod_{\psi_c \in C_p} \psi_c(\boldsymbol{i}_c, \boldsymbol{v}_c, \boldsymbol{\theta}_c, \boldsymbol{\omega}_c) \tag{6.3}$$

The function $Z(x)$ represents a normalising constant which is required to ensure that the distribution sums to one given the observed image $\boldsymbol{i}$ and velocity $\boldsymbol{v}$. The labels, which consist of the estimated gaze directions θ^t and angular velocities ω^t, are both discretised to allow efficient inference. Gaze directions are represented as a distribution over 32 bins each representing an 11.25 degree range of angles.

The probability distribution for the angular velocity ω^t is represented as a vector of three weights $\mathbf{w}^t = (\mathrm{w}^+, \mathrm{w}^0, \mathrm{w}^-)^\mathsf{T}$ which define a mixture of the three components shown in figure 6.8.

The first component is a Gaussian to represent the peak in the centre of the distribution corresponding to no head rotation. The other two components are

log-Gaussians to represent rotations in the positive and negative directions. We determined the parameters for these three components from the model data using Expectation Maximisation.

The following sections will define how the four types of factor function model the interactions between random variables.

6.2.1 Angular Velocity Factors

We begin by defining the factor representing transitions between angular velocities. The component weights $\mathbf{w}^t$ for the distribution over ω^t are expected to be correlated with those for ω^{t+1} because head movements usually last for more than one frame. The 3×3 matrix A represents the expected angular acceleration of the head in terms of the component weights, so we can predict $\mathbf{w}^{t+1}$ from $\mathbf{w}^t$, resulting in the following factor function definition:

$$\psi_c(\omega^t, \omega^{t+1}) \propto P(\omega^{t+1}|\omega^t) \tag{6.4}$$

$$\propto (\mathbf{w}^t)^\mathsf{T} A \mathbf{w}^{t+1} \tag{6.5}$$

Element j, k of A represents the probability of mixture component k representing ω^{t+1} if component j represents ω^t. The elements of A were estimated from the model data.

The angular velocity factors model the tendency for head movements to be spread over a number of frames. Since head movements are relatively rare, when an individual does move their head, they tend to move for more than one frame. If the frame-to-frame state transitions were modelled as being independent, as in an HMM, the probability of large head movements lasting multiple frames would be significantly underestimated. The angular acceleration factors resolve this problem by modelling the second order interactions.

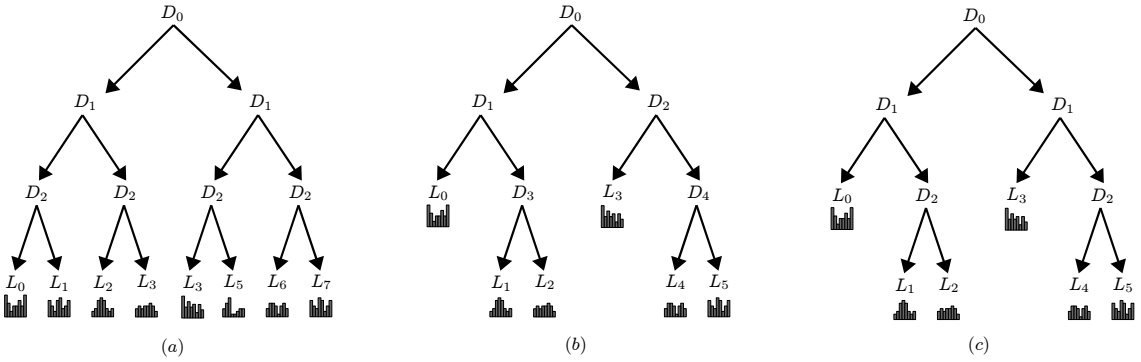


Figure 6.9: Examples showing the difference between standard trees and ferns and the hybrid structure that was used. A fern is shown in (a), a standard tree in (b) and our hybrid in (c). The hybrid has leaves at different depths like a tree, but has the same branch tests at every depth like a fern.

6.2.2 Image Classification Factors

The next factor is that which relates head images to directions. Initially we do not have any information on the mapping between images and gaze directions, however we do know that similar images are more likely to represent the same direction. This property is modelled using a forest of randomised tree classifiers, with each leaf node containing a histogram over the 32 direction bins, initially representing a uniform distribution. The histograms are scene-specific parameters that are inferred during the automatic learning process.

The randomised trees are a slight modification of the standard construction, and represent a mid-point between randomised trees and randomised ferns, as illustrated in figure 6.9. Since the true labels for images are not known in advance, the branch tests are selected at random, but all branch tests at the same depth are made equal. This gives the trees some of the performance advantages of ferns, because the same branch tests are made for all images and only one test needs to be stored for each depth in the tree. Since trees only expand leaves containing data, the advantages of trees in terms of storage requirements are retained, which allows training to a greater depth than would be possible with ferns. Branches are split until there are fewer than one hundred examples in each leaf node. One further advantage of the

hybrid trees is discussed briefly in the conclusion at the end of the chapter.

A total of forty trees were trained using the HOG and CTC features described in section 4.3.2, with the factor function averaging over all of the histograms to obtain the required probability estimate:

$$\psi_c(\theta^t, i^t) = \frac{1}{n} \sum_{k=1}^{n} P(\theta^t | D^k(i^t)) \tag{6.6}$$

The notation $D^k(i^t)$ is used to represent the branch outcomes from passing i^t down the kth tree in the forest, from which the conditional probability is estimated using the histogram at the corresponding leaf. The class histograms for the leaf nodes in the trees, which we will represent as $\mathcal{H}$, are scene-specific model parameters that are automatically inferred. The histograms were all initialised to represent a uniform distribution over the classes before the first iteration.

Randomised tree classifiers were chosen over other types of classifier because they require very little time to retrain if the image data remains the same and only the corresponding class distributions change. They also have a direct probabilistic representation, so do not require a mapping function to be learned as with other types of classifier such as SVMs.

6.2.3 Changing Image Factors

The next type of factor is intended to represent the correlation between the gaze direction changing and the observed image changing. Although a wide variety of appearances could represent the same gaze direction, as modelled by the classification factors, a particular person looking in the same direction should have a relatively consistent appearance over short time periods (e.g. a fraction of a second) because the variables affecting their appearance are unlikely to have changed.

The distance between two head images $d(i^t, i^{t+1})$ is measured as the number of randomised trees in which the two images reached different leaves, if δ represents

the Kronecker delta function then this is defined as:

$$d(i^t, i^{t+1}) = \sum_{k=1}^{n} \delta(D^k(i^t), D^k(i^{t+1})) \tag{6.7}$$

The vector ϕ of length $n+1$ represents the expected weight on the stationary angular velocity component for each number of trees where the leaves reached differ, so $\phi_{d(i^t,i^{t+1})}$ is the probability that ω^t will be represented by the stationary mixture component, and $1 - \phi_{d(i^t,i^{t+1})}$ is the probability that ω^t will be represented by one of the two moving components. The probabilities for the moving components are assumed to be equal so are not modelled separately. This results in the following factor definition:

$$\psi_c(\omega^t, i^t, i^{t+1}) = P(\omega^t | i^t, i^{t+1}) \tag{6.8}$$

$$\propto \mathrm{w}^0 \phi_{d(i^t,i^{t+1})} + (1 - \mathrm{w}^0) \left(\frac{1 - \phi_{d(i^t,i^{t+1})}}{2} \right) \tag{6.9}$$

The elements of ϕ are scene-specific parameters that are learned automatically and w^0 is the element of $\mathbf{w}^t$ corresponding to the probability of ω^t being represented by the stationary component.

6.2.4 Head Motion Factors

Lastly we describe the factors C_T which cover the transitions between pairs of gaze directions. The factor function is defined in terms of a prior transition matrix T and a matrix of angular velocity marginals M:

$$\psi_c(\theta^t, \theta^{t+1}, v^t, v^{t+1}, \omega^t) \propto P(\theta^{t+1} | \theta^t, v^t, v^{t+1}, \omega^t) \tag{6.10}$$

$$\propto (\theta^{*t})^{\mathsf{T}} (T \oplus M) \theta^{*t+1} \tag{6.11}$$

The operator $\oplus$ is used here to denote the element-wise product of two matrices and θ^{*t} represents the vector θ^t rotated so that it is measured relative to v^t. M is

a cyclic matrix where elements m_{ij} represent the probability of the angular velocity required to transition from state i to state j given the estimated angular velocity parameters ω^t.

The matrix T represents our prior knowledge of how the gaze direction should change between each pair of frames. Many of the elements in T could be learned from the observations, however some of the elements represent transitions that are expected to occur very infrequently, such as rapid head movements or transitions between backward facing directions. These elements would be impractical to estimate empirically. To avoid estimating all 32^2 elements of T directly, the number of degrees of freedom was reduced by parameterising T, as described in the following section, before fitting to the transitions in the model data using a constrained optimisation.

In the event that either v^t or v^{t+1} has a magnitude of less than half the mean human walking speed ($0.7ms^{-1}$), the walking direction is considered to be unreliable and the head motion factor is evaluated in the absolute frame of reference. This differs from equation 6.11 in that θ^t and θ^{t+1} are used rather than the relative versions and a small modification is made to the steady state for T, which will be described in the next section.

Transition Matrix Parameterisation

From any state, we represent the probability of travelling to a destination state as a mixture $\boldsymbol{z}$ of the three components illustrated in figure 6.8. It is these mixture coefficients on the three components of the velocity distribution that constitute the parameterisation.

When an individual is facing their direction of travel, there is an equal probability that they will move their gaze direction to the left or to the right, however if they are already looking sideways it is more likely that they will move their gaze direction towards the direction of travel. This observation has been previously used

to aid high resolution head tracking [45]. We model this by allowing the weights for the three components to vary depending on the current angle of the head, so the parameterisation consists of three vectors of n weights $\boldsymbol{z}^+$, $\boldsymbol{z}^0$ and $\boldsymbol{z}^-$ corresponding to positive, stationary and negative rotation probabilities respectively. Each of the three components were quantised to give a probability distribution q over discrete states so that the members of T could be easily calculated efficiently:

$$t_{ij} = z_i^+ q^+_{j-i \pmod n} + z_i^- q^-_{j-i \pmod n} + z_i^0 q^0_{j-i \pmod n} \tag{6.12}$$

The values for the weights are optimised to ensure that they represent a transition matrix that is consistent with both the prior angular velocity mixture weights w^t_{pr} and some general properties of head motion, both of which were learned from the hand labelled motion data. Specifically, the following constraints must be met for a transition matrix to be valid:

Total Weight To ensure that the probability mass represented by the velocity distribution sums to one, each set of three component weights must total one:

$$C_i^{\Sigma}(\boldsymbol{z}) = z_i^+ + z_i^0 + z_i^- - 1 = 0 \tag{6.13}$$

Positive Weights To prevent negative transition probabilities, inequality constraints must be added to ensure that each of the $3n$ component weights are not negative:

$$C_i^{p^+}(\boldsymbol{z}) = \max(-z_i^+, 0) = 0 \tag{6.14}$$

$$C_i^{p^0}(\boldsymbol{z}) = \max(-z_i^0, 0) = 0 \tag{6.15}$$

$$C_i^{p^-}(\boldsymbol{z}) = \max(-z_i^-, 0) = 0 \tag{6.16}$$

Steady State The gaze directions of pedestrians are highly correlated with their walking directions, since people tend to look in their direction of travel and rarely look behind themselves. The gaze distribution of an individual observed for long enough should tend towards the distribution s shown in figure 6.4. We expect the transition matrix to satisfy this condition if s is its steady state:

$$T^{\mathsf{T}} s = s \tag{6.17}$$

To enforce the steady state distribution, constraints were introduced with the following form:

$$C_i^S(\boldsymbol{z}) = \frac{\sum_j t_{ji} s_j}{s_i} - 1 = 0 \tag{6.18}$$

If an individual is not walking and the motion factor is evaluated in the absolute frame of reference, s is set to be uniform to reflect the lack of prior knowledge on potential gaze directions.

The reason for incorporating the prior in this way is to avoid imposing any preference on the overall distribution of gaze directions for an individual. Although we expect the gaze direction to converge on s when an individual is observed over a long period of time, we only observe people over relatively short intervals, so we do not expect an individual's gaze distribution to resemble s over the time for which they are observed. The distribution s and some individual gaze direction distributions are shown in figure 6.4.

Objective Function The requirements above constrain $2n$ degrees of freedom, however $\boldsymbol{z}$ has $3n$ variables. A constraint on the relative values of z_i^0 could be introduced to provide an additional $n-1$ constraints, but doing so would make it often impossible to satisfy all of the constraints. Instead, an objective function was used to regularise the solution by imposing the preference of having the weights for

each state equal to the angular velocity mixture weight priors:

$$f(\boldsymbol{z}) = -\sum_i (z_i^- - \mathrm{w}_{pr}^-)^2 + (z_i^0 - \mathrm{w}_{pr}^0)^2 + (z_i^+ - \mathrm{w}_{pr}^+)^2 \tag{6.19}$$

Although we do not expect the weights to be equal to the mean, they are expected to deviate by only a small amount. In practice this regularisation guides the optimisation towards the solution that we require; without it the optimisation sometimes finds undesirable solutions. These solutions satisfy the constraints but have unrealistic properties such as requiring the head to constantly spin in one direction whilst slowing down for forwards facing directions to maintain the steady state distribution.

Transition Matrix Optimisation

To find the optimal parameter values for the matrix T, the constraints were combined with the objective function using the quadratic penalty method [88]. Every constraint introduces a penalty term which is zero when satisfied.

$$F(\boldsymbol{z};\mu) = f(\boldsymbol{z}) - \frac{1}{2\mu}\sum_i \Big(C_i^{\Sigma}(\boldsymbol{z})^2 + C_i^{S}(\boldsymbol{z})^2 + C_i^{p^+}(\boldsymbol{z})^2 + C_i^{p^0}(\boldsymbol{z})^2 + C_i^{p^-}(\boldsymbol{z})^2 \Big) \tag{6.20}$$

Although the constraints have the same quadratic form as the objective function, the parameter μ is reduced during the optimisation to make the penalty functions dominate over the objective function to ensure that the constraints are satisfied. The objective function has discontinuous second derivatives resulting from the inequality constraints, so the method of nonlinear conjugate gradients was used to perform the optimisation instead of a Newton-based method. The following equations show the general form of the derivatives:

$$\begin{aligned}\frac{\partial F}{\partial z_k^-} = -2(z_k^- - \mathrm{w}_{pr}^-) - \frac{1}{\mu}\Bigg(& \big(z_k^- + z_k^0 + z_k^+ - 1\big) \\ & + \sum_i \left(\frac{q_{i-k}^- s_k}{s_i}\right)\left(\frac{\sum_j t_{ji} s_j}{s_i} - 1\right) + \max(-z_k^-, 0)\Bigg)\end{aligned} \tag{6.21}$$

$$\frac{\partial F}{\partial z_k^0} = -2(z_k^0 - \mathrm{w}_{pr}^0) - \frac{1}{\mu}\Bigg(\left(z_k^- + z_k^0 + z_k^+ - 1\right) + \sum_i \left(\frac{q_{i-k}^0 s_k}{s_i}\right)\left(\frac{\sum_j t_{ji} s_j}{s_i} - 1\right) + \max(-z_k^0, 0)\Bigg) \quad (6.22)$$

$$\frac{\partial F}{\partial z_k^+} = -2(z_k^+ - \mathrm{w}_{pr}^+) - \frac{1}{\mu}\Bigg(\left(z_k^- + z_k^0 + z_k^+ - 1\right) + \sum_i \left(\frac{q_{i-k}^+ s_k}{s_i}\right)\left(\frac{\sum_j t_{ji} s_j}{s_i} - 1\right) + \max(-z_k^+, 0)\Bigg) \quad (6.23)$$

6.3 Model Optimisation

Having described the structure of the individual factor functions, we now consider the optimisation of the scene-specific parameters $\boldsymbol{\phi}$ and $\mathcal{H}$ and also distributions over $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$, which we consider to be latent variables for the purpose of learning the scene parameters. The learning uses the EM algorithm, which alternates between calculating an expectation over the latent variables and maximising the parameters given the expectation. The method is equivalent to the use of the Baum-Welch algorithm to optimise HMM parameters.

Expectation The first step requires the calculation of $P(\boldsymbol{\theta}, \boldsymbol{\omega} | \boldsymbol{\phi}^t, \mathcal{H}^t, \boldsymbol{v}, \boldsymbol{i})$, which is the expected distribution over the latent variables given the parameter estimate and the observations at iteration t. Since our CRF contains cycles the expectation cannot be calculated exactly in any reasonable amount of time, so we approximate it using Loopy Belief Propagation (LBP), for which a brief overview can be found in section 6.1.2. Our CRF is chain-structured, so we use a message passing schedule which propagates messages in alternating forwards and backwards passes in a similar way to the Forwards-Backwards algorithm for HMMs. The CRF has disconnected components corresponding to each of the people in the dataset, so LBP was applied to each individually.

Figure 6.10: Sample frames from the two scenes on which the system was tested. The frame on the left is from the Town Centre video, to which tracking was applied to create head image dataset C. Most people in this dataset are moving but in the second dataset most people are stationary. The image on the right is a frame from the Transport Terminal video, which was tracked to create head image dataset E. The Transport Terminal video also exhibits significant distortion, which we train for implicitly.

Maximisation In the maximisation step the probability histograms $\mathcal{H}$ in the leaves of the randomised trees and the vector $\boldsymbol{\phi}$ of motion probabilities from the image difference factors are both updated:

$$\{\boldsymbol{\phi}^{t+1}, \mathcal{H}^{t+1}\} = \underset{\boldsymbol{\phi}, \mathcal{H}}{\operatorname{argmax}} \sum_{\boldsymbol{\theta}, \boldsymbol{\omega}} P(\boldsymbol{\theta}, \boldsymbol{\omega} | \boldsymbol{\phi}^t, \mathcal{H}^t, \boldsymbol{v}, \boldsymbol{i}) \log P(\boldsymbol{\theta}, \boldsymbol{\omega}, \boldsymbol{\phi}, \mathcal{H} | \boldsymbol{v}, \boldsymbol{i}) \qquad (6.24)$$

The parameters in $\boldsymbol{\phi}$ and $\mathcal{H}$ are conditionally independent given $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$ so can be calculated separately. The leaf histograms are maximised when they are equal to the mean of the expected gaze directions for all of the images reaching the leaf, which is the standard training process for trees. The motion probabilities are maximised by marginalising over the factors for all pairs of images in the dataset.

6.4 Evaluation

The system was evaluated on the tracking output from two large video sequences of different scenes, shown in figure 6.10. Both scenes are public places where pedestrians exhibit a wide variety of appearances due to hats, sunglasses and different clothing. The mean size of the head images in both cases was 24×26 pixels.

Method	Test Dataset	
	Town Centre	Terminal
Our Unsupervised System	23.9	38.5
Supervised Ferns (chapter 4)	45.5	59.2
Walking Direction	25.9	78.4

Table 6.1: Gaze estimation performance (MAAE in degrees) of our unsupervised system compared with that of conventionally trained classifiers and the walking direction baseline. Our system outperforms the other approaches on both datasets.

The first dataset (Dataset C in section 2.6.2) is from an outdoor town centre scene where the majority of pedestrians are walking and consists of 473412 images from 2259 people. Every one hundredth image in the dataset was hand labelled to provide ground truth, a total of 4347 images. This sequence is publicly available and the images with ground truth are available to enable future comparisons. The second dataset (Dataset E) covers a busy transport terminal where the majority of people are stationary and consists of 639581 images from 3861 people.

The performance of the system was measured using the Mean Absolute Angular Error (MAAE), which is stated in degrees. A baseline performance measure was obtained by assuming that the gaze direction is the same as the walking direction. This baseline provides very good estimates for the Town Centre dataset, since most people look in their direction of travel, however in the Transport Terminal dataset there are many stationary people so the direction of travel is often incorrect.

The system was also compared with the HOG/CTC ferns that were developed in chapter 4, which were trained using 1500 head images that were cropped from still photos of different people (Dataset A). The results from the HOG/CTC ferns make use of the HMM filtering, but do not take the walking direction into account.

The results of applying the system to the two large video datasets are shown in table 6.1. In both datasets, our system significantly outperforms the supervised ferns, which demonstrates the value of learning the scene-specific classifier and using the walking direction in the gaze direction model. Since most people in the Town

Method	Training Dataset	Testing Dataset		
		Town Centre (C)	Terminal (E)	Still Images (A)
Unsupervised Trees	Town Centre	25.6	47.8	60.2
Unsupervised Trees	Terminal	64.9	42.9	71.4
Supervised Ferns	Still Images	45.5	59.2	43.5

Table 6.2: Comparison of the performance (MAAE in degrees) of the learned forest of randomised tree classifiers when trained and tested on the two video datasets and the still image dataset. For these experiments the CRF model was not used for testing. When the supervised ferns were tested on the still images (dataset A), 80% of the data was used for training and the remaining 20% for testing. The results show that the learned classifiers still outperform the supervised ferns even in the absence of motion information.

Centre dataset look in their direction of travel, there is not much to be learned, so we only marginally outperform the walking direction baseline. In the Transport Terminal dataset there are many stationary people so our system performs considerably better than the walking direction baseline, which is almost random.

Although one of the benefits of the unsupervised learning approach is the ability to learn scene-specific classifiers, to provide some additional insight the learned randomised forests were tested in isolation (without the CRF) on each of the two video datasets as well as the still images that were used to train the supervised classifier. It should be noted that this is not the intended usage of the system, since for any practical purpose the combination of the CRF model and the learned trees would be used rather than just the trees alone.

The results from testing the randomised forest in isolation are shown in table 6.2. An important conclusion from these results is that we can use the walking direction to learn a randomised forest classifier, but the classifier remains effective even when the walking direction is not used in the estimation at testing time.

In the absence of the CRF model, the randomised forests that were learned from the Town Centre and Transport Terminal datasets each perform best on the dataset from which they were learned, however the Town Centre forest appears to

generalise better and is more accurate when tested on the still image dataset. A likely reason for this is that although both video datasets are large, most of the information for learning the classifiers comes from people who are walking. The Town Centre dataset has 441048 images from moving people, compared to only 69512 images from moving people in the Transport Terminal dataset. Many of these images will be similar, since we acquire around 200 images from each person, making approximately 420 different informative people in the Transport Terminal dataset in comparison to approximately 2100 in the Town Centre dataset. It is likely that this variation in appearances during training is responsible for the better performance of the Town Centre classifier, however this is not a limitation of the approach in general because a deployed system would be able to constantly improve the classifier over many days or years by continuously refining the ϕ and $\mathcal{H}$ parameters.

Many of the errors in the Transport Terminal dataset were caused because people often walk backwards to improve their view of the departure board. Since the data that was used to learn the model parameters did not include situations like this, our model considers this to be almost impossible. The result is that an incorrect gaze estimate (usually the direction of motion) is fed back into the randomised forest, where it has a negative effect on the estimations for other people on subsequent iterations. This issue could be resolved either by using training data from a wider variety of scenes to learn the model parameters, or by modifying the tracker to detect abnormal walking patterns so that they can be omitted.

Graphs of the MAAE and expected log likelihood after each of the first ten iterations are shown in figure 6.11 for the two datasets. Although the likelihood increases with each iteration, in both cases the MAAE for the full model remains relatively unchanged after the second iteration and reaches a minimum after approximately five iterations before increasing again slightly. This suggests that there might be inaccuracies in the model causing a disparity between the actual accuracy and the accuracy estimated by the likelihood function.

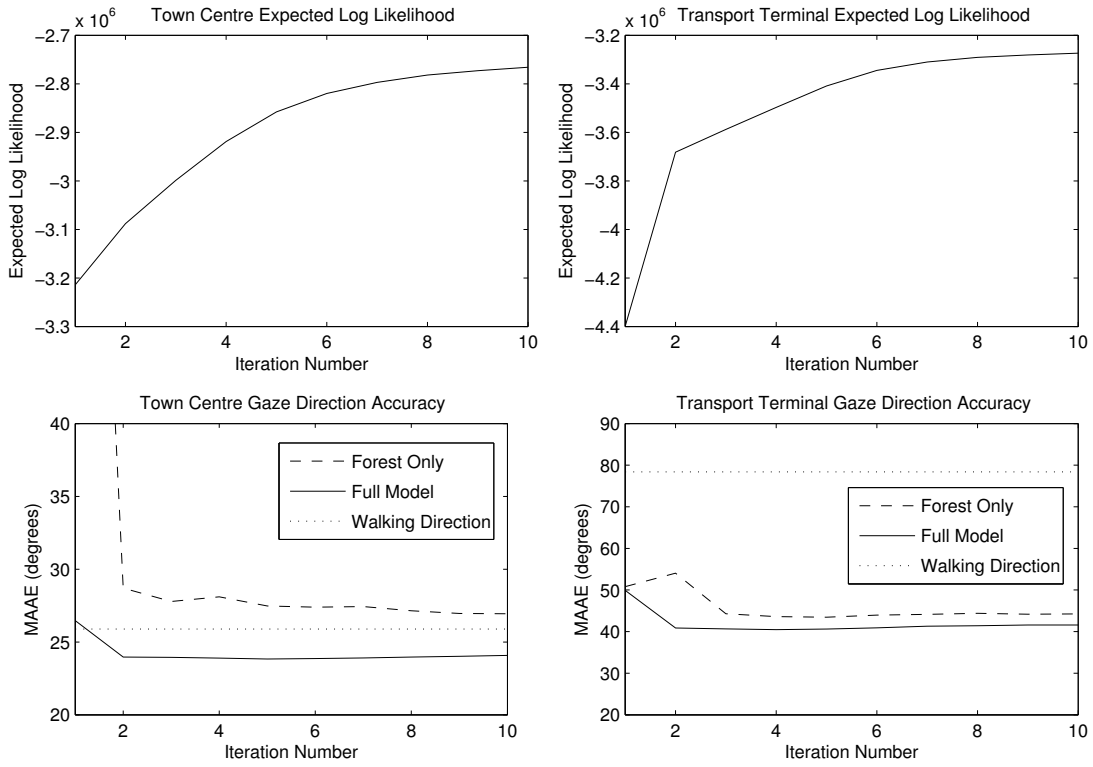


Figure 6.11: Graphs showing how the expected log likelihood (top) and the MAAE (bottom) change with each iteration. The MAAE plots compare the accuracy of the full CRF model and the forest of randomised trees in isolation with the walking direction baseline.

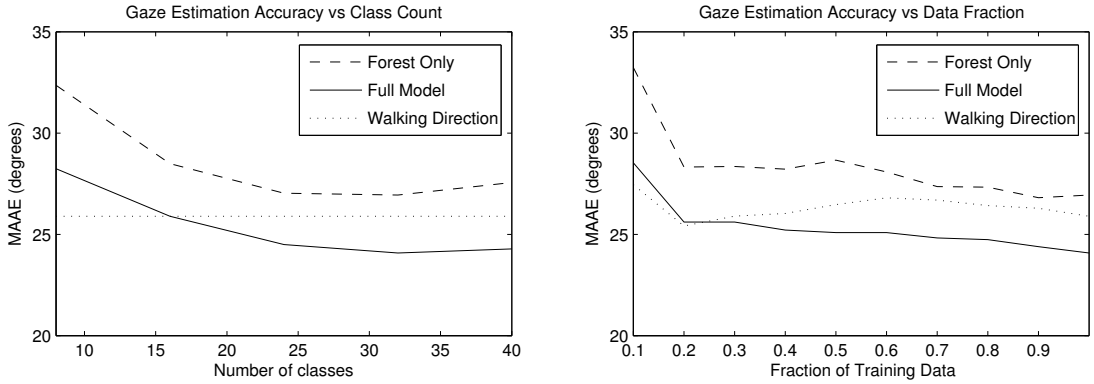


Figure 6.12: Graphs showing how the MAAE changes with the number of direction classes used (left) and the fraction of training data that was used (right).

Further experiments were used to test the effect of some of the parameters used for the optimisation. Figure 6.12 shows the results from experiments where the number of classes and the fraction of data used were each varied. The performance improved when the number of classes was increased up to 32, but the following test with 40 classes resulted in slightly lower performance. Experiments with more classes were not possible due to system memory limitations. It is possible that the reduction in performance was due to the data in the leaves being too sparse, so allowing leaves to hold more data before they are split might allow more classes to be used.

In the experiments where a subset of the training data was used, complete sequences for a subset of the people in the dataset were processed. The results show that the performance increases with the quantity of data which suggests that the method would benefit from larger datasets.

Some sample image sequences for individual people with the randomised forest estimations and expected state probability distributions are shown in figure 6.13. Figure 6.14 shows the automatically estimated gaze directions drawn onto the video sequences that they originate from.

6.5 Conclusion

We have developed a system which is capable of learning to estimate gaze directions in surveillance video without any human intervention. Our evaluation has shown that the performance of the scene-specific classifiers exceeds that of a conventionally trained classifier on the scene where they were learned. The following specific contributions have been made in this chapter:

- A CRF model was developed to accurately represent the interactions between gaze directions and walking directions.
- A complete system for inferring gaze directions and training a forest of ran-

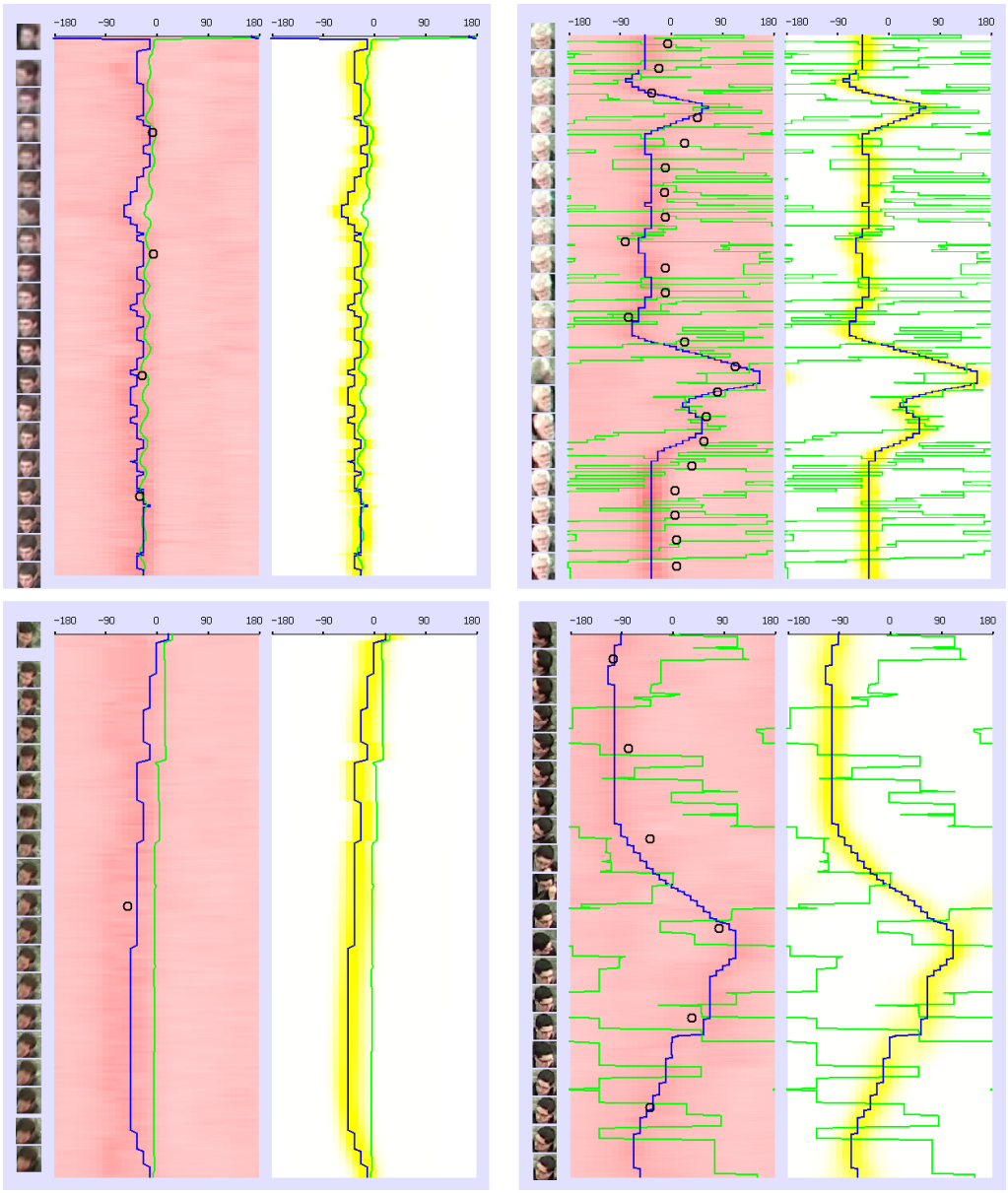


Figure 6.13: Four example sequences showing how head movements are correctly identified. The horizontal axis corresponds to the 360° range of gaze directions, with the centre representing 0 (looking at the camera). The vertical axis represents time, with the first frame in the sequence at the top and the last frame at the bottom. The red backgrounds show the observation probability (red is high) and yellow backgrounds show smoothed probabilities from the latent variables (yellow is high). The green lines are the direction in which the person is walking, blue is the maximum of the marginal distribution and circles represent ground truth labels. The left sequence shows a pedestrian who is moving, so the walking direction provides a strong prior for the gaze direction. The right sequence shows a pedestrian who is standing still, so the direction of motion is not helpful, but the learned randomised forest ensures that the gaze direction estimates are still reasonably accurate.

Figure 6.14: Frames from the two video datasets showing the estimated gaze directions drawn onto the people from which the automatic tracking extracted the head image sequences.

domised trees was demonstrated.

- A hybrid decision tree was proposed to allow efficient inference, with the potential for additional benefits in a distributed system.

Our implementation is able to learn classifiers from batches of data in approximately double the amount of time that is required to generate it, so a real-time online implementation is feasible. An online implementation would allow a potentially unlimited amount of training data to be used by accumulating the estimations in the randomised trees continuously over an unlimited period of time, which the experiments with different quantities of training data indicate would probably result in improved performance.

There is also potential for improving the performance by combining the result of the learning across a number of different installations. This would only require combining the histograms from the randomised trees, and would allow hundreds or thousands of times more data to be incorporated. Our hybrid randomised trees would be of particular value in this case, because multiple forests of decision trees could be easily combined if the same set of random branch tests were chosen across all installations. The only difference between the trees would be the points at which branches were expanded, so by allowing additional histograms at non-terminal branches any number of trees could be combined with a minimal increase in evaluation cost.

Although the randomised trees were used as a component in the CRF model, they are also of value in isolation. The accuracy is lower in this case, but there is no need for object velocities and the evaluation time is considerably lower.

CHAPTER 7

Conclusion

7.1 Thesis Summary

The objective of the research described in this thesis has been to automatically estimate the gaze directions for pedestrians in surveillance video. Although the primary objective has been to minimise the error in the gaze direction estimates, the requirements for a gaze estimation system to be of practical value have also been taken into consideration. All of the algorithms that have been developed process video either in real-time or close to real-time and considerable effort has been directed at the specific problem of ensuring that the methods generalise to unknown scenes and people.

Chapter 3 described the development of predicate ferns for classifying gaze directions using low resolution head images. The predicate ferns allowed images to be classified without the prior assumptions about the skin and hair colour distributions

that other similar approaches have made in the past. The classifiers are optimised for very low resolution images, and are able to infer the person-specific skin and hair colour histograms to improve the estimations in subsequent video frames.

The next, chapter 4, covered the development of a stable tracking system which allowed the predicate ferns and other gaze direction classifiers to be compared with realistic data. The tracking algorithm was able to track small groups of people in PAL resolution video in real-time, or approximately twenty people in 1920x1080 resolution video at 5 FPS. The newly proposed HOG/CTC ferns were shown to outperform similar classifiers on a variety of video datasets. Both the tracking and gaze direction estimation components were combined, with the complete system used to build attention maps representing the amount of interest received by different areas of a scene. These attention maps were used to identify both static and transient regions where the pedestrians frequently looked. The results from the experiments motivated the approaches of the following two chapters.

An improved tracking algorithm based on MCMCDA within a sliding window was developed in chapter 5 to both improve the stability of the head regions and to provide guaranteed real-time performance. When evaluated in the context of general pedestrian tracking the system was able to reach accuracy levels comparable with state of the art pedestrian tracking systems, but unlike the others was able to process video in real-time. An unexpected consequence of accurately modelling the failure properties of the observations was the ability to track through total occlusions for short periods of time.

Chapter 6, the last technical chapter, described a method for estimating gaze directions without any hand labelled training data. The dynamics of head movements relative to the walking direction and the corresponding effect on the appearance of the head were modelled using a CRF. The learning algorithm was able to use vast quantities of automatically acquired data to learn appearance models that were customised for specific scenes. The resulting algorithm provided considerably more

accurate gaze direction estimations than conventional approaches.

7.2 Contributions

The specific contributions of the four technical chapters are summarised below.

Chapter 3:

- A randomised fern based image classifier with predicate based branches was proposed along with a corresponding algorithm for inference.
- A method for learning colour distributions corresponding to the abstract labels to improve estimation accuracy was developed.
- Modifications were proposed to the standard methods for combining decision tree estimates to make best use of the predicate ferns.

Chapter 4:

- A multi-target tracking system was developed for the specific purpose of obtaining stable image sequences
- Robust randomised fern based gaze classifiers were developed, which included the new CTC image measurement to provide invariance to lighting effects.
- A complete system for measuring attention in large scale scenes was demonstrated. This is believed to be the first system to measure attention in general scenarios where pedestrians are able to move freely.

Chapter 5:

- A principled objective function was developed, allowing accurate location estimates and robust data associations to be made, including the ability to track through total occlusions.

- A new move type for MCMCDA was introduced to allow the removal of false positives, which could be also extended to identify different object types.
- An efficient and scalable multi-threaded architecture was proposed to allow large crowds of pedestrians to be tracked in real-time.

Chapter 6:

- A CRF model was developed to accurately represent the interactions between gaze directions and walking directions.
- A complete system for inferring gaze directions and training a forest of randomised trees was demonstrated.
- A hybrid decision tree was proposed to allow efficient inference, with the potential for additional benefits in a distributed system.

7.3 Future Work

A variety of approaches have been developed to cope with the specific problem of coarse gaze estimation and there are many different directions in which the research could be continued. The following suggestions are categorised by application area.

7.3.1 Gaze Direction Estimation

Online Gaze Learning The system for estimating gaze directions without hand labelled image data in chapter 6 processed images in batches that were accumulated over a short period of approximately twenty minutes. One limitation of this approach is that the amount of data that can be processed is limited by the amount of memory in the computer. The system would benefit from being adapted to operate using an online data source so that potentially unlimited amounts of data could be used. This could be achieved by including both fixed and variable histograms in the leaves

of the randomised trees. The data corresponding to recently observed people would be variable until the optimisation of their CRF had converged, at which point their contributions to the leaf histograms would become fixed. During the maximisation step of the optimisations, only the variable histograms would be adjusted.

7.3.2 Surveillance Tracking

Tracking Multiple Object Types The tracking model from chapter 5, distinguished genuine people from false positives occurring in the background by learning different models of how the objects move. The same approach could be used to distinguish between different types of pedestrian behaviour such as running, jogging and cycling by including separate models for each. A model which assumes gradual acceleration and accurate predictions from a constant velocity model would allow the movement of a cyclist to be encoded more efficiently than that of a pedestrian. If the HOG detector was replaced with a more general detection method such as background subtraction it would also be possible to distinguish vehicles from people using a similar approach.

Tracking Feedback The MCMCDA tracking system described in chapter 5 samples from the data association hypotheses to locate the mode before applying a simple hill-climbing algorithm to find the single most likely hypothesis. Although the aim is to find the most likely hypothesis, some situations are ambiguous and have a small set of hypotheses that are almost equally likely, so there is a reasonable chance that the wrong one will be chosen.

The asynchronous architecture that was proposed would allow this problem to be solved by using feedback from the data association to the HOG detection module. The MCMCDA component of the system could identify ambiguous associations between detections by measuring the frequency with which the associations change when the hypotheses are sampled. The result would be used to instruct the HOG de-

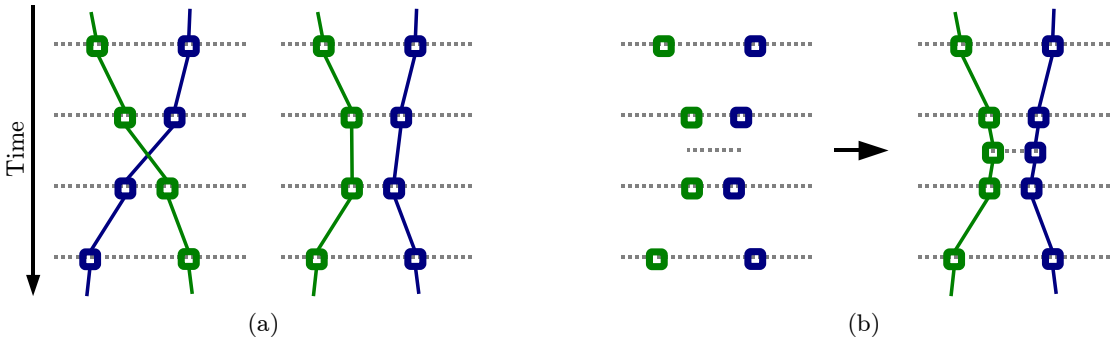


Figure 7.1: An example of how MCMCDA feedback could be used to resolve ambiguous data associations. Rectangles represent head detections, solid lines represent data associations, and dotted lines show the time at which the HOG detector was applied to the image. Two potential hypotheses with a similar likelihood are shown in (a). The extra detections resulting from performing an additional search over an intermediate frame as shown in (b) allows the ambiguity to be resolved.

tection module to perform additional searches, enabling the ambiguity to be resolved (see figure 7.1). The additional searches would only cover a small image region and scale range, so the extra time required would be insignificant when compared to the cost of searching over the entire image.

7.3.3 General Image Classification

Trees and Ferns with Latent Variables The performance of the predicate-based tree classifiers of chapter 3 is believed to have been limited by the use of segmented images as input. An obvious solution to this problem would be to replace the hard grouping of pixels into clusters and instead consider the likelihood of pixels belonging to the same logical label. To cope with this, the inference algorithm could be replaced by a branch-and-bound search where the bound would be based on the joint probability of both the labelling hypothesis and the likelihood of the pixel colours belonging to their assigned label histogram. This would also remove the limit on the number of labels that could be used, since there would be no need to search over all possible combinations, which would allow the method to be applied to a wider variety of classification problems.

The general principle of having branches in decision trees depending on latent variables could be extended to allow branch tests to be arbitrary functions of the observed data and the latent variables. The advantage of including latent variables is to allow some of the natural variation in appearance to be removed from the classification process. Although the colour of an object is a variable to which invariance is frequently required, other parameters such as the lighting direction could also be included as latent variables. This would allow randomised trees to be constructed with branches testing for properties such as the skin surface orientation, with the likelihood of the branch being determined by the probability of the observed colour being produced given the latent variables and the surface orientation required by the branch outcome. The application domain of such a system would not be limited to just image classification; it would be feasible to train a set of randomised trees to estimate the orientation of an arbitrary 3D object to allow tracking in images that are too small for geometric tracking methods to be applied.

CHAPTER 8

Appendices

8.1 List of Publications

The work described in this thesis contributed to the following publications:

Unsupervised Learning of a Scene-Specific Coarse Gaze Estimator
B. Benfold and I. D. Reid
Proceedings of the 13th International Conference on Computer Vision (ICCV), Barcelona, November 2011

Stable Multi-Target Tracking in Real-Time Surveillance Video
B. Benfold and I. D. Reid
Proceedings of the 24th International Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, June 2011

Gaze Directed Camera Control for Face Image Acquisition
E. Sommerlade and B. Benfold and I. D. Reid
Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, May 2011

Guiding Visual Surveillance by Tracking Human Attention
B. Benfold and I. D. Reid
Proceedings of the 20th British Machine Vision Conference (BMVC), London, September 2009
Best Student Poster Award

A Distributed Camera System for Multi-Resolution Surveillance
N. Bellotto, E. Sommerlade, B. Benfold, C. Bibby, I. Reid, D. Roth, C. Fernandez, L. Van Gool and J. Gonzalez
Proceedings of the 3rd ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC) 2009

Colour Invariant Head Pose Classification in Low Resolution Video
B. Benfold and I. D. Reid
Proceedings of the 19th British Machine Vision Conference (BMVC), Leeds, September 2008

8.2 Complete Experimental Results

The result of training and testing each of the gaze classification algorithms described in table 8.1 on every combination of head image datasets (see section 2.6.2) can be found in tables 8.2 and 8.3. In cases where training and testing datasets are the same, 80% of the data was used for training and the remaining 20% was used for testing, with the exception of algorithm e which did require any training data. In some cases the combination of testing and training dataset produces results that are not representative of what could be achieved in a genuine deployed system. Unrealistic experiments are shown with a grey background.

Algorithm	Method	Description
a	HOG/CTC Ferns	Twenty randomised ferns of depth 16 trained with either HOG or CTC features. (see section 4.3.2)
b	Predicate Ferns	Twenty randomised ferns of depth 16 trained with predicate features. (see chapter 3)
c	Mean Template / SVM	Mean Template difference vectors classified with a multi-class SVM (see [91])
d	HOG-CTC / SVM	HOG and CTC vectors classified with a multi-class SVM. (see section 4.3.3)
e	Unsupervised CRF Learning	Forty randomised trees with either HOG or DC features, trained using unsupervised learning with the CRF Model (see chapter 6)

Table 8.1: Algorithms that were tested and their corresponding identifiers.

Table 8.2: Mean Absolute Angular Error (MAAE) resulting from the methods described in table 8.1 when tested and trained on all compatible datasets from table 2.4 in section 2.6.2. Shaded results are from unrealistic training and testing conditions which would not be suitable for practical applications.

Training Dataset	Test Dataset: A	B	C	D	E	F
A	(a) 43.5 ± 2.8	(a) 37.7 ± 0.7	(a) 45.5 ± 1.5	(a) 64.5 ± 1.4	(a) 59.2 ± 1.9	(a) 58.5 ± 1.2
	(c) 57.1 ± 2.1	(c) 52.1	(c) 65.6	(c) 65.5	(c) 61.3	(c) 61.2
	(d) 47.5 ± 2.6	(d) 45.0 ± 0.2	(d) 48.2 ± 0.2	(d) 64.6 ± 0.5	(d) 51.2 ± 0.4	(d) 59.7 ± 0.2
B	(a) 62.0 ± 1.5	(a) 11.9 ± 0.3	(a) 58.8 ± 1.3	(a) 63.6 ± 1.7	(a) 60.6 ± 0.8	(a) 56.5 ± 1.2
	(c) 67.3	(c) 19.3 ± 0.5	(c) 79.2	(c) 54.8	(c) 64.1	(c) 63.8
	(d) 68.2 ± 0.3	(d) 11.7 ± 0.3	(d) 57.8 ± 0.3	(d) 61.2 ± 0.4	(d) 56.3 ± 0.4	(d) 63.3 ± 0.3
C	(a) 56.0 ± 1.1	(a) 47.2 ± 1.7	(a) 32.1 ± 0.9	(a) 51.9 ± 0.9	(a) 41.0 ± 1.3	(a) 56.5 ± 1.2
	(c) 68.9	(c) 65.0	(c) 51.2 ± 1.9	(c) 69.2 ± 0.0	(c) 58.2	(c) 69.6
	(d) 63.1 ± 0.2	(d) 61.6 ± 0.4	(d) 32.9 ± 1.0	(d) 62.8 ± 0.3	(d) 44.2 ± 0.3	(d) 66.8 ± 0.3
	(e) 60.2	(e) 55.6	(e) 23.9	(e) 63.4	(e) 47.5	(e) 57.5
D	(a) 65.0 ± 1.3	(a) 67.0 ± 1.4	(a) 50.2 ± 0.9	(a) 29.7 ± 3.2	(a) 44.3 ± 0.7	(a) 64.5 ± 1.5
	(c) 64.8	(c) 56.4 ± 0.0	(c) 78.4	(c) 40.7 ± 2.5	(c) 56.7	(c) 66.1
	(d) 78.1 ± 0.4	(d) 77.0 ± 0.3	(d) 61.5 ± 0.3	(d) 28.9 ± 2.0	(d) 47.3 ± 0.3	(d) 74.7 ± 0.3
E	(a) 66.2 ± 1.5	(a) 56.4 ± 3.1	(a) 53.4 ± 1.5	(a) 59.9 ± 2.0	(a) 24.2 ± 1.1	(a) 56.9 ± 0.7
	(c) 63.9	(c) 52.3 ± 0.0	(c) 67.3	(c) 51.3	(c) 41.0 ± 2.4	(c) 52.5
	(d) 74.1 ± 0.4	(d) 62.0 ± 0.3	(d) 57.7 ± 0.2	(d) 64.8 ± 0.5	(d) 25.8 ± 1.6	(d) 64.7 ± 0.3
	(e) 71.5	(e) 67.9	(e) 62.8	(e) 69.8	(e) 38.5	(e) 65.4
F	(a) 58.6 ± 1.1	(a) 33.0 ± 1.0	(a) 51.7 ± 0.6	(a) 65.8 ± 1.2	(a) 51.4 ± 1.9	(a) 39.6 ± 3.0
	(b) 71.4 ± 1.0	(b) 34.8 ± 2.1	(b) 67.4 ± 0.6	(b) 68.8 ± 1.8	(b) 59.2 ± 2.8	(b) 55.1 ± 2.1
	(c) 71.4	(c) 51.3	(c) 73.0	(c) 71.2	(c) 56.8	(c) 53.9 ± 1.6
	(d) 59.7 ± 0.4	(d) 43.3 ± 0.2	(d) 54.0 ± 0.3	(d) 58.9 ± 0.5	(d) 48.0 ± 0.1	(d) 40.6 ± 2.0

Table 8.3: Eight-class classification accuracy (% correct) resulting from the methods described in table 8.1 when tested and trained on all compatible datasets from table 2.4 in section 2.6.2. Shaded results are from unrealistic training and testing conditions which would not be suitable for practical applications.

		Test Dataset					
		A	B	C	D	E	F
Training Dataset	A	(a) 46.0 ± 2.9 (c) 32.0 ± 2.5 (d) 44.2 ± 2.6	(a) 43.5 ± 0.9 (c) 34.9 (d) 43.3 ± 0.1	(a) 37.7 ± 0.8 (c) 23.5 (d) 36.7 ± 0.1	(a) 24.1 ± 1.4 (c) 23.5 ± 0.0 (d) 25.5 ± 0.3	(a) 28.5 ± 1.1 (c) 21.5 (d) 33.4 ± 0.3	(a) 36.7 ± 0.8 (c) 27.8 (d) 35.6 ± 0.1
	B	(a) 29.4 ± 0.5 (c) 25.2 (d) 25.7 ± 0.2	(a) 86.1 ± 0.5 (c) 81.5 ± 0.9 (d) 94.4 ± 0.3	(a) 31.7 ± 0.6 (c) 22.1 (d) 33.2 ± 0.2	(a) 30.9 ± 1.3 (c) 28.9 (d) 29.9 ± 0.3	(a) 29.5 ± 0.7 (c) 24.8 (d) 32.7 ± 0.3	(a) 38.6 ± 1.0 (c) 26.8 (d) 32.3 ± 0.3
	C	(a) 30.6 ± 1.0 (c) 23.7 (d) 27.9 ± 0.3 (e) 26.6	(a) 36.4 ± 1.2 (c) 26.0 (d) 26.9 ± 0.3 (e) 33.3	(a) 53.2 ± 1.5 (c) 36.9 ± 1.6 (d) 54.3 ± 1.2 (e) 51.9	(a) 37.7 ± 0.9 (c) 24.9 (d) 32.8 ± 0.2 (e) 25.2	(a) 45.2 ± 0.6 (c) 30.2 (d) 44.4 ± 0.2 (e) 30.0	(a) 32.7 ± 0.7 (c) 23.9 (d) 26.4 ± 0.2 (e) 33.3
	D	(a) 26.3 ± 0.8 (c) 23.5 (d) 22.7 ± 0.2	(a) 29.5 ± 2.8 (c) 30.9 (d) 25.1 ± 0.2	(a) 35.8 ± 0.5 (c) 20.3 (d) 29.9 ± 0.1	(a) 61.1 ± 4.1 (c) 48.8 ± 2.0 (d) 62.5 ± 2.5	(a) 39.8 ± 0.9 (c) 30.1 (d) 37.1 ± 0.3	(a) 32.1 ± 1.0 (c) 25.0 (d) 24.7 ± 0.3
	E	(a) 23.2 ± 1.4 (c) 26.2 (d) 23.2 ± 0.3 (e) 22.9	(a) 34.8 ± 1.6 (c) 35.8 (d) 28.9 ± 0.3 (e) 27.7	(a) 36.1 ± 0.9 (c) 25.4 (d) 33.2 ± 0.2 (e) 29.2	(a) 35.3 ± 1.0 (c) 32.0 (d) 32.1 ± 0.3 (e) 20.5	(a) 63.6 ± 2.4 (c) 43.3 ± 3.3 (d) 67.5 ± 2.4 (e) 40.7	(a) 35.7 ± 0.7 (c) 32.9 (d) 30.3 ± 0.3 (e) 27.7
	F	(a) 34.1 ± 1.0 (b) 24.8 ± 0.6 (c) 19.7 (d) 33.1 ± 0.3	(a) 52.7 ± 1.7 (b) 44.5 ± 0.9 (c) 33.9 (d) 45.2 ± 0.2	(a) 36.5 ± 0.4 (b) 23.9 ± 0.8 (c) 19.5 (d) 36.1 ± 0.2	(a) 30.2 ± 1.1 (b) 28.3 ± 0.9 (c) 21.8 (d) 36.1 ± 0.4	(a) 36.1 ± 1.1 (b) 27.6 ± 1.5 (c) 26.6 (d) 41.3 ± 0.3	(a) 55.4 ± 2.5 (b) 38.4 ± 1.6 (c) 37.0 ± 1.8 (d) 55.6 ± 2.0

8.3 Assumed Background Knowledge

A description of the Linear Kalman Filter is provided for reference.

8.3.1 Linear Kalman Filter

The linear Kalman filter is a well established method for updating state estimates using new observations with Gaussian error. The brief explanation in this appendix is adapted from [130]; a more detailed explanation can be found in Kalman's paper [55].

The state and covariance of the state at time step i using the information up to and including that at time step j are represented by $x_{i|j}$ and $P_{i|j}$ respectively. At each time step, the matrix F_k provides a prediction of the state given the previous one using the system evolution model. The matrix B_{k-1} maps the effect of a known control vector u_{k-1} into the state space and the error of the control input and prediction is Q_{k-1}. The predicted state and covariance are:

$$x_{k|k-1} = F_k x_{k-1|k-1} + B_{k-1} u_{k-1} \tag{8.1}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_{k-1} \tag{8.2}$$

The predictions are then updated using the observations that are made at time k. The vector z_k represents the location of the observation, R_k represents the covariance of the observation, and H_k is a matrix to map from the state space to the observation space. The following five calculations are made to update the state estimate for each observation. First the innovation y_k, which measures how much the observation differs from the predicted value, is calculated along with its covariance S_k:

$$y_k = z_k - H_k x_{k|k-1} \tag{8.3}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{8.4}$$

Next the optimal Kalman gain is calculated:

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{8.5}$$

The Kalman gain specifies the fraction of the observation that should be used to update the state:

$$X_{k|k} = x_{k|k-1} + K_k y_k \tag{8.6}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{8.7}$$

If there are no observations then multiple predictions can be made without any state updates.

Bibliography

[1] J. Aghajanian and S.J.D. Prince. Face pose estimation in uncontrolled environments. In *Proceedings of the 20th British Machine Vision Conference (BMVC)*, September 2009. 17, 22

[2] Irshad Ali and Matthew Dailey. Multiple human tracking in high-density crowds. In *Advanced Concepts for Intelligent Vision Systems*, volume 5807 of *LNCS*, pages 540–549. Springer, 2009. 28

[3] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997. 34

[4] Sileye O. Ba and Jean-Marc Odobez. Evaluation of multiple cue head pose estimation algorithms in natural environments. In *ICME*, pages 1330–1333. IEEE, July 2005. 17, 20, 22, 49

[5] Sileye O. Ba and Jean-Marc Odobez. Multiperson visual focus of attention from head pose and meeting contextual cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(1):101–116, 2011. 25

[6] Vineeth Nallure Balasubramanian, Sreekar Krishna, and Sethuraman Panchanathan. Person-independent head pose estimation using biased manifold embedding. *EURASIP Journal on Image and Video Processing*, 2008, 2008. 14, 22, 27

[7] Yaakov Bar-Shalom and Edison Tse. Tracking in a cluttered environment with probabilistic data association. *Automatica*, 11(5):451 – 460, 1975. 118

[8] Chiraz BenAbdelkader. Robust head pose estimation using supervised manifold learning. In *ECCV (6)*, volume 6316 of *LNCS*, pages 518–531. Springer, September 2010. 14, 22, 27

[9] J. Berclaz, F. Fleuret, and P. Fua. Multiple object tracking using flow linear programming. In *Proceedings of the 12th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (Winter-PETS)*, Snowbird, Utah, December 2009. 32

[10] N. Bergman and A. Doucet. Markov chain monte carlo data association for target tracking. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages II705 –II708 vol.2, 2000. 119

[11] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. In *EURASIP Journal on Image and Video Processing*, volume 2008. 2008. 117, 134

[12] C. Bibby and I. Reid. Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association. In *Proceedings of Robotics Science and Systems*, 2007. 117, 118

[13] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. 11, 148

[14] A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *Proceedings of the 11th IEEE International Conference on Computer Vision*, pages 1–8, October 2007. 34

[15] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. 35

[16] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. 35

[17] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984. 34

[18] M.D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *Proceedings of the 12th IEEE International Conference Computer Vision (ICCV)*, pages 1515 –1522, September 2009. 31, 41, 139

[19] Lisa M. Brown and Ying-Li Tian. Comparative study of coarse head pose estimation. In *Proceedings of the Workshop on Motion and Video Computing (MOTION '02)*, page 125, Washington, DC, USA, 2002. IEEE Computer Society. 12, 22

[20] Roberto Brunelli. Estimation of pose and illumination direction. AI Memo 1499, Massachusetts Institute of Technology, November 1994. 15

[21] Cristian Canton-Ferrer, Josep R. Casas, and Montse Pardàs. Head orientation estimation using particle filtering in multiview scenarios. In *CLEAR*, volume 4625 of *LNCS*, pages 317–327. Springer, 2007. 16, 22

[22] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991. 84, 121

[23] Longbin Chen, Lei Zhang, Yuxiao Hu, Mingjing Li, and HongJiang Zhang. Head pose estimation using fisher manifold learning. In *AMFG*, pages 203–207. IEEE Computer Society, 2003. 14, 22

[24] Qian Chen, Tetsuo Shimada, Haiyuan Wu, and Tadayoshi Shioyama. Head pose estimation using both color and feature information. In *ICPR*, pages 2842–2841, 2000. 11

[25] Qian Chen, Haiyuan Wu, Takeshi Fukumoto, and Masahiko Yachida. 3D head pose estimation without feature tracking. In *FG*, pages 88–93. IEEE Computer Society, April 1998. 15, 49

[26] Shinko Y. Cheng, Sangho Park, and Mohan M. Trivedi. Multi-spectral and multi-perspective video arrays for driver body tracking and activity analysis. *Computer Vision and Image Understanding*, 106(2-3):245–257, 2007. 25

[27] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, May 1968. 64

[28] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1197–1203, 1999. 27, 82

[29] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 886–893. IEEE Computer Society, June 2005. 27, 29, 42, 82, 93, 119, 121

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977. 148

[31] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 30

[32] Thomas E. Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Multi-target tracking using joint probabilistic data association. In *Proceedings of the 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, volume 19, pages 807–812, December 1980. 118

[33] Nir Friedman, Dan Geiger, and Moisés Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997. 64

[34] Yun Fu and Thomas S. Huang. Graph embedded analysis for head pose estimation. In *FG*, pages 3–8. IEEE Computer Society, April 2006. 14, 22, 27

[35] Weina Ge and Robert T. Collins. Multi-target data association by tracklets with unsupervised parameter estimation. In *Proceedings of the 19th British Machine Vision Conference (BMVC)*, 2008. 119, 132

[36] Graeme Gerrard and Richard Thompson. Two million cameras in the uk. *CCTV Image Magazine*, (42):10–12, 2011. 1

[37] Martin Gill and Angela Spriggs. Assessing the impact of CCTV. Technical Report 292, UK Home Office, February 2005. 2

[38] Martin Gill, Angela Spriggs, Jenna Allen, Martin Hemming, Patricia Jessiman, Deena Kara, Jonathan Kilworth, Ross Little, and Daniel Swain. Control room operation: findings from control room observations. Technical Report 14/05, UK Home Office, 2005. 1

[39] Shaogang Gong, Stephen McKenna, and John J. Collins. An investigation into face pose distributions. In *Proceedings of the 2nd International Conference on*

Automatic Face and Gesture Recognition (FG '96), pages 265–, Washington, DC, USA, 1996. IEEE Computer Society. 13

[40] Shaogang Gong, Tao Xiang, and Somboon Hongeng. Learning human pose in crowd. In *Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis*, MPVA '10, pages 47–52, New York, NY, USA, 2010. ACM. 29

[41] J. González, F. Xavier Roca, and J. José Villanueva. Hermes: A research project on human sequence evaluation. In *Computational Vision and Medical Image Processing (VipIMAGE'2007)*, 2007. 44, 67

[42] Nicolas Gourier, Jérôme Maisonnasse, Daniela Hall, and James L. Crowley. Head pose estimation on low resolution images. In *CLEAR*, volume 4122 of *LNCS*, pages 270–280. Springer, 2006. 12, 22, 24

[43] Zhibo Guo, Huajun Liu, Qiong Wang, and Jingyu Yang. A fast algorithm face detection and head pose estimation for driver assistant system. In *8th International Conference on Signal Processing*, 2006. 13, 22, 25

[44] Ismail Haritaoglu and Myron Flickner. Attentive billboards: Towards to video based customer behavior. In *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision*, WACV '02, pages 127–, Washington, DC, USA, 2002. IEEE Computer Society. 26

[45] J.J. Heuring and D.W. Murray. Modeling and copying human head movements. *IEEE Transactions on Robotics and Automation*, 15(6):1095–1108, December 1999. 158

[46] Tin Kam Ho. Random decision forests. In *ICDAR*, pages 278–, 1995. 35

[47] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. 35

[48] Tin Kam Ho. A data complexity analysis of comparative advantages of decision forest constructors. *Pattern Analysis and Applications*, 5(2):102–112, 2002. 35

[49] Nan Hu, Weimin Huang, and Surendra Ranganath. Head pose estimation by non-linear embedding and mapping. In *ICIP (2)*, pages 342–345, 2005. 14, 22

[50] Chang Huang, Bo Wu, and Ramakant Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV (2)*, volume 5303 of *LNCS*, pages 788–801. Springer, October 2008. 32

[51] J. Huang, X. Shao, and H. Wechsler. Face pose discrimination using support vector machines (SVM). In *Proceedings of the 14th International Conference on Pattern Recognition (ICPR)*, volume 1, pages 154 –156 vol.1, August 1998. 19, 22

[52] Y. Huang, S. Lin, H.Q. Lu, and H.Y. Shum. Face alignment using intrinsic information. In *ICIP*, pages V: 3307–3310, 2004. 11

[53] Christopher Jaynes, Amit Kale, Nathaniel Sanders, and Etienne Grossmann. The terrascope dataset: A scripted multi-camera indoor video surveillance dataset with ground-truth. In *Proceedings of the IEEE Workshop on VS PETS*, October 2005. 44, 67

[54] Fan Jiang, Junsong Yuan, Sotirios A. Tsaftaris, and Aggelos K. Katsaggelos. Anomalous video event detection using spatiotemporal context. *Computer Vision and Image Understanding*, 115(3):323–333, 2011. 3

[55] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960. 183

[56] Hina Keval and Martina Angela Sasse. Not the usual suspects: A study of factors reducing the effectiveness of CCTV. *Security Journal*, 23:134–154, October 2008. 1

[57] Hannes Kruppa, Modesto Castrillon-Santana, and Bernt Schiele. Fast and robust face finding via local context. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, October 2003. 30

[58] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498 – 519, February 2001. 149

[59] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955. 118

[60] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289. Morgan Kaufmann, 2001. 147

[61] Oswald Lanz. Approximate bayesian multibody tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9):1436–1449, 2006. 28

[62] Andreas Launila. Real-time head pose estimation in low-resolution football footage. Master's thesis, School of Computer Science and Communication, Royal Institute of Technology, Stockholm, Sweden, 2009. 27, 79

[63] Andreas Launila and Josephine Sullivan. Contextual features for head pose estimation in football games. In *ICPR*, pages 340–343. IEEE, 2010. 18, 22

[64] Mi-Suen Lee. Detecting people in cluttered indoor scenes. In *CVPR*, pages 1804–1809. IEEE Computer Society, June 2000. 28

[65] Bastian Leibe, Ales Leonardis, and Bernt Schiele. An implicit shape model for combined object categorization and segmentation. In *Toward Category-Level Object Recognition*, volume 4170 of *LNCS*, pages 508–524. Springer, 2006. 31

[66] Bastian Leibe, Konrad Schindler, and Luc J. Van Gool. Coupled detection and trajectory estimation for multi-object tracking. In *Proceedings of the 11th IEEE International Conference on Computer Vision*, pages 1–8. IEEE, October 2007. 32, 124

[67] Christian Leistner, Amir Saffari, and Horst Bischof. Miforests: Multiple-instance learning with randomized trees. In *ECCV (6)*, volume 6316 of *LNCS*, pages 29–42. Springer, September 2010. 147

[68] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9):1465–1479, 2006. 34, 37

[69] Yuan Li, Haizhou Ai, Chang Huang, and Shihong Lao. Robust head tracking with particles based on multiple cues fusion. In *ECCV Workshop on HCI*, volume 3979 of *LNCS*, pages 29–39. Springer, 2006. 28

[70] Yuan Li, Chang Huang, and Ram Nevatia. Learning to associate: Hybrid-boosted multi-target tracker for crowded scene. In *CVPR*, pages 2953–2960. IEEE, June 2009. 32

[71] Zhu Li, Yun Fu, Junsong Yuan, Thomas S. Huang, and Ying Wu. Query driven localized linear discriminant models for head pose estimation. In *ICME*, pages 1810–1813. IEEE, July 2007. 14, 22

[72] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *ICIP (1)*, pages 900–903, 2002. 30

[73] J. Liu, X. Tong, W. Li, T. Wang, Y. Zhang, H. Wang, B. Yang, L. Sun, and S. Yang. Automatic player detection, labeling and tracking in broadcast soccer video. In *Proceedings of the 18th British Machine Vision Conference (BMVC)*, 2007. 119

[74] Xiaoming Liu, Nils Krahnstoever, Ting Yu, and Peter H. Tu. What are customers looking at? In *AVSS*, pages 405–410. IEEE Computer Society, 2007. 26

[75] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In Patrick J. Hayes, editor, *IJCAI*, pages 674–679. William Kaufmann, August 1981. 84, 121

[76] A. Tavano M. Farenzena, L. Bazzani, D. Tosato, G.Paggetti, G. Menegaz, V. Murino, and M.Cristani. Social interactions by visual focus of attention in a three-dimensional environment. In *Workshop on Pattern Recognition and Artificial Intelligence for Human Behaviour Analysis (PRAI-HBA)*, December 2009. 25, 28

[77] Bingpeng Ma, Wenchao Zhang, Shiguang Shan, Xilin Chen, and Wen Gao. Robust head pose estimation using LGBP. In *ICPR (2)*, pages 512–515. IEEE Computer Society, 2006. 19, 22

[78] S. J. McKenna and S. Gong. Real-time face pose estimation. *Real-Time Imaging*, 4:333–347, October 1998. 13

[79] R. Mehran, A. Oyama, and M. Shah. Abnormal crowd behavior detection using social force model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 935 –942. IEEE, June 2009. 3

[80] Ramin Mehran, Brian Moore, and Mubarak Shah. A streakline representation of flow in crowded scenes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6313 of *LNCS*, pages 439–452, 2010. 3

[81] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104:90–126, October 2006. 29

[82] Louis-Philippe Morency, Patrik Sundberg, and Trevor Darrell. Pose estimation using 3D view-based eigenspaces. In *AMFG*, pages 45–52. IEEE Computer Society, 2003. 13, 22

[83] Erik Murphy-Chutorian and Mohan M. Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(4):607–626, 2009. 21, 46

[84] Erik Murphy-Chutorian and Mohan M. Trivedi. Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):300–311, 2010. 25

[85] Jeffrey Ng and Shaogang Gong. Multi-view face detection and pose estimation using a composite support vector machine across the view sphere. In *Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, RATFG-RTS '99, pages 14–, Washington, DC, USA, 1999. IEEE Computer Society. 19, 22

[86] A. Nikolaidis and I. Pitas. Facial feature extraction and pose determination. *PR*, 33(11):1783–1791, November 2000. 11

[87] Sourabh Niyogi and William T. Freeman. Example-based head tracking. In *FG*, pages 374–378. IEEE Computer Society, 1996. 16, 22

[88] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, USA, August 2000. 160

[89] Jean-Marc Odobez and Sileye O. Ba. A cognitive and unsupervised map adaptation approach to the recognition of the focus of attention from head pose. In *ICME*, pages 1379–1382. IEEE, July 2007. 25

[90] Songhwai Oh, Stuart Russell, and Shankar Sastry. Markov chain monte carlo data association for general multiple-target tracking problems. In *ICDC*, 2004. 119

[91] Javier Orozco, Shaogang Gong, and Tao Xiang. Head pose classification in crowded scenes. In *Proceedings of the 20th British Machine Vision Conference*, September 2009. 18, 20, 22, 28, 45, 92, 95, 97, 180

[92] O. Ozturk, T. Yamasaki, and K. Aizawa. Tracking of humans and estimation of body/head orientation from top-view single camera for visual focus of attention analysis. In *Proceedings of the 2nd IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), ICCV*, pages 1020 –1027, 2009. 11, 28

[93] Mustafa Özuysal, Michael Calonder, Vincent Lepetit, and Pascal Fua. Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(3):448–461, 2010. 38

[94] Mustafa Özuysal, Pascal Fua, and Vincent Lepetit. Fast keypoint recognition in ten lines of code. In *CVPR*. IEEE Computer Society, 2007. 37

[95] Ravikanth Pappu and Paul A. Beardsley. A qualitative approach to classifying gaze direction. In *FG*, pages 160–165. IEEE Computer Society, April 1998. 15, 23, 25, 27

[96] Hanna Pasula, Stuart J. Russell, Michael Ostland, and Yaacov Ritov. Tracking many objects with many sensors. In Thomas Dean, editor, *IJCAI*, pages 1160–1171. Morgan Kaufmann, July 1999. 119

[97] Alonso Patron, Marcin Marszalek, Andrew Zisserman, and Ian Reid. High five: Recognising human interactions in TV shows. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 50.1–50.11. BMVA Press, 2010. doi:10.5244/C.24.50. 26

[98] Judea Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pages 133–136, 1982. 148

[99] Victor Prisacariu and Ian Reid. fastHOG - a real-time GPU implementation of HOG. Technical Report 2310/09, Department of Engineering Science, Oxford University, 2009. 106, 119, 121

[100] Robert Rae and Helge Ritter. Recognition of human head orientation based on artificial neural networks. *IEEE Transactions on Neural Networks*, 9(2):257–265, 1998. 12, 23, 27, 49

[101] Deva Ramanan, David A. Forsyth, and Andrew Zisserman. Tracking people by learning their appearance. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(1):65–81, 2007. 50

[102] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843 – 854, December 1979. 118

[103] Ian D. Reid and David W. Murray. Active tracking of foveated feature clusters using affine structure. *International Journal of Computer Vision*, 18(1):41–60, 1996. 88

[104] N.M. Robertson, I. Reid, and M. Brady. Behaviour recognition and explanation for video surveillance. *The IET Conference on Crime and Security*, pages 458–463, June 2006. 25

[105] N.M. Robertson and I.D. Reid. Estimating gaze direction from low-resolution faces in video. In *ECCV (2)*, volume 3952 of *LNCS*, pages 402–415. Springer, May 2006. 17, 23, 27, 49

[106] N.M. Robertson, I.D. Reid, and J.M. Brady. What are you looking at? gaze estimation in medium-scale images. In *Proceedings of the Workshop on Human Activity Recognition and Modelling (HAREM), BMVC*, volume 1, September 2005. 17, 25, 27

[107] Stuart Russel and Peter Norvig. *Artificial Intelligence - A Modern Approach*, chapter 15.3, pages 549–551. Prentice Hall, 2003. 60

[108] E. Seemann, K. Nickel, and R. Stiefelhagen. Head pose estimation using stereo vision for human-robot interaction. In *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 626 – 631, May 2004. 12, 23, 29

[109] Jamie Sherrah and Shaogang Gong. Fusion of 2D face alignment and 3D head pose estimation for robust and real-time performance. In *Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 24 –30, 1999. 15

[110] Jamie Sherrah and Shaogang Gong. Fusion of perceptual cues for robust tracking of head pose and position. *Pattern Recognition*, 34(8):1565 – 1572, 2001. 15

[111] Jamie Sherrah, Shaogang Gong, A. Jonathan Howell, and Hilary Buxton. Interpretation of group behavior in visually mediated interaction. In *ICPR*, pages 1266–1269, 2000. 25

[112] Jianbo Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, June 1994. 84

[113] Teera Siriteerakul, Daisuke Sugimura, and Yoichi Sato. Head pose classification from low resolution images using pairwise non-local intensity and color differences. In *Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, November 2010. 19, 23, 98

[114] Kevin Smith, Sileye O. Ba, Jean-Marc Odobez, and Daniel Gatica-Perez. Tracking the visual focus of attention for a varying number of wandering

people. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(7):1212–1229, 2008. 17, 26, 28

[115] Bi Song, Ting-Yueh Jeng, Elliot Staudt, and Amit K. Roy-Chowdhury. A stochastic graph evolution framework for robust multi-target tracking. In *ECCV 2010*, volume 6311 of *LNCS*, pages 605–619. Springer, September 2010. 119

[116] S. Srinivasan and K.L. Boyer. Head pose estimation using view based eigenspaces. In *Proceedings of the 16th International Conference on Pattern Recognition (ICPR)*, volume 4, pages 302 – 305 vol.4, 2002. 13, 23, 27

[117] Severin Stalder, Helmut Grabner, and Luc J. Van Gool. Cascaded confidence filtering for improved tracking-by-detection. In *ECCV (1)*, volume 6311 of *LNCS*, pages 369–382. Springer, September 2010. 32, 41, 134, 139

[118] Rainer Stiefelhagen. Tracking focus of attention in meetings. In *ICMI*, pages 273–280. IEEE Computer Society, October 2002. 24

[119] Rainer Stiefelhagen. Estimating head pose with neural networks - results on the Pointing04 ICPR workshop evaluation data. In *Pointing '04 ICPR workshop*, August 2004. 12

[120] Rainer Stiefelhagen, Michael Finke, Jie Yang, and Alex Waibel. From gaze to focus of attention. In *VISUAL*, volume 1614 of *LNCS*, pages 761–768. Springer, 1999. 12, 23, 49

[121] Charles Sutton and Andrew McCallum. An introduction to conditional random fields. November 2010. 152

[122] Ying-Li Tian, Lisa Brown, Jonathan H. Connell, Sharath Pankanti, Arun Hampapur, Andrew W. Senior, and Ruud M. Bolle. Absolute head pose estimation from overhead wide-angle cameras. In *AMFG*, pages 92–99. IEEE Computer Society, 2003. 12, 23, 27

[123] Antonio Torralba and Pawan Sinha. Detecting faces in impoverished images. AI Memo 2001-028, Massachusetts Institute of Technology, November 2001. 48

[124] Paul A. Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR) (1)*, pages 511–518. IEEE Computer Society, December 2001. 16, 28, 29

[125] Michael Voit, Kai Nickel, and Rainer Stiefelhagen. A bayesian approach for multi-view head pose estimation. *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 31–34, September 2006. 12, 23, 27

[126] Michael Voit and Rainer Stiefelhagen. Tracking head pose and focus of attention with multiple far-field cameras. In *Proceedings of the 8th international conference on Multimodal interfaces*, ICMI '06, pages 281–286, New York, NY, USA, 2006. ACM. 12, 23, 27

[127] Ce Wang and Michael Brandstein. Robust head pose estimation by machine learning. In *ICIP*, 2000. 20, 23

[128] Xianwang Wang, Xinyu Huang, Jizhou Gao, and Ruigang Yang. Illumination and person-insensitive head pose estimation using distance metric learning. In *ECCV (2)*, volume 5303 of *LNCS*, pages 624–637. Springer, October 2008. 14, 23

[129] Yucheng Wei, Ludovic Fradet, and Tieniu Tan. Head pose estimation using gabor eigenspace modeling. In *ICIP (1)*, pages 281–284, 2002. 13, 23, 25

[130] Wikipedia. Kalman filter — wikipedia, the free encyclopedia, 2011. [Online; accessed 10-August-2011]. 183

[131] B. Williams, G. Klein, and I. Reid. Real-time SLAM relocalisation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007. 37

[132] David S. Wooding. Fixation maps: quantifying eye-movement traces. In *ETRA*, pages 31–36. ACM, 2002. 103

[133] Bo Wu and Ram Nevatia. Tracking of multiple, partially occluded humans based on static body part detection. In *CVPR (1)*, pages 951–958. IEEE Computer Society, 2006. 32, 82

[134] Ying Wu and Kentaro Toyama. Wide-range, person- and illumination-insensitive head orientation estimation. In *FG*, pages 183–188. IEEE Computer Society, March 2000. 16, 23, 27

[135] Qian Yu, Gérard G. Medioni, and Isaac Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *CVPR*. IEEE Computer Society, 2007. 119

[136] Xenophon Zabulis, Thomas Sarmis, and Antonis A. Argyros. 3D head pose estimation from multiple distant views. In *Proceedings of the 20th British Machine Vision Conference (BMVC)*, September 2009. 16, 23, 27

[137] Liang Zhao, Gopal Sarma Pingali, and Ingrid Carlbom. Real-time head orientation estimation using neural networks. In *ICIP (1)*, pages 297–300, 2002. 11, 23