Tracking and Modelling of Team Game Interactions

by

Christopher James Needham

Submitted in accordance with the requirements for the degree of Doctor of Philosophy.

The University of Leeds
School of Computing

October 2003

The candidate confirms that the work submitted is his own and that the appropriate credit has been given where reference has been made to the work of others.

Abstract

Team games are complicated activities, which involve much interaction between players. Analysing these interactions is of considerable interest; however, it is time consuming, error prone, and unreliable to manually obtain the positions of the players throughout a game. Automating this process could produce efficient and repeatable results.

Multiple object tracking in large, congested, rapidly changing, and frequently occluded domains (for example a soccer pitch) is a complex problem, particularly given the non-linear nature of each player's movements. This thesis presents a stochastic sampling based multiple object tracker, capable of tracking objects from a single camera, in the complex domain of sports games. Sports players' shapes vary dramatically, presenting challenges to existing techniques. Multi-resolution template based feature descriptors are learned from example players' shape data, providing a mechanism for identifying players' locations in images. Sports scenes are often busy, in the sense that there may be many players close to each other, causing occlusion of one or more players. The use of multiple cameras to resolve these ambiguities is investigated.

Performance evaluation of computer vision systems is an important and often understudied activity. The performance evaluation in this thesis focuses on positional performance evaluation. New metrics and statistics are presented which provide an important insight into how well a tracking system is performing (and why it may not be).

Analysing the movements of the players over time allows a behaviour model of their movements and interactions to be learned. Positional player data is represented and described using density estimation methods. An emergent approach to identifying players is presented. Each player's response from a set of learned Gaussian mixture models is used in a graph partitioning scheme. This allows the identification of each player's 'position'. Behaviour and interaction models have potential uses for analysing tactics, identifying good or atypical players, and most powerfully to be incorporated into a multiple object tracking system to govern the expected dynamics of the players.

Acknowledgements

I would like to thank my colleagues in the Vision Group for many stimulating and profitable discussions. My supervisor Roger Boyle has encouraged me to complete this work and has always been there whenever I have needed him.

I would also like to thank the School of Sport and Exercise Science and staff at the Sports Centre at the University of Leeds for their support in obtaining video footage, and the EPSRC for the financial support of a PhD studentship award.

Many thanks are also due to the running community in Leeds for keeping me sane (or allowing me not to be) through the hours spent training and racing over the years.

Declarations

Some parts of the work presented in this thesis have been published in the following articles:

C. J. Needham and R. D. Boyle, "Tracking multiple sports players through occlusion, congestion and scale", In *Proc. British Machine Vision Conference*, pages 93–102, Manchester, UK, 2001. BMVA.

C. J. Needham and R. D. Boyle, "Performance evaluation metrics and statistics for positional tracker evaluation", In *Proc. Third Intl. Conference on Computer Vision Systems*, number 2626 in LNCS, pages 278–289, Graz, Austria, 2003. Springer.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, there has been a growth of activity in the areas of multiple object tracking and the modelling of interactions. Advances in computer hardware and the adoption and development of theoretical methodologies from related disciplines allow more computationally challenging problems to be approached. This has increased interest in surveillance systems and the interpretation of actions identified by such systems, and the possibility of real-time tracking applications are now realistic. This thesis aims to explore the feasibility of a computer vision system for tracking sports players.

1.1 Aims and motivations

Sport is a rich and diverse domain. It captivates the minds of the general public and sports are followed by millions of people around the world. To the computer vision researcher, it provides an abundant source of captivating footage from which to work. This thesis has two goals: to track the movements of sports (specifically football) players on an indoor pitch and to model the behaviour of the sports players within team games. Alongside these two goals, there are two primary motivations for this work.

Firstly, the sports science industry is very interested in being able to know how much ground athletes have covered, and how quickly they have moved, during the course of a game. This information would allow more specific training to be designed to suit individual players. An example of the extent to which professional football teams are interested in knowing where their players have been during a game is shown by the amount of money which is spent collecting statistics during the game, including numbers of completed passes, interceptions, corners, attempts on goal, etc. In the case of several teams a company called ProZone have been employed to film the games using many cameras, and then manually marking up the game overnight, allowing statistics of the positions, movements and speeds of the players to be identified. Figure 1.1 shows an example graphic produced by ProZone, illustrating the movements of David Beckham throughout a game.

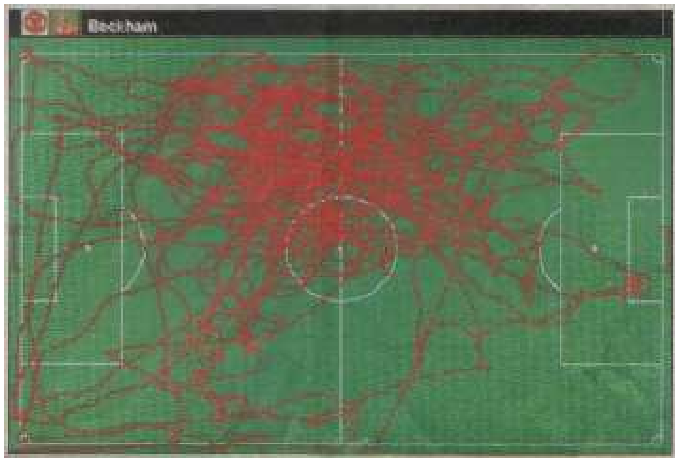


Figure 1.1: Graphic of Beckham's trajectory during a game. Produced by ProZone, appeared in the Times, Oct 2001.

With regard to computer vision, the tracking of sports players from video presents a challenging domain in which many people interact, occlude, make sudden body movements, and move in a non-linear fashion covering a large area of ground.

Secondly, the analysis of positional data from such a system to identify team game interactions is a fascinating research area. Sports games that involve two teams of players provide a rich environment for modelling cooperative, collaborative and adversarial actions of individuals and for modelling the behaviour of the teams as a whole. FA Premiership clubs are becoming more aware of the possibilities of team and individual performance analysis of football matches. Appendix A contains a recent press article discussing new technology in use at Leeds United FC.

1.2 The soccer domain

This thesis examines team games in the form of indoor 5-a-side soccer, with a view to creating a tracking system capable of extension to other domains, through consideration of the constraints imposed in the real world. These constraints take various forms: cost, efficiency, choice of camera location, and purpose.

Example footage that is used for tracking is shown in Figure 1.2. Examining this image, it can be seen that 5-a-side soccer is a limited domain; a maximum of ten players will be in view at any time, players are confined to the pitch, and the players are organised into two teams identified by coloured shirts. However, the soccer domain exhibits many challenging aspects: the size of the pitch means that the resolution of an image of the game varies greatly between the nearest and the furthest parts of the pitch; sports games are busy areas; sports players' shapes vary significantly, often in short periods; and the players move at variable speeds, often suddenly changing direction, which makes their movements hard to predict. In addition, environmental variations can be considerable, even indoors. Players cast shadows on the floor, and the lighting varies and also produces some areas of 'glare' at the camera end of the pitch.

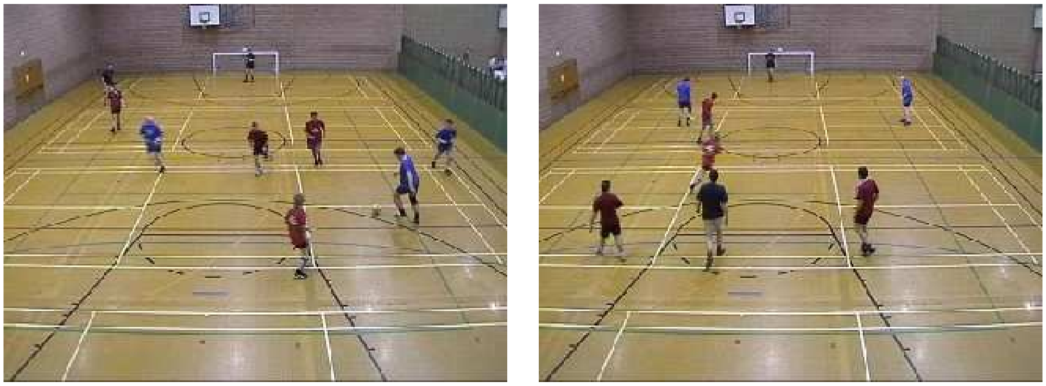


Figure 1.2: Example footage of soccer players.

Tracking multiple target objects from footage of a soccer game presents many occasions on which part of a player occludes another in the image. These occurrences of players overlapping can cause problems for image processing and low-level machine vision techniques (those which rely solely on image information).

It would be possible to use a set of 99 calibrated cameras to allow a high resolution image of each player to be obtained, or even to mount many cameras directly overhead reducing the problem to blob-tracking. Such an approach would not be readily transferable to sports stadia on the grounds of cost or practicality respectively.

Team games are complicated activities, which involve a lot of interaction between the players. This multi-player activity allows us to explore the relations and interactions, both between players, and the teams as a whole. The ultimate goal is a real-time, accurate, automatic tracking system of all objects on the sports pitch and a system capable of behaviour analysis.

Such a non-intrusive system used during a competitive match could help to decide which players are tiring earlier than normal (do players get injured when they tire atypically?) or even to decide which tactics are most successful against the opposition.

Other tracking systems developed do not appear to be robust to the challenges of this domain. The main problems foreseen are:

- perspective; soccer covers a large area, and due to the perspective effects, the resolution of players on different parts of the image is significantly different.
- shape; soccer players' shapes vary considerably and change quickly, relative to pedestrians or cars in typical surveillance applications.
- occlusion; commonly players obscure each other as viewed from the same camera, they overlap on the image, increasing the complexity of the tracking.
- evaluation; how should a tracker (or machine vision algorithms) be evaluated?
- behaviour; how to represent interactions between groups or individuals, it is an under developed research area.

This thesis will explore these areas in search of insights or solutions, whilst considering practical issues of computational efficiency and the attractiveness of a single uncalibrated camera (or as few cameras as possible).

1.3 Thesis overview

This section details how the remainder of this thesis is organised.

Chapter 2 identifies our needs in greater detail. A review of relevant work, in the varied areas of low-level computer vision; incorporating image processing and shape modelling techniques, to higher level computer vision techniques; reviewing tracking and behaviour modelling. Approaches to evaluating algorithms and systems are discussed, and a variety of current trackers and tracking systems are explored. Throughout this chapter conclusions are drawn about the best approaches to take in order to fulfil the aims of this thesis.

A multiple object tracker is presented in Chapter 3. The perspective effects of the image are examined, and methods for image transformations are introduced. Within a CONDENSATION [47] based tracking scheme, positions of the players are represented in the real world coordinate system, since this provides greater separation between players than in the image plane where the players frequently occlude. Incorporation of a set of Kalman filters into the CONDENSATION framework is shown to help group the samples back together, and improve the tracking results.

Discussion about the plausibility of shape models for representing the numerous poses of a sports player appears in Chapter 4, along with a multi-resolution texture based shape model learned from example data. Several methods of employing this approach are compared and evaluated. The method is also demonstrated on video footage of pedestrians, and its potential use for modelling the transitions between poses is shown.

Chapter 5 presents work on trajectory description. The need for tracker evaluation is of paramount importance, and ways of describing and comparing trajectories are presented which allow evaluation of a tracking system as a whole.

Tracking multiple sports players from multiple cameras is the focus of Chapter 6. The framework set up in Chapter 3 is extended to establish a common coordinate system, through calibrating each of the images used, and consideration is given to the calculation of the update parameters of the Kalman filters. The multi-resolution texture based shape model of Chapter 4 is incorporated into the tracker as the fitness function, and methods of combining data from multiple views is discussed. Example tracked footage is shown, alongside evaluation and discussion about the tracker's successes and failings.

Chapter 7 discusses the collection of data for behaviour modelling, and creates probability density maps of real world data from a 5-a-side soccer game using a Gaussian mixture model, and secondly using vector quantisation. A system for generating synthetic data (with endless supply) is designed to provide a greater quantity of data with which to work. Emergent methods are used on this data to identify players' positions, and approaches to a variety of tasks involving behaviour modelling and interactions are discussed.

Finally, conclusions and further work are discussed in Chapter 8.

Chapter 2

Computer vision and behaviour modelling: A review

This chapter identifies the areas of research relevant to multiple object tracking from video sequences, looking at each stage from the initial image processing and segmentation of the video, the fitting of shape models to the extracted foreground shapes, filtering and tracking methods used to predict future locations/shapes, methods for tracking multiple targets, and the fusion of information from multiple cameras/views. Finally, event recognition and behaviour modelling research is explored.

Computer vision systems are generally modular in their approach. It is usual for the process to begin by applying a general processing or image filtering technique to the whole image, to either reduce the dimensionality (3 colour channels to a single channel foreground-background image), or to highlight features in the image, to aid subsequent algorithms.

2.1 The segmentation task

There are many ways of representing colour. Most video formats use RGB (red, green, blue), but for some applications other *perceptual* colour spaces such as HSI (hue, saturation, intensity) are more appropriate, since in such a space, perceptually similar colours are 'closer' to each other. Vandenbroucke *et al.* [111] experiment with a hybrid colour space for classifying pixels into one of a number of categories. The method is applied to the soccer domain, identifying thresholds for the background, and choosing three colour representations from a large set, which give the best separation between sets of training foreground pixels (images of soccer players on two teams), which provides a good segmentation and team identification. Due to the non-time-dependent approach, this method is robust to small camera jitters, and stationary objects. Their footage is taken on an outdoor grass football pitch, and the method relies upon an empty similarly coloured background. The sports centre floor that is used in this work is covered in multi-coloured pitch markings for many different sporting activities, which makes the segmentation task more challenging. Incorporating an image segmentation scheme which can cope with these conditions will lead to a tracker with greater generality. Such a tracker will be more robust to different situations and should be less dependent on video footage being similar to the scenario on which it is developed.

Video analysis allows information from previous image frames to be used to aid segmentation. Maintaining a temporal background model and performing background subtraction has been shown to be a fast and efficient method of extracting moving objects from a scene [3]. This performs best in relatively empty scenes through which objects are moving, however, sporting activities do not fit into this category. In busy scenes, where sports players are always on the pitch (in the field of view), it is difficult to adjust the update parameters to allow both fast moving objects to be segmented and objects which stay still for a period of time, as sports players (particularly in sports like netball) often have a tactical position in which they stand for short lengths of time, and can become incorporated into the background model through dynamic background maintenance. If a static background model is used to combat this, it may not allow for changes in the lighting conditions or small camera movements.

Recently, more complex (and computationally expensive) background subtraction methods have been developed, notably adaptive background subtraction [71] which uses the colour information in the image to dynamically maintain a mean and standard deviation for each colour channel (R,G,B) for each pixel, and after a training period, identifies the pixel as foreground if the RGB values at the pixel are outside α standard deviations of the mean value, with α typically equal to 3. This method produces a well segmented foreground image, in general, however it is found to be oversensitive to camera jitter, particularly with respect to the multi-coloured pitch markings.

Colour mixture models [84] may be used to locate and track objects in dynamic scenes. The ‘Grimson method’ of adaptive background mixture models [99] models each pixel as a mixture of Gaussians, and considers whether a pixel value fits the background model. This method is stable, computationally efficient, and is employed within the work described in Chapter 6, with further details and discussion of parameters.

2.2 Shape modelling

Early approaches to extracting shapes of objects from images used *snakes* [98]; deformable lines iteratively fitted to features extracted through edge detection or similar routines. Various techniques have been applied to improve this technique, *gradient vector flow* (GVF) [119] and *colorimetric* attributes of target objects [113] have improved these methods, although the main drawback of a snake based approach is that the resulting shape (contour) may not represent the target object sufficiently well, due to missed features, or noisy data, and is particularly poor for occlusion reasoning.

Contour models vastly increase the power of an iterative fitting approach, and have been used in a wide variety of machine vision applications when the objects to be identified are similar in nature, for example industrial inspection of resistors [20], pedestrians in car parks [3], and leaves on bushes [65]. Flexible shape models such as the *point distribution model* (PDM) [21] allow a model of the target object to be built from sets of examples of labelled points on the outline of the desired object. To create a PDM, the sets of examples first need to be aligned, so they correspond as closely as possible. (This requires a normalisation of scale, rotation and translation). The statistics of the object can then be captured; a mean shape can be calculated and *principal components analysis* (PCA) [55] can be applied to the deviations of each example from the mean, to identify the modes of variation in the shape model. One major advantage of using PCA is that the dimensionality of the model can be reduced significantly, by utilising only the major modes of variation in the model (i.e. those which capture 99% or 95% of the variation).

Learning shape models from *training data* constrains the shape of the target object to be similar to those seen. Example shapes can be generated from the model for use by *active shape models* (ASM) [20] which are a technique for fitting PDMs to image data, iteratively adapting towards the best fit to the image object, whilst remaining within the bounds of the model. The power behind the PDM approach lies in having similar shaped objects, which can be represented well by a set of *landmark* points. Magee [67] employs a particular PDM derivative, a vector based PDM, for cow tracking. The outline of a cow (viewed from the side) contains sufficiently many landmark points for a usable model to be fitted to the images of cows on route to the milking parlour. In later work, Magee builds three separate models for the cows' shape (Figure 2.1), one for each phase of the cows' gait cycle allowing the models to capture the cow's shapes better [68], and uses these within a cyclic hidden Markov model to identify abnormality in the cows' gait (indicating lameness) [69].

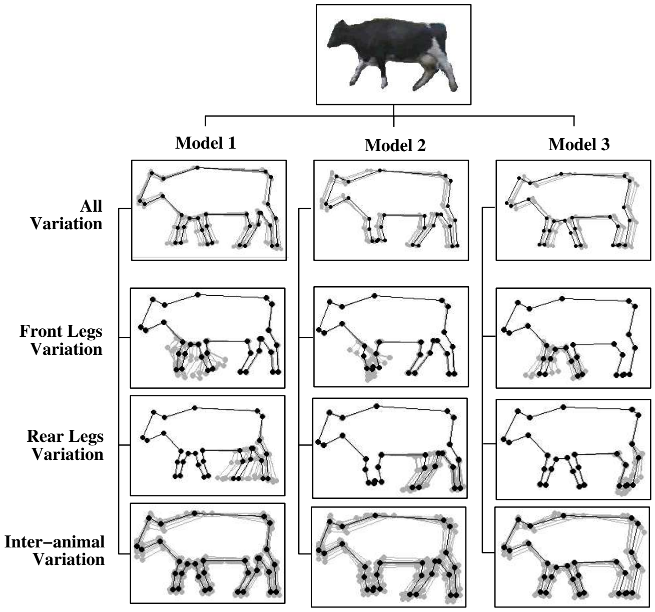


Figure 2.1: The three models used by Magee for cow tracking; cows are always in one of the three configurations shown. Model 1: All legs are separate. Model 2: The front legs are together (or occluding), and their outline is modelled as one. Model 3: The rear legs are together.

There may be scope for such a concept to be employed for modelling the shape of sports players, with a number of models being used to represent when players: stand with their legs closed; stand with their legs open; are running, creating a diagonal shape; or have their arms out. Figure 2.2 shows such examples.

Other variations of the PDM have included extending to polar coordinates [40] to allow non-linear (rotational) variation to be captured by the model, whilst reducing the likelihood of implausible shapes being generated, which can arise when a PDM is created from a set of examples in which non-linear variation occurs, or if the shapes are not similar enough. Further work on non-linear PCA has also included work by Heap [41] who builds a model in which several linear PCA are performed on the data. Jumps (*wormholes in shape space*) from one PCA cluster to another are identified, which are used for hand tracking; with a single linear PCA discontinuities would occur in the shape space of a hand when fingers parted or joined. Recent approaches to achieving non-linear dimensionality reduction have introduced *locally linear embedding* (LLE) [90] and a *global geometric framework* [106] to alleviate the problem of generating implausible shapes from models built using PCA when non-linear variation occurs.

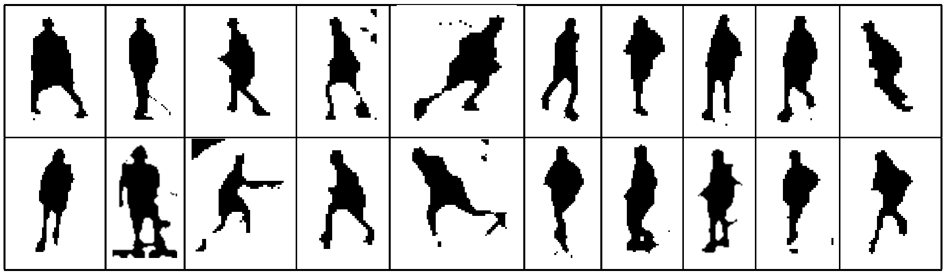


Figure 2.2: The variation in shape of soccer players.

Baumberg [3, 4] tracks pedestrians with an ASM using a flexible shape model comprising of a B-spline with a fixed number of equally spaced control points (similar to the PDM's landmark points) around the object's outline, which sufficiently captures the variation in pedestrian shapes without implausible shapes becoming incorporated in the model. Figure 2.2 illustrates the variation in the outlines of extracted soccer players, which raise questions about the most suitable way of tracking these shapes, since it is hard to identify a suitable set of landmark points for such varying shapes.

Notable alternatives to these 2D shape outline methods are the use of a set of exemplars [15, 107] (or combinations of exemplars) to represent all possible shapes that may appear, the use of a 3D wire frame model as used in a vehicle tracking scheme for visual surveillance of moving vehicles [30], or the use of a full 3D model for tracking human figures [96] which has the drawback of being slow, due to its very high computational cost. PCA has been applied to spatial templates of segmented people walking, followed by *canonical analysis* (CA) to the lower dimensionality (PCA transformed) data; this increases class separation, and reduces within class separation allowing individuals to be identified through their gait [43], since each cluster within this feature space corresponds to a unique person. Rigoll [89] has created pseudo-2D HMMs to capture the shape of a person in an image and used these with Kalman filtering to estimate the position of the bounding box around a person at all future times.

Appearance models based on the texture or pixel intensities within a region of the image have been used as an alternative to the shape modelling approaches detailed. *Eigen-Tracking* was developed by Black [9] for tracking and recognising hand gestures in video, extending the work of Turk and Pentland [110] on *Eigenfaces*. An eigen-model can be created from a training set of fixed size feature vectors of pixel intensities from an image region. The major modes of variation in the model can be found using PCA, allowing the training set to be represented by a linear combination of a reduced set of orthogonal eigen-images. The proximity of an image region to a known individual's face or hand configuration when projected into eigen-space can be used for tracking and recognition. Cootes *et al.* [19] have developed the *active appearance model* which is an integrated model of shape and appearance (generally grey-levels). Once a shape model is identified, the intensity values within defined regions of the shape can be incorporated into a model for PCA. Such a model has been demonstrated to successfully track faces by iteratively comparing an image generated by the model to the image and adjusting the parameters automatically to produce the best fit.

Kang [56] devises a novel appearance model for identifying and tracking soccer players. A circle around a player is constructed and at equal intervals around this circle a set of points are identified. About each point a set of concentric circles within the outer circle are constructed. A colour model, built from observing RGB vales in the sets of radial bins, creates a rotational- and scale-invariant appearance model. This is incorporated into a joint probability model along with image and real-world motion models to track soccer players from two video streams on an outdoor pitch. Liebe [64] compares the properties of appearance and contour based approaches for categorising objects in an image database, and finds that different cues (colour, texture, global and local shape) each perform better for different types of objects.

To summarise, the popular shape models in use are contour models (such as PDMs or spline models which rely on a set of points on the extracted outline of the shape) and appearance models (incorporating image information in the form of colour or texture possibly in addition to shape). For *similar* shapes these cluster well, and PCA can be used to reduce the dimensionality, thus identifying the major features or characteristics of the shapes.

2.3 Tracking methods

Object tracking is tackled in many ways, with approaches dependent upon the object(s) and scene(s) in question. Using *a priori* information about the object(s) and scene(s) in a particular application tends to lead to more fruitful results. This section begins by discussing filtering techniques used in object tracking, then looks at several other tracking concepts, and moves onto multiple object tracking, fusing information from multiple cues and views, finally covering multiple camera approaches to articulated body motion capture, where a full 3D reconstruction is desired.

Filtering techniques

Kalman filtering [18,25,33,85,114] is widely used to estimate a time process. The Kalman filter cycle resembles a predictor-corrector algorithm in that there are two sets of equations, the time update equations for *predicting* the state in a future time, and the measurement update equations for *correcting* the estimate by an actual measurement at that time. A powerful extension to the Kalman filter is the *interacting multiple model* (IMM), which maintains a set of different system models, allowing easy adaptation to different dynamics [14]. Stauffer and Grimson [99] successfully track both cars and people using *adaptive background mixture models*, extracting regions of connected components, and maintaining sets of Kalman filters (with different 'Kalman models') for multiple hypothesis tracking and to explain the movement of objects between frames.

The use of *Monte Carlo* methods and *particle filtering*, which use probabilistic simulation and sampling to heuristically estimate a solution, led to the use of CONDENSATION - conditional density propagation over time [47], which uses stochastic sampling to track curves in cluttered environments. The algorithm stochastically samples from a probability density function (PDF) a set of N possible particles (which are feature vectors, parameterising the target object), applies predictive dynamics to each particle, and evaluates each particle to create a new PDF for the next time step. This technique allows fast tracking of *an object* in cluttered scenes. Instantiating two or more independent CONDENSATION trackers to track more than one object in a scene often leads to the object which 'evaluates' best being tracked by more than one tracker, whilst other objects are not tracked [76]. Multiple object tracking techniques based on the CONDENSATION algorithm have been developed. MacCormick [65] introduces a *probabilistic exclusion principle* and a new *partitioned sampling* algorithm to track multiple targets. Partitioned sampling allows for a reduction in the number of samples needed, particularly in high dimensional spaces (such as those associated with multiple objects). Tao [105], and Koller-Meier [61] also identify frameworks to allow CONDENSATION-like tracking of a varying number of objects in a scene; objects can be counted, created and deleted. The main difference between these methods and those presented in Chapter 3 is that in our method the positional information in the samples is stored as ground plane positions, as opposed to image plane, which gives the advantage that the proximity of objects relates to real world distances, rather than image distances; this aids tracking and occlusion reasoning, given that our domain is large and perspective effects are considerable. In addition our method presented in Chapter 3 integrates the use of a Kalman filter for each player, within the tracking framework.

The efficiency of Monte Carlo methods is exploited by Sullivan *et al.* [101, 102] for *Bayesian localisation* of objects in image sequences. A Bayesian approach to learning the expected responses of a bank of filters, applied to the images of training data, allows a likelihood to be calculated for probabilistic inference as to the object's location, which may be propagated over time.

Tracking methods

Alternative approaches to tracking moving objects include using PDE-based level sets [80] which are computationally intensive, the *closed-worlds* of Intille and Bobick [45] (to be discussed in Section 2.6), and split-and-merge contour models (snakes without a shape model) for real-time tracking on DSP boards [2]. Toyama [107] uses a *metric mixture* (M^2) allowing the use of exemplars in place of a parameterised model in a probabilistic framework. The metric mixture alleviates the problem of comparison between an example test image and each exemplar (to identify the most appropriate). This is analogous to the distance between an active shape model (or deformable contour) fitted to an example test image and a shape model example in shape space (the eigen-space into which the feature vectors are projected for PCA). It does this by learning a probabilistic model from a set of training data. M^2 involves matching example image features exploiting the *chamfer* distance as used by Gavrila [32]. A chamfer distance is the average distance between an image and a template where each pixel that is `on` in the binary template is matched to the nearest pixel that is above a threshold in the image. The temporal sequence of images in the training set is used to learn likelihoods and state transition dynamics for the model.

Multiple cues

Robust tracking may be achieved by fusing together information from several simple cues, as opposed to the computationally expensive shape fitting methods previously described. Triesch [108] presents a face tracking system which integrates different cues (several simple cues are chosen; intensity change, colour analysis, motion continuity, and contrast range), in a self organising manner, by re-calibrating the sensitivity of poorly performing cues (those which differ significantly from the desired result). This *demographic integration* strives for maximal coherence between the different cues, with the assumption that a minority of cues will be affected by environmental changes at any one time. Sherrah [95] presents a *continuous global evidence-based Bayesian modality fusion network* (CBMF), with an example use for multiple face tracking using three cues (modalities); skin colour, frame differencing (motion) and ellipse fitting, where the inexpensive modalities of skin colour and frame differencing are applied on a per pixel basis, and the CBMF performs selective calculation of an expensive modality, like ellipse fitting.

Articulated body motion capture

Articulated body motion capture has received increasing attention in recent years. It has been pursued by those wishing to gain 3D pose information from a single camera (though methods generally are extensible), and those wishing to devise an integrated multiple camera system. Brand [15] analyses sequences of 2D shadows to identify 3D poses by learning a function between 'paths' in the two systems, which are later used to resolve ambiguities over the length of the clip. Deutscher *et al.* [23] introduce *annealed particle filtering* as a technique for tracking in high dimensional configuration spaces (using a model with 29 degrees of freedom). Within a Bayesian framework, Sidenbladh *et al.* [96] track 3D figures using 2D image motion. A prior probability distribution of pose and angle parameters is used, and a first order Markov model updates the temporal dynamics of the change in shape and velocity of the model of the 3D human. Bottino and Laurentini [11] use *volume intersection* (VI) to reconstruct 3D shapes from 2D silhouettes, which again is a non-invasive technique, not requiring the use of markers, or other specialist commercial magnetic tracking systems.

Multiple cameras

The use of multiple cameras for object tracking has become increasingly popular, particularly for surveillance applications. These include reconstructing the paths of people through a set of non-overlapping cameras [58], the *camera handoff* problem of passing the 2D tracking task from one video stream to another, as the target leaves one field of view and enters another [50], using colour information to re-identify a pedestrian if he/she re-appears in the scene, or in the view of an independent neighbouring camera [79], and to aid occlusion reasoning by fusing the information from multiple views of the same scene, using *Bayesian belief networks* [24]. Ellis [28] has learned the topology of an arbitrary network of cameras, through statistical analysis of many observations of pedestrians walking through the scene(s). Stein [63, 100] has devised an excellent method for establishing a *common coordinate frame* between multiple video streams. Planar geometric constraints are applied to moving objects in the scene, rather than using photometric properties, which can vary between images and cameras. This is also used to align video from multiple un-synchronised cameras in time.

Multiple cameras are exploited more significantly for full 3D reconstruction. Bowden [12] reconstructs 3D pose from a single camera view through learning a non-linear point distribution model of a human's upper body. Ong and Gong [78] extend this approach to track a human body using two cameras. A hybrid 2D-3D model (outlines plus skeleton) is learned using hierarchical PCA, and a CONDENSATION tracker fits the model to each view. Gait analysis for medical purposes is the focus of Marzani *et al.* [70], who find that more than three cameras are needed to disambiguate all occlusions. Jennings [51] makes use of stereo range images for 3D finger tracking, as does Harville [39] for people tracking, using the depth information to create a *plan view statistic*. Also of note is Kanade's use of 30 cameras at Super Bowl XXXV allowing reconstruction from any viewpoint using Virtualized Reality[tm] and also of 49 cameras in 'The 3D Room' [91].

This section has reviewed object tracking techniques, generally applied with/after methods reviewed in the previous two sections, on image processing and shape modelling. A wealth of approaches have been pursued, often dependent upon the domain in question. In the next section, *tracking systems* will be reviewed, specifically systems relating to the tracking of people.

2.4 People trackers

Many different people trackers with various purposes have been developed in recent years. This section identifies seven established people tracking systems and considers whether these approaches are useful for a sports player tracking system.

One of the first systems, designed for pedestrian surveillance, was the *Leeds People Tracker* [3, 4] which employs contour tracking, active shape models, and Kalman filtering to track multiple people from a single camera. This relies upon a B-spline model built from examples of similarly shaped pedestrians, however sports players are more animated, and their silhouettes vary more than pedestrians'. The *ADVISOR* system [97] incorporates a people tracker which is an extension of the Leeds People Tracker. Tracking robustness has been increased, through fusion of data from four tracking modules: motion detection, region tracking, head detection and shape tracking.

Some systems allow a determination of the body pose, and real-time tracking of head and hands, such as *Pfinder* [117]. Pfinder is a 'person-finder' which uses a multi-class statistical model of colour and shape to create a blob representation of a tracked person. This will only work when there is a single person in the scene, and produces a more detailed model than is needed to obtain the position of sports players.

W^4 [37] is a real-time system for detecting and tracking people outdoors using a *temporal texture template* which dynamically changes during tracking. W^4 has been extended to work using a controllable pan-tilt-zoom camera [35], and for the tracking of groups of people in the *Hydra* system [36], by using/building appearance models and segmenting groups before, during and after occlusions for each person, to make sure tracking is maintained. Isard and MacCormick [49] have implemented a *Bayesian Multiple-Blob Tracker* (BraMBLe) which uses a Bayesian multiple-object filter allowing comparison between hypotheses containing different numbers of objects, where people are represented as a set of generalised cylinders. Recent work by McKenna *et al.* [71] performs tracking on three levels of abstraction: regions, people and groups. Strong use is made of colour information in this system, to assist in coping with shadows and disambiguating occlusions in pedestrian scenes.

Aiming to track sports players for use in behaviour modelling requires knowledge of the position of each player, since any behaviour analysis where more than one player is represented at a single point may compromise the results (since it cannot happen in the real world). This means that an approach to model groups (more than one person) is not one that should be followed in this application. Many tracking systems in the world are good, and fit for use, for the domain on which they were designed; specifically designed trackers enjoy much greater success than any generic tracker.

2.5 Tracker evaluation

Designing performance evaluation methods for computer vision applications is a challenging task. Even for one of the simplest examples of a computer vision application: *an object classifier*. Evaluation methods for a tomato classifier are now considered.

Figure 2.3: Example tomatoes and apple, for use with a tomato classifier.

Is $P(\texttt{classification=tomato} \mid \texttt{object=tomato})$ a good way to evaluate the classifier? At first glance, it may seem to be fine, given 100 instances of a tomato, if the classifier classifies every tomato as a tomato, then it will be correct 100% of the time, however if given 1 instance of a tomato, and 99 of an apple, then this scheme for evaluating the classification could appear to give correct classification 100% of the time, if it classified *every* object as a tomato, when in fact it was *incorrect* 99% of the time.

Is $P(\texttt{classification correct})$ a good way to evaluate the classifier? This may seem more appropriate, given 100 instances of a tomato, if the classifier classifies every tomato as a tomato, then it will be correct 100% of the time, however if given 1 instance of a tomato, and 99 of an apple, then this scheme for evaluating the classification could appear to give correct classification 99% of the time if the classifer classified every object as an apple, however, since it is a tomato classifier, it performs very poorly, it is correct 0% of the time in the task of classifying tomatoes.

This example shows how *context specific* performance evaluation methods need to be. The accepted way to evaluate a classifier is to use Bayes' Risk, which associates costs with each type of error that can be made: $Risk = Cost(\texttt{error1}) \times P(\texttt{error1}) + Cost(\texttt{error2}) \times P(\texttt{error2}) + \ldots$ How these costs are decided depends largely on the context of the task; in some situations, for example medical, some costs may be high (associated with an error which results in death).

In the context of evaluating a sports player tracker, there are many ways in which the performance of such a computer vision system can be evaluated, although few metrics exist for *positional* tracker evaluation. Often little evaluation on how *precisely* a target is tracked is presented in the literature, with the authors tending to say for what percentage of the time the target was tracked. This issue is now emerging as a key aspect of tracker performance evaluation, and an annual workshop on performance evaluation of tracking and surveillance [74] has recently begun (2000).

Performance evaluation is a wide topic and covers many aspects of computer vision. Ellis discusses approaches to performance evaluation [27], and identifies many aspects which include; how algorithms cope with different physical conditions in the scene, i.e. weather, illumination and irrelevant motion, assessing performance through ground truthing, and the need to compare tracked data to marked up data, whether this be targets' positions, 2D shape models, or classification of some description.

For evaluating the performance of a 'sports player tracker', the aim is to evaluate how well a tracker is able to determine the position of a target object.

2.6 Football oriented work

Sport has a massive influence on our lives and culture, as Bill Shankly said in an interview with the Sunday Times on Oct. 4, 1981: 'Some people think football is a matter of life and death ... I can assure them it is much more serious than that.' The Internet has increased exposure to sport, allowing 24 hour access to information, gambling, videos, games and more [44]. The entertainment industry is also making use of the latest computer vision techniques, notably at Super Bowl XXXV. Kanade [91] used 30 cameras to allow a reconstruction of the action from any virtual viewpoint, including being able to freeze time and view the action from a moving viewpoint.

Football related work has been inspired by several different ambitions, including annotation, action recognition, game reconstruction, and play evaluation. Early work in the soccer domain focused on the idea of creating an automatic commentary on the game [1, 87, 104], and of being able to parse the game to identify highlights of the action [34, 120].

Tracking of sports players has often involved a different approach to pedestrian tracking; Intille and Bobick use *closed-worlds* [45] for video-annotation of American football footage. Tracking using closed-worlds involves giving a context to the scene, by saying that in a region of space and time, all pixels are able to be associated with one object in the scene. In the American football example, each pixel must be associated with: one of a known number of players, the pitch markings, or grass. Using this information, the internal state of the closed-world must be determined from the video footage, to identify the positions of the players. This was followed by work aiming to identify actions from visual evidence, namely American football plays [46]. A simplified set of American football plays were identified, with the aim of identifying which of these took place in a video sequence. In contrast, this thesis focuses on learning the movements and actions of sports players from actual matches, rather than recognising when pre-determined moves have taken place.

SoccerMan [6, 7] is a (soccer) game reconstruction system: various techniques are used in the tracking of players, and then a virtual 3D world with playground texture and textured player shapes is formed. This can be viewed from any virtual viewpoint, and video from two views is necessary. The main drawback of the system is that a large amount of manual intervention is needed to track the players. Before tracking, each players' head is identified manually in every tenth frame, in both video streams, and a 3D spline is fitted to this data, as a basis for identifying the players' positions. Textures for each player are extracted, and attention is given to separating those player textures that would overlap. This ensures individual player texture representations are obtained and allows subsequent reconstruction of the game for visualisation from different viewpoints.

Soccer and other team games take place on a pitch too large to be covered entirely by one camera. Multiple camera approaches have been discussed briefly in Section 2.3, however other approaches have incorporated the use of overhead cameras with fish-eye lenses to cover an indoor basketball scene [81], and the use of *image mosaicing* [6,59,93], where a single non-static camera is used, and features within the images are identified to match the images, or to calibrate the system to allow transformation to a single world coordinate system.

Colour has also played a part in sports tracking [75, 93, 112], due to the near uniform colour of the pitch, and the two teams wearing distinguishably coloured kit. Ok [76] has employed a CONDENSATION tracker with *occlusion alarm probability* which aims to resolve the coalescence of multiple players (into a single player). When two or more players come close to each other, or occlude, the occlusion alarm probability used in calculating the weighting function for the CONDENSATION algorithm repels particles, to avoid multiple sets of particles tracking the same object. Kim [59] has tackled the issue of tracking the ball, making use of the physics which governs its motion, to identify its 3D position.

Interpreting useful information from team games has seen Taki *et al.* [103] look at evaluating teamwork in soccer games, by investigating space advantage on the pitch. A *minimum moving time* (MMT) is discussed which identifies which player is able to get to each point on the pitch first. MMT takes into account each players' position and velocity, and is similar to *Voronoi* regions used in computational geometry. KUBO [62] synthesises a soccer game using a multi-agent system to learn coordinated behaviours.

Sports games have been the subject of attention for computer vision researchers due to their potential applications, the challenging domain, and the structure behind the games. Rather than trying to identify particular events, such as corners, penalties, goals, or set plays, the aim here is to identify the positions of the players to a high enough degree of accuracy for behaviour analysis (to be discussed in Section 2.8).

2.7 Commercial video match analysis systems

There is a number of commercial match analysis systems on the market, each offering a different perspective on how to analyse a game. This section will briefly introduce the functionality and aims of several of these systems, before discussing their applicability to the goal of this thesis.

SoftSport inc. concentrate on passes of the ball. They have two products on offer, which they advertise as:

- 'Second Look 3P's[tm] (Player, Performance, Profile) is a software product that analyzes the full ninety minutes of the player's performance and provides detailed reports of his completed and lost passes. Second Look 3P's[tm] provides a concise, unbiased evaluation of the player.'

- 'Second Look[tm] for Soccer is a software product that analyzes the full ninety minutes of a soccer match and provides many detailed reports on team performance, and all individual player performance. It gives an overall view of the game's tactics and strategy. Second Look[tm] for Soccer provides a concise, unbiased evaluation of the game's players and opponents.'

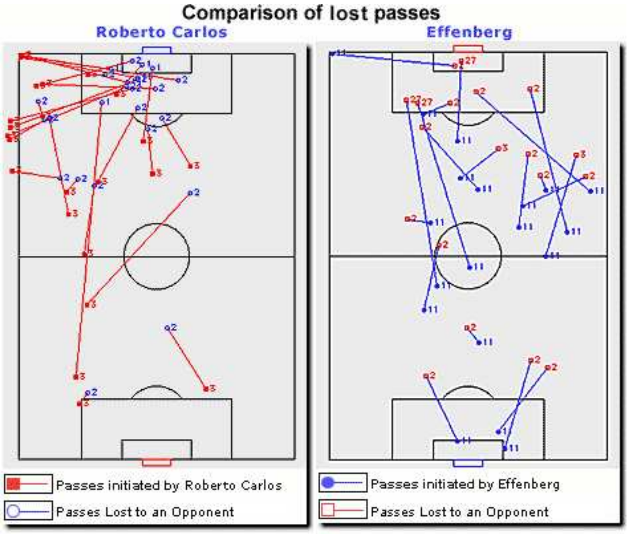

Comparison of lost passes
Roberto Carlos
Effenberg
Passes initiated by Roberto Carlos
Passes Lost to an Opponent
Passes initiated by Effenberg
Passes Lost to an Opponent

Figure 2.4: Example SoftSport graphic showing completed passes comparison of two players from Real Madrid vs Bayern Munich - March, 2000. Graphic reproduced with permission of Zvi Friedman. © SoftSport Inc.

This (system) requires a coach to enter all the data using the software, which can be time consuming. The software allows overhead images (Figure 2.4) of player comparisons to be viewed, and looks at the following statistics:

Goals Scored
Impact Passes/Shots
No. of Lost Passes
Shots On Goal
Total No. of Passes
Completion Rate
Impact Passes/Goals
No. of Completed Passes
No. of Intercepted Passes

TrackSYS offer a software package 'The Observer Video-Pro' (developed by Noldus Information Technology) used for video annotation and presentation. A set of events and players are defined before the game, and the game is annotated as it occurs, or afterwards from video. This data can be analysed to show the basic statistics of the game, as a time-event plot, or a series of video clips can be produced to display the results.

Sports Analytica provide a video match analysis service, which is geared up for use by an individual footballer. The footballer can send off videos of four matches of himself playing, and the Sports Analytica team will process the video on the computer. They categorise the games and use a non-linear video system (Figure 2.5) to allow video clips to be played in any order, to highlight the capabilities of the player.

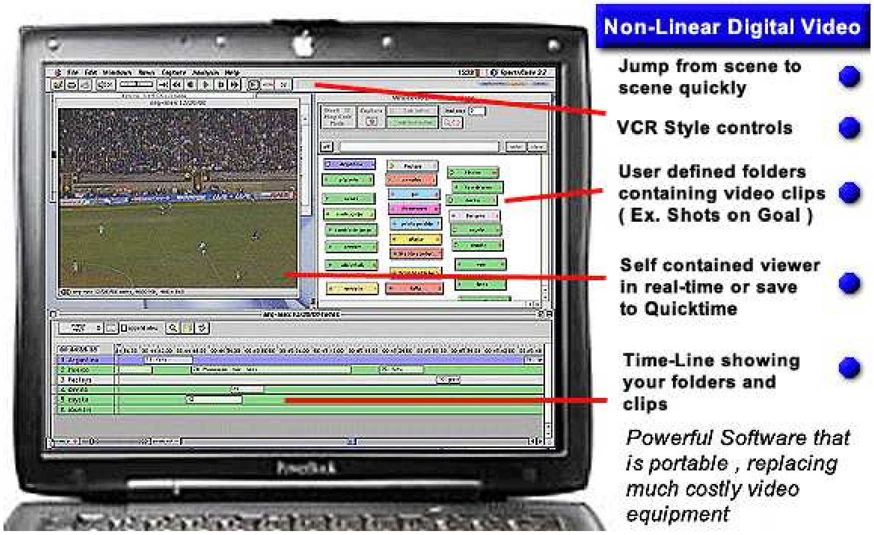

Non-Linear Digital Video
Jump from scene to scene quickly
VCR Style controls
User defined folders containing video clips (Ex. Shots on Goal)
Self contained viewer in real-time or save to Quicktime
Time-Line showing your folders and clips
Powerful Software that is portable , replacing much costly video equipment

Figure 2.5: Example graphic showing the Sports Analytica system in operation. Graphic reproduced with permission. © Sports Analytica.

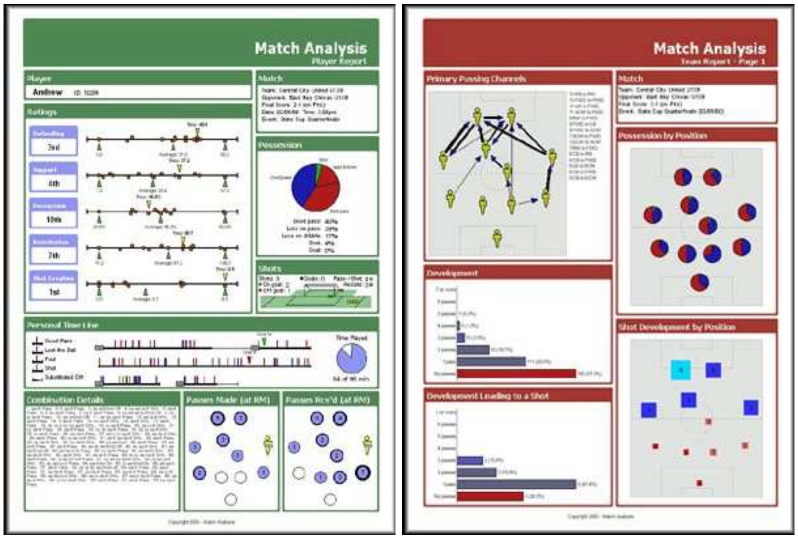


Figure 2.6: Example player and team report cards from Match Analysis. Reproduced by permission. © Match Analysis.

Match Analysis allows a coach to send off a narrated video of the game, and their specialists will methodically enter into the computer many events as they happen. Within a couple of days a glossy set of ‘Player Reports’, ‘Team Reports’ and a ‘Coaches Summary’ are sent out (Figure 2.6) , which highlight the following:

Primary Passing Channels
Variation in Pace over Time
Shots On-goal vs Off-goal
Position of Shooters
Loss on Pass vs Dribble
Possession % by Player
Assists
Passes Made by Player
Time-line of Player Events
Distribution Rating
Support Ratings
and more...

ProZone is a professional company who fit several cameras in the soccer ground, and overnight manually mark-up the position of each player on video footage. This provides information of the ground plane position of each player throughout the game, and is used to present information about the movements of the players throughout the whole game, and can provide valuable statistics to the coach about the time spent moving at different speeds, distance covered as well as a plot of the player's path throughout the game.

Several of these packages may be useful for post match video analysis of events, for example, for the team to see each corner they took and how successful different corner taking tactics were - no longer would a video recorder need to be fast forwarded through to the next corner - the digital video clips classified as corners could just be played. It should be noted that each of these products are not automatic in the computer vision sense; they are software products/services requiring much operator time and interaction. ProZone show the closest resemblance to the positional analysis in which we are interested, however it currently takes over 34 hours to manually mark-up the positions of the 22 players and the ball. Automating this process is our interest.

Positional analysis of sports players is of interest to coaches and trainers, since it allows a quantifiable way of analysing a players' performance, aids the design of better training regimes (also allowing quantifiable evaluation of such schemes on performance), and provides information for tactical analysis. It is important to obtain the positions of each player throughout the game to the highest possible level of accuracy and to a quantifiable level of accuracy. It is necessary to know how good the data obtained is. This data can be used to calculate the velocities and accelerations of each player, allowing the profile of a player's speed throughout the game to be identified. Coaches can use this to see how many short sprints the player runs in the game, how much time he spends stood still, or how quickly he can change pace. However, the main research interest here is in the behaviour modelling of the interactions between the players, once the positional data is obtained.

2.8 Behaviour modelling

Analysing the movements of the players over time allows a behaviour model of their movements and interactions to be learned. Such a model has potential uses for analysing tactics, identifying good or atypical players, and most powerfully to be incorporated into a multiple object tracking system to govern the expected dynamics of the players, which should vastly increase tracker performance. Analysis of the players' movements could answer the following questions:

- Can emergent behaviour be learned from observing real sports games?
- Can team membership be identified for each player?
- How do players interact with their team mates, and the opposition?
- Can the tactics be encapsulated in an understandable way?
- Can a generative model be created?
- Can the rules of the game be learned?

Behaviour models that can be built in order to assist in answering these questions include mixture models, Markov models, hidden Markov models (HMMs), and agent based approaches. These methods will now be discussed.

Gaussian mixture models (GMMs) [8] can be used to estimate probability density. A set of Gaussians can be used to represent the data, typically a set of *feature vectors*. The parameters of the Gaussians must be estimated, for which the *expectation-maximisation* (EM) algorithm is commonly used. Now any vector in this feature vector space can be represented as a mixture of Gaussians. The centres of the Gaussians may represent different clusters of data, although separation between cluster centres may not exist. Methods for achieving a form of clustering in which feature vectors are closely linked to a single cluster centre or *prototype vector* to classify it, are often useful in state based models.

Vector quantisation (VQ) can be applied to cluster a set of feature vectors into a set of distinct prototypical vectors. Johnson [52, 53] implements a VQ scheme for learning cluster prototypes whose point density approximates the probability density of the training data. This has been demonstrated on pedestrian trajectory data and produces a well partitioned set of prototypes, typifying the sample data.

Given a stochastic system, which may be in any one of a number of states, it is possible to represent the system as a graphical structure in which the nodes correspond to the states and the (directed) edges to the probability of moving from one state to another, or of remaining in the same state. One the simplest forms of this is a Markov chain, where the model may only stay in the same state, or move into the next state. Other models roughly in ascending order of complexity are; Markov models, 2nd order Markov models, nth order Markov models, variable length Markov models, hidden Markov models, Bayesian networks, and probabilistic graphical models, which encompass all of the above.

A *Markov model* can be represented as a graphical structure in which a network of states is formed, connected by edges corresponding to transition probabilities of moving from one state to another. In a first order Markov model, the state transitions are dependent only on the current state of the system. A transition matrix of the probabilities of moving from one state to another may be estimated by the frequencies of the event being observed in training data. Second order Markov models depend not only on the current state of the system, but also on the previous state. Graphically, this can be represented as each node being made up of the set of compounded states at the previous and current time step.

Markov models of the nth order are formed by creating meta-states which represent the system at the previous n time steps. Probabilities of moving from one meta-state to another may be estimated from the frequencies of the event being observed in training data, however, since incorporating this temporal 'memory' aspect, the number of meta-states grows rapidly and vast quantities of data are necessary to train the model in order for it to be representative of the system. Figure 2.7 illustrates directed graphs of a system with only two states A and B. The number of meta-states equals the number of states raised to the power n.

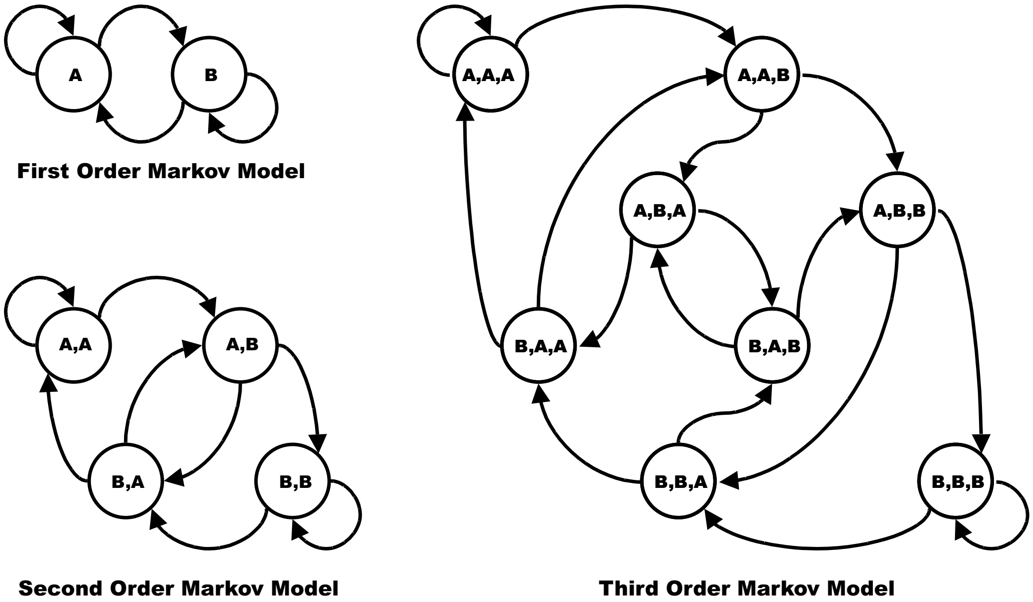


Figure 2.7: Markov models of the first, second and third order are shown on a system with only two states A and B. As the order of the Markov model increases the number of meta-states and possible transitions grows rapidly.

Two methods have been developed to overcome this problem. *Variable length Markov models* (VLMMs) allow for a variable length memory to be encoded into the model [31]. *Hidden Markov models* (HMMs) are first order Markov models which have been extended by introducing a set of hidden states, which are probabilistically linked to the observable states. The topology of HMMs can encode higher order dependencies, and are commonly employed in speech recognition applications [83].

Probabilistic graphical models can be used to represent sequentially changing systems. These can be represented by Bayesian networks (BN) [16,42,73] and provide a framework in which many commonly used statistical methods can be formulated, such as mixture models, HMMs, and Kalman filters. Bayesian networks are sometimes called Bayesian belief networks (BBN) since the Bayesian concept is to allow an existing belief to be altered in light of new evidence. Bayes' rule $[P(E|A) = P(A|E) \times P(E)/P(A)]$ is used for inference in Bayesian networks, which allows the prior estimates to be continually revised given new observations. In a generic sense, Bayesian networks can be used in situations for relating a set of inputs to a set of outputs.

Emergent global behaviour can be observed in a system governed by local interaction rules. Reynolds [88] demonstrates an agent-based approach to modelling the dynamics of a flock of birds for computer animation. The flock is represented as a set of *boids* (agents) each of which has the same simple rule set: to avoid collision, to match velocity, and to stay close to nearby boids. These simple interactions between boids produces a realistic simulated flocking behaviour.

Agent-based approaches have also been employed in the surveillance domain. Remagnino [86] uses a Bayesian network to model the behaviour of pedestrians and vehicles in a scene. A behaviour agent is used to provide a textual description of dynamic scene activity. Interactions at the object and inter-object (close proximity) level are demonstrated in a carpark surveillance application. Pedflow [115, 116] aims to model the movements of pedestrians on a street, and learn how they interact when a change in the environment layout occurs, leading to the use of microsimulation models to assess the impact of layout changes before they are implemented in the real world. Fernyhough [29] builds *qualitative event models* using qualitative spatial reasoning to identify interactions between two moving vehicles, allowing the events that occur to be described in a natural language description.

Recognition of activities has also been an area of significant interest for computer vision researchers. It is not the subject of this thesis, as the aim is to model the behaviour of the sports players, rather than to recognise any pre-defined actions they may perform, however it is of such a similar nature that several projects are worthy of discussion.

VIGOUR [94] is a computer vision system for tracking and recognising the actions/gestures of people for visually mediated interactions; recognising simple movements like waving and pointing to transfer the focus of the system (and camera) from one person to another. Skin colour and motion are used as a basis for tracking, and a vector representing a trajectory of features in successive frames is analysed and matched to a set of novel gestures learned from training sequences.

Bowden [13] applies a HMM-like approach for modelling the transitions between states in the application of tracking finger spelt American sign language (ASL). A discrete PDF for transitions between states (letters) in the English language can be created and incorporated in a Markov model for transitions between clusters in a shape model feature space, creating a hidden Markov model. This is used within a CONDENSATION tracking scheme, and improves the tracker's performance and reduces the number of samples needed to propagate the various hypotheses as to the shape of the hand for tracking at the next time instant.

Visual surveillance of pedestrians is of particular interest for computer vision researchers. Applications have ranged from identifying atypical behaviour in car park scenes [72] to using HMMs and coupled HMMs to recognise pedestrian's interactions [77]. Recognition of human movement through the use of *motion history images*, which encode a temporal aspect in addition to spatial motion, produces vivid temporal templates of current and past movements [10]. Matching these templates allows movements such as sitting or standing to be identified, and has been demonstrated to discriminate between 18 aerobic exercises.

2.9 Summary

This chapter has reviewed work in the computer vision and behaviour modelling domain, and discussed its relevance to the task in question: tracking and modelling team game interactions.

The ultimate goal would be to track robustly the players of sports games and to learn their behaviour. Having created a behaviour model, this could be used to improve the performance of the multiple object tracking system, by governing the expected dynamics of the players. A vicious circle exists: a robust tracker is needed to collect data in order to create a behaviour model *and* a behaviour model is needed in order to create a robust multi-object tracker.

Chapter 3

Tracking multiple objects

This chapter presents a framework for multi-object tracking, using a CONDENSATION based approach. Each player being tracked is independently fitted to a model, and the sampling probability for the group of samples is calculated as a function of the fitness score of each player. This function rewards consistently good scores, but punishes a group of some very good and some very bad fitness scores. Ground plane information is used throughout, and the predictive stage of the algorithm is improved to incorporate estimates of position from Kalman filters. This helps group the estimated positions of each player, and to aid in tracking through occlusions.

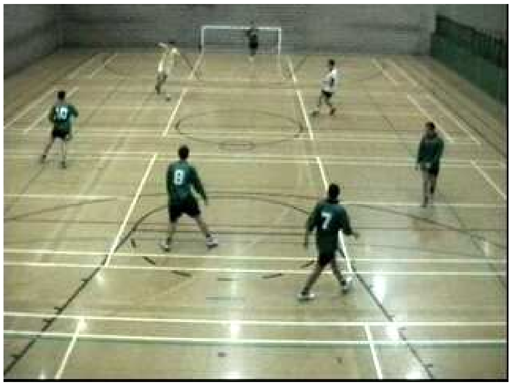


Figure 3.1: Example footage of soccer players.

The aim of this work is to produce a tracker which will automatically track sports players, and identify their real world positions for positional behaviour analysis, as opposed to recognising whether they are running, kicking the ball, or involved in a set play. The precision requirements for behaviour modelling would ideally aim for as detailed positional information as possible. Considering the size of the domain (typically 18 × 32 metres) and the space that a player can occupy it is sensible to aim to be accurate to within 1m of the player, and data less accurate than this can be considered as of little use for behaviour modelling.

3.1 Theoretical aspects

3.1.1 Image perspective

Tracking multiple objects through busy cluttered scenes remains a challenging problem. Figure 3.1 shows a typical scene from an indoor 5-a-side soccer game. All the action is constrained to the pitch, however the perspective of the image highlights several issues. The size of the pitch (18 × 32 metres) means that the resolution of an image of the game varies greatly between the nearest and the furthest parts of the pitch. Analysis reveals that in a typical image (e.g., Figure 3.1 which is 320 × 240 pixels in size), if two vertically adjacent pixels on the image plane are projected onto the ground plane, then pixels in the nearest part of the image are 3cm apart, whereas those in the far goal mouth are over 45cm apart. In the area of the image representing the nearest part of the pitch, 3 metres of ground plane covers 72 pixels, compared with only 8 pixels at the far end of the pitch. Figure 3.2 illustrates this perspective effect over the pitch, which is covered in equal sized two metre squares.

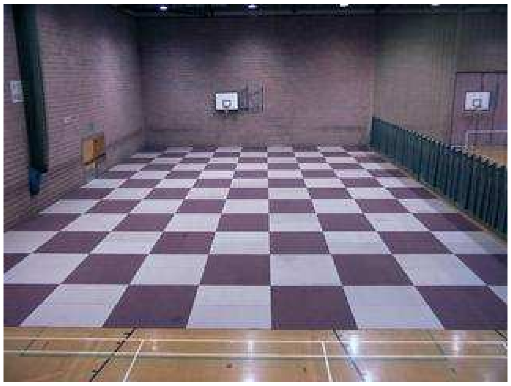


Figure 3.2: Perspective effects on a chequerboard pitch.

This emphasises the importance of considering the depth information in the image. It becomes important for the tracking to be performed using the ground plane coordinates, which take into account the amount of ground that a player is physically able to cover over time. The corresponding distance in image pixels varies greatly over the image. Knowledge of the players' position on the ground plane aids with resolving occlusions, particularly in scenes from such a perspective view, since often one player occludes part of another player when they are more than a metre away from each other, as shown in Figure 3.3.

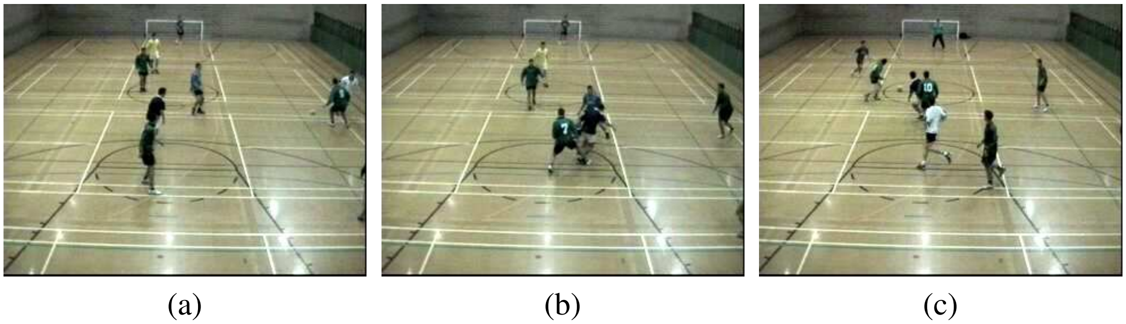

(a) (b) (c)

Figure 3.3: Examples of soccer players occluding each other. (a) Three pairs of players causing occlusion. (b) The group of three players in the centre looks very much like a group of only two. (c) The player in white occludes the bottom half of player 10, who in turn hides almost all of another player, except for his head and an arm.

The main feature of interest of a player is the position of the feet, which is why this is used instead of the players' centroid, and it is this position which we wish to determine with the greatest accuracy for use when modelling the players' behaviour in future work. It is assumed that the players' feet are in contact with the floor when calculating their world position from the image. Later in this thesis, issues with how well a computer vision system, or indeed a human, is able to locate the position of a player are discussed. Players' shapes vary greatly (to be discussed in Section 3.1.4) and every method for identifying the 'true' position of a player must have some tolerance in its estimate.

3.1.2 Image transformation

In this chapter a straightforward image plane to ground plane transformation (2D to 2D) is used to determine where the players are on the ground plane from where they appear on the image. Four pairs of corresponding points on the image and known real world ground plane points are used to construct the 3×3 transformation matrices. In later chapters, the more sophisticated approach of camera calibration is used to calibrate the image, which allows transformations between 2D image coordinates and 3D world coordinates (Section

6.2).

To be able to transform from one set of coordinates to the other, an image plane to ground plane (IPTOGP) transformation and a ground plane to image plane (GPTOIP) need to be calculated. For this, a projective transformation is used. Details of the fundamentals of projective geometry, what a projective transformation is, and proof of the uniqueness of the projective transform between two quadrangles in $\mathbb{P}^2$ are presented in Appendix B. The method presented is adapted from [17] and in depth coverage of geometry for computer vision can be found in [38].

The image plane is calibrated by identifying the image coordinates of four known real world points, which lie on the ground plane. This provides two quadrangles (a quadrangle is a set of 4 distinct points in $\mathbb{P}^2$, no 3 of which are collinear). Thus for each ground plane point (x_i, y_i) there is a corresponding image plane coordinate (u_i, v_i) for $i = 1, \ldots, 4$. Each of these coordinates can be written as a projective point P_i, $(x_i, y_i)^T \leftrightarrow [x_i, y_i, 1]$. Row reduction is performed on (P_1, P_2, P_3, P_4) (3.1), demonstrating that P_1, P_2, P_3 form a basis (3.2):

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.1}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ 0 & y'_2 & y'_3 & y'_4 \\ 0 & 0 & z'_3 & z'_4 \end{bmatrix} \tag{3.2}$$

From this row-reduced form it is trivial to find α, β, γ for which $P_4 = \alpha P_1 + \beta P_2 + \gamma P_3$. The projective transformation matrix for a point $P \in \mathbb{P}^2$ to the ground plane is:

$$\mathcal{M}_{gp} = \begin{bmatrix} \alpha x_1 & \beta x_2 & \gamma x_3 \\ \alpha y_1 & \beta y_2 & \gamma y_3 \\ \alpha & \beta & \gamma \end{bmatrix} \tag{3.3}$$

The projective transformation matrix $\mathcal{M}_{ip}$ for a point $P \in \mathbb{P}^2$ to the image plane can be found in the same way. Now the IPTOGP matrix can be calculated as:

$$\mathcal{M}_{iptogp} = \mathcal{M}_{gp}(\mathcal{M}_{ip})^{-1} \tag{3.4}$$

and the GPTOIP matrix is the inverse of this

$$\mathcal{M}_{gptoip} = (\mathcal{M}_{iptogp})^{-1} \tag{3.5}$$

Thus, we have a projective transformation $\phi : \mathbb{P}^2 \to \mathbb{P}^2$. $\phi[P] = [\mathcal{M}_{iptogp}P]$ from the image to the world coordinate system (and ϕ^{-1} from world to image). Since the 2D image coordinates lie on a plane, as do the 2D world coordinates, to transform an image point (u, v) it must be represented in $\mathbb{P}^2$ as $[u, v, 1]$, and projected to:

$$\phi[u, v, 1] = \mathcal{M}_{iptogp}[u, v, 1] = [x', y', z'] \tag{3.6}$$

Now $[x', y', z'] \in \mathbb{P}^2$ represents a line in $\mathbb{R}^3$. The intersection of this line (thought of as a ray) with the ground plane must now be calculated. The world point $(x, y)^T$ in $\mathbb{P}^2$ is $[x, y, 1]$, thus $x = \frac{x'}{z'}$ and $y = \frac{y'}{z'}$. The same procedure must be applied when transforming world points to image points.

3.1.3 Image segmentation

There are many possible approaches to image segmentation, as discussed in Section 2.1, which work to varying degrees of success, often related to their computational cost. Many of these were tried on video streams from a soccer game, and suffered from a range of problems, notably:

- those with a temporal aspect [3] struggle to cope with both fast moving players and those who remain stationary for a period of time. Both scenarios are common in the sports domain.
- those with a per-pixel model [71] produced a segmented image, but were very sensitive to camera jitter.

Thus a novel efficient scheme for producing a well segmented foreground image in this situation is developed. This section details the methods employed.

A foreground and a background PCA model are created. Each pixel in an image is then compared to these models and assigned a probability of being foreground. The foreground (and background) model in HSI space is built offline from a sample of pixels from regions of the images identified as being foreground (background respectively) before tracking begins. In each model a cluster is formed. The single cluster for each model is used to represent the entire image as opposed having a model for each individual pixel. This allows an example pixel to be compared to each cluster centre. The Mahalanobis distance of an example from a centre can be calculated. (The Mahalanobis distance is the Euclidean distance when transformed to an eigen-space.) HSI space is used because the separation between the foreground and background clusters is greater in this space than in other possible spaces such as RGB, or chromicity values.

For each pixel in an image, a probability of being foreground can be assigned. This is given by the likelihood that the pixel belongs to the foreground or background and is calculated:

$$p(fore) = d_b/(d_f + d_b) \tag{3.7}$$

where d_f and d_b are the Mahalanobis distance of the pixel from the respective cluster centres. This creates a noisy image, with player regions being fragmented, particularly the players' legs. Segmentation is improved by performing probabilistic relaxation on the image using

$$\begin{aligned} p(fore) &= p(fore) + \delta \quad \text{if median value of neighbouring pixels} > 0.5 \qquad (3.8) \\ p(fore) &= p(fore) - \delta \quad \text{otherwise} \qquad (3.9) \end{aligned}$$

and choosing δ such that a pixel is able to change from foreground to background (or vice-versa) after a suitable number of applications of the relaxation. Using 3 applications of the probabilistic relaxation with δ = 0.2 is judged to produce a well segmented foreground. The players' shapes in Figure 3.4 are extracted using this scheme. There is still a quantity of noise in the foreground image, which will be investigated later in the thesis, when the segmentation becomes more important to the shape fitting functions utilised.

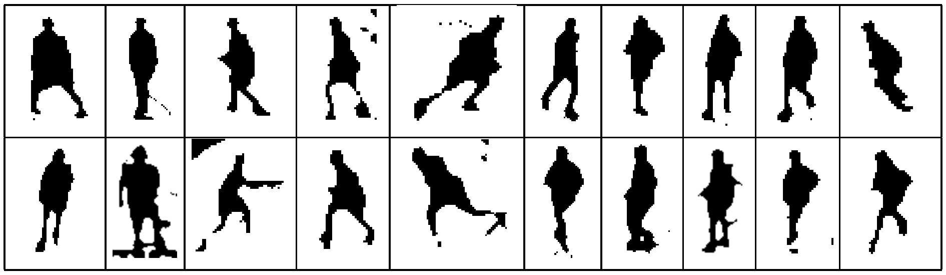


Figure 3.4: The variation in shape of soccer players.

3.1.4 Shape models

During the course of a game, sports players take on a wide variety of shapes. Players perform a wide range of activities. They often stand still, run towards the camera, across the pitch, wave their arms in the air, and kick the ball, to name just a few. Their silhouettes therefore also take on a wide variety of shapes.

In this chapter, a very simplistic approach is taken to shape modelling. A bounding box is fitted to each silhouette. How well the bounding box fits the image data is evaluated by finding the proportion of pixels in the box which have been classified as foreground during image segmentation. A more complex model could be used if information about the players' pose and orientation was the aim. The purpose of tracking the sports players is for analysis of their movements and positional behaviours, thus the most important feature is their feet.

In Chapter 4 complex shape models are discussed. Contour models are investigated on images of soccer players, and a novel shape descriptor is devised, which is employed in Chapter 6.

3.2 Multiple object tracking

The aim is for a tracker of multiple objects, as opposed to multiple trackers of objects. Multiple CONDENSATION trackers have the flaw that they each tend to coalesce towards the best fitting target, rather than each target in the scene.

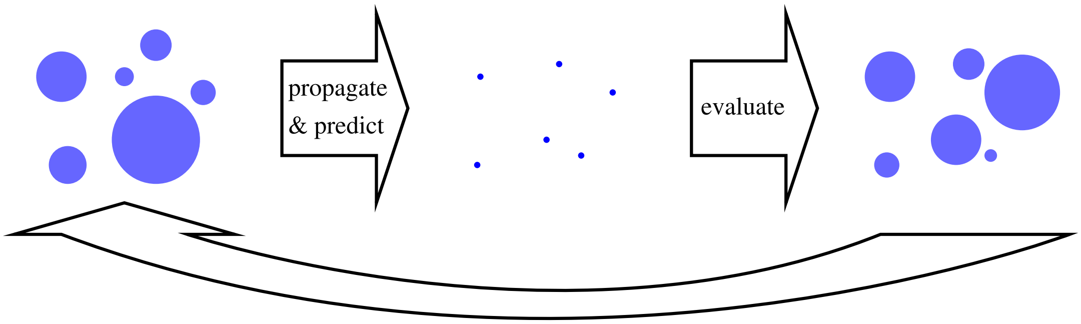


Figure 3.5: A single object CONDENSATION tracker.

A single CONDENSATION tracker can be explained well by considering Figure 3.5. If we have many estimates (samples) of where an object may be, and at each position evaluate using some method (to be discussed later) how well the object fits a model, a PDF can be created, which is shown graphically on the left-hand side of Figure 3.5 (where larger circles indicate a greater likelihood of the target being at the central position). Next a number of samples is taken from this distribution, and each has some predictive dynamics applied to it. This results in a new set of positions for the samples, which are evaluated again, to form a new PDF for use at the next time step.

The framework for multiple object tracking being introduced firstly shows the *structure* of the samples, and how they make up samplesets. Then the *propagation* of the samplesets, the application of the *predictive dynamics*, and the *evaluation* of the fitness of the samples to the image information are discussed.

3.2.1 Structure

Multi-object tracking adds an extra level to the structure of the algorithms. Here, a *sample* represents an instance of a player, a *sampleset* represents a collection of samples (players being tracked) at an instance, and a *supersampleset* represents a collection of samplesets.

It is the contact point of the players' feet with the floor that we wish to identify with the greatest accuracy. Image coordinates (u, v) can be used to represent the position of the players' contact point with the floor; calibrating the image plane allows image points to be projected to ground plane (world) coordinates. In this work, ground plane coordinates (x, y) are used throughout the computations, with image positions calculated from these.

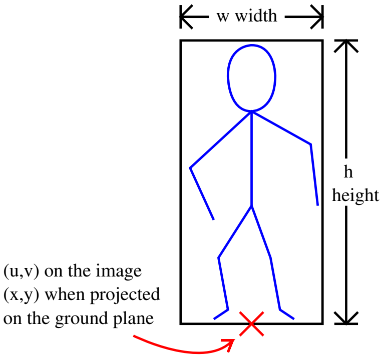


Figure 3.6: Sample representation of target player.

To calculate a bounding box for a player, first the single world point is projected onto the image plane to a point (u, v); next a bounding box of width w and height h is constructed, assuming that this point is the midpoint of the base of the bounding box. On creation, an identification number, id, is included for use in determining the player to which trajectories belong. Thus each player can be represented as

$$\mathbf{x} = (x, y, h, w, id) \tag{3.10}$$

Let $\mathbf{x}_t{}^i$ be an instance of a sample at time t. A sampleset $\mathbf{s}_t{}^j$ can be formed, which consists of an instance of each different object being tracked, along with a corresponding sampling probability ${\pi_t}^j$.

$$\mathbf{s}_t{}^j = (\mathbf{x}_t{}^1, \mathbf{x}_t{}^2, \ldots, \mathbf{x}_t{}^{n_j}, \pi_t{}^j) \tag{3.11}$$

where n_j is the number of objects in the sampleset $\mathbf{s}_t{}^j$.

A 'supersampleset' $\mathbf{S}_t = (\mathbf{s}_t{}^1, \mathbf{s}_t{}^2, \ldots, \mathbf{s}_t{}^N)$ is created to store each of these samplesets, where N is the predefined number of samples used in the CONDENSATION algorithm.

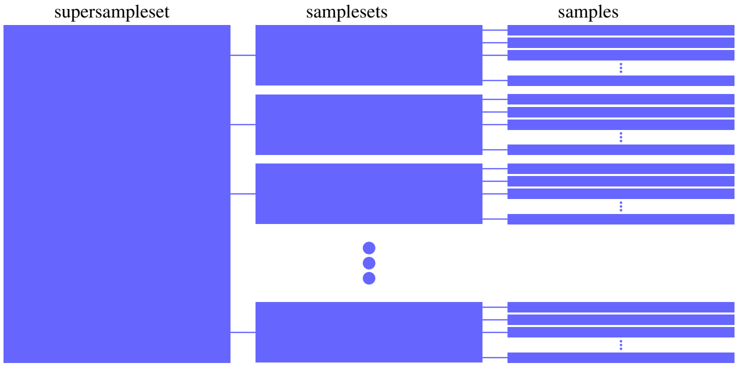


Figure 3.7: The hierarchical structure of the supersamplesets, samplesets and samples.

3.2.2 Propagation

The samplesets within the supersampleset are propagated in the usual way (as in Figure 3.5), that is N samplesets, $\mathbf{s}'_t$, are generated from $p(\mathbf{s}_t|\zeta_t)$ at each time step, where ζ_t is foreground image probability data, i.e., $\mathbf{s}'_t$ is drawn randomly from $p(\mathbf{s}_t|\zeta_t)$. Then $\mathbf{s}'_{t+1}$ is drawn randomly from $p(\mathbf{s}_{t+1}|\mathbf{s}_t = \mathbf{s}'_t)$ and π_{t+1} is calculated as $p(\zeta_{t+1}|\mathbf{s}_{t+1} = \mathbf{s}'_{t+1})$.

The probabilities are assigned by rescaling the weights assigned to each sampleset from a fitness function which assesses how well each bounding box fits its target (see Section 3.2.4). The sampleset with the highest sampling probability is used as the 'best' sampleset for representing the players.

3.2.3 Predictive dynamics

A simple model to use to predict each sampleset $\mathbf{s}_t{}^j \in \mathbf{S}_t$ from $\mathbf{s}_{t-1}{}^j \in \mathbf{S}_{t-1}$ is:

$$\begin{aligned} x_t{}^i &= x_{t-1}{}^i + \varepsilon_x \\ y_t{}^i &= y_{t-1}{}^i + \varepsilon_y \\ h_t{}^i &= h_{t-1}{}^i + \varepsilon_h \\ w_t{}^i &= w_{t-1}{}^i + \varepsilon_w \end{aligned} \tag{3.12}$$

for $i = \{1, \ldots, n_j\}$, and where ε_x and $\varepsilon_y \sim \mathcal{N}(0, \sigma_1)$ with σ_1 typically in the region of 100mm, given that the maximum distance on the ground plane that a sports player will move will be in the order of $3\sigma_1$ (300mm per 25^{th} of a second). This allows for the tracking of a player who moves at a speed of $7.5\ ms^{-1}$. Velocity information of the players could be incorporated at this stage, however the nature of the sport often involves the players making sharp, sudden changes of direction, which may mislead a tracker dependent upon velocity information.

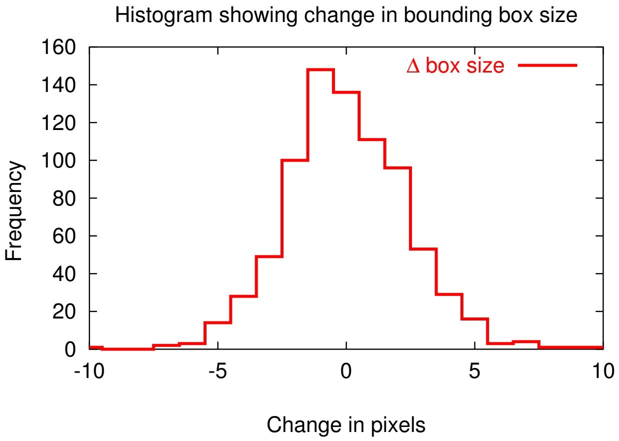


Figure 3.8: Histogram of changes in bounding box size of players from one frame to the next. Created from a hand marked-up set of bounding boxes of 6 players over 400 frames.

Due to the rapid change in shape of a player that can occur when they raise arms to attract attention, or open stride when running, the height and width of the bounding box must be allowed to react quickly to this change, thus adding Gaussian noise to the height and width with ε_h and $\varepsilon_w \sim \mathcal{N}(0, \sigma_2)$ with $\sigma_2 = 2$ pixels allows such a change. Empirical evidence of the changes in bounding box size, through manual mark-up of bounding boxes on example footage, confirm that this is a reasonable variation to allow. On a set of 400 marked-up bounding boxes around players the mean variation from frame to frame was found to be 0.0 with a standard deviation of 2.4. The distribution of the changes in bounding box size is illustrated in Figure 3.8, and approximates a bell-shaped curve, which characterises Normally distributed data. Doing this has the drawback that the samples in a sampleset may no longer each correspond to different players, for example, when one sample locks to another target in close proximity, which is already being tracked.

3.2.4 Evaluation

In this chapter, a very simple fitness function is used to evaluate how well the samples fit the image data. Models used in later work are described in Section 4.2. Here the fitness score for a player (sample) is evaluated as the average value of the probability foreground image within the bounding box.

If the fitness score for each player sample within a particular sampleset is similar, then the overall score for the sampleset is increased (rewarded). If one or more of the samples is a poor fit, then the overall score is reduced (punished). This aims to aid the propagation of the samplesets with the best overall fit of the n_j objects, but not those for which one or more objects fit very well, when there are some that don't fit well at all.

These weightings of the samplesets are then normalised to create a probability density function, which is then sampled for propagation of the 'better' samplesets to the next time step.

3.3 Improving with Kalman filtering

Altering the predictive step of the CONDENSATION algorithm can prevent the samples from straying too far from the other samples representing the same target. The position of a player on the ground plane can be predicted for the next time step, given previous states. Here, n_j Kalman filters are used, one for each player. They are updated using the observed value of the position of each player (sample) in the 'best' sampleset.

Kalman filters are being used because they address the problem of estimating the position $\mathbf{x_t} = (x, y) \in \mathbb{R}^2$ of the player at the next discrete time step. A simple linear stochastic difference equation governs this process

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{w}_{t-1} \tag{3.13}$$

with a measurement $\mathbf{z} \in \mathbb{R}^2$ which directly relates to $\mathbf{x}$ that is

$$\mathbf{z}_t = \mathbf{x}_k + \mathbf{v}_{t-1} \tag{3.14}$$

The independent random variables $\mathbf{w}_t$ and $\mathbf{v}_t$ represent the process and measurement noise, and have Normal probability distributions

$$p(\mathbf{w}) \sim \mathcal{N}(0, \mathcal{Q}) \tag{3.15}$$

$$p(\mathbf{v}) \sim \mathcal{N}(0, \mathcal{R}) \tag{3.16}$$

Currently constant $\mathcal{Q}$ (process noise covariance) and $\mathcal{R}$ (measurement noise covariance) are used. However, in later work these are used to assess the certainty of the estimates being made, which will improve the 'trust' in the estimate of the player's position from the Kalman filter, compared to the observation $\mathbf{z}$ from the image, when resolving occlusions. The process noise covariance, $\mathcal{Q}$, reflects the uncertainty of the location of the player's position within a bounding box, and is set to 300mm in the x and y direction, since the position of the mid-point of the base of the bounding box is within 300mm of the true location of the player. Due to the stochastic nature of the CONDENSATION algorithm employed, and the fact that the Kalman filter is updated using only the player's position from the 'best' sampleset (which leads to the update observations being 'jumpy'), the measurement noise covariance matrix, $\mathcal{R}$, is chosen to be an order of magnitude greater than $\mathcal{Q}$. (The measurement noise covariance is discussed in more detail in Section 6.4). Thus, the covariance matrices used are:

$$\mathcal{Q} = \begin{bmatrix} 300 & \sqrt{300} \\ \sqrt{300} & 300 \end{bmatrix} \quad \mathcal{R} = \begin{bmatrix} (300)^2 & 300 \\ 300 & (300)^2 \end{bmatrix} \tag{3.17}$$

At each time step, a Kalman estimate $\hat{\mathbf{x}}_\mathbf{t} = (\hat{x}_t, \hat{y}_t)$ of the position of each player is calculated, and each sampleset $\mathbf{s}^j{}_t \in \mathbf{S}_t$ is predicted from $\mathbf{s}^j{}_{t-1} \in \mathbf{S}_{t-1}$ using

$$\begin{aligned} {x_t}^i &= (\hat{x}_t + {x_{t-1}}^i)/2 + \varepsilon_x \\ {y_t}^i &= (\hat{y}_t + {y_{t-1}}^i)/2 + \varepsilon_y \\ {h_t}^i &= {h_{t-1}}^i + \varepsilon_h \\ {w_t}^i &= {w_{t-1}}^i + \varepsilon_w \end{aligned} \tag{3.18}$$

for $i = \{1, \ldots, n_j\}$, and the observed player positions $\mathbf{z}_t$ from the 'best' sampleset are used to update each discrete Kalman filter. This has the effect of grouping the samples corresponding to each player within the CONDENSATION algorithm, since each sample is drawn towards the predicted $\hat{\mathbf{x}}_\mathbf{t}$ for that player. This prevents the samples for a player splitting up into two or more groups, which might have allowed the 'best' sample for a player to jump between the groups, or lock onto a different player instead.

The use of velocity in the CONDENSATION algorithm was not explored as the main issue with the initial approach appeared to be the spreading of samples over too large an area. Observation of tracked footage revealed that the bounding boxes jumped often, as the best hypothesis for a player's location moved from one sample to another. A method of keeping the samples in a closer group, whilst still supporting multiple hypotheses was desired. Incorporating the Kalman filter, largely as smoothing filter, provided the desired result. The predictive estimate of the player's dynamic Kalman filter (with constant velocity assumption) was used, as samples are evaluated at that time step.

3.4 Evaluation and results

Considering the implications of Section 3.1.1 and 3.1.2 there is probably a limited area of the ground plane for which the comparison of differences in ground plane positions is valid, since on parts of the image neighbouring pixels are almost half a metre apart. Also using the assumption that the players' position is at the midpoint of the base of the bounding box may not be valid when considering asymmetric shapes, for example when a player leans to one side. However, here it is assumed these are usable enough to be valid.

In order to evaluate the tracking, the true ground plane positions of the players need to be determined. A sequence was independently hand marked-up 4 times, and analysis of each resulting trajectory with each of the other trajectories has been performed. Figure 3.9(a) shows the trajectories of a single footballer over 835 frames. Figure 3.9(b) shows the distribution of the Euclidean differences between each trajectory, which were calculated as the distance between two players' positions at each time step. Analysis of the six pairwise permutations of the four hand-tracked trajectories shows a mean difference of 312.2 mm between positions, a standard deviation of 239.7 mm, and a mode between 200 and 300 mm. Thus, it is reasonable to take the mean of four hand marked-up trajectories as the 'true' trajectory of the player, to which automatically tracked trajectories are compared.

For the modelling of sports players' behaviour, zero error in players' positions would be ideal, although given the variability in human performance, an error of up to 0.5 m might be regarded as acceptable in hand-tracked data. It is expected that data will be usable for behaviour analysis if it is within 1 m of true position, so trajectories will be regarded as acceptable if they are within 1 m of the mean hand-tracked position.

Tracking has been performed on a short sequence of indoor 5-a-side football; four players were tracked in the sequence. Firstly, the tracking was performed using the multi-target CONDENSATION described in Section 3.2, using N = 1000 samples, which resulted

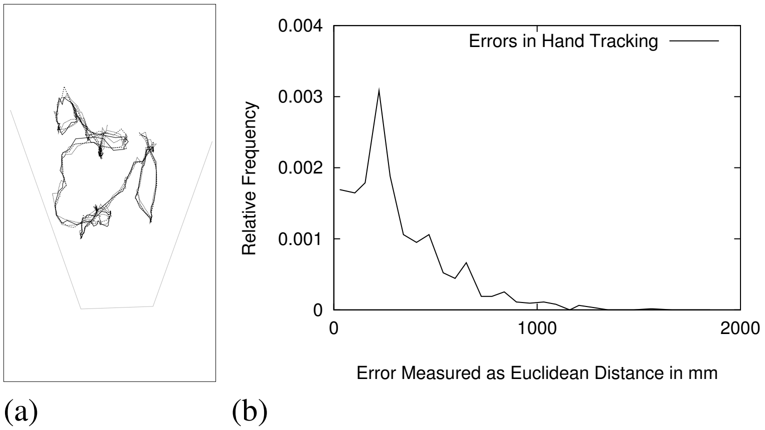
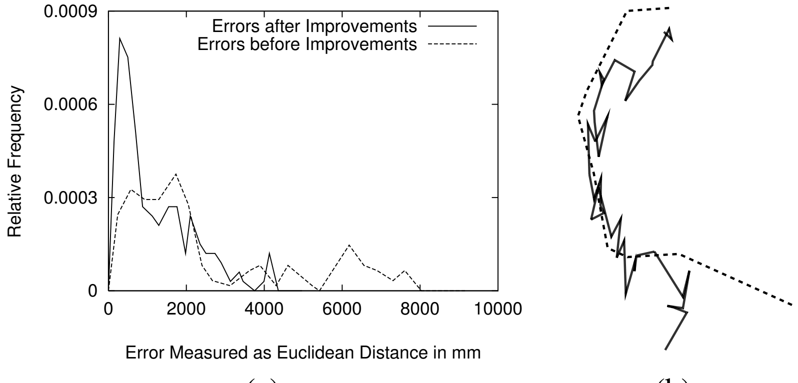


Figure 3.9: (a) Four independent hand-tracked trajectories of the same soccer player over 835 frames, every fifth frame. (b) Euclidean difference between player position, in the six pairwise permutations of the four hand-tracked trajectories.

in the position of the player being allowed to jump around as multiple hypotheses of each player were propagated. Samples were observed switching targets from frame to frame, and not consistently locking onto a specific target. Comparing the trajectories to hand marked-up trajectories revealed a mean error of 2.5 metres for this imperfect system, shown in Figure 3.10(a). The system is not particularly sensitive to the number of samples used. With many more samples $(1000 \ll N)$ the system performs the same, only slower, and it performs less well with considerably fewer samples. [Importance sampling (ICONDENSATION [48]) which allows the unification of low- and high-level tracking methods, using fewer samples to increase speed, would not be of use here, since no high-level shape- or contour-based tracking method is being employed. (This may be of more use to increase tracking speed in Chapter 6, although the shape descriptor developed in Chapter 4 is still relatively low-level)].

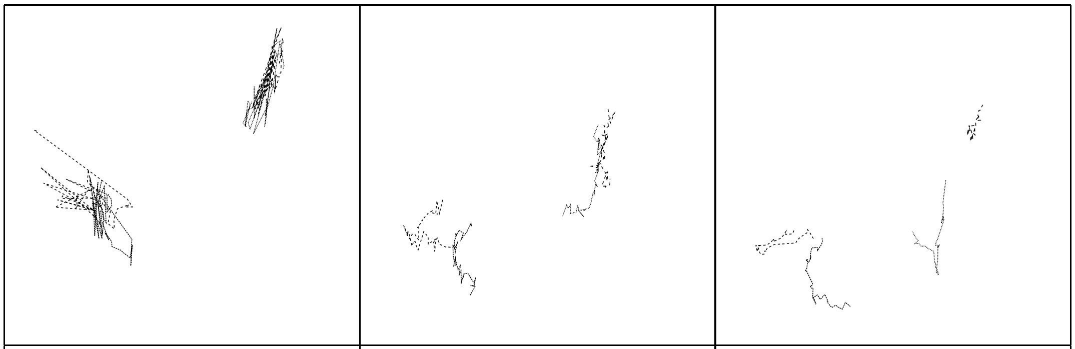
The tracking was performed again after the Kalman filtering extension improvements detailed in Section 3.3. This time, the samples locked onto the four players much better, without swapping players, or having multiple samples tracking a single player. This reduced the mean error in the positions to 1.16 metres, and the modal value to below 400 mm. Figure 3.10(a) shows the error distances, and highlights the improvement in the new tracking system. Figure 3.10(b) shows close up how noisy the trajectories are, indicating that applying an additional smoothing filter to the trajectories after tracking is complete would produce better results.

Figure 3.11 displays the trajectories obtained for both tracking schemes, alongside the hand marked-up trajectories. The bounding boxes marking the tracked players are illustrated in Figure 3.12.



(a) (b)

Figure 3.10: (a) Reduction in the size of the Euclidean error of the tracked footballers on the ground plane after the improvements over 40 frames, compared to the same hand-tracked sequence. (b) Comparison of trajectories of soccer players. The solid line represents the automatically tracked player. The dashed line represents the hand tracked player.



(a) Initial results without Kalman improvements (b) Tracking results with improvements (c) Hand tracked players

Figure 3.11: Comparison of trajectories of soccer players over 40 frames.

3.5 Conclusions

This chapter has presented a novel framework for multi-object tracking. The initial scheme gave 28% of the tracking as usable. With the improvements, 56% of the trajectories are within a metre of the hand marked-up trajectory, and hence usable for behaviour modelling. The errors in the tracker are characterised by the bounding box not fitting to the feet well enough, whilst player tracking is maintained.

Areas for further study identified in this chapter are: improving the fitness function, resolving occlusions, and positional tracker evaluation. The development of a shape model/descriptor for use in evaluating the fitness function within the tracking scheme should improve the localisation of the players. Multiple cameras should assist in resolving occlusion, since they would provide additional information. Positional tracker evaluation should be able to provide more useful information than a simple mean, allowing an insight into how the system is performing. These issues are addressed in the coming chapters.

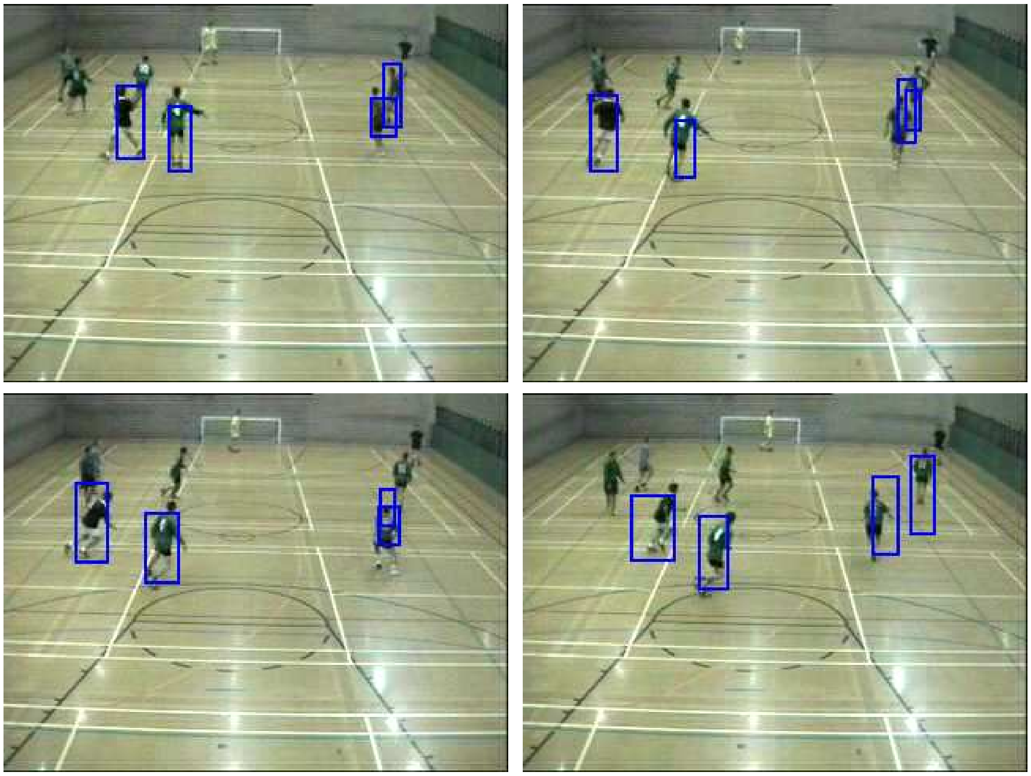


Figure 3.12: Example tracked footage of soccer players, frames 30,40,50,60.

Chapter 4

Feature description

Feature description methods in computer vision are used for a variety of tasks. Being able to describe object's features or characteristics allows them to be categorised or recognised. The comparison of two or more objects can identify how similar or different they are. Describing the features of a sports player is not a trivial task. Sports players' poses and shapes vary enormously. Creating a description to capture the range of configurations of a player's pose is explored in this chapter.

Predict and sample tracking schemes (such as CONDENSATION) are only as powerful as the fitness function used to evaluate how well a sample represents the target object. This chapter focuses on methods that can be used to characterise the shape or features of a target object. Firstly, experiments with B-spline contour models trained on pedestrians and on soccer players are performed. Secondly, a new multi-resolution kernel to locate soccer players is presented; template matching is introduced, followed by an overview of the kernel method and three possible schemes in which it could be used. Following the implementation details for these schemes, evaluation of the schemes is performed. Further evaluation is then performed in depth on an example test image of a soccer game to highlight the properties of the multi-resolution templates.

The final section in this chapter applies the multi-resolution template method developed to sequences of pedestrians walking demonstrating its effectiveness in a different domain and in addition showing the potential use in behaviour modelling: through temporal analysis of pose transitions in a Markov model.

4.1 Contour methods for shape description

Contour based tracking methods are commonly used to model the shape of similar target objects, such as resistors [20, 21], where a point distribution model (PDM) or spline is fitted to a set of landmark points on the outline of the target object. The ‘Leeds People Tracker’ (LPT) of Baumberg [3,4] employs a B-spline model trained from a set of example segmented pedestrians. The top of each pedestrian’s head is identified in the image, and control points evenly spread around the contour. The tracker performs well at finding the contours of similarly shaped people, for example successfully tracking pedestrians in outdoor car parks.

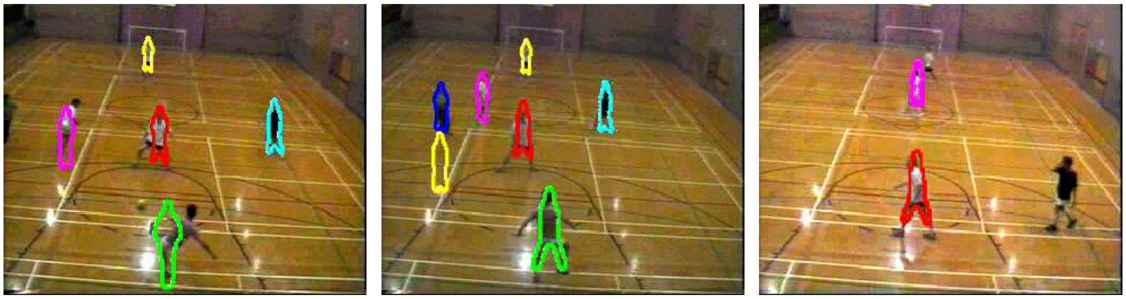


Figure 4.1: Examples of tracking soccer players with Baumberg’s pedestrian B-spline model. The spline struggles to accurately fit to the soccer players outlines, due to the opening of arms or legs.

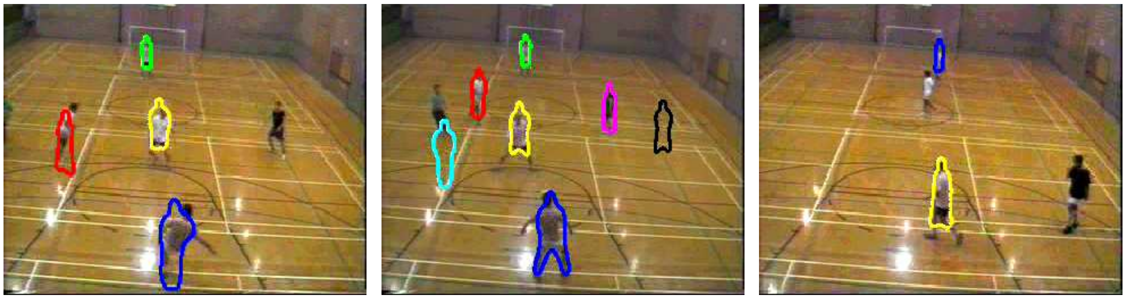


Figure 4.2: Examples of tracking soccer players with the B-spline soccer model. The broader spline fits closer to the players’ shapes, yet still misses feet and arms.

As a starting point for investigating contour shape models in the sports domain, the Leeds People Tracker is used. Firstly, the tracker is used to track soccer players using the pedestrian shape model, then a new shape model is learned to model the shape of soccer players. Figure 4.1 shows the LPT tracking soccer players with the pedestrian spline model, which can be compared to Figure 4.2 showing the LPT adapted to use a spline model trained from segmented examples of soccer players.

The *soccer* model allows for a greater variation in the player's shape. Generally the spline is wider, the major mode of variation is in the separation of the players legs, then the overall width change, followed by the third and fourth major modes of variation being the dropping of the player's left and right shoulders, capturing the players' running motion. Figures 4.3 and 4.4 show close-ups of the pedestrian and soccer B-spline models, respectively, fitted to the image data. Both of the models struggle to capture the players' arms. The soccer model captures the pose where the players' legs are wide apart more successfully, however both tend to 'ride up' the soccer players' images, missing the feet.

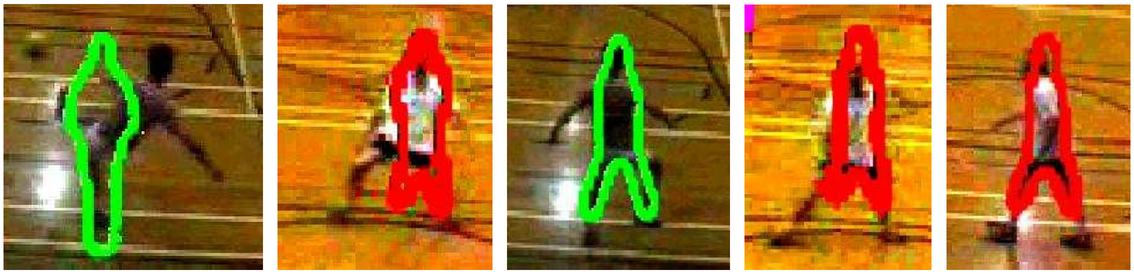


Figure 4.3: Examples of Baumberg's pedestrian B-spline model fitted to soccer players.

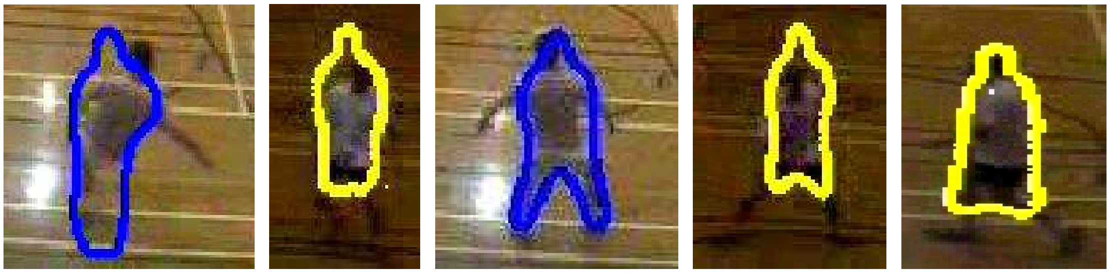


Figure 4.4: Examples of the B-spline soccer model fitted to soccer players.

The soccer shape model makes a small (and rather disappointing) improvement to the performance of the pedestrian tracker on soccer footage. The spline model is unable to adjust sufficiently to fit to the varied shapes of the soccer players. It may have been possible to incorporate a spatiotemporal model [5] to increase the success of fitting the splines to the players in the images. This may have helped to deal with the issues of modelling a player's shape when their arms are open, or legs wide apart. However, this approach was not followed further, since the systems' parameters are extensive and difficult to 'tweak' to cope with the range of conditions present, for example in Figures 4.1 and 4.2 'ghost' players can be seen; the tracker tracks a non-existent player, due to the fact that a player has been stationary in that position in a large number of preceding frames has been incorporated into the background model.

4.2 A multi-resolution people finding kernel

This section describes a landmark-independent scheme for use as the fitness function within the CONDENSATION based tracking of football players. The underlying principles of template matching are introduced, an overview of the approach to creating a set of templates to represent the target objects is presented, and details of the implementation are described.

4.2.1 Introduction to template matching

It is possible to tailor a convolution mask (template) to identify a specific feature. Intuitively the best (highest) response from the convolution of a mask with an image will be where large image values are multiplied by large mask values. Convolution masks which sum to one smooth the image and filter out noise. Convolution masks which sum to zero (hence also having a mean value of zero,) "suppress the effects of varying levels of illumination in the first order" [22, p246].

To detect vertical single pixel width high intensity lines in an image, a convolution mask could be formed by taking a relevant section of the image. For example, taking a suitable image patch:

$$\begin{array}{|c|c|c|}\hline 0 & 255 & 0 \\ \hline 0 & 255 & 0 \\ \hline 0 & 255 & 0 \\ \hline \end{array} \quad \text{we can create Mask 1} \quad = \frac{1}{3}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

Convolving this mask with the image will *highlight* thin vertical lines. Mask 1 looks solely at how well the data fits a vertical single pixel width line.

Creating a template which *distinguishes* between vertical single pixel width lines and other image features involves looking also at how well the mask does *not* fit the data. This can be expressed by forming a mask which sums to zero. The template associated with Mask 1 would be:

$$\frac{1}{9}\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

Each entry in this template is calculated by subtracting $\frac{1}{9}$ from the value of the entry in Mask 1. This makes the sum of the entries zero. This method can be used to create templates of any size. For example to create an $m \times n$ template, an $m \times n$ image patch of the desired feature must be obtained. These values must be normalised so that the entries in the mask sum to one, and finally $\frac{1}{mn}$ must be taken from each entry, ensuring the entries sum to zero.

Table 4.1 shows the effect of applying the two masks to a selection of four possible image patches. The first mask incorrectly indicates that the first high intensity image patch is as much like a vertical single pixel width line as the second image patch which is actually a vertical single pixel width line. The second mask eliminates this problem by also taking into account how well the mask doesn't fit the data. The last two columns in the table show the results of convolving the masks with a horizontal single pixel width high intensity line, and with a vertical single pixel width low intensity line.

image mask	250 250 250 250 250 250 250 250 250	10 250 10 10 250 10 10 250 10	10 10 10 250 250 250 10 10 10	250 10 250 250 10 250 250 10 250
0 $\frac{1}{3}$ 0 0 $\frac{1}{3}$ 0 0 $\frac{1}{3}$ 0	250	250	90	10
$\frac{-1}{9}$ $\frac{2}{9}$ $\frac{-1}{9}$ $\frac{-1}{9}$ $\frac{2}{9}$ $\frac{-1}{9}$ $\frac{-1}{9}$ $\frac{2}{9}$ $\frac{-1}{9}$	0	160	0	-160

Table 4.1: Identifying vertical single pixel lines using convolution masks on example image patches.

4.2.2 Overview of the approach

A template matching approach is employed since it is a more sophisticated way of accurately identifying the location of the footballers. The previous approach of fitting a bounding box to regions of high probability foreground, often had the effect of chopping off the players legs, which are harder to identify as foreground. The multi-resolution people-finding kernel reduces this problem. Throughout this section the word 'kernel' is used to mean 'template represented as a convolution mask'.

The method employed here to determine the fitness of a bounding box on the image to a player is analogous to convolving the portion of the image within the bounding box with a pre-determined kernel, which represents a player. However, the image within the bounding box must be scaled to be of the same resolution as the kernel it is being convolved with. The variation in size (in image pixels) of the players is illustrated in Figure 4.5, and is due to the perspective effects, since the players are different distances away from the camera (similarly, the kernel could be interpolated instead).

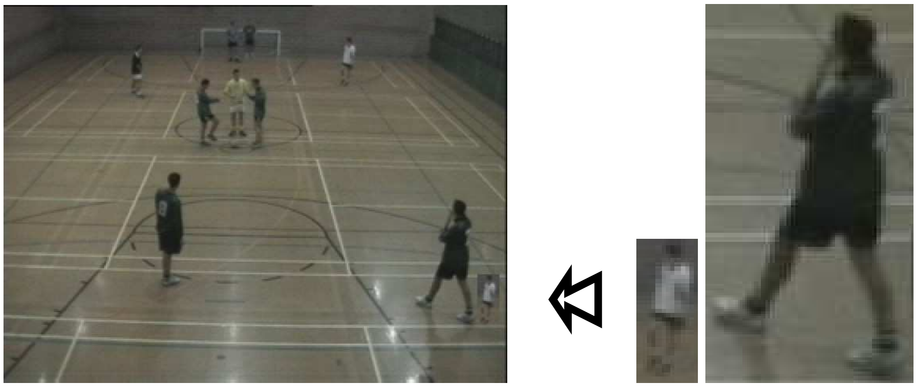


Figure 4.5: Illustration of the difference in pixel sizes of players. Note the player in the white top who appears as the same pixel size superimposed on the near-right, indicated by the arrow. On the right, two players are extracted and enlarged from the image, maintaining the same proportional difference in size.

Once we have created a training set of bounding boxes containing players, there are several approaches that we could take:

A From this set, create a model of ‘player boxes’ (for example using PCA), and from this model generate an example kernel to test against the image example within the bounding box (selecting examples which are less than a Mahalanobis distance of 3 from the mean). Used within the stochastic sampling of the CONDENSATION based scheme this is a sensible approach.

B From this set, choose one example kernel (from the training set) to test against the image example within the bounding box. This doesn’t allow for anything not in the training set to be generated.

C Cluster this data, and select a number of exemplar kernels to use to test against the image example within the bounding box, testing each one, and taking the best result.

4.2.3 Implementation

An offline model of a player is computed from data obtained from a set of (120) hand annotated bounding boxes containing players.

A colour foreground/background model is used to extract the players, which assigns to each pixel a probability of being foreground (belonging to a player). The monochrome ‘foreground probability image’ within the bounding box of a player, of size $w \times h$, is scaled down so that the resolution becomes $a \times b$, by sub-sampling using a Gaussian window function. This removes the obstacle of the differences in scale between players close to the camera, and those further away. This image can now be represented by $\mathbf{g}$, an $a \times b$ vector, such that $\mathbf{g}_j$ equals the value of the foreground probability at pixel j (for $j = 1, \ldots, ab$). A normalised *feature vector*, $\mathbf{f}$, of length ab can be created by taking $\mathbf{f}_j = \frac{\mathbf{g}_j}{\sum_{i=1}^{ab} \mathbf{g}_i}$.

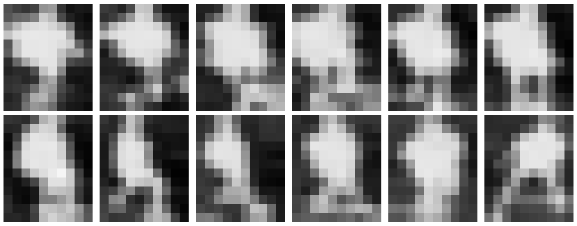


Figure 4.6: Sub-sampled images (10x12 pixels) of soccer players.

This normalised feature vector $\mathbf{f}$ may be represented as a kernel $\mathbf{k}$ with $\mathbf{k}_j = \mathbf{f}_j - \frac{1}{ab}$. The image region within the bounding box may now be evaluated by convolving with the kernel $\mathbf{k}$. The test vector $\mathbf{g}$ represents the sub-sampled (scaled down) image information from the test image within the bounding box. Calculating $\sum_{j=1}^{ab} \mathbf{k}_j \mathbf{g}_j$ gives the response of kernel $\mathbf{k}$ to the image region represented by $\mathbf{g}$. The reasoning behind this approach is explained in Section 4.2.1.

For the first approach (A), the fitness function of an individual player is evaluated by generating an example $\mathbf{k}$ (from an eigen-model of kernel templates of players created from the training data) and calculating $\sum_{j=1}^{ab} \mathbf{k}_j \mathbf{g}_j$.

The second approach (B) involves selecting a $\mathbf{k}$ from the training set and calculating $\sum_{j=1}^{ab} \mathbf{k}_j \mathbf{g}_j$.

For the third approach (C) it is noted that the footballers are often in different poses, and that the entire training set may be well represented by a small number of exemplars. Firstly, we must identify how many clusters it is sensible to use to represent this data. The data is clustered using the k-means algorithm (using various runs, and choosing the best one, i.e. the clustering with the lowest cost function), and Figure 4.7 shows the cost of the k-means clustering against the number of clusters. This is used to choose how many clusters to use. The gradient of the graph begins to become less steep at around the value 5, indicating that incorporating more clusters gains less of a decrease in k-means cost (the RMS distance from each vector to the closest cluster centre), which approaches zero as more clusters are used. Thus five clusters are chosen to represent the data.

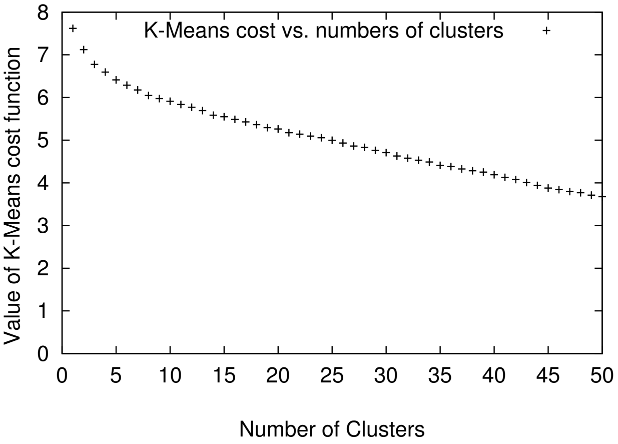


Figure 4.7: From this graph of the k-means cost against the number of clusters, a value of five clusters was chosen, as the point where the gradient of the graph starts to become less steep.

The data clustered is the normalised and shifted 'kernel style' data, along with the additional condition that $\mathbf{k}_j$ is clipped below $\frac{1}{ab}$, i.e. $\mathbf{k}_j < \frac{1}{ab}$. Without this final condition, the cluster centres are found not to represent the data very well, since they are distorted by a few large values, where an example image had few high intensity pixels and many low intensity pixels. With this condition, the cluster centres look plausible (Figure 4.8).

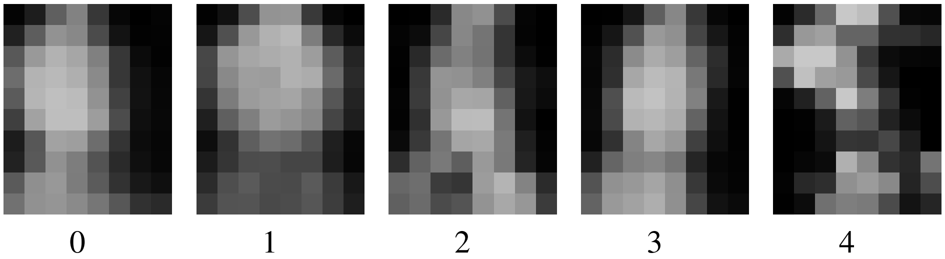


Figure 4.8: The five footballer template kernels.

Five clusters are chosen to represent the various poses, and five templates $(\mathbf{t}^1, \ldots, \mathbf{t}^5)$ are created to represent the feature vectors at the centre of the clusters. For each template,

$\sum_{j=1}^{ab} \mathbf{t}_j^i \mathbf{g}_j$ is calculated, and the largest of these is chosen as the fitness for the player. In each case above the fitness score is set to zero if it is negative (since it is more unlike the model than like it), leaving all fitness values in the range [0,1].

4.2.4 Evaluation

This section evaluates the three methods for applying a multi-resolution kernel as introduced in Section 4.2.2. The evaluation is undertaken in two parts. Firstly, the three methods (A, B & C) are compared. Secondly, the robustness of locating a suitable bounding box around a player is demonstrated on a single test image for the preferred method chosen in the first half of the evaluation.

The three suggested approaches for incorporating the multi-resolution convolution kernel based method into a tracking system were:

A PCA model. Generate an example from the model.

B An example. Randomly choose an example from the training set.

C Five Exemplars. Cluster training set into exemplars and apply each one.

Each of these approaches is tested on a set of (124) positive examples of bounding boxes well-centred on a player (Figure 4.9) and also on a set of (98) negative examples of bounding boxes not well-centred on a player (Figure 4.10). For each approach the foreground extracted image within the bounding box is sub-sampled to an 8 × 10 image, which is then convolved with the kernels.

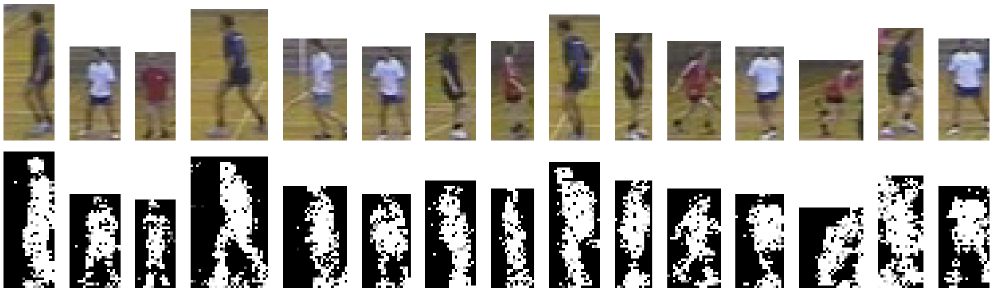


Figure 4.9: A sample from the set of positive examples of bounding boxes well-centred on a player and corresponding foreground/background images.

The negative examples all feature a player, but have chopped off the player's legs, the right-hand side of the body, the player's head, or are judged to be not well fitting to the player's image. Ideally, comparing the difference between the means for the positive

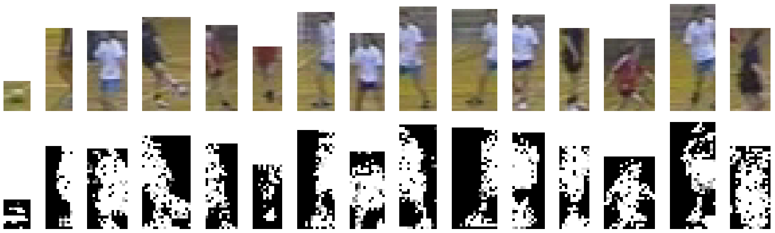


Figure 4.10: A sample from the set of negative examples of bounding boxes not well-centred on a player and corresponding foreground/background images.

and negative examples for each method will identify the method for which the separation between gaining a high response (for a positive example) from the kernel, and a poor response (for a negative example) is greatest. All three methods give a very low response (generally zero) when the bounding box is not near a player or contains only a fraction of a player. Although this is an important property these trivial cases are not of interest here and are not contained in the set of negative examples. Histograms of the results of each method on the two sets are shown in Figure 4.11. Method B appears to perform less well than A and C. This is to be expected, since the example chosen from the training set may well be dissimilar to the test image it is applied to. Method A produces a reasonably evenly spread set of responses for the positive examples, whereas method C produces a negatively skewed distribution of responses, which is preferable for this application; it is better not to gain as many low responses to positive examples. With regard to the negative examples, method A has many more in the very low response region, compared to method C, which can be explained since method C takes the best response for a set of 5 exemplars, one of which may gain a positive response.

Analysis of the means of the three methods on both sets of test images (Table 4.2) shows that the difference between the means is small for method B. For both methods A and C a similar separation between the means is observed, 0.100 and 0.089 respectively.

Method	A	B	C
mean (and s.d.) of positive examples	0.190 (0.081)	0.096 (0.056)	0.222 (0.068)
mean (and s.d.) of negative examples	0.090 (0.091)	0.043 (0.056)	0.133 (0.101)
difference between means	0.100	0.053	0.089

Table 4.2: Comparison of methods A, B, and C as a template fitness function.

Following these observations it is decided to employ method C in the remainder of

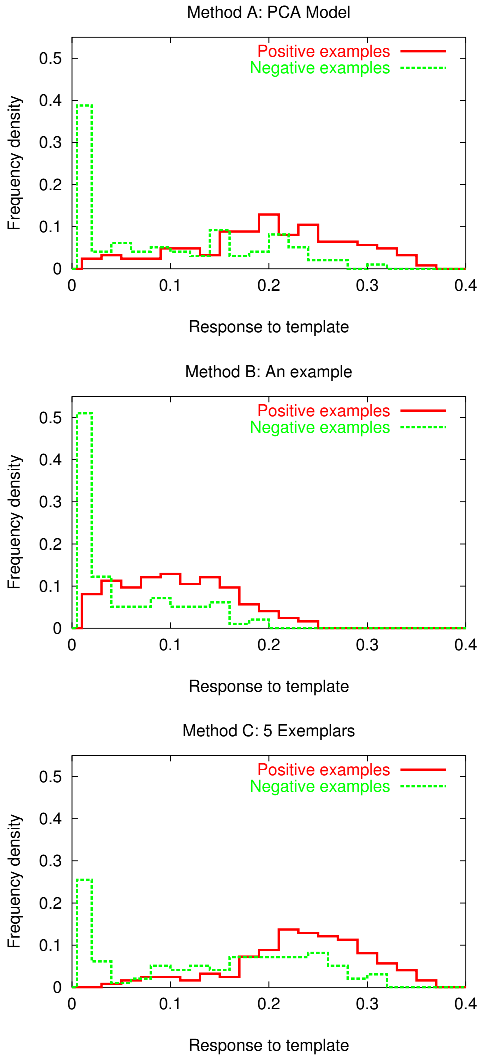


Figure 4.11: Graphs of the response of each of the methods A, B, and C on a set of positive examples of bounding boxes well-centred on a player, and a set of negative examples of bounding boxes not well-centred on a player.

this work. The advantage of this method over method B is clear. Over method A it has the advantage that an example (which may not be representative of any of the training examples) from a PCA model does not need to be generated for each comparison. An added advantage is that pose identification/categorisation may be possible, as is demonstrated in Section 4.3.

For the second part of this section, a scene from a soccer game is used to demonstrate the robustness of locating a suitable bounding box around a player. In particular, the sensitivity of the bounding box is evaluated with respect to vertical positioning, horizontal positioning, changes in width, changes in height, and changes in scale. Figure

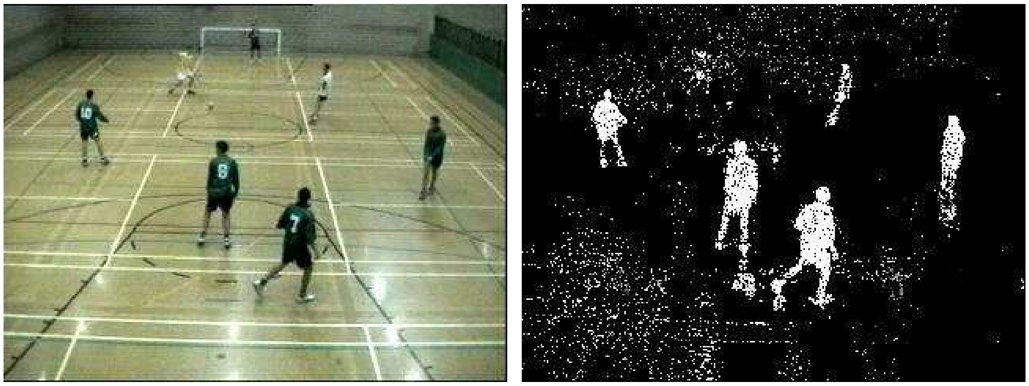


Figure 4.12: Example image from a soccer game with a corresponding foreground-background image used in the analysis of templates' fitness on 'Player 7'.

4.12 shows the image used, and also the foreground segmented from the background. It is this monochrome image to which the templates are applied. As detailed in the previous section, the five template kernels shown in Figure 4.8 are each convolved with the image to gain a response which can be used as a *fitness function* in the CONDENSATION tracking scheme. Inspection of the image reveals that 'Player 7' appears at position $(x, y) = (184, 190)$ with a bounding box of width 38 pixels and height 75 pixels. This bounding box is marked on Figure 4.13, and as in the previous chapter, the position (x, y) is taken as the mid-point of the base of the bounding box. Each bounding box is subsampled to form a 8×10 image, to which the template kernels are applied. In addition to evaluating method C, the performance of the method without the condition that all the templates are clipped below $\frac{1}{80}$ $(\frac{1}{ab})$ is also discussed.

Ideally the fitness function should peak at the value which most agrees with the player's position, should be piecewise continuous and give a small response away from the player's position. The degree of sharpness of the function, in the area surrounding the peak will affect the performance when employed in a tracking scheme.

Figure 4.13 illustrates the shape of the templates' response when the fixed-size (38×75) bounding box is translated horizontally across the image and when it is translated vertically on the image. This demonstrates the sensitivity of the bounding box with respect to vertical and horizontal positioning, respectively. The template responds strongly when vertically centred over the player, however a strong response is also obtained when the bounding box is 20 pixels higher on the image and it is chopping off the player's legs and covering some empty space above the player's head. This is wrong. It may be expected that the lower third of the bounding box should contain mainly background (and two thin legs as foreground), however the templates (Figure 4.8) learned from training

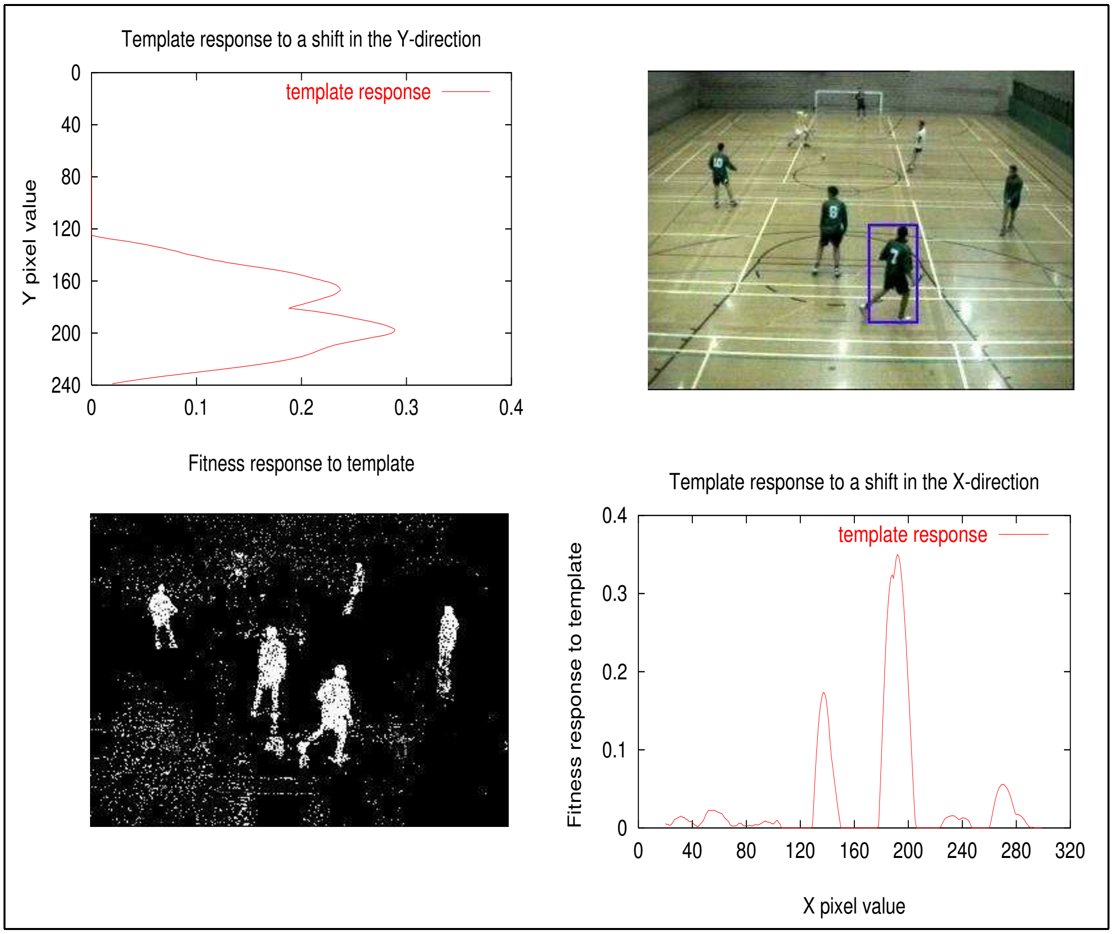


Figure 4.13: Templates' fitness functions illustrating the sensitivity/robustness to vertical and horizontal positioning of the bounding box around 'Player 7'. Each graph lines up with both images allowing analysis of the curve to be explained through comparison with the relevant part of the image.

examples do not reflect this; template 3 in particular has a large amount of foreground in the lower third of the box which may be explained by this template representing examples of thin bounding boxes around stationary players without their arms sticking out. It is hypothesised that a fixed aspect ratio bounding box may solve this problem. It must be noted that the clipped templates perform better than the unclipped templates in this case. Figure 4.14 shows that the unclipped templates give a *greater* response 20 pixels above, which is worse. Horizontally a narrower peaked function is observed. Close to the player, the area slightly to the right receives a stronger response, due to the majority of the player's weight being on the right. A second, smaller peak is observed around $x = 140$ as a response to the legs of 'Player 8', and particularly the shadow cast by the player. This area appears sensitive to lighting changes, due to the glare created by the bright overhead

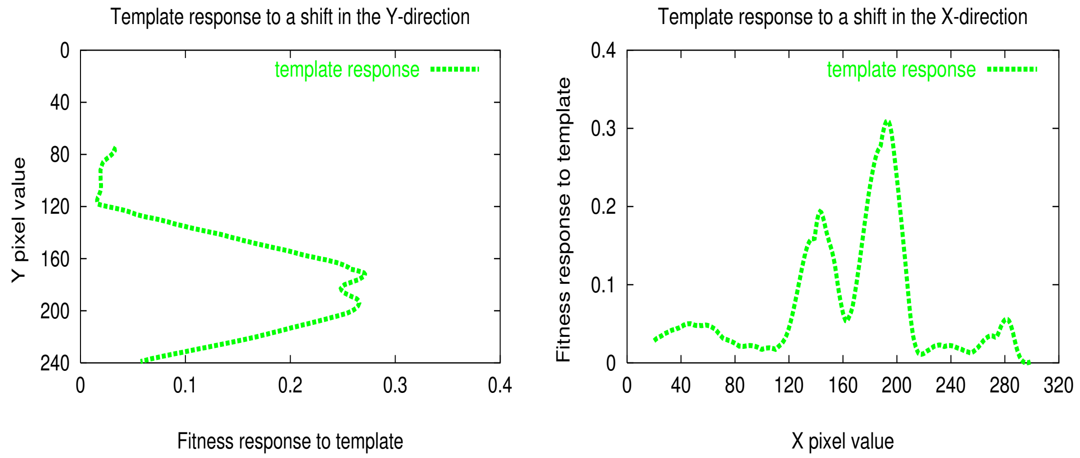
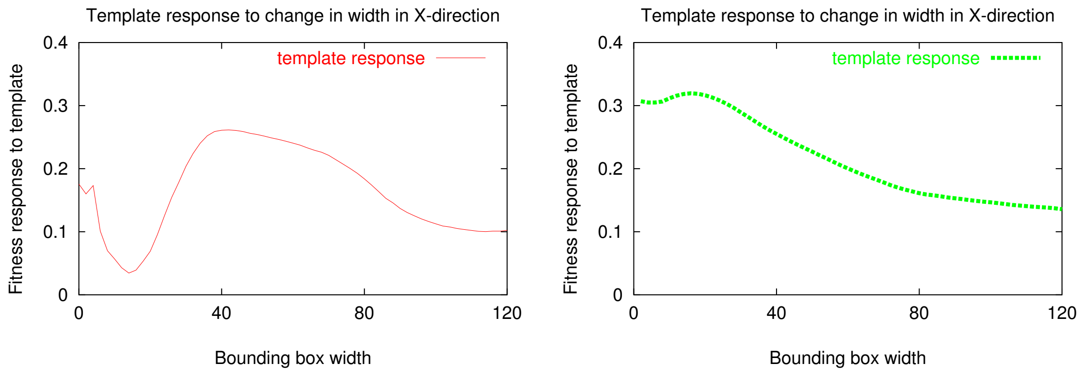


Figure 4.14: Templates' fitness functions illustrating the sensitivity/robustness to vertical and horizontal positioning of the bounding box around 'Player 7' when the exemplar kernels are not clipped below $\frac{1}{80}$.

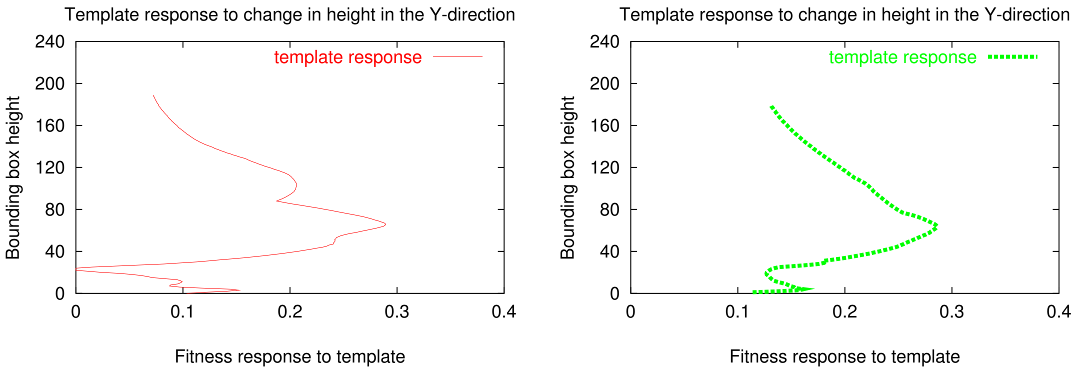
lighting. These peaks are narrower than in the unclipped template version (Figure 4.14) and give a lower response when not well centred on a player.



(a) clipped templates (b) unclipped templates

Figure 4.15: Analysis of templates' fitness with different width bounding boxes. The function is insensitive to small changes in the width of the bounding box. (a) Showing a suitable width of 38 pixels has been chosen. (b) Showing a preference for a narrower box than the 38 pixel wide bounding box marked on Figure 4.13.

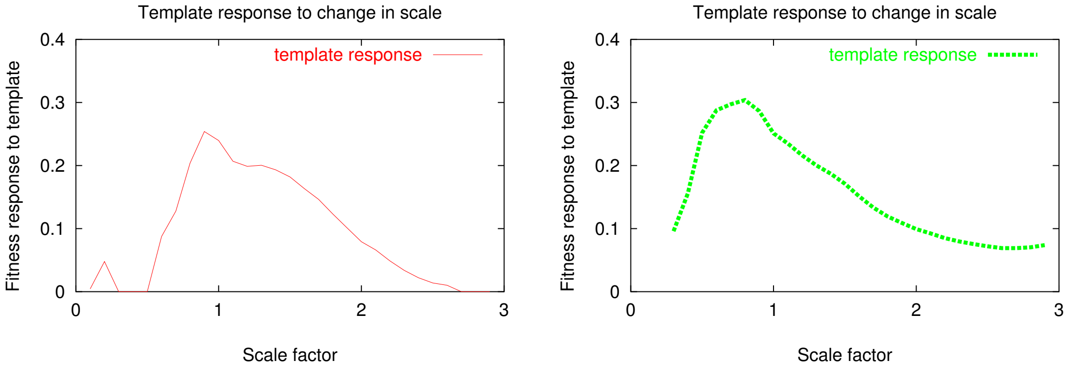
Variations in responses of the templates' fitness functions to changes in width and height of the bounding box are shown in Figures 4.15 and 4.16, respectively. In this example, the fitness function gives a larger response with a narrow bounding box, of around half the width of the suggested 38 pixel wide box. By fixing the location of the player to $(184, 190)$ and varying the height of the bounding box, the fitness function



(a) clipped templates (b) unclipped templates

Figure 4.16: Analysis of templates' fitness with different height bounding boxes. The bounding box illustrated in Figure 4.13 is 65 pixels high, which is at the peak of both graphs.

shown in Figure 4.16 confirms that 65 pixels is a suitable height for the bounding box (in both versions of the templates).



(a) clipped templates (b) unclipped templates

Figure 4.17: Analysis of templates' fitness with different scale bounding boxes (aspect ratio preserved). (a) The templates agree with a scale factor of 1.0. (b) The unclipped templates appear to prefer a smaller bounding box; chopping off the player's head, and horizontal extremities.

The final evaluation investigates the robustness of scale when applying the templates to subsampled players in differing sized bounding boxes. Fixing the (x, y) position and scaling the bounding box by a scale factor in the x and y directions preserves the aspect ratio. The graph in Figure 4.17(a) peaks at approx 0.95 indicating an agreement in the scale of the bounding box, whereas the unclipped version favours smaller bounding boxes.

This brief evaluation has demonstrated the performance of each of the three approaches on a set of positive and negative examples, and identified the most appropriate to use in further work. The robustness of method C has been shown on an example soccer player in an example scene, as a demonstration of the shape of the distribution of the fitness function when the bounding box size or shape varies.

4.3 Kernel function for pedestrian pose analysis

This section examines the use of a multi-resolution kernel for analysis of the pattern of movement through different poses of pedestrians walking, as a basis for demonstrating its potential use in the more complex domain of sports players on a large sports pitch.

4.3.1 Training

To train a model eighteen training sequences of a pedestrian walking have been used (six pedestrians, each appearing in three clips). In the clips (see Figure 4.18), the pedestrian walks normally from right to left as viewed. A Grimson [99] background extraction method is applied to the images, to produce a well segmented foreground image of the pedestrian (Figure 4.18). There is little clutter in the scene, and the single pedestrian can be easily tracked, identifying a suitable bounding box, which can be used for analysis. In total 1206 frames of pedestrians in bounding boxes were identified, giving a large set of pedestrian images to use. Only one target object is in the scene at each time. The pedestrian can easily be extracted by fitting a box around all foreground pixels which are not classified as noise (i.e. with less than 3 neighbouring pixels also foreground).

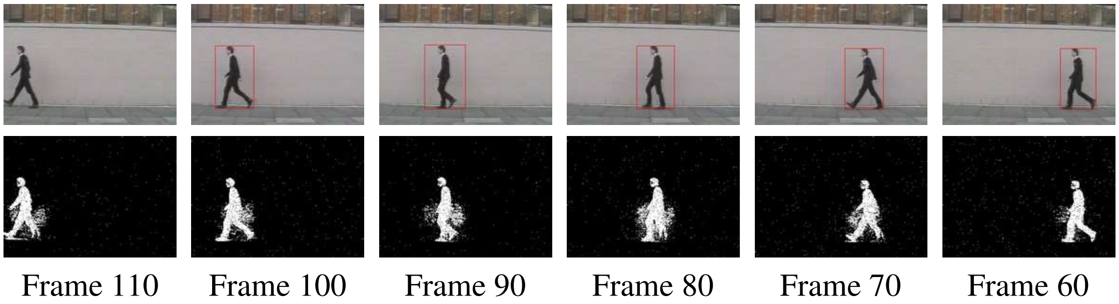


Figure 4.18: Pedestrians: example footage, tracking and foreground extraction.

These bounding boxes are all of different sizes, which can make them hard to analyse since each bounding box contains a different number of pixels. An additional complication is the high dimensionality of the data. [e.g. a pedestrian who covers 100 pixels in height and 50 pixels in width is represented as a vector of length 5000 in a monochrome image, and a vector of length 15000 in a colour (RGB) image].

Applying the multi-resolution kernel to each of the images within the bounding box allows a lower dimensionality vector to be obtained, and importantly for each example pedestrian to be represented by a vector/image of the same size. In this example, the images are scaled down to a 12×15 image, equivalent to a 180 dimensional vector. This reduces the dimensionality by a factor of approximately 80. $(80 \times 180 = 14400 \sim 15000)$.

4.3.2 Clustering

Once the 1206 training examples were obtained, the k-means clustering algorithm was applied to the data. The results of the k-means cost (RMS distances from each vector to closest cluster centre) versus number of clusters is plotted in Figure 4.19. This graph is used to decide that it is sensible to use six clusters to represent the data, as the curve becomes less steep. This is supported by the fact that including more templates was judged not to produce additional templates which were noticeably different.

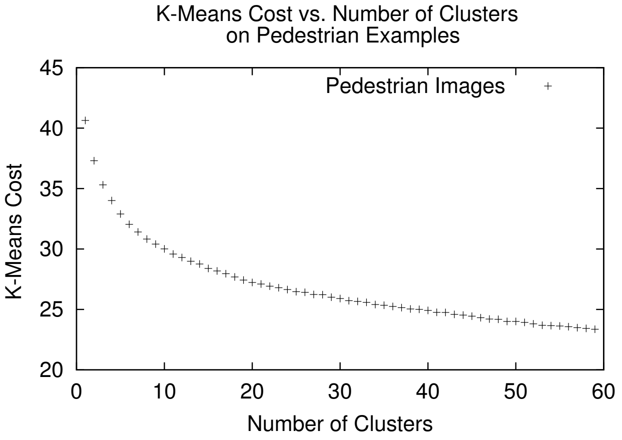


Figure 4.19: K-means cost of different numbers of clusters of pedestrian kernels.

The training examples are clustered using the k-means algorithm into six clusters (each of which contain between 75 and 370 of the original 1206 training examples) revealing the six templates which can be used to identify the different poses. Images representing the feature vectors at the centre of the clusters are shown in Figure 4.20.

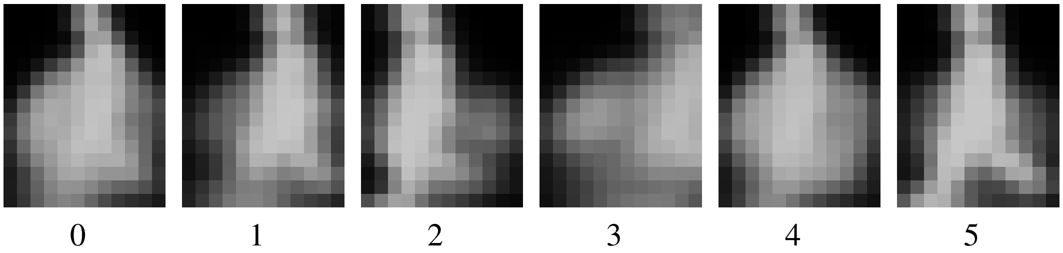


Figure 4.20: The six pedestrian template kernels. Templates 1 and 5 represent pedestrians when striding out with legs apart, templates 0 and 4 represent when the legs are reasonably close together and some shadow is cast, template 2 represents the pedestrian with legs close together, and template 3 looks least like a person and is formed from the start of image sequences as the pedestrians enter the scene from the right.

4.3.3 Pose transitions

Given the six template kernels, each of the 1206 examples can now be convolved with each of these kernels, and assigned as closest to the one which returns the highest score. Since these example images are taken as part of a sequence, it is possible to create a transition matrix (Table 4.3) of the movements from one pose/template to the next. From the 18 sequences 1188 pose transitions are observed $(1206 - 18 = 1188)$. Figure 4.21 illustrates graphically the frequency of transitions between poses observed in the training sequences. For example, a pedestrian may enter the scene and be represented by ‘template 3’ (cross reference with Figure 4.20) in the bottom-left of Figure 4.21. After a few frames in this state, a state transition may occur as the pedestrian becomes represented better by ‘template 0’ for a period of time. Following this the pedestrian may move to being represented by ‘template 5’ in the state at the top-middle of the figure, with legs fully open, followed by ‘template 2’ as the legs begin to close, and returning to ‘template 0’ as the legs become close together. Cycles through such behaviours was generally observed as the pedestrians walked from right to left in the video footage. The amount of shadow cast by the pedestrian also affected the pose transitions of the pedestrians. Tables 4.4 and 4.5 show a test sequence with the fitness scores for each kernel. Each kernel is ranked by fitness score (result when convolved with the subsampled image) with the best (highest score) highest in the table, for each frame.

Having used k-means clustering to identify six poses (states) that a pedestrian can be in at any one time, it is possible to create a Markov model of the pedestrian walking. *Markov models* (first order) allow an outcome of an independent process to depend *only* on the state at the generative time step. Thus, by analysing the changes between poses in the training set (18 sequences in total; 3 sequences of 6 pedestrians), a Markov model can

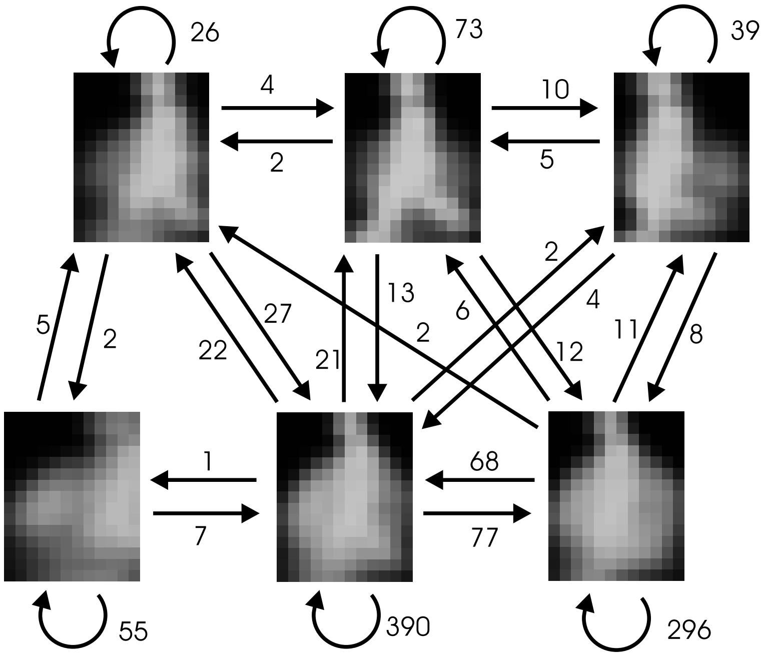


Figure 4.21: Pedestrian pose transition graph.

	0	1	2	3	4	5	Total
0	390	22	2	1	77	21	513
1	27	26	0	2	0	4	59
2	4	0	39	0	8	5	56
3	7	5	0	55	0	0	67
4	68	2	11	0	296	6	383
5	13	2	10	0	12	73	110
Total	509	57	62	58	393	109	1188

Table 4.3: Transition graph between pedestrian templates 0-5.

be created. A stochastic matrix, $P = p(i,j)$, of transition probabilities is formed based on the frequency of transitions from one pose/state to another observed in the training data. If the frequency of transitions (Table 4.3) from state i to state j is denoted $freq(i,j)$, then:

$$p(i,j) = \frac{freq(i,j) + 1}{\Sigma_k freq(i,k) + N} \tag{4.1}$$

where N = the number of states, here $N = 6$. This preserves the need for $\Sigma_k p(i,k) = 1$ to create a stochastic matrix, and allows for all possible transitions to occur.

$$P = \begin{pmatrix} 0.753 & 0.044 & 0.006 & 0.004 & 0.150 & 0.042 \\ 0.431 & 0.415 & 0.015 & 0.046 & 0.015 & 0.076 \\ 0.081 & 0.016 & 0.645 & 0.016 & 0.145 & 0.097 \\ 0.110 & 0.082 & 0.014 & 0.767 & 0.014 & 0.014 \\ 0.177 & 0.008 & 0.031 & 0.003 & 0.763 & 0.018 \\ 0.121 & 0.026 & 0.095 & 0.009 & 0.112 & 0.638 \end{pmatrix} \tag{4.2}$$

It is now possible to:

- generate a sequence, by taking a probabilistic walk through the model
- evaluate how likely a test sequence of a pedestrian is (in comparison to the training sequences)

This allows the state or pose of the pedestrian at the next time step to be predicted. Within the stochastic sampling of the CONDENSATION tracking, or indeed any ‘predict and sample’ tracking scheme, this would allow the samples to be partitioned, so that a corresponding percentage of samples of each pose type could be used for tracking. i.e. tracking a pedestrian at time t, who is in pose 1, it would be expected at time $t+1$ to be in pose 0 or 1, where most of our samples can be directed, with a few for the remaining poses/states.

4.4 Summary

A novel fitness function for use in a CONDENSATION based tracking framework has been presented. A set of exemplar poses are learned from subsampled example images of soccer players, creating a set of multi-resolution template kernels which when convolved with the image respond suitably. This assists with the localisation of target players in the tracking application. The same technique has also been applied to pedestrians, identifying their poses as they walk.

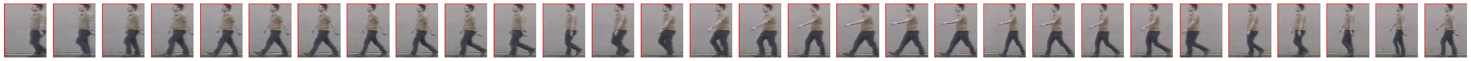
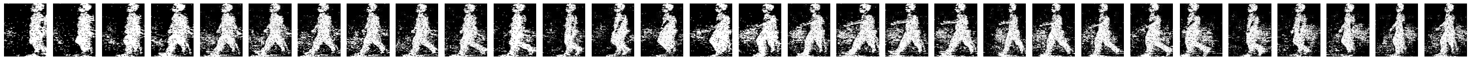
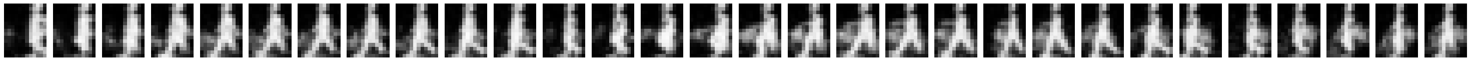
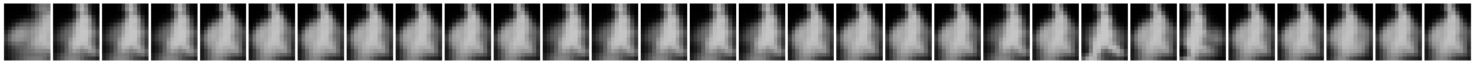
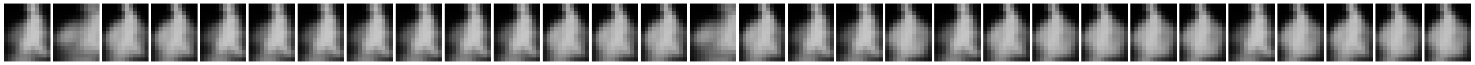
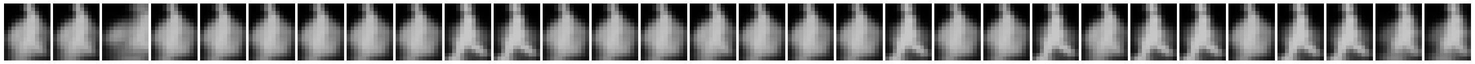
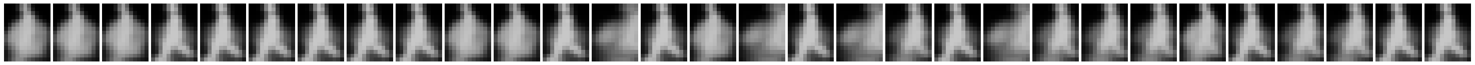
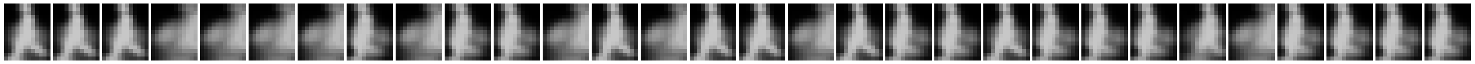
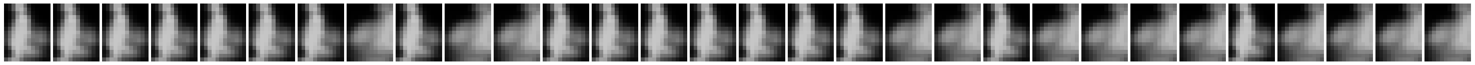
Frame	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Image																														
Foreground Image																														
Subsampled Image																														
1st Best Fit																														
Template No.	3	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	0	5	0	2	0	0	4	0	0
Fitness Score	28	28	34	42	42	42	42	42	40	41	41	34	36	38	31	38	42	37	42	41	37	42	42	41	40	35	36	36	37	42
2nd Best Fit																														
Template No.	1	3	0	0	1	1	1	1	1	1	1	0	0	0	3	0	1	1	4	1	0	4	4	4	4	1	4	0	4	4
Fitness Score	19	22	25	40	40	39	37	36	39	38	38	28	30	36	30	37	39	36	36	35	32	39	40	36	28	33	35	36	33	37
3rd Best Fit																														
Template No.	0	0	3	4	4	4	4	4	4	5	5	4	4	4	0	4	4	4	5	4	4	5	0	5	5	4	5	5	1	1
Fitness Score	5	17	22	29	31	32	34	34	30	35	36	17	19	26	23	27	32	28	34	33	23	38	40	35	22	30	32	32	32	35
4th Best Fit																														
Template No.	4	4	4	5	5	5	5	5	5	4	4	5	3	5	4	3	5	3	1	5	3	1	1	1	0	5	1	1	5	5
Fitness Score	-6	5	13	22	26	30	32	33	29	34	35	13	15	19	13	25	23	19	33	30	20	29	25	35	21	28	29	27	29	33
5th Best Fit																														
Template No.	5	5	5	3	3	3	3	2	3	2	2	3	5	3	5	5	3	5	2	2	5	2	2	2	1	3	2	2	2	2
Fitness Score	-15	-4	2	13	11	9	8	10	10	8	10	13	14	14	1	12	16	18	15	10	17	19	25	12	-2	11	16	19	13	15
6th Best Fit																														
Template No.	2	2	2	2	2	2	2	3	2	3	3	2	2	2	2	2	2	2	3	3	2	3	3	3	3	2	3	3	3	3
Fitness Score	-32	-23	-16	-2	3	6	8	7	3	8	8	-9	-9	-1	-14	-1	7	3	7	10	1	5	3	10	-7	5	8	8	9	11

Table 4.4: Ranked templates on an example image sequence (frames 5-34). The fitness score for each template is shown in descending order.

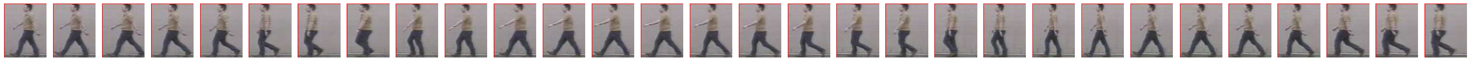
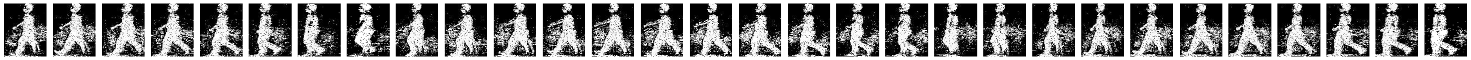
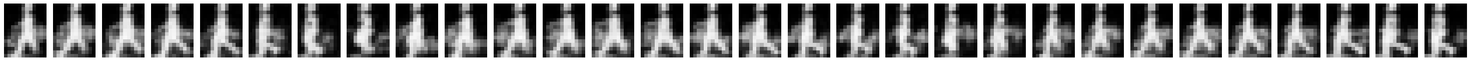
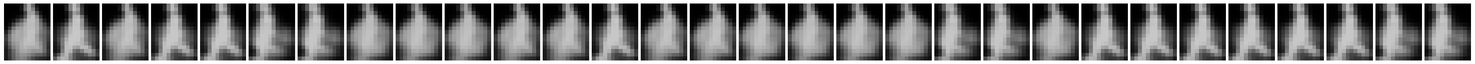
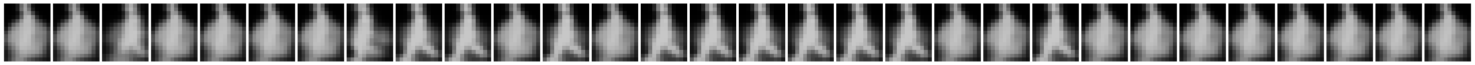
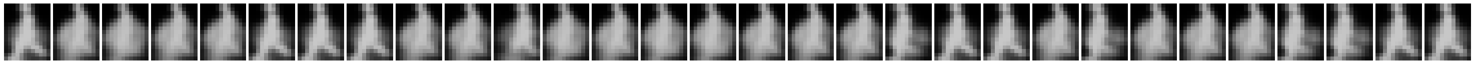
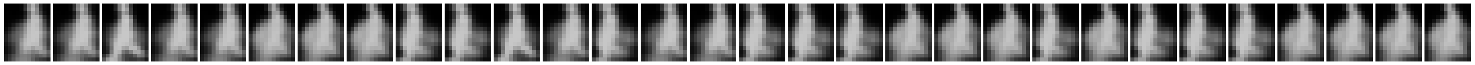
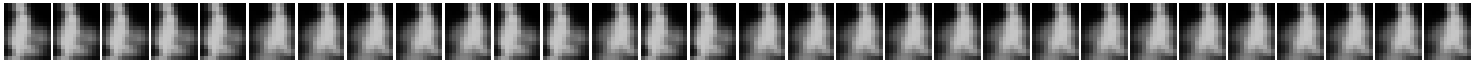
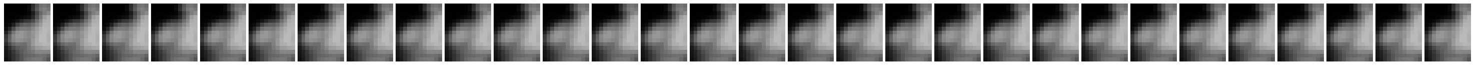
Frame	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
Image																														
Foreground Image																														
Subsampled Image																														
1st Best Fit																														
Template No.	0	5	0	5	5	2	2	4	4	4	0	0	5	0	0	4	4	4	4	2	2	4	5	5	5	5	5	5	2	2
Fitness Score	42	41	44	46	44	41	39	37	44	44	45	44	41	42	42	42	40	40	37	36	40	40	38	42	42	43	42	38	41	41
2nd Best Fit																														
Template No.	4	4	1	4	4	4	4	2	5	5	4	5	4	5	5	5	5	5	5	4	4	5	4	4	4	4	4	4	4	4
Fitness Score	38	41	37	42	41	34	24	36	41	43	37	39	39	40	42	41	40	36	32	36	36	39	38	39	40	40	39	36	31	27
3rd Best Fit																														
Template No.	5	0	4	0	0	5	5	5	0	0	1	4	0	4	4	0	0	0	2	5	5	0	2	0	0	0	2	2	5	5
Fitness Score	37	41	37	41	40	33	19	34	40	40	36	38	39	39	42	40	35	35	31	31	31	36	33	37	38	35	34	35	29	22
4th Best Fit																														
Template No.	1	1	5	1	1	0	0	0	2	2	5	1	2	1	1	2	2	2	0	0	0	2	0	2	2	2	0	0	0	0
Fitness Score	33	25	36	27	28	25	16	28	29	31	34	33	24	29	26	26	29	26	29	27	27	28	31	27	26	30	33	28	21	16
5th Best Fit																														
Template No.	2	2	2	2	2	1	1	1	1	1	2	2	1	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Fitness Score	17	23	12	26	24	7	-4	10	24	20	14	18	22	20	24	22	18	20	14	11	8	23	15	23	24	20	16	9	-2	-6
6th Best Fit																														
Template No.	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Fitness Score	8	5	12	6	5	-1	-7	-0	4	0	9	8	1	6	6	5	4	7	6	3	1	6	3	6	8	7	5	2	-4	-6

Table 4.5: Ranked templates on an example image sequence (frames 35-64). The fitness score for each template is shown in descending order.

Chapter 5

Performance evaluation of positional tracking systems

There are many ways in which the performance of a computer vision system can be evaluated. For evaluating the performance of a 'sports player tracker', the aim is to evaluate how well a tracker is able to determine the position of a target object. Few metrics exist for positional tracker evaluation; here the fundamental issues of trajectory comparison are addressed, and metrics are presented which allow the key features to be described.

In Chapter 3, means and standard deviations of errors in tracked data from manually marked up data have been presented, with simple plots. Harville [39] presents similar positional analysis when evaluating the results of person tracking using plan-view algorithms on footage from stereo cameras. In certain situations Dynamic Programming can be applied to align patterns in feature vectors, for example in the speech recognition domain, known as Dynamic Time Warping (DTW) [92]. In this work trajectory evaluation builds upon comparing equal length (in time) trajectories having frame by frame time steps with direct correspondences.

When undertaking performance evaluation of a computer vision system, it is important to consider the requirements of the system. Common applications include detection (simply identifying if the target object is present), coarse tracking (for surveillance applications), tracking (where reasonably accurate locations of target objects are identified), and high-precision tracking (for medical applications, reconstructing 3D body movements). This section focuses on methods behind *positional* tracker evaluation, the aim being to evaluate how well a tracker is able to determine the position of a target object, for use in tracking and high-precision tracking as described above. The metrics developed are applied to real trajectories for positional tracker evaluation. Data obtained from the sports player tracker on video of a 5-a-side soccer game, and from a vehicle tracker, is analysed. These give quantitative positional evaluation of the performance of computer vision tracking systems, and provides a framework for comparison of different methods and systems on benchmark datasets.

5.1 Metrics and statistics for trajectory comparison

A *trajectory* is a sequence of positions over time. The general definition of a trajectory T is a sequence of positions (x_i, y_i) and corresponding times, t_i:

$$T = \{(x_1, y_1, t_1), (x_2, x_2, t_2), \ldots, (x_n, y_n, t_n)\} \tag{5.1}$$

In the computer vision domain, when using video footage, time steps are usually equal, and measured in frames. Thus, t_n may be dropped, as the subscript on the positions can be taken as time, and Equation 5.1 becomes:

$$T = \{(x_1, y_1), (x_2, x_2), \ldots, (x_n, y_n)\} \tag{5.2}$$

i.e. trajectory T is a sequence of (x_i, y_i) positions at time step i, as illustrated in Figure 5.1. *Paths* are distinguished from trajectories by defining a path as a trajectory not parameterised by time.

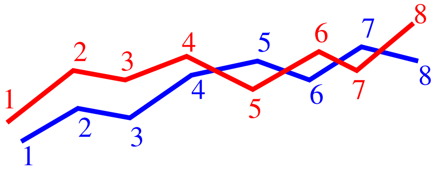


Figure 5.1: Example of a pair of trajectories.

To evaluate the performance of the tracker, metrics comparing two trajectories need to be devised. We have two trajectories T_A and T_B which represent the trajectory of a target from the tracker, and the ground truth trajectory - which is usually marked up manually from the footage. Metrics comparing the trajectories allow us to identify how *similar*, or how *different* they are.

5.1.1 Comparison of trajectories

Consider two trajectories composed of 2D positions at a sequence of time steps. Let positions on trajectory T_A be (x_i, y_i), and on trajectory T_B be (p_i, q_i), for each time step i. The displacement between positions at time step i is given by $\mathbf{d}_i$:

$$\mathbf{d}_i = (p_i, q_i) - (x_i, y_i) = (p_i - x_i, q_i - y_i) \tag{5.3}$$

And the distances between the positions at time step i are given by d_i:

$$d_i = |\mathbf{d}_i| = \sqrt{(p_i - x_i)^2 + (q_i - y_i)^2} \tag{5.4}$$

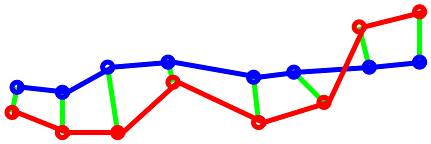


Figure 5.2: Comparison of displacement between two trajectories.

A metric commonly used for tracker evaluation is the mean of these distances, as used in Section 3.4. We shall call this metric m_1.

$$m_1 = \mu(d_i) = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{5.5}$$

m_1 gives the average distance between positions at each time step. Figure 5.2 shows two trajectories and identifies the distance between corresponding positions. The distribution of these distances is also of significance, as it shows how the distances between trajectories (tracker error) are spread, as illustrated in Figure 5.3, where a skewed distribution can be seen. Other statistics provide quantitative information about the distribution. Here we identify the mean, median (expected to be lower than the mean, due to the contribution to the mean of the furthest outliers), standard deviation, minimum and maximum values as useful statistics for describing the data. Let us define $\mathcal{D}(T_A, T_B)$ to be the set of distances d_i between trajectory A and B. The above statistics can be applied to this set:

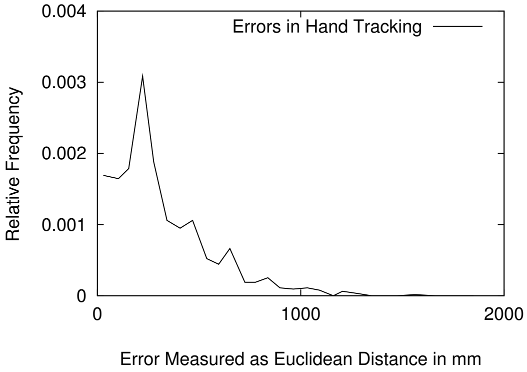


Figure 5.3: Distribution of distances between positions.

$$
\begin{array}{llcll}
\text{Mean} & \mu(\,\mathcal{D}(T_A, T_B)\,) & = & \frac{1}{n}\sum_{i=1}^{n} d_i & \\
\text{Median} & median(\,\mathcal{D}(T_A, T_B)\,) & = & d_{\frac{n+1}{2}} & \text{if } n \text{ odd,} \\
 & & = & \frac{1}{2}(d_{\frac{n}{2}} + d_{\frac{n}{2}+1}) & \text{if } n \text{ even} \\
\text{Standard deviation} & \sigma(\,\mathcal{D}(T_A, T_B)\,) & = & \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i - \mu(d_i))^2} & \\
\text{Minimum} & min(\,\mathcal{D}(T_A, T_B)\,) & = & \text{the smallest } d_i & \\
\text{Maximum} & max(\,\mathcal{D}(T_A, T_B)\,) & = & \text{the largest } d_i &
\end{array}
\tag{5.6}
$$

5.1.2 Spatially separated trajectories

Some pairs of trajectories may be very similar, except for a constant difference in some spatial direction (Figure 5.4). Defining a metric which takes this into account may reveal a closer relationship between two trajectories. Given the two trajectories T_A and T_B, it is possible to calculate the optimal spatial translation $\hat{\mathbf{d}}$ (shift) of T_A towards T_B, for which m_1 is minimised. $\hat{\mathbf{d}}$ is the average displacement between the trajectories, and is calculated as:

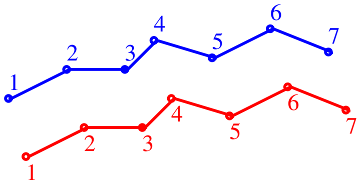


Figure 5.4: Two spatially separated trajectories.

$$\hat{\mathbf{d}} = \mu(\mathbf{d}_i) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{d}_i \tag{5.7}$$

Now we can define $\mathcal{D}(T_A+\hat{\mathbf{d}}, T_B)$ to be the set of distances between a translated trajectory T_A (by $\hat{\mathbf{d}}$) and T_B. The same statistics can be applied to this set, $\mathcal{D}(T_A + \hat{\mathbf{d}}, T_B)$, to describe the distances. $\mu(\ \mathcal{D}(T_A + \hat{\mathbf{d}}, T_B)\ ) < \mu(\ \mathcal{D}(T_A, T_B)\ )$ in all cases, except when the trajectories are already optimally spatially aligned.

When $\mu(\ \mathcal{D}(T_A + \hat{\mathbf{d}}, T_B)\ )$ is notably lower than $\mu(\ \mathcal{D}(T_A, T_B)\ )$, it may highlight a tracking error of a consistent spatial difference between the true position of the target, and the tracked position.

5.1.3 Temporally separated trajectories

Some pairs of trajectories may be very similar, except for a constant time difference (Figure 5.4). Defining a metric which takes this into account may reveal a closer relationship between two trajectories. Given the two trajectories T_A and T_B, it is possible to calculate

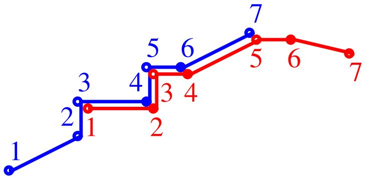


Figure 5.5: Two temporally separated trajectories.

the optimal temporal translation j (shift) of T_A towards T_B, for which m_1 is minimised. When the time-shift j is positive $T_{A,i}$ is best paired with $T_{B,i+j}$, and when j is positive $T_{A,i+j}$ is best paired with $T_{B,i}$. Time-shift j is calculated as:

$$j = \arg\,\min_{\mathrm{k}} \Big(\frac{1}{n-|k|} \sum_{i=Q}^{R} \big|(p_{i+k}, q_{i+k}) - (x_i, y_i)\big| \Big) \tag{5.8}$$

$$\text{if } k \geqslant 0 \text{ then } Q = 0 \text{ else } Q = -k.\ \ R = Q + n - |k|.$$

Now we can define $\mathcal{D}(T_A, T_B, j)$ to be the set of distances between a temporally translated trajectory T_A or T_B, depending on j's sign. The same statistics as before can be applied to this set, $\mathcal{D}(T_A, T_B, j)$, to describe the distances. $\mu(\ \mathcal{D}(T_A, T_B, j)\ ) < \mu(\ \mathcal{D}(T_A, T_B)\ )$ in all cases, except when the trajectories are already optimally temporally aligned.

When $\mu(\ \mathcal{D}(T_A, T_B, j)\ )$ is significantly lower than $\mu(\ \mathcal{D}(T_A, T_B)\ )$, it may highlight a tracking error of a consistent temporal difference between the true position of the target, and the tracked position. In practice j should be small; it may highlight a lag in the tracked position (Figure 5.5).

5.1.4 Spatio-Temporally separated trajectories

Combining the spatial and temporal alignment process identifies a fourth distance statistic. We define $\mathcal{D}(T_A + \hat{\mathbf{d}}', T_B, j)$ to be the set of distances between the spatially and temporally optimally aligned trajectories, where $\hat{\mathbf{d}}' = \hat{\mathbf{d}}(T_A, T_B, j)$ is the optimal spatial shift between the temporally shifted (by j time steps) trajectories.

The procedure for defining this set is similar to above; calculate the optimal j for which the mean distance between space (translation of $\hat{\mathbf{d}}'$) and time (time-shift of j) shifted positions is minimised, using an exhaustive search. Once j has been calculated, the set of distances $\mathcal{D}(T_A + \hat{\mathbf{d}}', T_B, j)$ can be formed, and the usual statistics can be calculated.

When the trajectories are spatio-temporally aligned, the mean value, $\mu(\ \mathcal{D}(T_A + \hat{\mathbf{d}}', T_B, j)\ )$ is less than or equal to the mean value of the three other sets of distances; when the trajectories are unaltered, spatially aligned, or temporally aligned.

5.1.5 Area between trajectories

The area between two trajectories provides time independent information. The trajectories must be treated as paths whose direction of travel is known. *Paths* are distinguished from *trajectories* by not being parameterised by time.

Given two paths A and B, the area between them is calculated by firstly calculating the set of crossing points where path A and path B intersect. These crossing points are then used to define a set of regions. If a path crosses itself *within* a region, then the loop created is discarded by deleting the edge points on the path between where the path crosses itself. This resolves the problem of calculating the area if a situation where a path crosses itself many times occurs, as illustrated in Figure 5.6. Now the area between the paths can be calculated as the summation of the areas of the separate regions. The area of each region is calculated by treating each region as an n-sided polygon defined by edge points (x_i, y_i) for $i = 1, \ldots, n$, where the first point is the intersection point, the next points follow those

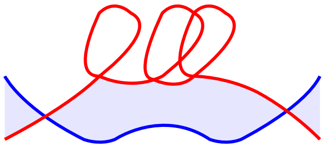


Figure 5.6: Regions with self crossing trajectories. The shaded regions show the area calculated.

on path A, then the second crossover point, back along path B to the first point. i.e. the perimeter of the polygon is traced. Tracing the polygon, the area under each edge segment is calculated as a trapezoid; each of these is either added to or subtracted from the total, depending on its sign, which results from the calculation of $(x_{i+1} - x_i)(y_i + y_{i+1})/2$ as the area between the x-axis and the edge segment from (x_i, y_i) to (x_{i+1}, y_{i+1}). After rearrangement Equation 5.9 shows the area of such a region. (It does not matter which way the polygon is traced, since in our computation the modulus of the result is taken).

$$A_{region} = \left| \frac{1}{2} \left(\left((\sum_{i=1}^{n-1} x_{i+1} y_i) + x_1 y_n \right) - \left((\sum_{i=1}^{n-1} x_1 y_{i+1}) + x_n y_1 \right) \right) \right| \tag{5.9}$$

The areas of each of the regions added together give the total area between the paths, and has dimensionality L^2 i.e. mm^2. To obtain a useful value for the area metric, the area calculated is normalised by the average length of the paths. This gives the ‘area’ metric on the same scale as the other distance statistics. It represents the average time independent distance (in mm) between the two trajectories, and is a *continuous average distance*, rather than the earlier discrete average distance.

5.2 Evaluation, results and discussion

Performance evaluation is performed on two tracking systems; the sports player tracker, and a vehicle tracker [66]. Figure 5.7 shows example footage used in each system. First, the variability between two hand marked up trajectories is discussed.

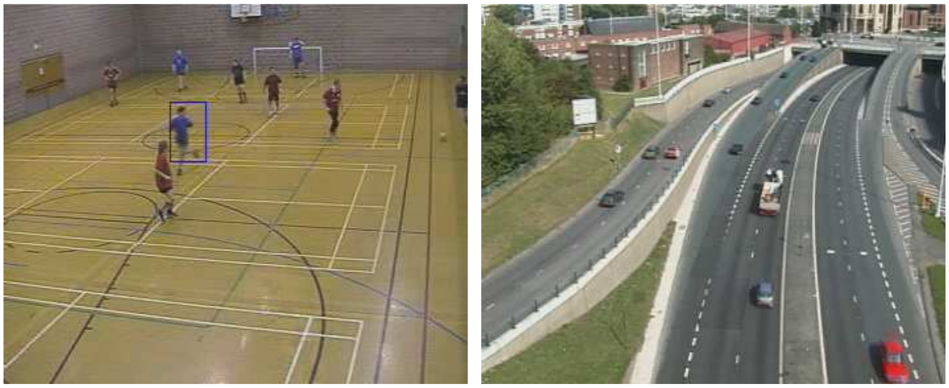


Figure 5.7: Example footage used for tracking.

5.2.1 Comparison of two hand marked up trajectories

This section compares two independently hand marked up trajectories of the same soccer player during an attacking run. There are small differences in the trajectories, and they cross each other many times. The results are shown in Table 5.1, and the trajectories are shown graphically in Figure 5.8 with the area between the two paths shaded, and dark lines connecting the positions on the trajectories at each time step. The second row of Table 5.1 identifies an improvement in the similarity of the two trajectories if a small spatial shift of $\hat{\mathbf{d}} = (-58, 74)$ in mm, is applied to the first trajectory. As expected in hand marked up data, the two trajectories are optimally aligned in time (time-shift $j = 0$).

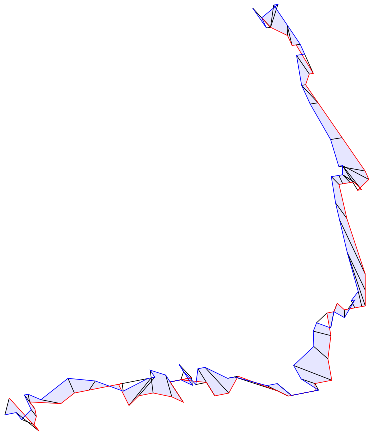


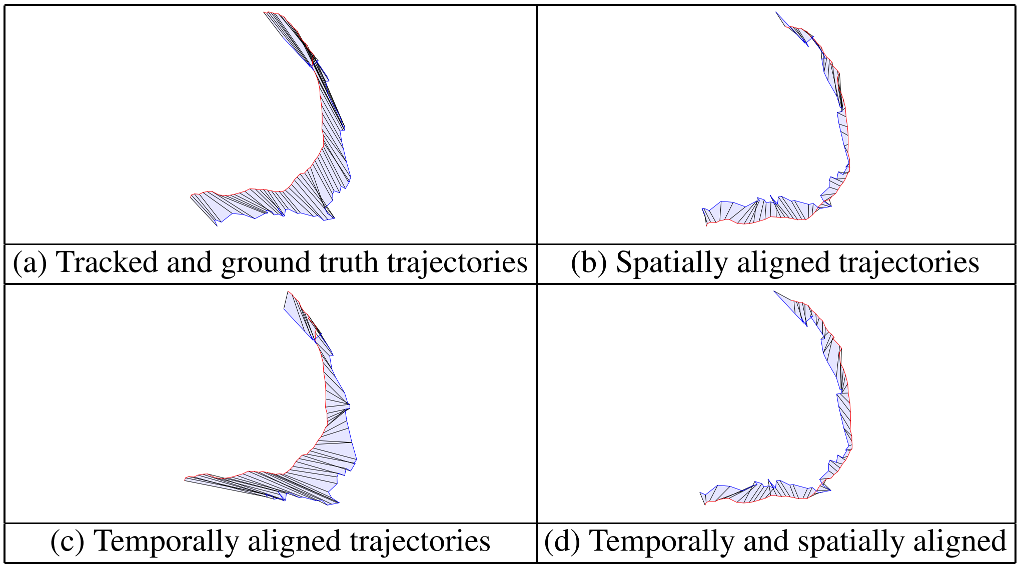
Figure 5.8: Two example hand marked up trajectories, showing the area between them, and the displacements between positions at each time step.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	134	115	0	444	89	56
$\mathcal{D}(T_A + (-55, 74), T_B)$	110	92	10	355	72	42
$\mathcal{D}(T_A, T_B, 0)$	134	115	0	444	89	56
$\mathcal{D}(T_A + (-55, 74), T_B, 0)$	110	92	11	355	72	42

Table 5.1: Results of trajectory evaluation. All distances are in mm.

5.2.2 Sports player tracker example

This section compares a tracked trajectory, T_C, to a hand marked up trajectory T_B. Our sports player tracker identifies the ground plane position of the players, which is taken as the mid-point of the base of the bounding box around the player, and is generally where the players' feet make contact with the floor. Figure 5.9 qualitatively illustrates the shifted trajectories, whilst Table 5.2 quantitatively highlights the systematic error present in this sequence. If T_A is shifted by 500mm in the x-direction, and $600 - 700$mm in the y-direction, the differences between the trajectories fall significantly. This may be due to an invalid assumption that the position of the tracked players is the mid-point of the base of the bounding box around the player. This may be due to the player's shape in these frames, tracker error, or human mark up of the single point representing the player at each time step.



(a) Tracked and ground truth trajectories

(b) Spatially aligned trajectories

(c) Temporally aligned trajectories

(d) Temporally and spatially aligned

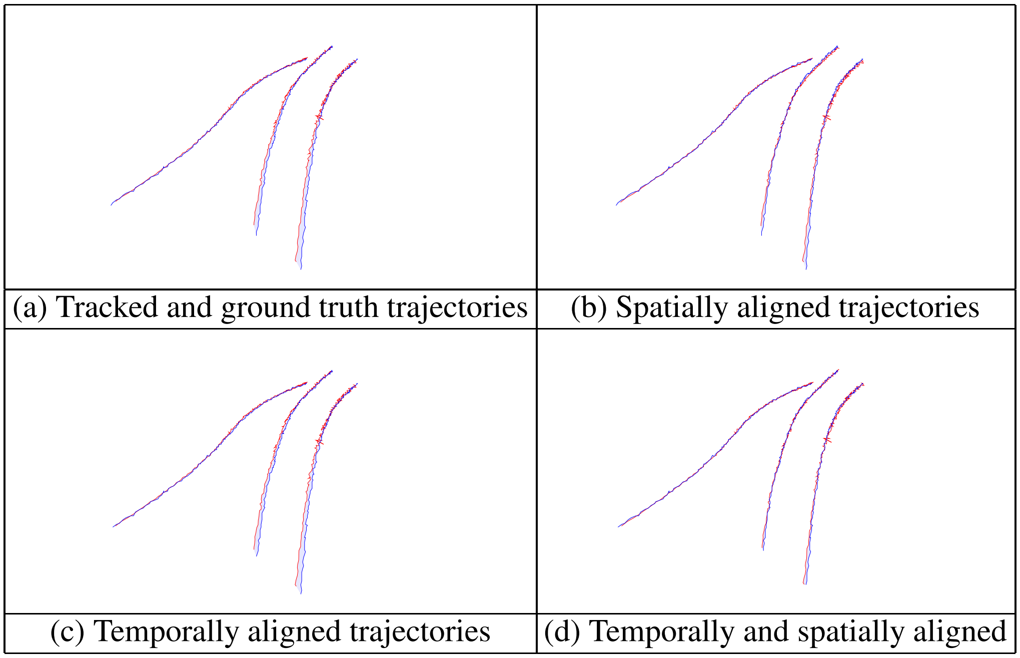
Figure 5.9: (a)-(d) Example trajectories over 70 frames. Trajectory T_C from tracker compared to T_B - the hand marked up trajectory. The figures show the area between them, and the displacements between positions at each time step.

Metric	mean	median	min	max	s.d	'area'
$\mathcal{D}(T_C, T_B)$	890	859	393	1607	267	326
$\mathcal{D}(T_C + (510, -710), T_B)$	279	256	40	67	145	133
$\mathcal{D}(T_C, T_B, -9)$	803	785	311	1428	237	317
$\mathcal{D}(T_C + (551, -618), T_B, -2)$	263	230	52	673	129	138

Table 5.2: Results of trajectory evaluation. All distances are in mm.

5.2.3 Car tracker example

This section compares trajectories from a car tracker [66] with manually marked up ground truth positions. In this example, the evaluation is performed in image plane coordinates (using 352×288 resolution images), on a sequence of cars on an inner city bypass. A sample view is shown in Figure 5.7.



(a) Tracked and ground truth trajectories

(b) Spatially aligned trajectories

(c) Temporally aligned trajectories

(d) Temporally and spatially aligned

Figure 5.10: (a)-(d) Three pairs of example trajectories over 200 frames. Trajectory T_A from tracker compared to T_B - the hand marked up trajectory, with the area between them shaded.

Trajectory comparison is performed on three trajectories of cars in the scene, each over 200 frames in length. Figure 5.10 displays these trajectories along with the ground truth, and Table 5.3 details the quantitative results, from which it can be seen that there is little systematic error in the system, with each car's centroid generally being accurate to between 1 and 3 pixels.

5.3 Summary and conclusions

Quantitative evaluation of the performance of computer vision systems allows their comparison on benchmark datasets. It must be appreciated that algorithms can be evaluated in many ways, and we must not lose target of the aim of the evaluation. Here, a set of metrics for positional evaluation and comparison of trajectories has been presented. The specific aim has been to compare two trajectories. This is useful when evaluating the performance of a tracker, for quantifying the effects of algorithmic improvements. The spatio/temporally separated metrics give a useful measure for identifying the precision of a trajectory, once the systematic error is removed, which may be present due to a time lag,

Left Path

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1.7	1.3	0.1	7.1	1.3	0.6
$\mathcal{D}(T_A + (-0.2, -0.8), T_B)$	1.6	1.3	0.2	6.5	1.1	0.5
$\mathcal{D}(T_A, T_B, 1)$	1.5	1.3	0.1	5.1	0.8	0.6
$\mathcal{D}(T_A + (0.8, -0.1), T_B, 1)$	1.3	1.2	0.1	5.2	0.8	0.5

Middle Path

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	3.0	2.3	0.4	12.4	2.2	1.8
$\mathcal{D}(T_A + (1.9, -0.9), T_B)$	2.3	1.9	0.1	11.2	2.0	0.9
$\mathcal{D}(T_A, T_B, 1)$	2.9	2.3	0.5	8.7	1.4	1.8
$\mathcal{D}(T_A + (3.1, 1.8), T_B, 3)$	1.3	1.3	0.1	3.6	0.7	0.6

Right Path

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	3.2	2.9	0.3	9.7	1.8	2.1
$\mathcal{D}(T_A + (2.3, -0.2), T_B)$	2.5	2.3	0.1	8.6	1.4	1.2
$\mathcal{D}(T_A, T_B, 0)$	3.2	2.3	0.3	9.7	1.8	2.1
$\mathcal{D}(T_A + (2.9, 2.0), T_B, 2)$	1.7	1.6	0.1	6.0	0.9	1.0

Table 5.3: Results of trajectory evaluation. All distances are in pixel units.

or constant spatial shift. There are many potential obvious uses for trajectory comparison in tracker evaluation, for example comparison of a tracker with and without Kalman filtering (clearly this affects any assumption of independence).

An interesting and valuable project would be to develop a system for automatically interpreting the statistics of the position evaluation of a tracking system. These and other metrics could form the basis of such a system that would describe quantitatively and in addition speculate as to the most probable reason for the results, be these temporal lags, spatial displacements, or noisy tracked data. This qualitative description of performance is then useful in identifying areas for improvement in tracking systems.

It is also important to consider how accurate we require a computer vision system to be (this may vary between detection of a target in the scene and precise location of a targets' features). Human mark up of ground truth data is also subjective, and there are differences between ground truth sets marked up by different individuals. If we require a system that is at least as good as a human, in this case, the tracked trajectories should be compared to how well humans can mark-up the trajectories, and a statistical test performed to identify if they are significantly different.

Chapter 6

Multiple camera systems

One of the greatest challenges to multiple object tracking in busy dynamic scenes is occlusion. When one object becomes partially occluded or hidden by another object resolving each objects' position and shape is a challenging task, particularly from a single viewpoint. Incorporating information from multiple views is an approach which should improve the performance of a tracking system.

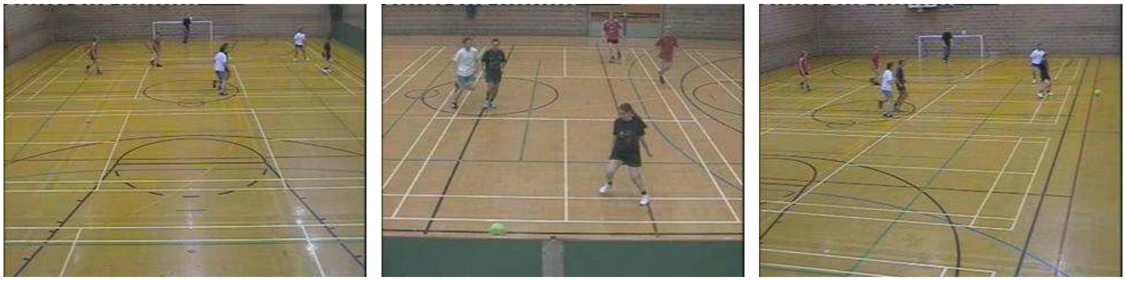


Figure 6.1: Three views of a soccer game at the same instant.

In this chapter, a number of alterations are made to the tracker presented in Chapter 3. The background image is statistically modelled using adaptive mixture models producing an improved foreground segmentation. The template tracking methods introduced in Chapter 4 are incorporated into the tracker as the new fitness function.

Footage from multiple cameras (Figure 6.1) is used in the tracking system. For this numerous issues are identified, and changes are made to the tracker. A common coordinate system must be established in order for transformations between the real world coordinate system and the images from each camera view. A more sophisticated 3D transformation is used to calibrate the images in place of the 2D to 2D ground plane transformation of Chapter 3. The measurement error covariance matrix $\mathcal{R}$ used in Kalman filtering to control the sensitivity of the confidence associated with update measurements, and level of trust of internal estimates, needs to be calculated to take into account the different views from which video streams are taken, since change in 'pixel depth' is much greater along the principal axis of the camera than perpendicular to it.

There are many ways, and at many stages, that information from the video stream can be fused together. Desirable properties of such fusion schema are investigated. Positional evaluation of the tracking system is undertaken using the metrics developed in Section 5; evaluating trajectories from the tracker using multiple video streams of the same event, in comparison to using just a single stream. Finally, difficulties and future possibilities are discussed.

6.1 Adaptive mixture models for image segmentation

Foreground extraction from busy scenes in which objects move at a variety of speeds, shadows are cast, and lighting varies has received much attention over the years. In [99] Stauffer and Grimson present an *adaptive background mixture model*. This computationally efficient method is adopted here, and provides a robust foreground extraction, through modelling the background as a set of Gaussians on a per-pixel basis. Gaussian mixture models (GMMs) are covered in more detail in Section 7.2. Here, the methods for adaptively updating the GMMs in image sequences are discussed.

For each pixel, a mixture of k Gaussians is formed on the RGB pixel data. A weight $w_{k,t}$ is associated with each mixture k and is updated at each time step t:

$$w_{k,t} = (1-\alpha)w_{k,t-1} + \alpha(M_{k,t}) \tag{6.1}$$

where α is the learning rate, and $M_{k,t}$ is 1 for the mixture which matches, and 0 for the remaining mixtures. (A match is found if the pixel is within 3 s.d. of a mixture centre). These weights are then re-normalised, and the mean and variance of the matching Gaussian is adaptively updated with the new observation.

During a short setup period, at each pixel in the image, a set of 4 Gaussians is formed. Using $k = 4$ Gaussians is sufficient to capture any multi-modal distribution of background pixels. Particularly, the areas of floor on which 'glare' from the lighting is sometimes present do not show up as foreground. A set of 25 frames is used to initialise the model, taking every fifth frame. During this five second training phase, a learning rate of $\alpha = 0.1$ is used, allowing the model to adapt quickly to the scene. Once this training is complete, a learning rate of $\alpha = 0.02$ is used. This allows for changes in lighting or camera jitter to be incorporated into the background, yet α is chosen to be small enough that stationary players are not incorporated. Now, a foreground image can be calculated as those pixels which are not within 3 s.d. of any of the mixture centres. The proportion of the data that should be accounted for by the background model has been set to $T = 0.8$ in this work. Further details and discussion about the 'Grimson method', as it has become known, can be found in [99]. Figure 6.2 shows the foreground pixels identified in an image of a 5-a-side soccer game.

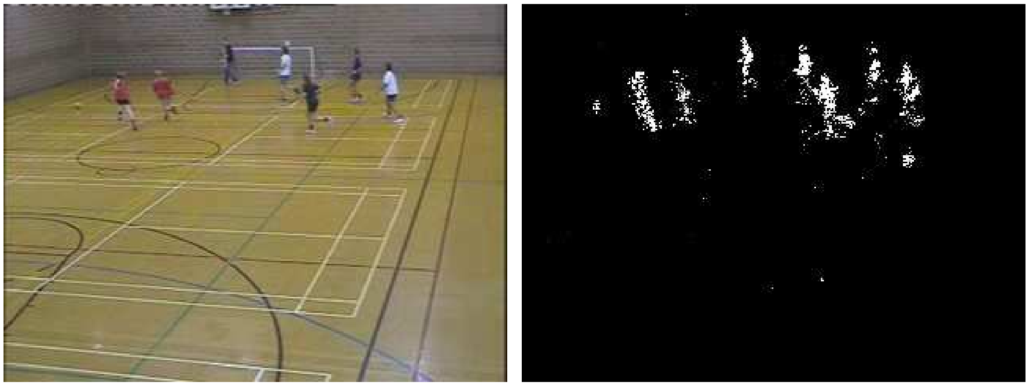
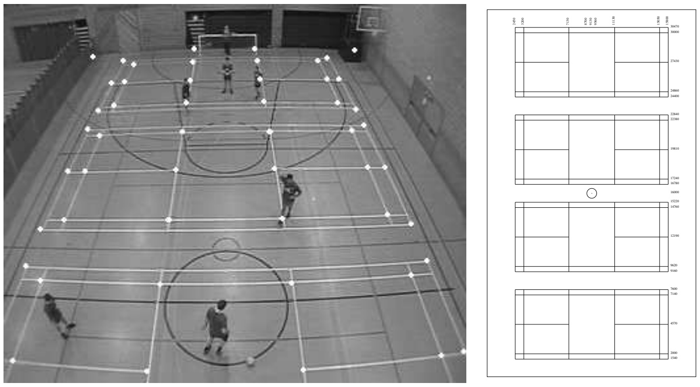


Figure 6.2: An example scene and corresponding foreground image.

6.2 Establishing a common coordinate system

To improve the camera calibration, a 3D calibration is performed. Unlike the system used in Chapter 2, this involves using correspondences between many points on the image and in the real world. A 3D calibration allows not only points on the ground plane (x, y) to be identified on an image (u, v), but also for three-dimensional points (x, y, z) to be identified on an image (u, v). This now allows for (u_{feet}, v_{feet}) and (u_{head}, v_{head}) on the image to be identified from the real world positions $(x, y, 0)$ and $(x, y, z_{player\ height})$ where (x, y) is the ground plane position of the player.

Calibrating each of the images from the different views allows points to be transformed between points in one image, points in the real world, and the corresponding points in another image of the same scene. The methods of Tsai [109] are employed to calibrate the images. These methods perform a perspective projection, using a pin-hole camera model. Many pairs of (image,world) points are needed to calibrate the image, and should be well spread over the image. As many points as are identifiable in the scene are used, typically around 50. Figure 6.3 shows an example image used for calibration, where the 51 points used in the calibration have been superimposed on top. The specific location from which the footage is taken is ideal for calibration, since there are many markings on the sports hall floor which, once measured, can easily be identified on images of the scene from any view. Without such points to use, it may be necessary to construct a large calibration frame. This convenience is not really contrived, especially in the sports domain.



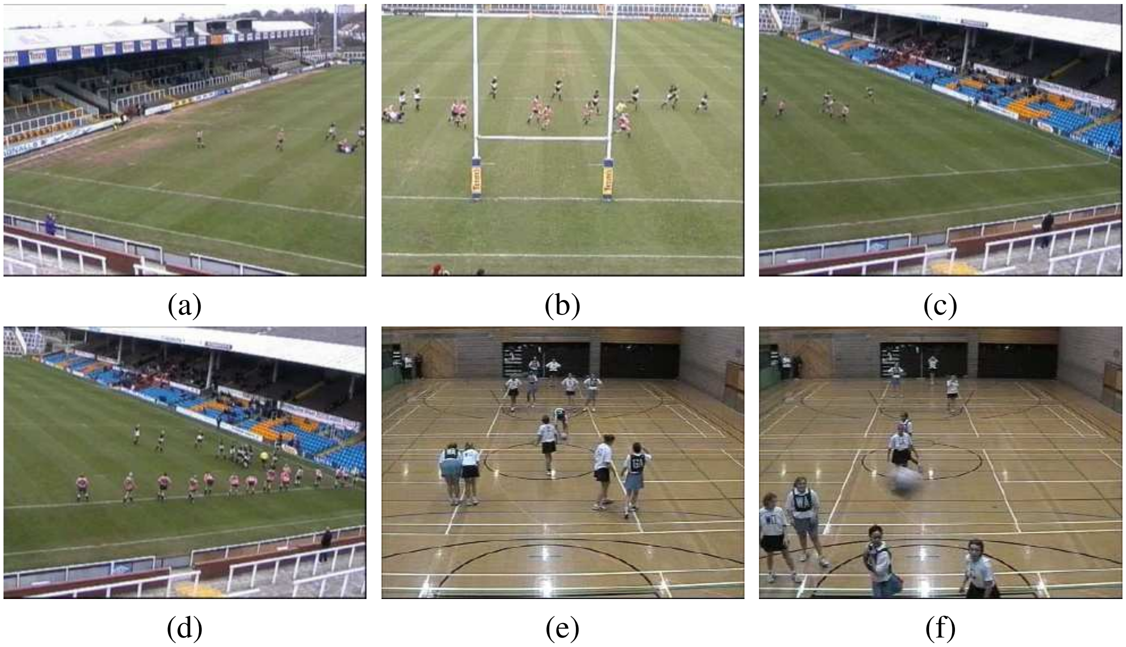
(a) (b)

Figure 6.3: (a) Calibration image: showing the point correspondences. Footage taken with a fish-eye lens. (b) Showing measured features of the badminton court lines on the sports hall floor, used for calibration.

6.3 Camera setup and view choice

Experimentation with *fish-eye* lenses, to allow coverage of a larger area of the pitch, and mounting cameras as high as possible, has mixed results. The field of view of a conventional camera is not wide enough for the whole of the pitch to be covered, even when placed several metres high in the rafters of a sports hall roof. Figure 6.3 shows an example image taken with a camera fitted with a wide angle conversion ×0.7 fish-eye lens. The camera calibration provides an accurate real world to image coordinate transformation, successfully dealing with the radial distortion effects that appear in the image. Image plane errors on the calibration set for the fish-eye lens example were found to have a mean of 1.5 pixels, compared to 0.9 pixels with a normal camera lens. Similarly, object space errors were found to be roughly twice as large, yet still acceptable, with a mean of 87mm compared to 45mm. Such footage however, did not work well with the template tracking scheme, since player's shapes varied considerably; those near the camera were viewed from almost above, with mainly heads and shoulders visible; those in the middle covered the most pixels and the players looked similar to those in the traditional view used from the balcony with a normal lens; and those furthest from the camera occupied very few pixels. For this reason, video captured using a fish-eye lens was not used further in the tracking experiments.

Ideally, using an overhead camera pointing straight down from very high above would remove almost all of the problems encountered. The whole pitch would be in the field of view of a single camera, and each player (seen from above) would be represented by a distinct 'blob' disconnected from all the other players. This, however, is impractical; the ceiling is too low and any tracker developed would not be extensible to use in outdoor situations. By choosing a camera angle similar to that used in broadcast football, there is the possibility of extending a successful tracking system to cope with an outdoor 11-a-side football game, covered by multiple cameras.



(a) (b) (c)

(d) (e) (f)

Figure 6.4: (a)-(c) Three views of a rugby game to cover the whole pitch. (d) Rugby viewed from the side is problematic for computer vision approaches, since the players frequently line up across the pitch, creating many instances of multiple occlusion. (e) Netballers mark each other very tightly; the players appear to move in pairs. (f) Consideration should be given to how likely the ball is to come into contact with the camera.

Other sports such as rugby and netball exhibit radically different characteristics and thought needs to be given as to suitable camera locations. Figure 6.4 (a)-(c) shows three views of a rugby game needed to cover the whole pitch (with possibly three more at the other end for higher resolution images of that half of the pitch). Rugby games (of both codes) involve the teams frequently lining up across the width of the pitch. Viewed from the side it is hard to distinguish separate players. Figure 6.4 (d) gives an example of this, viewed from the end. If the cameras were at the side of the pitch it would be difficult to extract any useful information. Viewed from the end of the pitch, there is more spatial separation between the players in the image, and this aids machine vision algorithms.

Netball is played in a fascinatingly different way to soccer. Both games are similar; two teams play on a confined pitch with a round ball with a goal at each end, and aim to score as many goals as possible. However, netballers mark their opponents *much* closer than in soccer. Typically, in every image, each player will be next to their opposing player (Figure 6.4 (e)). This presents many cases of occlusion and not just from one side; the players will overlap on images from many viewpoints. A final consideration for camera placement is to locate the camera in a position where it is safe from being hit by a ball. Figure 6.4 (f) shows the ball coming close to the camera during a netball game.

6.4 Estimating measurement error covariance

Kalman filtering is performed using the ground plane positions of the players. The measurement error covariance, $\mathcal{R}$, varies depending where on the image the point is. Filtering performance is improved when $\mathcal{R}$ changes with respect to where the player is on the ground plane. Figure 6.5 illustrates how the equal-sized chessboard squares on the ground plane appear to be of very different sizes on the image.

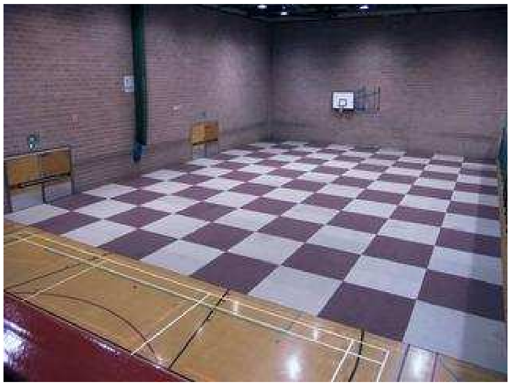


Figure 6.5: Sports hall carpeted with 2 metre squares.

Closer points can be identified with a greater accuracy than those further away, which are at a lower resolution. From this type of view, of a large scene, a pixel at the front may represent 0.003m of the ground plane, whereas far away it may represent 0.5m.

On the image plane the bounding box around the player is represented by an image position (u, v), a height h, and a width w. By empirical observation, and supported by the analysis of Section 4.2.4, it has been found that on the image the template correctly fits to the true position of the player with a margin of error of up to a quarter of the width of the bounding box horizontally about the player's position, and vertically from the player's position by up to ten percent of the height of the bounding box (Figure 6.6). Thus if (u_0, v_0) is the midpoint of the baseline, then:

$$\begin{aligned} u_0 - \delta u < u < u_0 + \delta u \\ v_0 - \delta v < v < v_0 + \delta v \end{aligned} \tag{6.2}$$

where $\delta u = 0.25w$ and $\delta v = 0.1h$.

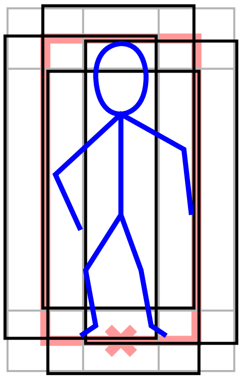


Figure 6.6: Illustration of variability in a typical bounding box fit on a player.

Building upon the reasoning in Section 3.3, the measurement error covariance matrix $\mathcal{R}$ represents the covariance of errors of *measuring* the position (as opposed to the process error, $\mathcal{Q}$, which measures the uncertainty that the mid-point of the base of the bounding box represents the player). $\mathcal{R}$ describes a symmetric ellipse or a symmetric bivariate Gaussian distribution of the deviations about the u- and v-axes. On the image plane the measurement error covariance matrix takes the form:

$$\mathcal{R}_{image} = \begin{bmatrix} (\delta u)^2 & 0 \\ 0 & (\delta v)^2 \end{bmatrix} \tag{6.3}$$

$\mathcal{R}_{image}$ varies over the image and is dependent on where the player is, and the size of their bounding box. Unfortunately this covariance matrix cannot be transformed directly to the ground plane, due to the non-linear image plane to ground plane transformation being used. The ground plane measurement error covariance matrix at any point (x, y) can be estimated by transforming to the image coordinate (u, v), and taking four points on the image plane; $(u + \delta u, v), (u - \delta u, v), (u, v + \delta v), (u, v + \delta v)$ which represent the extremities of acceptable deviation from (u, v). Transforming each of these to ground plane positions (x_i, y_i) for $i = 1, \ldots, 4$ respectively, allows $\mathcal{R}$ to be estimated as:

$$\mathcal{R} = \begin{bmatrix} \frac{1}{4}\sum_{i=1}^{4}(x_i - x)^2 & \frac{1}{4}\sum_{i=1}^{4}(x_i - x)(y_i - y) \\ \frac{1}{4}\sum_{i=1}^{4}(x_i - x)(y_i - y) & \frac{1}{4}\sum_{i=1}^{4}(y_i - y)^2 \end{bmatrix} \tag{6.4}$$

This seems a little crude, yet provides sensible results. The uncertainty in the measurement (units in millimetres) increases as players are further from the camera, where each pixel represents a greater area of possible ground plane. Testing on a typical image gave:

$$\mathcal{R} = \begin{bmatrix} (144)^2 & (50)^2 \\ (50)^2 & (194)^2 \end{bmatrix} \quad \mathcal{R} = \begin{bmatrix} (212)^2 & -(65)^2 \\ -(65)^2 & (346)^2 \end{bmatrix} \quad \mathcal{R} = \begin{bmatrix} (226)^2 & (300)^2 \\ (300)^2 & (700)^2 \end{bmatrix}$$

(a) at the front (b) near the middle (c) near the back

(6.5)

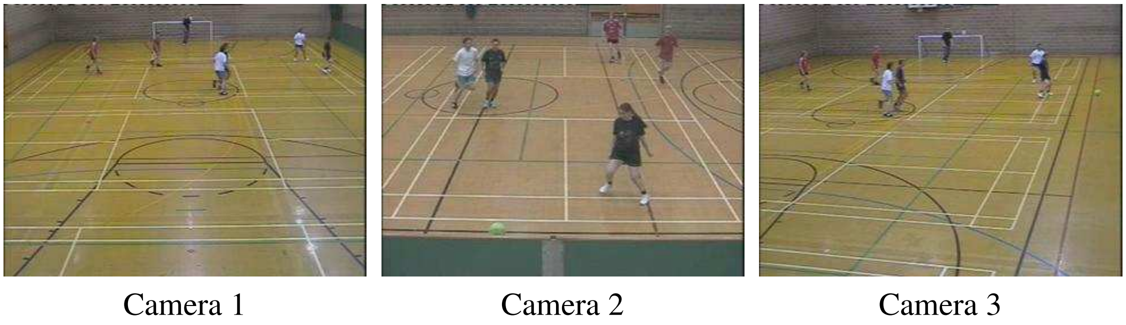
The current tracking method involves updating a Kalman filter for each player at each time step, and using an estimate of the next position to predict the new location of the player. For each Kalman filter, $\mathcal{R}$ is calculated using the above procedure with (u, v), w and h being the values of the sample for each player, within the selected ‘best’ sampleset.

6.5 Information fusion for multiple view tracking

The methods employed in Chapter 3 can be extended to incorporate additional information from multiple views. Given multiple views of the pitch, such as those in Figure 6.7, there will be some regions of the pitch on which each player will be visible from 0,1,2, … ,k views, where k is the maximum number of views available, as illustrated in Figure 6.8 for $k = 3$.

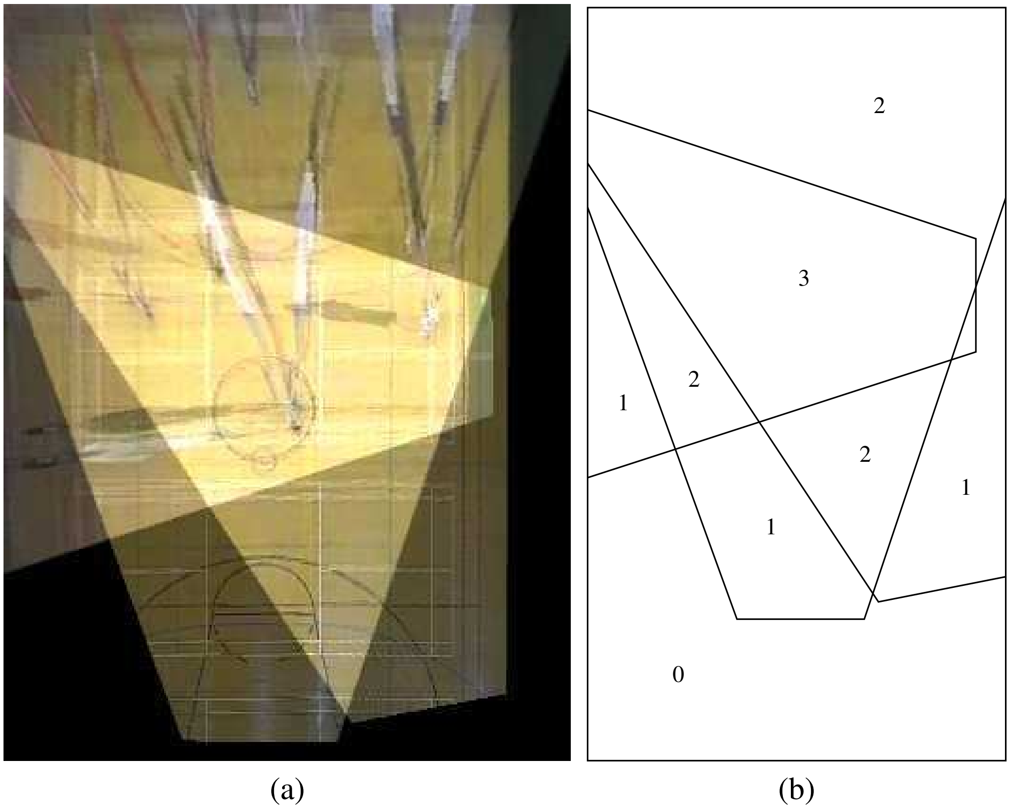
The aim is to track each player in as many views as possible, to gain accurate positional information, as opposed to using multiple cameras to cover the whole field in order to track players from a single view and then follow them between views.

The framework for multiple object tracking using multiple cameras adapts the framework introduced in Section 3.2. Again, the *structure* of the samples, and how they make



Camera 1 Camera 2 Camera 3

Figure 6.7: Three views of a soccer game at the same instant.



(a) (b)

Figure 6.8: (a) Rectified View of the three images in Figure 6.7. Illustrating the regions of the ground plane in which one, two or three views of a player are visible. (b) Diagram labelling each of the different regions with the number of views it is visible from.

up samplesets for use in a CONDENSATION based tracker is discussed. Then the *propagation* of the samplesets, the application of the *predictive dynamics*, and the evaluation of the *fitness* (using the multi-resolution template scheme of Chapter 4) of the samples to the image information are discussed.

6.5.1 Structure

Given k views of a scene, it is possible to represent the set of p players in a sampleset containing p samples of the form $(x, y, H, h[k], w[k], id)$ where H is the player's real-world height, $h[k]$ and $w[k]$ are arrays of height and width dimensions of the bounding box for each view, and id is a player identity label. Each of the cameras is calibrated, allowing (x, y), the ground plane position, to be projected using a perspective projection onto each image j, at (u_j, v_j). Thus, only a single estimate for a player's position is needed in each sample, rather than using several image positions.

6.5.2 Propagation

Using the new structure, the samplesets can be propagated in the usual way, as described in Section 3.2.2.

6.5.3 Prediction

The same predictive dynamics as in Section 3.2 are used, although now each Kalman filter must be updated with the value of the player's position from the 'best' sampleset, and use a measurement error covariance, $\mathcal{R}$, which reflects that the measurements have been taken from multiple views.

The method described previously to estimate $\mathcal{R}$ for a player in a single view, can be applied from each view giving estimates for the measurement error, $\mathcal{R}_{i,j}$, for each player i in view j. A Boolean variable $\rho_{i,j}$ is used to identify if player i can be seen from view j.

$$\rho_{i,j} = \begin{cases} 1 & \text{if player } i \text{ is visible from view } j \\ 0 & \text{if player } i \text{ is not visible from view } j \end{cases}$$

Now $\mathcal{R}_i$, the measurement error covariance of player i, can be calculated by convolving all valid estimates of the bivariate Gaussian distributions that the covariances represent, i.e. those for which the player is in view. Details of the convolution can be found in Appendix C, giving:

$$\mathcal{R}_i = (\Sigma_j \rho_{i,j} \mathcal{R}_{i,j}^{-1})^{-1} \tag{6.6}$$

6.5.4 Fitness function

A major change is needed to the way in which the sampling probability of a sampleset is calculated. The fitness scores for each player can be calculated using the multi-resolution templates (method C) learned in Section 4.2. Let $t_{i,j}$ be the fitness of player i in view j found by this method, i.e. the highest response from convolving a sub-sampled image within the bounding box of the player with each of the five template kernels. In addition, the scores for the player from each view must be combined to create a fitness score t_i for each player i. There are several ways that this can be done.

The fitness scores *could* be combined in a way which implies that evidence from an extra view always increases the player's fitness score t_i. For example, let $t_{i,j}$ be the fitness of player i on image j, then one method for combining the scores, would be: $t_i = 1 - \prod_j (1 - t_{i,j})$. However, this has the drawback that when a supplementary view contains information indicating the absence of a player from a given location, then the player's fitness t_i will not be decreased.

A more sensible scheme involves devising a system in which evidence from an extra view suggesting that there is no player at the given position leads to a reduction in the associated score for the sample, and if it provides confirming evidence, then the score for the sample is increased. Figure 6.9 illustrates a property of the two schemes. This second approach is taken in combining information from multiple views and is now detailed further.

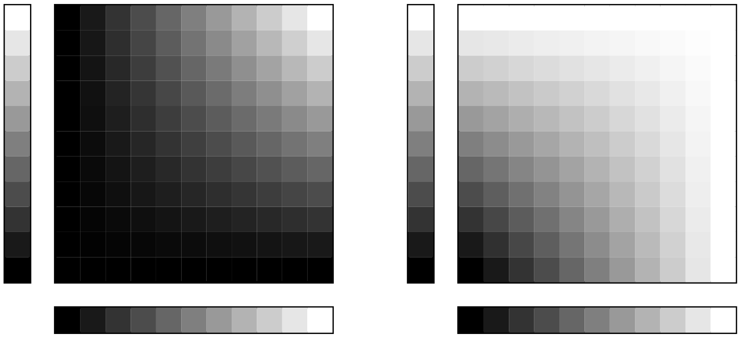


Figure 6.9: Abstract representation of two ways of combining two scores. Black = 0 (poor), White = 1 (good). The graphic on the left represents the combination scheme in use; two good scores combine to create a good score, a good and a poor score become an average score, and two poor scores remain a poor score. The graphic on the right represents an alternative combination scheme discussed and rejected; two good scores combine to create a good score, a good and a poor score become a good score, and two poor scores become a slightly better poor score.

Let $t_{i,j}$ be the fitness of player i on image j, as found by the method in Section 4.2. Firstly, the average score μ_i over all views in which the player is visible can be calculated.

$$\mu_i = \frac{\sum_{j=1}^{n} t_{i,j}}{\sum_{k=1}^{n} \rho_{i,k}} \tag{6.7}$$

For those views from which the player is not visible, the player's fitness score is now set to the average score for that player from all remaining views.

$$\text{if } \rho_{i,j} = 0 \quad \text{then set } \ t_{i,j} = \mu_i \tag{6.8}$$

Now the fitness t_i of player i is calculated as the average of the squared differences between the mean μ_i and each fitness score $t_{i,j}$ (for player i from view j), taken away from one. This represents an 'agreement factor' between the views (if they are all the same, then it will equal one, otherwise, it will be less than one, equal to three-quarters at the limit of the disagreement between views). This is then multiplied by the product of the fitness scores from the different views. If one of these is poor, it decreases the overall fitness score t_i:

$$t_i = (\prod_{j=1}^{n} t_{i,j})(1 - \frac{1}{\sum_{k=1}^{n} \rho_{i,k}} \sum_{l=1}^{n} \rho_{i,l}(t_{i,l} - \mu_i)^2) \tag{6.9}$$

The t_i's can be combined as before to give a sampling probability for the sampleset. It should be noted that $0 \leq t_{i,j} \leq 1$ and hence $0 \leq t_i \leq 1$. Evidence from an extra view may increase or decrease the score t_i associated with there being a player at the given position. If the extra view provides evidence that there is no player at the given position, it will lower the score associated with a sample, and if confirming evidence is provided then the score for that sample will increase. When player i is only visible in view j this procedure results in $t_i = t_{i,j}$ as expected.

6.6 Evaluation

The three views of a soccer game (Figure 6.7) were used for tracking. A comparison between tracking from a single view and two views is now presented. A set of six players are tracked for a short period of time from (i) Camera 1, (ii) Camera 3, and (iii) Cameras 1 and 3. Figures 6.10 and 6.11 show the tracker in action on the two video streams (scenario (iii)). The bounding boxes around each player are shown by the set of blue boxes. On each pair of images the mid-point of the base of the bounding box is at the same real world location.

Figure 6.10 shows 16 frames of tracked footage from cameras 1 and 3. The tracking appears to be working well and agreement as to player's positions between camera views

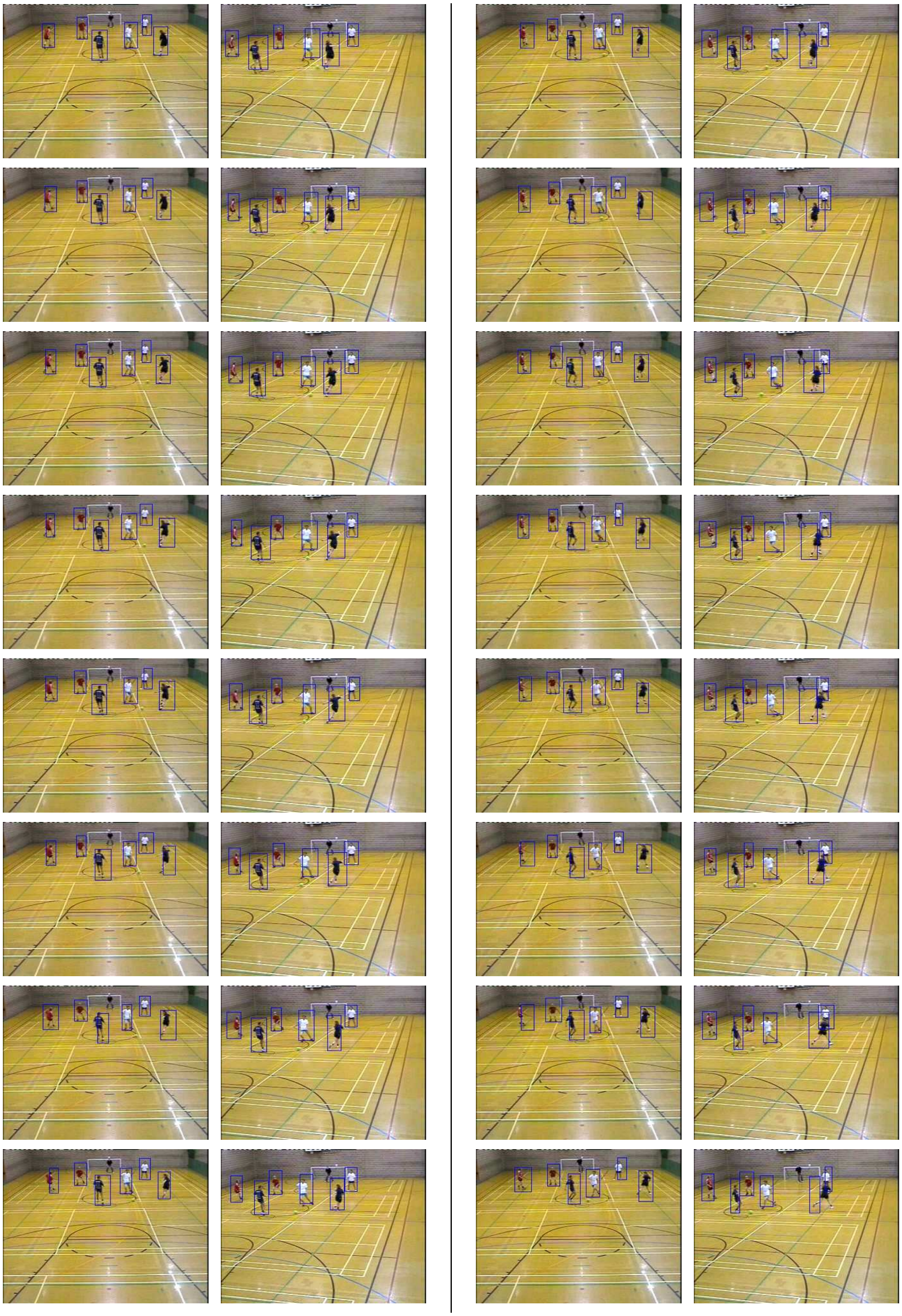


Figure 6.10: Tracked footage.

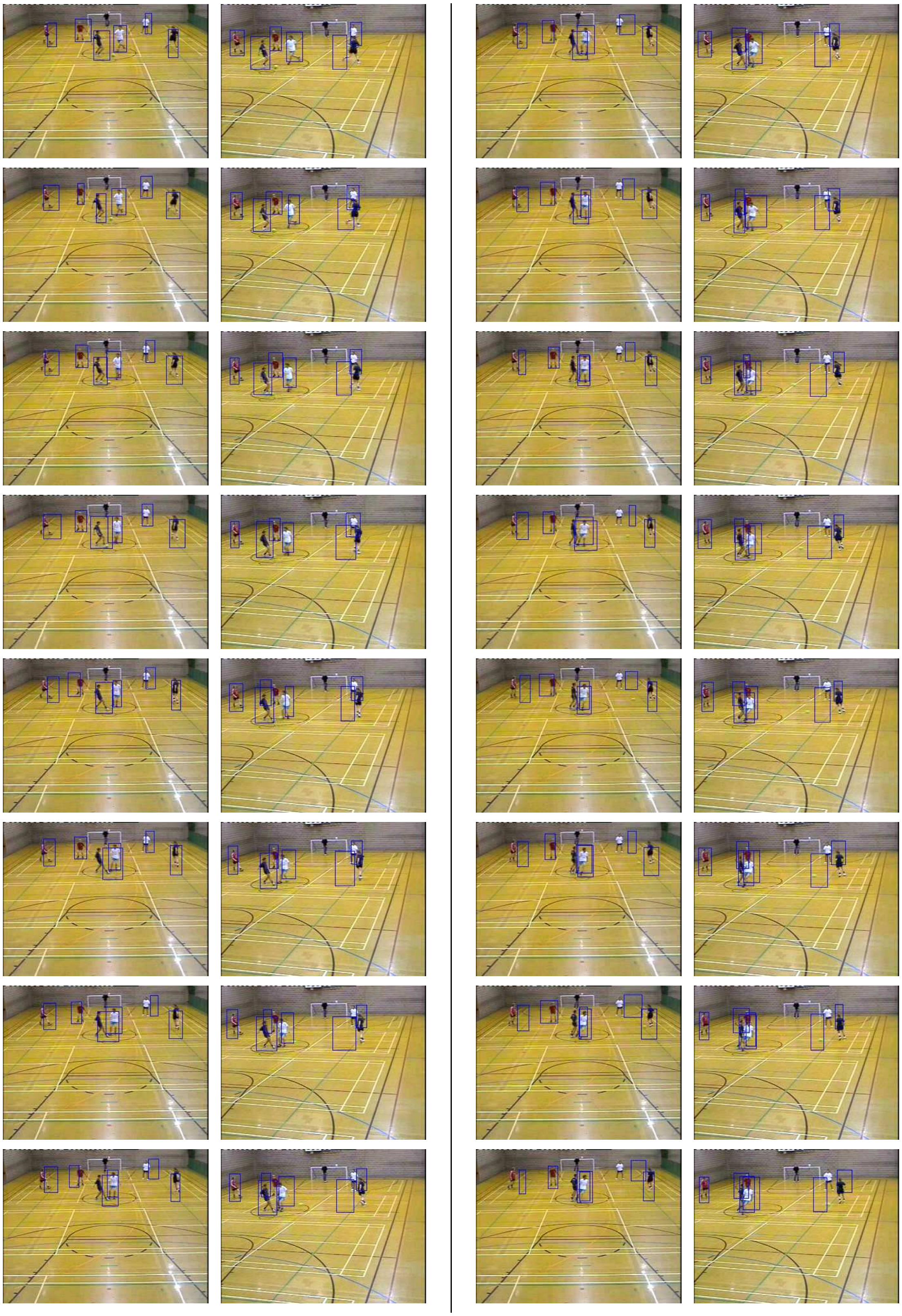


Figure 6.11: Tracked footage (2).

is good. In Figure 6.11 the tracking system begins to track less successfully; first, the box around a player fails to keep up with the player as the player moves up the pitch, this in turn causes the bounding box to fit to half of the player in one view (camera 1) and to fall to the side of the player, missing the player completely, in the other view; secondly, the two players in the centre get very close, though from Camera 3 it can be seen that the player in white is still tracked separately to the other player. By the end of this short clip, it can be seen that the tracker is not able to successfully continue tracking to a reliable degree of accuracy.

The performance evaluation metrics and statistics for positional tracking systems developed in Chapter 5 are applied to each player in each of the three experiments. Tables of these results are presented in Appendix D. There does not appear to be a significant difference between the three experiments. Combining information from multiple views in the way discussed above does not appear to significantly improve the tracking results. Figure 6.12 shows the trajectories for the six players in each of the three experiments, and compares them to the ground truth trajectory obtained by manually marking up the players from cameras 1 and 3, and averaging.

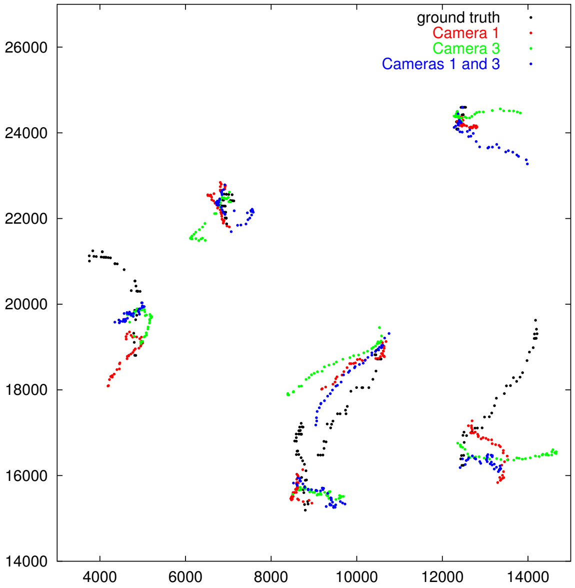


Figure 6.12: Trajectories from experiments (i), (ii) and (iii) on 6 players, compared to the ground truth.

Looking in more detail, it can be seen that the system tracks player 4 better when using both views. The average distance between the trajectories and the ground truth is reduced from 793mm (Camera 1) and 835mm (Camera 3) to 505mm when both cameras are used for tracking. The x-component of a spatial shift to optimally align the trajectories with the ground truth when both cameras are used is close to the average of when the cameras are used separately $((347 + -173)/2 = 87 \approx 82)$. The system appears to track player 1 less well when using two cameras. The average distance between the trajectories and the ground truth is increased from 1016mm (Camera 1) and 1083mm (Camera 3) to 1161mm when both cameras are used for tracking. The y-component of a spatial shift to optimally align the trajectories with the ground truth when both cameras are used is similar to that of the two single cameras, and the x-component resembles more closely that of Camera 3. Here the temporal aspect of the metrics is not particularly useful, however, it would be more useful on longer trajectories, or those misaligned in time.

6.7 Discussion

At the beginning of this chapter it was hypothesised that 'incorporating information from multiple views is an approach which should improve the performance of a tracking system'. In Section 6.6 examples of better and worse tracking when using two views over one are shown.

The main pitfalls of the approach used appear to be in the choice of fitness function. A 2D fitness function struggles to capture the 3D pose of a human player. If, from a single view, the tracker fits the bounding box (determined by the fitness function) slightly below the players feet, then it may track reasonably successfully for a period of time, always tracking a bit below the player (this may lead to the trajectory consistently being half a metre away). If two views are used, then this half metre out in one view may be significant enough to miss the player completely in another view. This may be one cause of poor multiple view performance. This may be reduced or eradicated by modelling the player as a 3D object; replacing the bounding box with a cuboid, generalised cylinder, multiple connected cylinders or skeletal model. This reason should also begin to account for the poor performance of the multiple camera tracking system when Camera 2 was used with either of the other views. This camera is setup orthogonal to Camera 1, and it was expected that this pairing would give the best results.

Combining or fusing information from multiple views needs to be investigated further. Here, a system in which information from an extra view always increased fitness was found not to work at all. The system which is used allows information from an extra view to increase or decrease the fitness, and is compatible with scores from players which are visible in a variety of number of views. It would be useful if a system could be devised in which the reliability of the fitness in each view could also be determined. For example, tracking is generally more reliable in a direction perpendicular to the camera axis. In Section 4.2.4 the template kernels were shown to able to identify more reliably the location of the target player in the x-direction (perpendicular to the camera from this view) than in the y-direction (along the principal axis of the camera in this case). The method used to estimate the measurement error covariance matrix $\mathcal{R}$ used in updating the Kalman filters does to some extent take this into account, though explicitly using this in a fusion schema would be worth investigating.

6.8 Summary

This chapter has built upon the tracking framework developed in Chapter 3, improving the background segmentation scheme to aid subsequent machine vision algorithms, in particular the multi-resolution templates used within the fitness function, and calibrating the images to allow for a common coordinate frame to be used, so that a player can be represented as being at a single real world position. This has allowed extension to tracking from multiple views. Camera positioning is found to be an important aspect in obtaining usable footage and depends on the sport that is being covered. Methods of fusing data for tracking from multiple views has been discussed and results of tracking multiple players from multiple views has been evaluated on a short sequence and found not be significantly different to single view tracking results.

Chapter 7

Team game interactions

Capturing the behaviour of a set of sports players would allow many exciting activities to be performed; identifying tactics, predicting future movements, recognising set-plays, identifying teams, and evaluating teamwork. An advanced model of the interactions of players may allow for speculative tactical attacking moves to be simulated by a team and for probable responses for the defending team to be generated. A behaviour model to aid in predicting movements of sports players in a tracking system is the ultimate goal. From a cognitive viewpoint, a model which could learn the rules of the game would be very powerful indeed.

Throughout this work, the aim has been towards learning behaviour from real world data, and Chapters 3 to 6 have concentrated primarily on extracting positional data from video streams of soccer team games. There are many approaches which may be taken when analysing a team game. At one end of the spectrum, it is possible to recognise set-plays from a set of player trajectories [46]; this task involves comparing a sequence of positions to several 'hand-crafted' set-plays and classifying whether one of the predefined set-plays has occurred. At the other end of the spectrum is a logical and cognitive strand of thought which would undoubtedly need to be explored and utilised in order to begin to solve the *really* hard tasks such as learning the rules of the game purely from positional or visual evidence. There are also many statistical methods which fit at various points on this spectrum. Supervised statistical methods may be used to group data or to learn a particular aspect of play, through training on a set of examples in which the play occurs. Unsupervised statistical methods may be used to cluster or characterise the data in some way, without being supplied with any prior knowledge in addition to the data.

This chapter discusses the collection of data for behaviour modelling, and creates probability density maps of real world data from a 5-a-side soccer game using a Gaussian mixture model, and secondly using vector quantisation. A system for generating synthetic data (with endless supply) is designed to provide a greater quantity of data with which to work. Emergent methods are used on this data to identify players' positions, and approaches to a variety of tasks involving behaviour modelling and interactions are discussed.

7.1 Data collection

Automatic tracking of multiple objects interacting in large, busy domains with considerable occlusions (both in frequency and in the number of objects occluding) has presented many challenges to visual tracking algorithms and is recognised as beyond the capabilities of current state-of-the-art trackers. For the purposes of modelling the behaviour and interactions of the players in a soccer game, a section of footage has been marked up manually. This provides data with which to work productively and which is sufficiently reliable.

A multiple camera system has been used to capture the whole area of the pitch, and the cameras were calibrated in an identical way to that described in Section 6.2. Each of the ten players has been identified in each frame in which they are visible from two video streams. The positions were then averaged between the views, for those times when the player was in the field of view of more than one camera. Figure 7.1 shows the marked up positions of players.

The location and velocity of the ball is a large factor in influencing the movements of sports players, along with where the rest of their team are, and where their opponents are. The ball is relatively small and moves very fast. It is often not in contact with the floor, travelling in a complex 3D trajectory through the air. (When spin is applied to the ball it deviates from a parabola, and its trajectory is no longer confined to a 2D plane.) This makes it harder to determine its position in the real world; if it is possible to reliably and robustly identify the ball in the image plane, then it is still not straight forward to relate this to the ball's true location, and errors associated with this may be unacceptably large. In the footage of soccer games used it is often (indeed for the majority of the time) not possible for a human observer to identify the ball when shown a single frame of the video. For these reasons, tracking or manual markup of the ball has not been undertaken in this work.

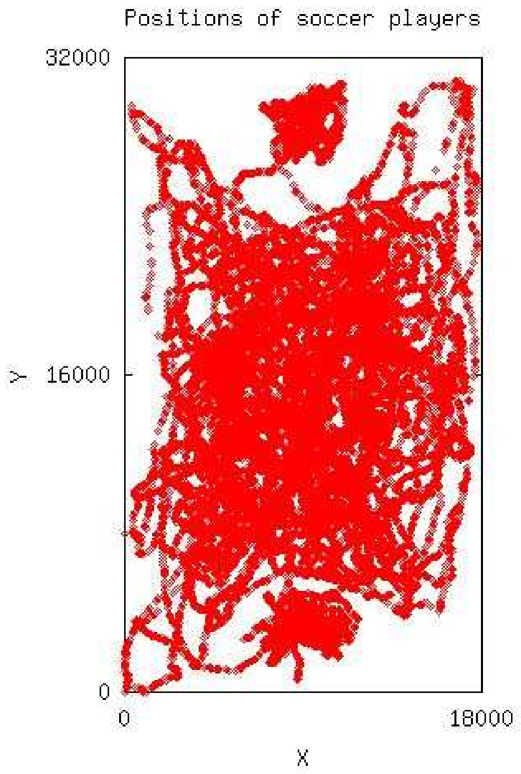


Figure 7.1: Representation of hand marked-up players' positions over the soccer pitch. Units are in millimetres.

Others have tried various approaches to the task of tracking the ball, notably; Kim uses a single moving camera, physics-based constraints (parabolic trajectories) and the shadow cast by the ball in soccer games [59] (though constrained to a motion in a single vertical plane); Ohno tracks the ball using colour and motion assuming there are no players around the ball [75] (occlusion of the ball by players in real footage is common); and Pingali uses sets of pairs of cameras for tracking a tennis ball using monochrome 60Hz cameras [82]. The first two of these are reported to have only been evaluated in a single experiment. The tennis ball tracking has been used in a number of broadcast tennis games to provide ball speeds and virtual replays. However, currently it is unable to handle player-ball interactions and occlusions, with which the other two systems also struggle. Ball tracking in the soccer domain is an open research topic in its own right.

7.2 Probability density estimation

Estimating the probability density of the positions of the players may provide additional information for the propagation of samples in the CONDENSATION tracking scheme. A simple probability density may be obtained by constructing a histogram of frequencies of players being in one of a number of bins that the pitch has been divided into. However, this lacks sophistication, and has the potential to be over-sensitive to the training data, creating a potentially unsmooth probability density function.

Multivariate data can be well represented for density estimation purposes by a Gaussian mixture model (GMM). Taking a set of M Gaussian distributions to represent the data (where $M \ll N$, the number of data items used for training), with appropriately chosen parameters, allows a PDF to be created. Adopting the notation of Bishop [8], the PDF can be written as a linear combination of the contributions from each Gaussian:

$$p(\mathbf{x}) = \sum_{j=1}^{M} p(\mathbf{x}|j)P(j) \tag{7.1}$$

where the mixing parameters $P(j)$ are the prior probabilities that the data point $\mathbf{x}$ has been generated from the jth Gaussian, and:

$$p(\mathbf{x}|j) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}\left(\mathbf{x}-\mu\right)^{T}\mathbf{\Sigma}^{\mathbf{-1}}\left(\mathbf{x}-\mu\right)\right\} \tag{7.2}$$

with $\mathbf{x}$ the data of dimensionality d, mean vector μ, and covariance matrix $\mathbf{\Sigma}$.

The *Expectation-Maximisation* (EM) algorithm is commonly used to estimate the parameters of the set of Gaussians which best represent the data. This iterative solution to estimating the Gaussians' parameters involves maximising the likelihood ($\mathcal{L}$) of the solution, given the data set. Equation 7.3 gives the negative log-likelihood used as an error function for the data set $\mathbf{x}^n$.

$$E = -\ln \mathcal{L} = -\sum_{n=1}^{N} \ln p(\mathbf{x}^n) = -\sum_{n=1}^{N} \ln \left\{ \sum_{j=1}^{M} p(\mathbf{x}^n|j)P(j) \right\} \tag{7.3}$$

Training a GMM on the marked up data of the soccer game creates the density estimate map illustrated in Figure 7.2. It shows a two dimensional mixture of 100 Gaussians, after 1000 iterations of the EM algorithm on a set of 48771 positions. This gives a good representation of the density of the players' positions. For any position, the contribution of each Gaussian can be identified; however, for the purposes of state representation and exploring the transitions between states, a GMM may not the best approach, since the

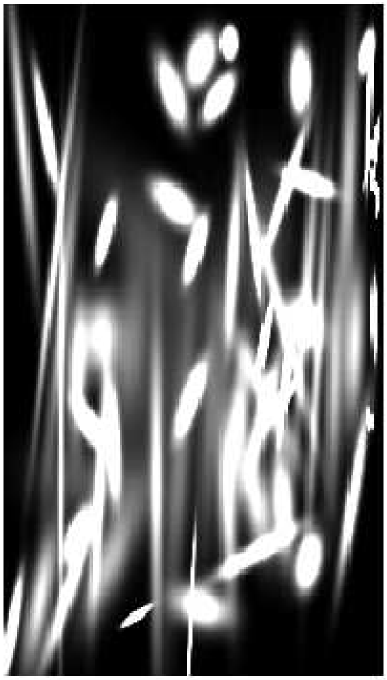


Figure 7.2: Probability density of players' positions over the soccer pitch, represented by a GMM with 100 components.

Gaussians do not correspond to different states, and many mixture components contribute to the probability for a position. A scheme which identifies more meaningful states is desired.

Incorporating the velocities of the players creates a GMM that is hard to visualise in 4D, though may be of use as part of a more complex scheme.

7.3 Vector quantisation

If a state based model is to be employed for modelling the system of players, then a method that can separate examples into one of a distinct number of states is preferable to one that will identify the contribution from a set of states, as may be done with the GMMs above.

Vector quantisation (VQ) is commonly used for data compression. The aim is to represent a large set of N d-dimensional vectors as a small set of *codebook* vectors, minimising the reconstruction error (detailed below).

The implementation used here to perform VQ is that of Johnson [52, 54]. A competitive learning neural network is formed, which learns in an unsupervised manner. The neural network has k output nodes; one to represent each prototype $\mathbf{m}_i$, and d input nodes; one for each dimension of the input training data.

The VQ algorithm

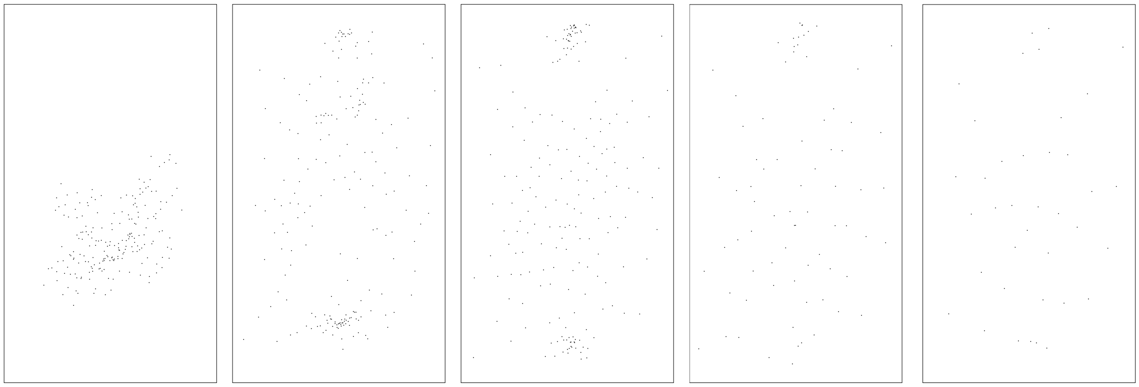
- Randomly place the k prototypes $\mathbf{m}_i$ $(i = 1, \ldots, k)$ in the feature space.
- Let $\mathbf{x}(t)$ be the feature vector for epoch t.
- Define $\alpha(t) = 1 - \frac{t}{T}$; a monotonically decreasing gain coefficient.
- To ensure the prototypes are representatively distributed, each node $\mathbf{m}_i$ has an associated sensitivity, S_i, initially zero: Set $S_i = 0 \; \forall \; i$
- Set a value for β, the node sensitivity adjustment parameter. This must be small in comparison to the feature space.
- Train for many (T) epochs, i.e. while ($t < T$)
  - find the prototype $\mathbf{m}_c(t)$ which is nearest to this input:
    set $c = \arg\min_i \left(\|\mathbf{x}(t) - \mathbf{m}_i(t)\| - S_i(t)\right)$
  - update the prototypes and sensitivities, for each i:

$$\begin{array}{rcl|rcl} \text{if } i & = & c & \text{if } i & \neq & c \\ \hline \mathbf{m}_c(t+1) & = & \mathbf{m}_c(t) + \alpha(t)\left[\mathbf{x}(t) - \mathbf{m}_c(t)\right] & \mathbf{m}_i(t+1) & = & \mathbf{m}_i(t) \\ S_c(t+1) & = & S_c(t) - \beta & S_i(t+1) & = & S_i(t) + \frac{\beta}{k-1} \end{array}$$

  - increment t, t++

The vector quantisation algorithm is quick and efficient in comparison to training GMMs. Here VQ is performed on the same data as in the previous section, firstly on 2D data of 48771 positions of players. The period, T, is set to $1,000,000$. The effects of varying the node sensitivity adjustment parameter, β, on the distribution of prototypical positions is shown in Figure 7.3, with $\beta = 0.0001$, a set of prototypes distributed in proportion to the data from which they are learned is observed. When β is larger (0.01), the prototype vectors do not cover the whole field, and when β is smaller (0.000001) there is a random spread of prototypes which are not representative of the data; there appear to be fewer prototype vectors in Figure 7.3(e) this is due to many of the vectors being positioned out of the field of play, far from any feature vectors in the training set, thus $\beta < 0.00001$ is clearly too small for practical use.

Incorporating velocity information into this representation can yield a more detailed description. Figure 7.4(a) shows the set of 200 prototypical vectors (4D) learned from the



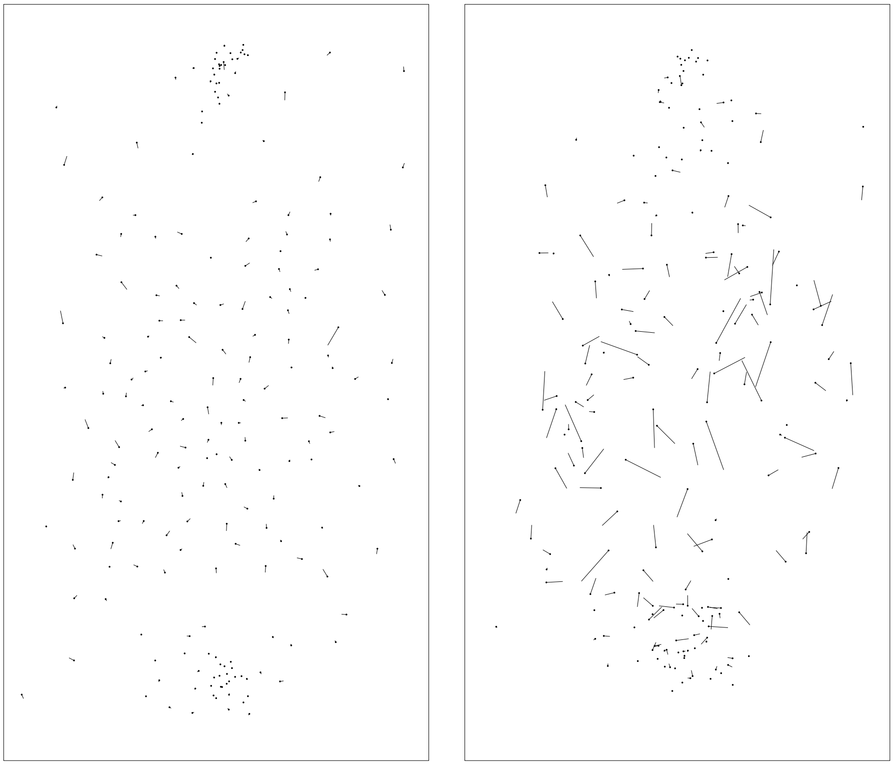
(a) $\beta = 0.01$ (b) $\beta = 0.001$ (c) $\beta = 0.0001$ (d) $\beta = 0.00001$ (e) $\beta = 0.000001$

Figure 7.3: Vector quantised positions (2D data) learned with different levels of the parameter β.

same training set, with $T = 1,000,000$ and $\beta = 0.001$, with velocities included. This clustering of the training set appears to give more weight to the positions of the players than to the velocities. The velocities are much smaller in magnitude than the positions. Scaling the data should alleviate this problem.

Repeating the experiment with the additional condition of normalising (through translation and scaling) the mean of each component of the feature vectors to 0.5 and a standard deviation of 0.08 ($\mathbf{x}_i \sim \mathcal{N}\ (0.5, (0.08)^2)$) gives a set of prototypical vectors whose velocity components vary more significantly than without the normalisation; the goalkeepers have very low speeds, and players move more quickly near the centre of the pitch (Figure 7.4(b)). These values were chosen since the implementation of the VQ algorithm requires all data to be in the range $(0, 1)$. Thus a mean of 0.5 is in the centre of this range, and a s.d. of 0.08 is sufficient for outlying data to be scaled to fall within $(0, 1)$.

The *reconstruction error* [60] is calculated as the sum of the minimum squared distance of the closest prototype to each input training vector. This cost can be used to identify the performance of the clustering when different numbers of clusters are used. Figure 7.5 shows the VQ reconstruction error obtained when between 1 and 400 prototypes were formed. The random initialisation of the prototypes, and the fact that each experiment was run only once for each number of clusters explains the non-monotonically decreasing reconstruction error as the number of clusters increases. When more than 150 prototypes are used, introducing more prototypes does little to further reduce the reconstruction error.



(a) without normalisation (b) normalised $x_i \sim \mathcal{N}(0.5, (0.08)^2)$

Figure 7.4: Illustrating the vector quantisation of the 4D data (positions and velocities) with 200 prototypes. (a) on raw data (b) with data scaled to mean 0.5 and s.d 0.08.

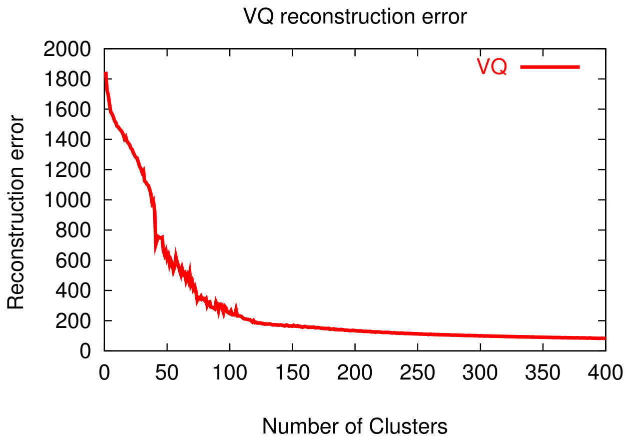


Figure 7.5: Plotting the reconstruction error of the VQ algorithm when clustered into different numbers of prototypes shows that after around 150 prototypes there is little improvement in the representation.

7.4 Emergent behaviour

Algorithms used for learning do so in two fashions: supervised and unsupervised. Supervised methods utilise additional knowledge of the scenario in their learning phase, such as knowledge of which group they belong to. Clustering algorithms are unsupervised methods commonly used to group together examples with similar features. They are often referred to as emergent since the result has come solely from the data without any predefined rules; the result has emerged from the data.

Reviewing related work and discussing possible aims for behavioural analysis of the players' movements in Chapter 2, it was asked if the following questions could be answered:

- Can emergent behaviour be learned from observing real sports games?
- Can team membership be identified for each player?
- How do players interact with their team mates, and the opposition?
- Can the tactics be encapsulated in an understandable way?
- Can a generative model be created?
- Can the rules of the game be learned?

The enormity of some of these tasks can now be realised. To take a simple example: given two unordered sets of ten players, then the calculation of a metric to indicate the similarity of the two instances may involve an exhaustive search over all possible combinations of player configurations. There are **10!** such possible combinations $(10! = 3,628,800)$. If this is used in a technique in which each example is compared to a set of cluster centres or prototypes, then the calculations quickly become overwhelming, and an alternative approach must be found.

It is possible to create a set of prototypes for the positions of the whole set of players. Figure 7.6 shows a set of 25 prototypes of all ten players' positions relative to one of the players. The above problem is encountered in this situation: to compare an example set of positions of players at a time instance involves a set of 10! comparisons with each cluster.

Prototypes created by utilising vector quantisation on 20 dimensional position data or even on 40 dimensional position and velocity data of the ten players could create a set of states ideal for use in a graphical model such as a Markov model or Bayesian network. However, initial experiments have shown that we do not have enough data to do this. Even with only 25 prototypes, commonly once a prototype/state has been left, it isn't returned to. With more data a model in which behaviour of cycling through sets of states could be observed combinations of which could be identified with types of play occurring; attacking, players bunching, or quick movements from end to end.

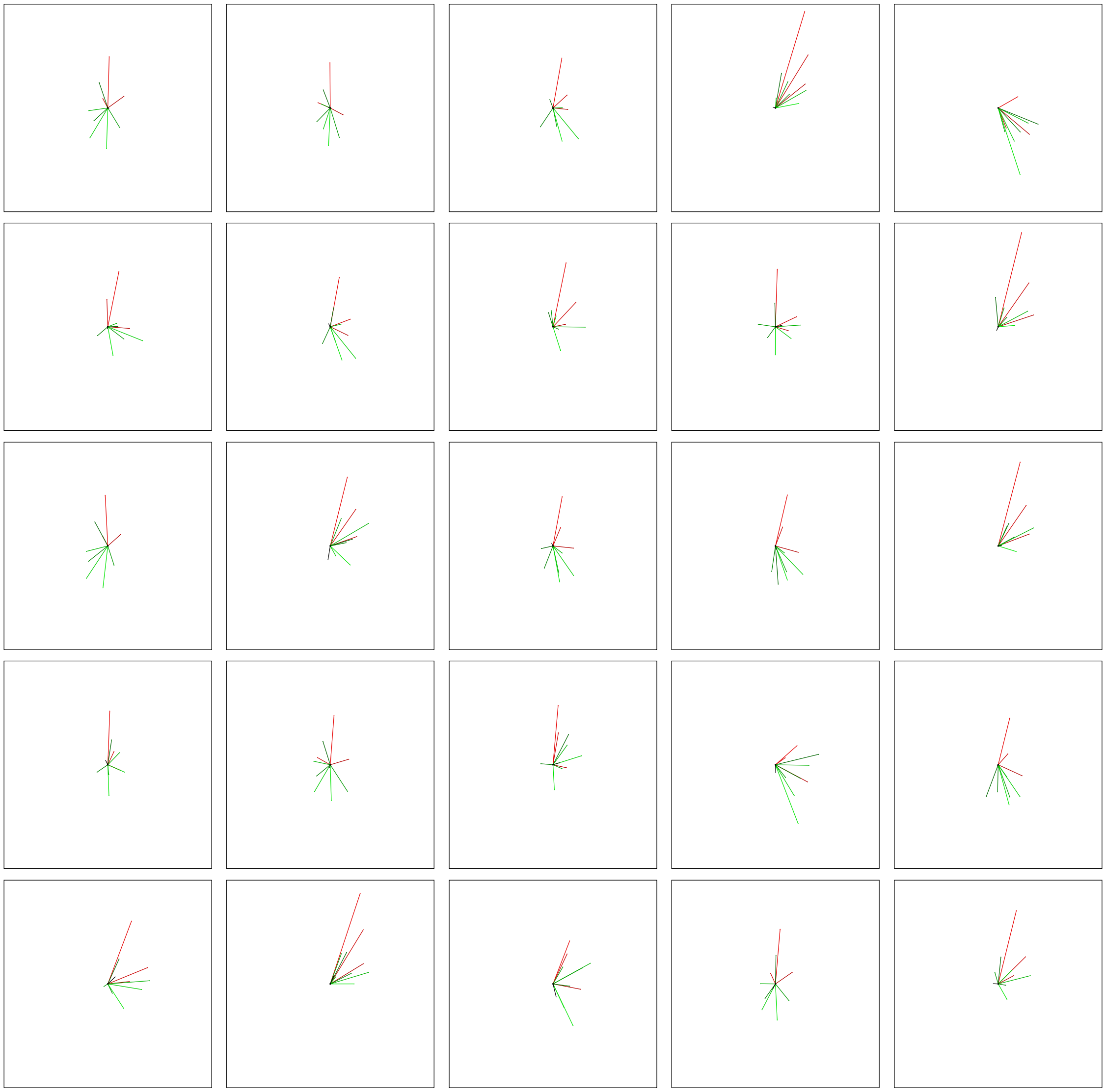


Figure 7.6: A set of 25 prototypes of the players' positions. Created through VQ on 20D positional data. Illustrated relative to Player 1. The same information is contained whichever player it is centred on. The red lines indicate the vectors between Player 1 and his team mates, the green lines indicate the vectors between Player 1 and players on the opposing team.

7.5 Creation of synthetic data

Machine learning techniques often require massive quantities of data on which to be trained. Specifically, a training set (of feature vectors) in which all configurations of the object space are represented is desired. To gain such a training set large quantities of data are required, so that everything that may be seen in a test sequence will have been seen before in training. In our case, this refers not only to all possible configurations of the players (where each of them may be on the pitch), but also to consider this over time. Incorporating the temporal aspect increases the quantity of data needed further, since it is necessary to observe the changes of positions of the players over time, in sufficient configurations for generalisation. Using enough data in the learning phases allows for modelling of all possible scenarios, and should remove any sensitivity to particular training instances which were either outliers or noise.

Due to the lack of large amounts of marked up, or automatically tracked training data, synthetic data of a similar nature will be used. Data created using a set of constraints and a stochastic input allows an endless supply of data. The aim is to learn behaviour from this data, (for which the rules are *known*), which could then be applied in a similar way to real positional data from sports games.

There are many ways that synthetic data could be created. Little experimentation has been performed on other approaches, since the method described here produces player interactions which are judged to have suitable properties and when visualised over time are engaging and patterns can be noticed (as though a game of some description is being played). In this chapter, synthetic positional team game data has been created governed by the following rules:

- 2 teams.
- 5 players on each team.
- There is a constrained pitch, with ‘walls’ at 0 and 1.
- At each iteration (time step), each player keeps their velocity, subject to the following additions/subtractions:
  - Players move away from the wall, when they are near it, governed by adding $0.2\exp(-20|\mathbf{x}-\mathbf{wall}|)$ to their velocity in a direction perpendicular to, and away from the wall (for all 4 walls).
  - Each player has a ‘home’, and shouldn’t stray too far from it, governed by adding $0.2\exp(2.5(|\mathbf{x}-\mathbf{home}|-\sqrt{2}))$ to their velocity in the direction of their ‘home’.
  - When a team has possession (of the nonexistent ball), their behaviour is that they move away from the nearest player on their team, governed by adding $0.1\exp(-10|\mathbf{x}-\mathbf{nearest\ teammate}|)$ to their velocity in a direction away from the nearest teammate.

- When a team doesn't have possession, their behaviour is that they move towards the nearest player on the opposing team governed by adding $0.5\exp(-2.5(|\mathbf{nearest\ opponent} - \mathbf{x}| - \sqrt{2}))$ to their velocity in a direction towards the nearest opponent.
- A small amount of uniform noise is added to the players velocity within a range $\pm\, 0.025$.

The probability of possession changing is p, here $p = \frac{1}{200}$ and is calculated by generating a random number r, (uniformly distributed) in the range (0,1), and changing possession if $r < p$. Thus on average a team has possession for 200 time steps.

The positions of the simulated players over a long period of time are shown in Figure 7.7. It shows that each player has been to most parts of the pitch, and is not confined to a particular sector. Analysing player movements governed by this rule set, through observation, confirms a realistic behaviour. The players do appear to be interacting, as though a game of some description is underway.

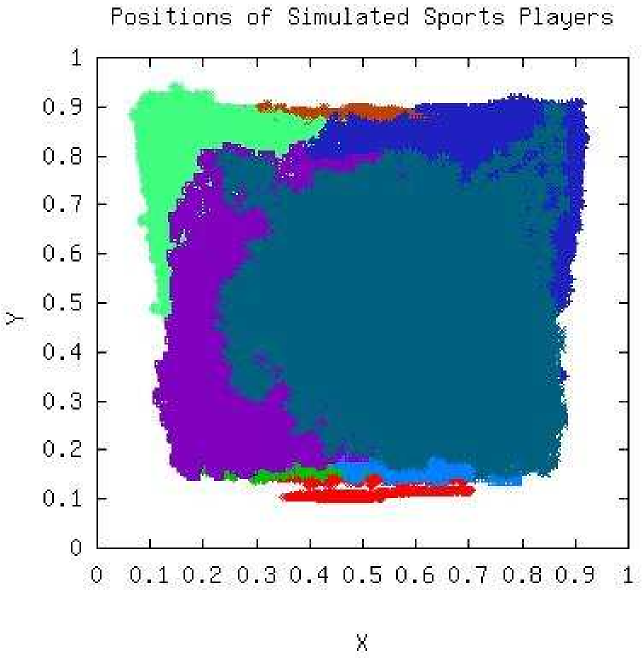


Figure 7.7: The positions of 10 simulated players, over 1,000,000 time-steps.

7.6 Team identification

If emergent methods are to be used to identify which team each player is on, then an unsupervised approach is necessary. Without providing any information as to which team each player is on (even in training data) a method would need to be developed which only used, say, the knowledge that there are 10 players, and there are 2 teams with 5 players on each, to discriminate between the two teams. More information or properties of the system must be known in order to make any decisions, which are hard to identify.

One such property that might be expected, is that when one team is rotated through 180 degrees about the centre of the pitch, then it will be similar to the other team. Using this 'property' alone fails to identify the team because it contains no information about the structure of a team and gives similar results for a left-right split of players; they look similar when rotated. A similar failing would occur if defenders were grouped with opposing attackers. For team identification a *supervised* approach must be adopted.

7.7 Player identification

Many approaches to the tasks of interest are thwarted by the need to employ an exhaustive search, involving 10! combinations of players and labels in order to identify each player. This section details an emergent approach to player identification without the need for exhaustive searches for finding a best match to be employed.

This approach involves training a set of Gaussian mixture models on the positions of each player. Then, taking an example set of players' positions a response (likelihood) from each GMM can be found for each player. A matrix (here 10×10) of these values can be formed which can be thought of as representing a weighted undirected bipartite graph, where the ten players form one set of vertices of the graph, and the labels 1-10 form the other set of ten vertices. The matrix contains the edge weights fully connecting the two sets of vertices. Using graph partitioning methods, edges can be removed until ten graphs remain; each containing two nodes and one edge. A correspondence between the players and the labels $1 - 10$ has now been established. (No label can be assigned to multiple players, and no player to multiple labels). Graph partitioning is an efficient and effective way to do this, since no exhaustive searches are needed.

Firstly, for each player i a GMM $\mathcal{G}_i$ with k components is formed (here $k = 30$ has been used). Given a player's position a response or likelihood from each GMM $\mathcal{G}_i$ can be calculated. For a particular example (at a time instant), the response $\mathcal{G}_i(\mathbf{x}_j)$ to each players' position $\mathbf{x}_j$ from each $\mathcal{G}_i$ can be calculated. A 10×10 matrix of these responses can be formed:

$$\begin{pmatrix} \mathcal{G}_1(\mathbf{x}_1) & \mathcal{G}_1(\mathbf{x}_2) & \dots & \mathcal{G}_1(\mathbf{x}_{10}) \\ \mathcal{G}_2(\mathbf{x}_1) & \mathcal{G}_2(\mathbf{x}_2) & \dots & \mathcal{G}_2(\mathbf{x}_{10}) \\ \vdots & \vdots & & \vdots \\ \mathcal{G}_{10}(\mathbf{x}_1) & \mathcal{G}_{10}(\mathbf{x}_2) & \dots & \mathcal{G}_{10}(\mathbf{x}_{10}) \end{pmatrix} \tag{7.4}$$

Simply choosing to associate a player's position $\mathbf{x}_j$ with the GMM $\mathcal{G}_i$ which gives the maximal response does not give a plausible result. Usually, more than one player will be associated with a label $\mathcal{G}_i$ which is something than must not be allowed to happen. Selecting an optimum labelling of the players with the numbers $1 - 10$ involves choosing the combination for which some metric is maximised (to find the most likely labelling). An exhaustive search of all possible combinations would entail evaluating $10!$ combinations.

The matrix of responses can be formulated as a graph; with 20 vertices (10 representing the players and 10 representing the GMMs corresponding to the player's labels $1 - 10$), the elements of the matrix give the weights of edges in the graph; the likelihood that label i corresponds to player j is calculated as $\mathcal{G}_i(\mathbf{x}_j)$. A fully connected bipartite graph is formed with each player connected to each label by a weighted edge. This is visualised in Figure 7.8.

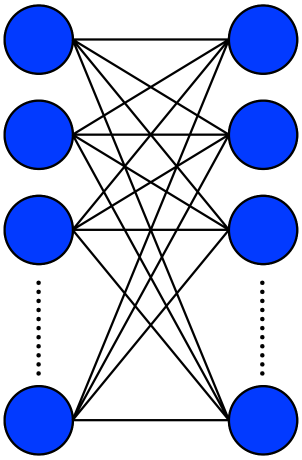


Figure 7.8: The graph to be partitioned.

Graph partitioning methods can now be employed to partition the graph into ten subgraphs. Partitioning the graph into ten should result in a direct association of each label with one and only one player. The methods of the METIS software package [57] are incorporated to perform the graph partitioning. Two methods of graph partition are experimented with: multilevel recursive bisection and multilevel k-way partitioning. Both of these try to minimise the cut with balancing constraints, and are approximate methods.

Multilevel recursive bisection partitions the graph by splitting the graph into two equally sized parts, and repeats this process on each remaining graph, until it has been split into the required number of partitions. Multilevel k-way partitioning is recommended for partitioning a graph into 8 or more partitions, and involves coarsening down the graph to a few hundred vertices, performing a bisection of this graph, and projecting back to the original graph. On the graph described (Figure 7.8 with twenty vertices, this method was found not to work in the task of partitioning the graph into ten equal parts in which each part contained two vertices and one edge. Often partitions contained one or three vertices. Multilevel recursive bisection is found to successfully partition the graph into ten partitions, allowing a correspondence between players and labels (GMMs) to be formed.

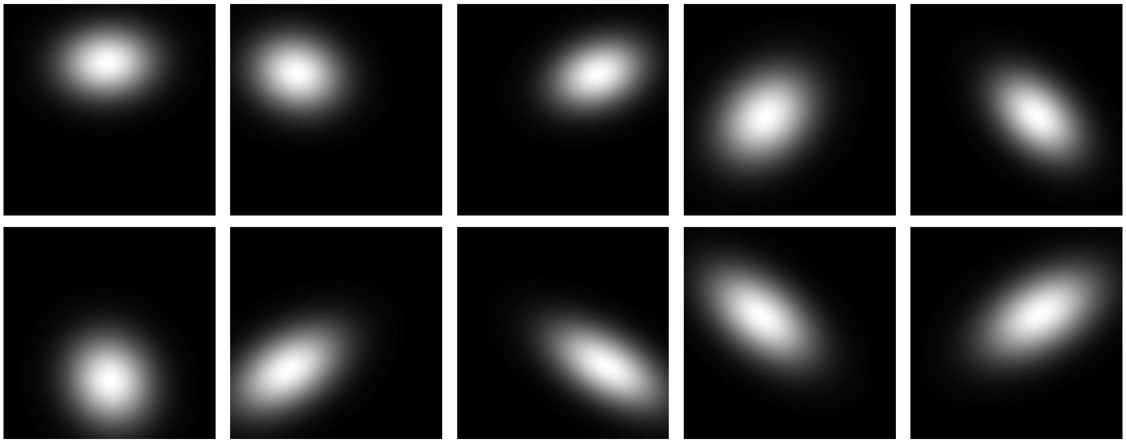


Figure 7.9: Ten GMMs trained on players positions over 5000 time steps.

A set of ten GMMs learned from 5000 sequential time steps of the synthetic data are visualised in Figure 7.9. Many positions $\mathbf{x}_j$ will produce a high response $\mathcal{G}_i(\mathbf{x}_j)$ (indicated in white) from more than one $\mathcal{G}_i$.

Using this method, overall performance of the system when labelling (classifying) the players at each time instance was correct in 1242 instances out of the 5000 time steps used for training the data. i.e. in only 1242 cases out of the 5000 was each label $1-10$ associated which the correct player. The maximum number of frames observed between correct labelling was 117. A breakdown of the results into performance for each player is detailed in Table 7.1. Each player is identified correctly in around three-quarters of all cases, and the maximum number of sequential frames in which a labelling is incorrect is much lower than 117, typically around 50.

Similar results are obtained when applied to unseen test data. Table 7.2 shows the results of the above trained system on a set of 5000 sequential test frames on a per player

Position	success/5000	% success	max. sequential incorrect labellings
1	4014	80.3	55
2	3958	79.2	82
3	3517	70.3	52
4	3943	78.9	17
5	3709	74.2	29
6	4108	82.2	48
7	3828	76.6	22
8	4202	84.0	49
9	3188	63.8	82
10	3477	69.5	50
Overall	1242	24.8	117

Table 7.1: Results on training data.

basis. Pleasingly, they are not dissimilar results, although the maximum time between correct labellings has increased.

Position	success/5000	% success	max. sequential incorrect labellings
1	3764	75.2	99
2	3519	70.3	99
3	3977	79.5	103
4	3628	72.5	30
5	4080	81.6	19
6	4512	90.2	14
7	4344	86.8	39
8	4098	81.9	41
9	3487	69.7	129
10	3939	78.7	18
Overall	1552	31.0	157

Table 7.2: Results on test data.

Incorporating time information, by building a histogram of frequency of labels assigned to each player, and choosing the label with the greatest number of occurrences over a window length of t_w improves the player identification system.

A comparison of the performance of the method, on a training and test set both of size 5000 is shown in Figure 7.10 with the window length t_w varying from 1 to 1000;

on unseen test data a behaviour similar to the training data is observed. The steps in the graph appear since the percentages correct have been calculated from the set of 1000 broken down into $1000/t_w$ distinct blocks of time.

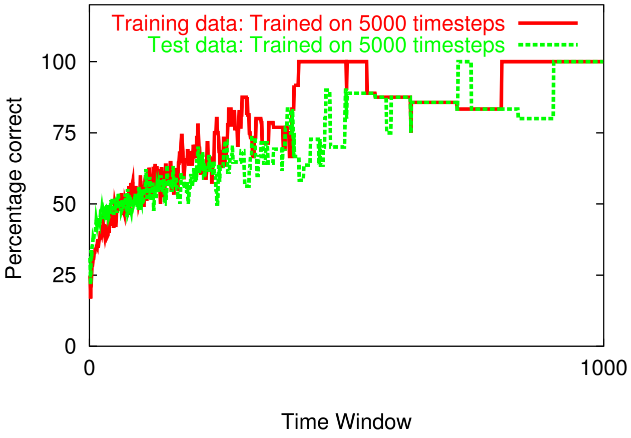


Figure 7.10: Comparison of performance on training and test data.

This method is now evaluated on an unseen test set comprising of 10,000 sequential frames of simulated play. Figure 7.11 and Table 7.3 display the results of the system when different sized training sets are used, when learning the GMMs, and for a range of time windows.

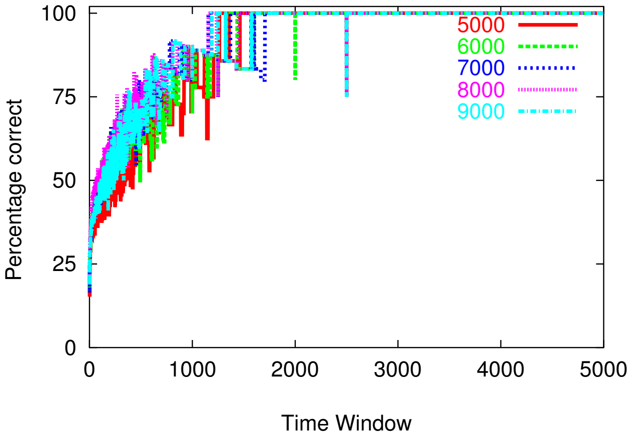


Figure 7.11: Comparison of performance on 10,000 time steps, when trained on various size training sets.

Once a time window of $t_w = 2000$ or greater is used, the system successfully identifies each player correctly. A steady improvement in the percentage correct is observed as

Window	Size of training set for GMMs				
length t_w	5000	6000	7000	8000	9000
1	22.8	24.7	24.4	28.3	25.4
2	15.3	16.5	16.3	20.0	18.6
3	23.9	25.6	25.5	29.7	26.4
4	22.3	25.5	24.1	29.3	25.0
5	25.0	28.1	27.2	33.0	29.1
10	28.5	31.0	32.0	36.0	31.4
20	32.0	36.8	35.6	41.2	34.6
50	36.5	41.0	43.0	45.0	41.5
100	43.0	49.0	46.0	47.0	41.0
200	42.0	56.0	56.0	52.0	54.0
500	68.4	70.0	75.0	65.0	68.4
1000	77.8	77.8	80.0	88.9	89.9
2000	100.0	100.0	100.0	100.0	100.0
5000	100.0	100.0	100.0	100.0	100.0

Table 7.3: Results.

the size of the window grows. With regard to the size of the training set, in this data the results are generally similar, though a little better with a larger training set. Figure 7.9 shows that the GMMs are being reduced down to very few Gaussians, which would explain this. Given more real world data, this approach could be tested more fully, and given the evidence in Figure 7.2 it is suspected that a mixture of many Gaussians would be formed to represent each player. In this case, a larger training set would provide a better representation of the players' movements.

The need to use information collected over a period of time in order to identify each player's "position" or *role* corresponds well with the approach that a human observer may take. In a related scenario: given a soccer game at an instance, in which the coloured strips of the two teams were replaced with a plain white shirt, it is speculated that it would not be possible to correctly identify which team each player is on in the majority of cases. Sometimes it may be possible, but there will be many times when a player passes the ball back to his goalkeeper, or two players tackle each other, for example, which makes it hard to define a set of rules to associate players to teams.

How many frames would be needed before a human observer could confidently identify who is on which team? Undertaking this task, a human observer may utilise his prior knowledge of player positioning in soccer games, but he will also draw strongly from many other cues; player orientation (which end they are facing, and whether they are facing the ball or moving into space); player velocities; player body movements (arms in the air wanting the ball, or hands on hips not involved in play); and even facial expressions.

7.8 Discussion

Providing a machine learning system with very limited knowledge, and trying to make results emerge form the data is a tricky task. It is much more common for a supervised approach to be used.

For example, a Bayesian network, hidden Markov model, or a neural network could be trained to provide a learned response (or likelihood of a range of responses) of an individual or team, where inputs to the system are each of the players' positions and velocities, and the output is the player/team position and velocity at the next time step. This approach would need knowledge of each player, and once trained would give an output which could be used for predicting future movements, or possibly used in a behavioural analysis to identify atypical play. A model which had learned from a set of experts playing, and been trained on hours of footage/positional data, could then be used when a novice (or less experienced) player was introduced into a team, to identify if they behave differently.

Supervised clustering methods may be a useful approach for modelling or classifying player movements with respect to their team and opponents, since a player or team behaves differently whether they are attacking or defending. Possession of the ball is a key factor in where the players position themselves, and in which way they move. Learning separate clusters or states, when the players are in similar configurations but possession is different should give a better representation of behaviour, since additional knowledge is incorporated into the model.

These approaches all require for each of the players to be identified (either with a unique label, or equivalently the order in which they appear in a feature vector). The work that has been presented in this chapter facilitates this. Given a set of players, the most likely labelling of the set to a learned set of GMMs can be found. From this stepping stone, harder tasks can now be considered, without having to incorporate as much handcrafted knowledge into the model.

A different branch of soccer analysis is appearing, driven by demand for analysing the *structure* of the game to allow for multimedia and limited bandwidth video streaming applications to choose which parts of a broadcast game are of interest. Ekin [26] uses cinematic features for automatic real-time event detection (such as *shot classification* (camera view) in the categories: long-shot, in-field medium shot, close-up shot or out of field shot). Other object based features are used to identify the penalty box and goals. Xie [118] parses the structure of a game into the two mutually exclusive categories of *play* and *break*, and trains sets of HMMs on features extracted from footage of plays and breaks, and uses dynamic programming to segment the two states. This type of work is very different to tracking and positional behaviour analysis, however it would be an important part in creating an integrated positional analysis tool from broadcast footage, where many camera views are used (and close-ups of players or the crowd may be of little use).

7.9 Summary

The modelling of behaviour and interactions, particularly with respect to positional data, is an under-researched area. This chapter has provided an insight into the behaviour modelling domain. It has explored density estimation methods for representing and describing data in the form of Gaussian mixture models and clustering using vector quantisation. Practical considerations of the unexpected complexity of matching ten player positions with ten player labels (or one configuration of players with another, for which there are 10! possibilities) led to the formulation of the task as a graph partitioning problem, an area in which much research has been undertaken. Combining this with a set of GMMs has resulted in an emergent method for identifying a player's "position" from the positional data alone.

Chapter 8

Conclusions

The work in this thesis was motivated by a desire to track sports players through the course of a team game and to analyse the resulting interactions.

8.1 Summary of work

In Chapter 3, a novel framework for a CONDENSATION based multiple object tracker, as opposed to multiple single object trackers, is developed. It is demonstrated that the performance of the tracker is improved when a set of Kalman filters are introduced to help regroup the samples which are stochastically propagated within the CONDENSATION framework. In this chapter, three main topics are identified as important; the need for a more sophisticated, yet computationally efficient, fitness function for player localisation capable of handling shapes which change greatly and quickly; the need for methods to resolve occlusions; and the need for a generally accepted scientific approach to positional tracker evaluation.

Development of a feature descriptor for use as a fitness function in the tracking system is the focus of Chapter 4. The use of splines for object tracking is explored. A novel multi-resolution people finding kernel is designed, which is robust to the perspective effects that are visible in this domain. Three possible methods for employing the multi-resolution kernels are devised and evaluated. The methods are evaluated against a set of positive examples and a set of negative examples of players. From this evaluation, a method in which a set of learned template kernels are applied is chosen. Further evaluation is performed on an example image to illustrate the properties of the method and its robustness and sensitivity to changes in horizontal and vertical positioning, changes in scale to the width and height of the bounding box, and fixed aspect ratio scaling. This novel method of learning template kernels from subsampled examples is also demonstrated on sequences of pedestrians walking. A set of six templates are formed once the training data has been clustered, and a Markov model of the pose transitions is created by observing the frequencies of transitions from one template to another over sequential frames of video of pedestrians walking. This could be employed to provide partitioned sampling in any ‘predict and sample’ tracking scheme.

Chapter 5 presents work on positional tracker evaluation which is notably lacking from computer vision literature. A set of metrics and statistics for trajectory comparison are presented with the goal of being able to compare two trajectories. The similarity of two trajectories cannot be encapsulated in a single number. For example, it may be of interest to consider the paths taken by two people which are identical, yet travelled at different speeds. The mean of the differences between the positions at each time step would be large, yet the area between the two paths would be zero.

Multiple camera systems are investigated in Chapter 6. Positioning of cameras is discussed and a common coordinate system is established. The measurement error covariance matrix of the Kalman filter for each player is estimated taking into account the player’s position on the pitch and the camera views from which they are visible. The multi-resolution template kernels are employed within the fitness function to identify players, and methods to fuse data from multiple views are discussed. Initial results for multiple camera tracking are less promising than expected and are comparable to single camera tracking.

Chapter 7 provides an insight into the behaviour modelling domain. It explores density estimation methods for representing and describing data in the form of Gaussian mixture models and clustering using vector quantisation. Practical considerations of the unexpected complexity of matching ten player positions with ten player labels (or one configuration of players with another, for which there are 10! possibilities) lead to the formulation of the task as a graph partitioning problem. Combining this with a set of GMMs results in an emergent method for identifying a player’s “position” from the positional data alone.

8.2 Contributions

The main contributions of this thesis are:

- An extension to the CONDENSATION algorithm to track sets of target objects as sets, rather than as individuals. Incorporation of a set of Kalman filters is shown to improve the tracking accuracy.
- A simple and robust shape descriptor capable of locating target objects in a range of sizes/resolutions and in a variety of different poses.
- Preliminary work on the potential use of Markov models for analysis of the transitions between poses.
- An overview of positional performance evaluation methods, with demonstrative use of metrics and statistics.
- Investigation of the use of multiple cameras in a sports hall, which was not as successful as intuition might expect.
- Preliminary exploration of behaviour analysis of team game interactions. Identification of the problem of combinatorial explosion associated with multiple objects and demonstration of a simple application to identify players.

8.3 Discussion

Research into a variety of topics within the computer vision domain has been presented in this thesis. In Section 1.2 a set of challenges expected were identified. These and other topics of interest are now discussed.

The *perspective effects* of the pitch have been accounted for in two ways. Firstly, all of the tracking has been performed using real world positions, rather than image coordinates, and the Kalman filters have been updated using a measurement error covariance matrix $\mathcal{R}$ which reflects the reliability of the measurement (proximity to camera and resolution of nearby pixels). Secondly, a resolution independent shape descriptor has been devised which is applied to sub-sampled image regions allowing it to be used on a large area of the image close to the camera or on a small region far from the camera.

The multi-resolution kernel provides a mechanism for *player localisation*, and can locate players in a variety of poses, without the need for a complex set of constraints. With more time it would be interesting and valuable to perform a comparison between player localisation/tracking using splines and the multi-resolution kernels. Additional investigation into choosing the size of the template should also be performed in order to determine the smallest size template which is still capable of tracking objects to the required degree of accuracy. The multi-resolution template kernels can be learned for any set of objects to be tracked which are always seen in a fixed (or limited) rotational orientation.

Resolving occlusions when one player obscures another player (or multiple players) intuitively should become an easier problem when information from multiple views is incorporated. In this work this has not been shown to be the case. For success in this area, more evaluation on each aspect of a tracking system would need to be performed as the system is developed, to ensure that each decision taken benefits the tracker. This should be done on two levels; evaluating the algorithm on its own, separate to the tracker; and also evaluating its effect on the tracking system as a whole on a bank of test sequences.

Performance evaluation is a large area of study which is essential to the development of computer vision systems. It is important to consider the context of the situation and the purpose of the evaluation. There are many ways in which aspects of a tracking system can be evaluated. This thesis has presented a range of evaluation; positional trajectory evaluation; comparison of three methods for applying multi-resolution kernels to players; an evaluation of the template kernels' sensitivity to changes in size, scale and shift of the bounding box around a player; development of metrics to advance the positional evaluation of tracking systems, applied to soccer and vehicles trajectories; brief positional evaluation of a tracker of multiple soccer players; and evaluation of an emergent system to identify players' positions in a simulated soccer game.

Behaviour modelling is a massive research area in its own right. A large amount of data able to represent the possible configurations of the feature space is necessary. Many tasks which at a first glance appear trivial are far from trivial once understood. Team games are structured activities in which players interact in a way that is generally understandable to a human observer. Many visual cues are used by humans in addition to the positions of the players. Machine learning methods which incorporate visual evidence in addition to positional and velocity data may be able to learn patterns of interactions between players in a realistic and engaging manner.

8.4 Future research

Much of the work in this thesis is applicable to many other areas of computer vision. This section discusses where the work undertaken in this thesis has the potential to lead to.

The work on pose transitions in Section 4.3 has scope for development. Applied in the soccer domain, it may be possible to use a different motion model for each pose; when a diagonal running pose is detected, then a fast moving player is likely; when a pose with the player's hands on their hips is detected, then a stationary player is likely.

Regarding evaluation, the vision community would ideally like to be able to quote a single number to identify the performance of a system or algorithm. In practice this is not possible. Chapter 5 identified a set of metrics and statistics for positional performance evaluation, and there is scope for building on top of these a system to interpret the numbers to highlight the important traits observed in the comparison of two trajectories.

On the behaviour side, there are many aspects open for further study. Statistically the behaviour of the individuals and teams could be captured in a meaningful way which could be used to evaluate tactics used in a game. A football team X may be able to decide what tactics it should use in the next game against opposing team Y, by analysing how different tactics have worked for team X against other opposition, and how team Y have coped with different tactics from other teams. This may give team X enough of an advantage to alter the outcome of the game.

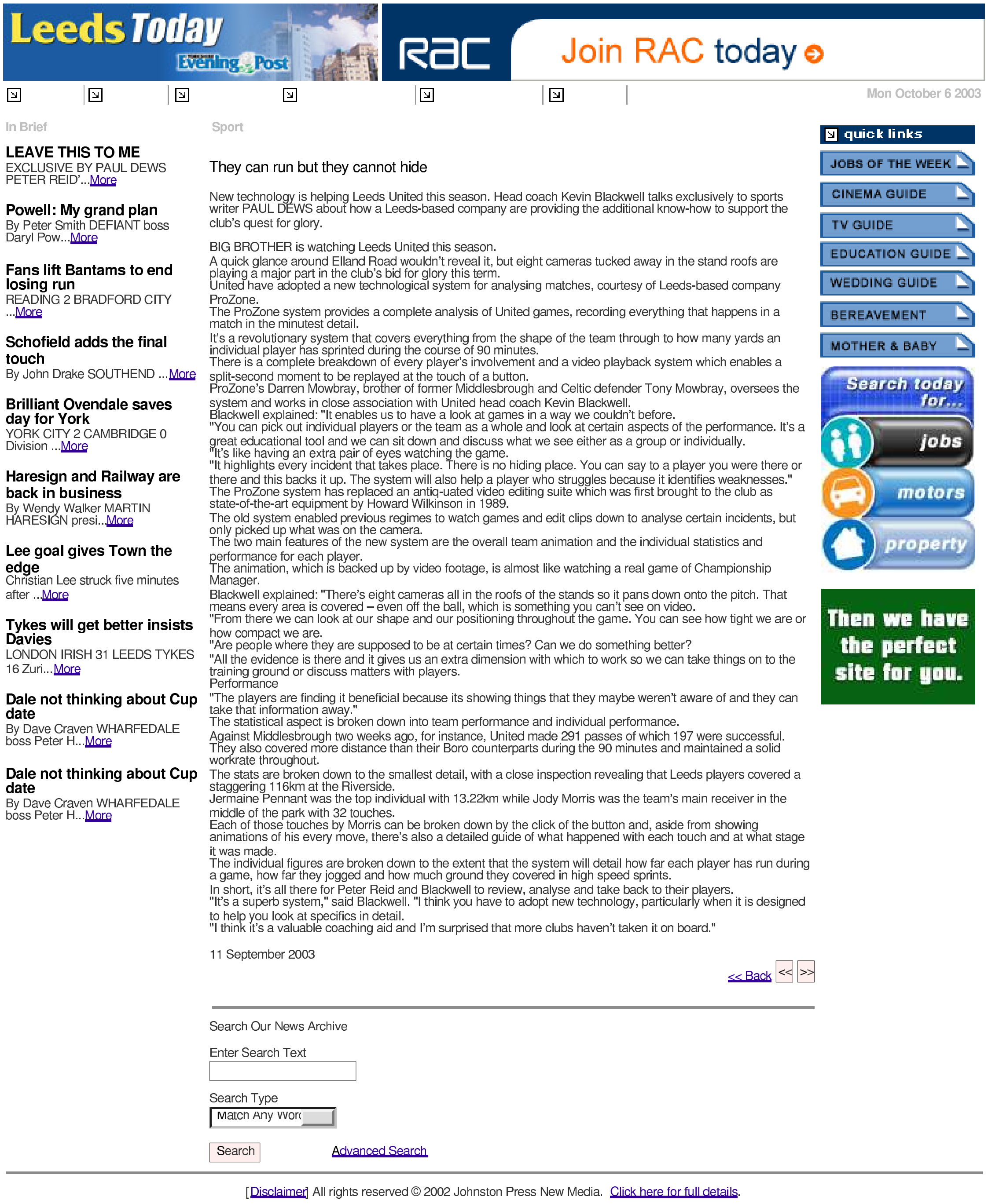
The areas of cognitive science and computer vision are becoming closer, and gaining understanding from image sequences is a developing research area. The combination of vision, logic and spatial reasoning could be applied to sports games, since there is an underlying structure behind the games. If methods could be developed to describe the game, then these methods could also be applied to understanding scenes where the structure is unknown, for example, the movements of people in a shopping centre or the intentions of pedestrians in a busy underground interchange.

Given suitable resources, the most exciting topic to address would be real-time tracking and behaviour modelling of a full outdoor soccer game. To accomplish this, an approach which incorporated a behaviour model into the tracker would be necessary. The results from the tracker could be used to create a more sophisticated behaviour model. At each iteration of adaptively updating the behaviour model the tracking results should improve. Such a system could revolutionise the tactical decisions made in the sports world.

Appendix A

Soccer clubs embrace new technology

This appendix contains a press cutting from the Leeds Today website of the Yorkshire Evening Post (http://www.leedstoday.net/ last accessed 06/10/03). This article describes the types of analysis of soccer games possible and now used in practice by FA Premiership side Leeds United.



In Brief

LEAVE THIS TO ME
EXCLUSIVE BY PAUL DEWS PETER REID'...More

Powell: My grand plan
By Peter Smith DEFIANT boss Daryl Pow...More

Fans lift Bantams to end losing run
READING 2 BRADFORD CITY ...More

Schofield adds the final touch
By John Drake SOUTHEND ...More

Brilliant Ovendale saves day for York
YORK CITY 2 CAMBRIDGE 0 Division ...More

Haresign and Railway are back in business
By Wendy Walker MARTIN HARESIGN presi...More

Lee goal gives Town the edge
Christian Lee struck five minutes after ...More

Tykes will get better insists Davies
LONDON IRISH 31 LEEDS TYKES 16 Zuri...More

Dale not thinking about Cup date
By Dave Craven WHARFEDALE boss Peter H...More

Dale not thinking about Cup date
By Dave Craven WHARFEDALE boss Peter H...More

Sport

They can run but they cannot hide

New technology is helping Leeds United this season. Head coach Kevin Blackwell talks exclusively to sports writer PAUL DEWS about how a Leeds-based company are providing the additional know-how to support the club's quest for glory.

BIG BROTHER is watching Leeds United this season.
A quick glance around Elland Road wouldn't reveal it, but eight cameras tucked away in the stand roofs are playing a major part in the club's bid for glory this term.
United have adopted a new technological system for analysing matches, courtesy of Leeds-based company ProZone.
The ProZone system provides a complete analysis of United games, recording everything that happens in a match in the minutest detail.
It's a revolutionary system that covers everything from the shape of the team through to how many yards an individual player has sprinted during the course of 90 minutes.
There is a complete breakdown of every player's involvement and a video playback system which enables a split-second moment to be replayed at the touch of a button.
ProZone's Darren Mowbray, brother of former Middlesbrough and Celtic defender Tony Mowbray, oversees the system and works in close association with United head coach Kevin Blackwell.
Blackwell explained: "It enables us to have a look at games in a way we couldn't before.
"You can pick out individual players or the team as a whole and look at certain aspects of the performance. It's a great educational tool and we can sit down and discuss what we see either as a group or individually.
"It's like having an extra pair of eyes watching the game.
"It highlights every incident that takes place. There is no hiding place. You can say to a player you were there or there and this backs it up. The system will also help a player who struggles because it identifies weaknesses."
The ProZone system has replaced an antiq-uated video editing suite which was first brought to the club as state-of-the-art equipment by Howard Wilkinson in 1989.
The old system enabled previous regimes to watch games and edit clips down to analyse certain incidents, but only picked up what was on the camera.
The two main features of the new system are the overall team animation and the individual statistics and performance for each player.
The animation, which is backed up by video footage, is almost like watching a real game of Championship Manager.
Blackwell explained: "There's eight cameras all in the roofs of the stands so it pans down onto the pitch. That means every area is covered – even off the ball, which is something you can't see on video.
"From there we can look at our shape and our positioning throughout the game. You can see how tight we are or how compact we are.
"Are people where they are supposed to be at certain times? Can we do something better?
"All the evidence is there and it gives us an extra dimension with which to work so we can take things on to the training ground or discuss matters with players.
Performance
"The players are finding it beneficial because its showing things that they maybe weren't aware of and they can take that information away."
The statistical aspect is broken down into team performance and individual performance.
Against Middlesbrough two weeks ago, for instance, United made 291 passes of which 197 were successful. They also covered more distance than their Boro counterparts during the 90 minutes and maintained a solid workrate throughout.
The stats are broken down to the smallest detail, with a close inspection revealing that Leeds players covered a staggering 116km at the Riverside.
Jermaine Pennant was the top individual with 13.22km while Jody Morris was the team's main receiver in the middle of the park with 32 touches.
Each of those touches by Morris can be broken down by the click of the button and, aside from showing animations of his every move, there's also a detailed guide of what happened with each touch and at what stage it was made.
The individual figures are broken down to the extent that the system will detail how far each player has run during a game, how far they jogged and how much ground they covered in high speed sprints.
In short, it's all there for Peter Reid and Blackwell to review, analyse and take back to their players.
"It's a superb system," said Blackwell. "I think you have to adopt new technology, particularly when it is designed to help you look at specifics in detail.
"I think it's a valuable coaching aid and I'm surprised that more clubs haven't taken it on board."

11 September 2003

<< Back << >>

Search Our News Archive

Enter Search Text

Search Type
Match Any Word

Search Advanced Search

[Disclaimer] All rights reserved © 2002 Johnston Press New Media. Click here for full details.

Figure A.1: Article on new technology at Leeds United, from Leeds Today website.

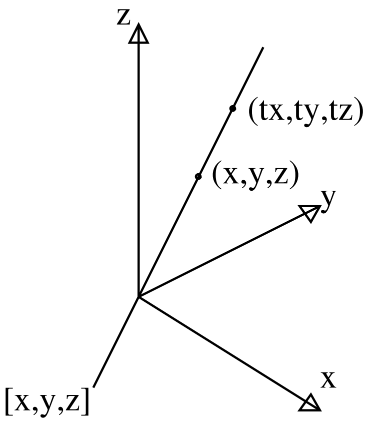
Appendix B

A projective transformation

This Appendix introduces the fundamental concepts of projective geometry, what a projective transformation is, and proves the uniqueness of the projective transform between two quadrangles in $\mathbb{P}^2$, which is a result necessary for the work undertaken in Chapter 3 - to transform points between the image and the world coordinate systems.

B.1 The projective plane

The *projective plane* (projective 2-space) $\mathbb{P}^2$, is the set of lines in $\mathbb{R}^3$ which pass through the origin,



$(x, y, z)^T \in \mathbb{R}^3 \backslash (0, 0, 0)$ determines a unique line through the origin.

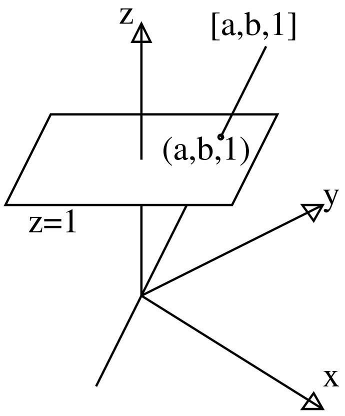
$$(tx, ty, tz) : t \in \mathbb{R}$$

This point of $\mathbb{P}^2$ is denoted by $[x, y, z]$.

Consider the plane $z = 1$ in $\mathbb{R}^3$. Each point in this plane determines a unique point of $\mathbb{P}^2$. $\mathbb{R}^2$ can be identified with a subset of $\mathbb{P}^2$

$$(x, y)^T \in \mathbb{R}^2 \leftrightarrow [x, y, 1] \in \mathbb{P}^2$$

Other points of $\mathbb{P}^2$ are of the form $[c, d, 0]$, and are lines in $z = 0$. Such a line corresponds to a 'direction' in $\mathbb{R}^2$.



B.2 What is a projective transformation?

Definition

$\phi : \mathbb{P}^n \to \mathbb{P}^n$ is a *projective transformation* if there is a non-singular $(n + 1) \times (n + 1)$ matrix $\mathcal{A}$ such that

$$\phi[P] = [\mathcal{A}P] \quad \forall\, [P] \in \mathbb{P}^n$$

Note

This is well defined, for if $t \neq 0$, $[tP]$ is the same point as $[P]$ and

$$\phi[tP] = [(\mathcal{A}t)P] = [\mathcal{A}(tP)] = [\mathcal{A}P] = \phi[P]$$

Also the matrix $t\mathcal{A}, t \neq 0$ gives the same transformation as $\mathcal{A}$.
ϕ is a transformation with inverse $\phi^{-1}[Q] = [\mathcal{A}^{-1}Q]$ as $\mathcal{A}^{-1}$ is non-singular.
The set of projective transformations of $\mathbb{P}^n$ is a group of transformations, for if $\phi[P] = [\mathcal{A}P]$ and $\psi[P] = [\mathcal{B}P] \quad \forall\, [P] \in \mathbb{P}^n$ with $\mathcal{A}, \mathcal{B}$ non-singular,
then: $\phi \circ \psi[P] = [\mathcal{A}\mathcal{B}P]$
So $\phi \circ \psi$ is a projective transform as $\mathcal{A}\mathcal{B}$ is non-singular.

B.3 A unique projective transformation between the image plane and the ground plane

Theorem

Given two quadrangles in $\mathbb{P}^2$ there is a unique projective transform which transforms one quadrangle to the other.

Proof

Put $X = (1, 0, 0)^T, Y = (0, 1, 0)^T, Z = (0, 0, 1)^T, E = (1, 1, 1)^T$.

$[X], [Y], [Z], [E]$ is a quadrangle since no 3 of X,Y,Z,E are linearly dependent.

It is sufficient to show that this quadrangle can be transformed to any other, say $[P_1], [P_2], [P_3], [P_4]$.

P_1, P_2, P_3 are linearly independent and therefore form a basis for $\mathbb{R}^3$

$\therefore P_4 = \alpha P_1 + \beta P_2 + \gamma P_3$ for some unique $\alpha, \beta, \gamma \in \mathbb{R}$.

α, β, γ are all non-zero for if not then 3 of P_1, P_2, P_3, P_4 would be linearly dependent.

Put $\mathcal{A} = (\alpha P_1, \beta P_2, \gamma P_3)$ and define $\psi[P] = [\mathcal{A}P] \quad \forall\, [P] \in \mathbb{P}^2$.

$\psi[X] = [\alpha P_1] = [P_1]$

$\psi[Y] = [\beta P_2] = [P_2]$

$\psi[Z] = [\gamma P_3] = [P_3]$

$\psi[E] = [\alpha P_1 + \beta P_2 + \gamma P_3] = [P_4]$

We now only need to show uniqueness:

If $\psi'[P] = [\mathcal{A}'P] \quad \forall\, P \in \mathbb{P}^2$ has the same properties as ψ then

$\psi'[X] = [\mathcal{A}'X] = [P_1] \Rightarrow \mathcal{A}'X = \alpha' P_1$

$\psi'[Y] = [\mathcal{A}'Y] = [P_2] \Rightarrow \mathcal{A}'Y = \beta' P_2$

$\psi'[Z] = [\mathcal{A}'Z] = [P_3] \Rightarrow \mathcal{A}'Z = \gamma' P_3 \quad$ for some α', β', γ'.

$\therefore \mathcal{A}' = (\alpha' P_1, \beta' P_2, \gamma' P_3)$.

Also $\psi'[E] = [\alpha' P_1 + \beta' P_2 + \gamma' P_3] = [P_4]$

$\therefore \alpha' P_1 + \beta' P_2 + \gamma' P_3 = \delta' P_4$ for some $\delta' \in \mathbb{R} \ (\delta' \neq 0)$.

$\therefore P_4 = \frac{\alpha'}{\delta'} P_1 + \frac{\beta'}{\delta'} P_2 + \frac{\gamma'}{\delta'} P_3$

By uniqueness of α, β, γ

$\alpha' = \delta' \alpha, \beta' = \delta' \beta, \gamma' = \delta' \gamma \quad \therefore \mathcal{A}' = \delta' \mathcal{A} \quad \therefore \psi' = \psi$.

Appendix C

Covariances and convolutions of Gaussian distributions

In Chapter 6, the measurement error covariance $\mathcal{R}_i$ for the Kalman filter for player i is calculated from the measurement error covariances $\mathcal{R}_{i,j}$ for player i in view j. This is worked out by combining all valid estimates of the bivariate Gaussian distributions that the covariances represent. This Appendix details how such covariances are combined.

Given n covariance matrices of dimension d, representing n multivariate Gaussian distributions all centred on the point $\boldsymbol{\mu}$ with covariance $\boldsymbol{\sigma}_i^2$. Summation of covariance matrices is commutative, as is multiplication in this case (since the covariance matrices are symmetric and d-square).

Take $f_i(\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}_i^2)$ for $i = 1 \dots n$.

$$f_i(\mathbf{x}) = \frac{\boldsymbol{\sigma}_i}{\sqrt{(2\pi)}} \cdot e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma_i}^{-2}(\mathbf{x}-\boldsymbol{\mu})} \tag{C.1}$$

$$\prod_{i=1}^{n} f_i(\mathbf{x}) = \frac{\prod_{i=1}^{n} \boldsymbol{\sigma}_i}{(\sqrt{(2\pi)})^n} \cdot \prod_{j=1}^{n} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma}_j^{-2}(\mathbf{x}-\boldsymbol{\mu})} \tag{C.2}$$

$$\prod_{i=1}^{n} f_i(\mathbf{x}) = \frac{\prod_{i=1}^{n} \boldsymbol{\sigma}_i}{(2\pi)^{\frac{n}{2}}} \cdot e^{-\frac{1}{2}\sum_{j=1}^{n}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma}_j^{-2}(\mathbf{x}-\boldsymbol{\mu})} \tag{C.3}$$

$$\prod_{i=1}^{n} f_i(\mathbf{x}) = \frac{\prod_{i=1}^{n} \boldsymbol{\sigma}_i}{(2\pi)^{\frac{n}{2}}} \cdot e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T (\sum_{j=1}^{n} \boldsymbol{\sigma}_j^{-2})(\mathbf{x}-\boldsymbol{\mu})} \tag{C.4}$$

$$\text{Let } \boldsymbol{\sigma}^{-2} = \sum_{j=1}^{n} \boldsymbol{\sigma}_j^{-2} \tag{C.5}$$

$$\Rightarrow \quad \boldsymbol{\sigma}^2 = (\sum_{j=1}^{n} \boldsymbol{\sigma}_j^{-2})^{-1} \tag{C.6}$$

Now

$$\prod_{i=1}^{n} f_i(\mathbf{x}) = \frac{\prod_{i=1}^{n} \boldsymbol{\sigma}_i}{(2\pi)^{\frac{n}{2}}} \cdot e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma}^{-2}(\mathbf{x}-\boldsymbol{\mu})} \tag{C.7}$$

$$\prod_{i=1}^{n} f_i(\mathbf{x}) = \frac{(\prod_{i=1}^{n} \boldsymbol{\sigma}_i)(\sum_{j=1}^{n} \boldsymbol{\sigma}_j)^{-\frac{1}{2}}}{(2\pi)^{\frac{n-1}{2}}} \cdot \frac{\boldsymbol{\sigma}^{-1}}{\sqrt{(2\pi)}} \cdot e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\sigma}^{-2}(\mathbf{x}-\boldsymbol{\mu})} \tag{C.8}$$

This shows that the convolution of several Gaussian distributions produce a new distribution proportional to $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$. Hence in our 2-D case $\boldsymbol{\sigma}_j^2 = \mathcal{R}_{i,j}$, and the covariance $\mathcal{R}_i$ resulting from convolving the Gaussian distributions which they represent has now been shown to be $\mathcal{R}_i = (\sum_{j=1}^{n} \mathcal{R}_{i,j}^{-1})^{-1}$. This is the value used as the measurement error covariance for a player in the multi-view tracker in Chapter 6.

Appendix D

Tracker results

This appendix contains the results of trajectory evaluation performed in Section 6.6 using the metrics developed in Chapter 5. Results are presented for six tracked players each when tracked using camera 1, camera 3 and cameras 1 and 3 together. All units are in millimetres.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1015	962	152	1881	536	118
$\mathcal{D}(T_A + (106, 792), T_B)$	704	676	56	1741	426	125
$\mathcal{D}(T_A, T_B, -16)$	697	673	253	1261	331	10
$\mathcal{D}(T_A + (42, 1118), T_B, 16)$	273	234	37	791	213	43

Table D.1: Camera 1 Track 1.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1754	1896	105	3062	942	668
$\mathcal{D}(T_A + (-186, 1671), T_B)$	914	830	140	2145	531	322
$\mathcal{D}(T_A, T_B, -16)$	1528	1141	126	2858	874	521
$\mathcal{D}(T_A + (-688, 1907), T_B, 16)$	372	375	58	1084	240	133

Table D.2: Camera 1 Track 2.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1522	1068	114	3891	1240	192
$\mathcal{D}(T_A + (218, 1221), T_B)$	1311	1313	33	2652	754	136
$\mathcal{D}(T_A, T_B, -16)$	1228	510	869	2082	372	292
$\mathcal{D}(T_A + (1014, 1694), T_B, 16)$	696	675	142	1292	329	183

Table D.3: Camera 1 Track 3.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	793	716	160	1589	505	321
$\mathcal{D}(T_A + (-172, -747), T_B)$	487	455	39	879	229	132
$\mathcal{D}(T_A, T_B, -9)$	423	431	29	788	260	366
$\mathcal{D}(T_A + (603, 80), T_B, -15)$	237	230	30	576	155	99

Table D.4: Camera 1 Track 4.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	294	281	18	582	172	82
$\mathcal{D}(T_A + (-171, 113), T_B)$	243	238	34	409	118	42
$\mathcal{D}(T_A, T_B, 11)$	261	197	35	519	170	138
$\mathcal{D}(T_A + (-345, 13), T_B, -16)$	122	113	40	260	51	12

Table D.5: Camera 1 Track 5.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	225	187	65	485	123	69
$\mathcal{D}(T_A + (141, -88), T_B)$	161	149	11	520	109	29
$\mathcal{D}(T_A, T_B, -1)$	219	187	63	501	122	69
$\mathcal{D}(T_A + (141, -154), T_B, 3)$	132	119	36	412	77	28

Table D.6: Camera 1 Track 6.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1082	1155	51	1971	628	333
$\mathcal{D}(T_A + (-452, 870), T_B)$	673	659	64	1447	378	294
$\mathcal{D}(T_A, T_B, -16)$	900	471	449	1510	398	298
$\mathcal{D}(T_A + (-263, 1300), T_B, 16)$	223	205	45	510	159	123

Table D.7: Camera 3 Track 1.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1191	1176	33	2340	793	350
$\mathcal{D}(T_A + (-531, 895), T_B)$	888	819	160	1805	403	393
$\mathcal{D}(T_A, T_B, -16)$	947	446	337	1934	450	136
$\mathcal{D}(T_A + (-755, 1271), T_B, 16)$	334	331	63	778	227	241

Table D.8: Camera 3 Track 2.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1481	1471	146	3107	938	819
$\mathcal{D}(T_A + (-270, 1358), T_B)$	926	833	166	1920	528	362
$\mathcal{D}(T_A, T_B, -6)$	1359	1032	233	2688	705	949
$\mathcal{D}(T_A + (-1494, 546), T_B, -16)$	402	333	106	858	242	155

Table D.9: Camera 3 Track 3.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	835	762	114	1615	456	628
$\mathcal{D}(T_A + (346, -743), T_B)$	429	460	9	800	211	229
$\mathcal{D}(T_A, T_B, -3)$	787	762	275	1513	398	662
$\mathcal{D}(T_A + (1438, 258), T_B, -16)$	335	318	43	792	190	102

Table D.10: Camera 3 Track 4.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	409	267	105	1285	352	59
$\mathcal{D}(T_A + (-281, -112), T_B)$	368	312	41	1026	250	80
$\mathcal{D}(T_A, T_B, 16)$	196	170	62	292	73	10
$\mathcal{D}(T_A + (80, 55), T_B, 16)$	180	178	114	282	40	27

Table D.11: Camera 3 Track 5.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	447	217	25	1009	372	96
$\mathcal{D}(T_A + (327, 244), T_B)$	395	377	173	606	111	69
$\mathcal{D}(T_A, T_B, 16)$	310	107	179	573	116	28
$\mathcal{D}(T_A + (17, -252), T_B, 16)$	195	179	75	331	84	43

Table D.12: Camera 3 Track 6.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1161	1063	152	2045	677	254
$\mathcal{D}(T_A + (-435, 881), T_B)$	800	818	55	1673	439	227
$\mathcal{D}(T_A, T_B, -16)$	1019	598	498	1665	392	204
$\mathcal{D}(T_A + (-217, 1180), T_B, 16)$	292	294	117	590	139	113

Table D.13: Cameras 1 and 3 Track 1.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1000	1016	116	1792	558	346
$\mathcal{D}(T_A + (-276, 747), T_B)$	716	761	135	1812	399	289
$\mathcal{D}(T_A, T_B, -16)$	777	533	67	1361	372	267
$\mathcal{D}(T_A + (-695, 1189), T_B, 16)$	382	365	130	949	209	141

Table D.14: Cameras 1 and 3 Track 2.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	1608	1333	91	3638	1149	325
$\mathcal{D}(T_A + (290, 1545), T_B)$	1060	1078	100	2070	533	103
$\mathcal{D}(T_A, T_B, -16)$	1069	716	624	1825	338	496
$\mathcal{D}(T_A + (1037, 2279), T_B, 16)$	537	546	102	895	206	148

Table D.15: Cameras 1 and 3 Track 3.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	504	505	105	894	231	295
$\mathcal{D}(T_A + (82, -472), T_B)$	245	230	39	582	127	84
$\mathcal{D}(T_A, T_B, -4)$	406	494	174	596	138	311
$\mathcal{D}(T_A + (182, -332), T_B, -2)$	231	211	47	571	113	76

Table D.16: Cameras 1 and 3 Track 4.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	683	426	92	1958	603	262
$\mathcal{D}(T_A + (-437, 372), T_B)$	618	635	91	1384	332	182
$\mathcal{D}(T_A, T_B, 16)$	330	184	22	636	219	165
$\mathcal{D}(T_A + (5, 247), T_B, 16)$	283	286	102	600	117	71

Table D.17: Cameras 1 and 3 Track 5.

Metric	mean	median	min	max	s.d	‘area’
$\mathcal{D}(T_A, T_B)$	329	277	35	699	208	97
$\mathcal{D}(T_A + (-138, 9), T_B)$	337	321	126	561	134	114
$\mathcal{D}(T_A, T_B, 7)$	259	210	32	693	208	29
$\mathcal{D}(T_A + (46, -237), T_B, 13)$	148	140	44	364	76	76

Table D.18: Cameras 1 and 3 Track 6.

Bibliography

[1] E. Andre, G. Herzog, and T. Rist. On the simultaneous interpretation of real world image sequences and their natural language description: The SOCCER system. In *Proc. European Conf. on Artifical Intelligence*, pages 449–454, 1988.

[2] S. Araki, T. Matsuoka, N. Yokoya, and H. Takemura. Real-time tracking of multiple moving object contours in a moving camera image sequence. *IEICE Trans. on Information and Systems*, E83-D:1583–1591, 2000.

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

[4] A. M. Baumberg and D. C. Hogg. An efficient method for contour tracking using active shape models. In *Proc. IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 194–199, Austin,Texas, 1994.

[5] A. M. Baumberg and D. C. Hogg. Learning spatiotemporal models from training examples. Technical Report Research Report 95.9, The University of Leeds, School of Computer Studies, 1995.

[6] T. Bebie and H. Bieri. SoccerMan - reconstructing soccer games from video sequences. In *Proc. Intl. Conf. on Image Processing*, pages 898–902, 1998.

[7] T. Bebie and H. Bieri. A video-based 3D-reconstruction of soccer games. In *Proc. Eurographics*, volume 19(3), 2000.

[8] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[9] M. J. Black and A. Jepsen. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *Intl. Journal of Computer Vision*, 26(1):63–84, 1998.

[10] A. F. Bobick and J. W. Davies. The recognition of human movement using temporal templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[11] A. Bottino and A. Laurentini. A silhouette based technique for the reconstruction of human movement. *Computer Vision and Image Understanding*, 83:79–95, 2001.

[12] R. Bowden, T. A. Mitchell, and M. Sarhadi. Reconstructing 3D pose and motion from a single camera view. In *Proc. British Machine Vision Conference*, pages 904–913, Southampton, UK, 1998.

[13] R. Bowden and M. Sarhadi. A non-linear model of shape and motion for tracking finger spelt American sign language. *Image and Vision Computing*, 20:597–607, 2002.

[14] K. J. Bradshaw, I. D. Reid, and D. W. Murray. The active recovery of 3D motion trajectories and their use in prediction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(3):219 –234, 1997.

[15] M. Brand. Shadow puppetry. In *Proc. Seventh IEEE Intl. Conf. on Computer Vision*, pages 1237–1244, Corfu, Greece, 1999.

[16] H. Buxton. Generative models for learning and understanding dynamic scene activity. In *First Intl. Workshop on Generative-Model-Based Vision*, pages 71–81, Copenhagen, 2002.

[17] S. Carter. MATH3231 Transformation Geometry. Lecture course in Mathematics at the University of Leeds, 1998.

[18] C. K. Chui and G. Chen. *Kalman Filtering with Real-Time Applications*. Springer, 1999.

[19] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *Proc. European Conf. on Computer Vision*, volume 2, pages 484–498, 1998.

[20] T. F. Cootes and C. J. Taylor. Active shape models - ‘smart snakes’. In *Proc. British Machine Vision Conference*, pages 266–275, 1992.

[21] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Training models of shape from sets of examples. In *Proc. British Machine Vision Conference*, pages 9–18, 1992.

[22] E. R. Davies. *Machine Vision*. Academic Press, 1997.

[23] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 126–133, 2000.

[24] S. L. Dockstader and A. M. Tekalp. Multiple camera fusion for multi-object tracking. In *Proc. IEEE Workshop on Multi-Object Tracking*, pages 95–102, 2001.

[25] R. M. du Plessis. *Poor Man's Explanation of Kalman Filtering or How I Stopped Worrying and Learned to Love Matrix Inversion*. Taygeta Scientific Incorporated, 1967.

[26] A. Ekin, A. M. Teklap, and R. Mehrotra. Automatic soccer video analysis and summarization. *IEEE Trans. on Image Processing*, 12(7):796–807, 2003.

[27] T. J. Ellis. Performance metrics and methods for tracking in surveillance. In *3rd IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, pages 26–31, Copenhagen, Denmark, 2002.

[28] T. J. Ellis, D. Makris, and J. K. Black. Learning a multiple-camera topology. In *Proc. Joint IEEE Intl. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[29] J. Fernyhough, A. G. Cohn, and D. C. Hogg. Building qualitative event models automatically from visual input. In *Proc. Sixth Intl. Conf. on Computer Vision*, Bombay, India, 1998.

[30] J. M. Ferryman, S. J. Maybank, and A. D. Worrall. Visual surveillance for moving vehicles. *Intl. Journal of Computer Vision*, 37(2):187–197, 2000.

[31] A. Galata, N. Johnson, and D. C. Hogg. Learning variable length Markov models of behaviour. *Computer Vision and Image Understanding*, 81:398–413, 2001.

[32] D. M. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proc. Seventh IEEE Intl. Conf. on Computer Vision*, pages 87–93, Kerkyra, 1999.

[33] A. Gelb and Technical Staff at The Analytical Sciences Corporation. *Applied Optimal Estimation*. The M.I.T. Press, 1974.

[34] Y. Gong, L. T. Sin, and C. H. Chuan. Automatic parsing of TV soccer programs. In *Proc. Intl. Conf. on Multimedia Computing and Systems*, pages 167–174, 1995.

[35] I. Haritaoglu, D. Harwood, and L. S. Davis. Active outdoor surveillance. In *Proc. Intl. Conf. on Image Analysis and Processing*, 1999.

[36] I. Haritaoglu, D. Harwood, and L. S. Davis. Hydra: Multiple people detection and tracking using silhouettes. In *Proc. Intl. Conf. on Image Analysis and Processing*, 1999.

[37] I. Haritaoglu, D. Harwood, and L. S. Davis. W^4: Real-time surveillance of people and their activities. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000.

[38] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[39] M. Harville. Stereo person tracking with adaptive plan-view statistical templates. In *Proc. ECCV Workshop on Statistical Methods in Video Processing*, pages 67–72, Copenhagen, Denmark, 2002.

[40] A. J. Heap and D. C. Hogg. Extending the point distribution method using polar coordinates. *Image and Vision Computing*, 14:589–600, 1996.

[41] T. Heap and D. C. Hogg. Wormholes in shape space: tracking through discontinuous changes in shape. In *Proc. 6th IEEE Intl. Conf. on Computer Vision*, pages 344–349, 1998.

[42] D. Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, March 1995.

[43] Pingsheng Huang. *Automatic Gait Recognition via Statistical Approaches*. PhD thesis, Department of Electronics and Computer Science, University of Southampton, 1999.

[44] S. S. Intille. Sport online. Technical report, M.I.T, May 1996.

[45] S. S. Intille and A. F. Bobick. Visual tracking using closed-worlds. In *Proc. Fifth Intl. Conf. on Computer Vision*, pages 672–678, Cambridge, MA, 1995.

[46] S. S. Intille and A. F. Bobick. A framework for recognizing multi-agent action from visual evidence. In *Proc. Nat. Conf. on Artificial Intelligence*, pages 518–525, 1999.

[47] M. Isard and A. Blake. Contour tracking by shochastic propagation of conditional density. In *Proc. European Conf. Computer Vision*, pages 343–356, Cambridge, UK, 1996.

[48] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. *Lecture Notes in Computer Science*, 1406:893–908, 1998.

[49] M. Isard and J. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *Proc. Eighth IEEE Intl. Conf. on Computer Vision*, volume 2, pages 34–41, Vancouver, Canada, 2001.

[50] O. Javed, S. Khan, Z. Rasheed, and M. Shah. Camera handoff: Tracking in multiple uncalibrated stationary cameras. In *Proc. IEEE Workshop on Human Motion*, pages 113–118, 2000.

[51] C. Jennings. Robust finger tracking with multiple cameras. In *Proc. Intl. Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems*.

[52] N. Johnson. *Learning Object Behaviour Models*. PhD thesis, School of Computer Studies, The University of Leeds, 1998.

[53] N. Johnson, A. Galata, and D. C. Hogg. The acquisition and use of interaction behaviour models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1998.

[54] N. Johnson and D. C. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14:609–615, 1996.

[55] I. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag, 1986.

[56] J. Kang, I. Cohen, and G. Medioni. Soccer player tracking across uncalibrated camera streams. In *Proc. Joint IEEE Intl. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pages 172–179, Nice, France, 2003.

[57] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*, 48:86–129, 1998.

[58] V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 253–259, Fort Collins, Colorado, 1999.

[59] T. Kim, Y. Seo, and K. Hong. Physics-based 3D position analysis of a soccer ball from monocular image sequences. In *Proc. Sixth IEEE Intl. Conf. on Computer Vision*, pages 721–726, Bombay, India, 1998.

[60] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78:1464–1480, 1990.

[61] E. Koller-Meier and F. Ade. Tracking multiple objects using the condensation algorithm. *Journal of Robotics and Autonomous Systems*, 34:93–105, 2001.

[62] M. Kubo and Y. Kakazu. Acquisition of the various coordinated motions of multi-agent system on soccer game. In *Proc. First IEEE Conf. Evolutionary Computation*, pages 686–691, 1994.

[63] L. Lee, R. Romano, and G.Stein. Monitoring activites from multiple video streams: Establishing a common coordinate frame. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):758–767, 2000.

[64] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Madison, USA, 2003.

[65] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proc. Seventh IEEE Intl. Conf. on Computer Vision*, pages 572–578, Corfu, Greece, 1999.

[66] D. R. Magee. Tracking multiple vehicles using foreground, background and motion models. In *Proc. ECCV Workshop on Statistical Methods in Video Processing*, pages 7–12, Copenhagen, Denmark, 2002.

[67] D. R. Magee and R. D. Boyle. Building shape models from image sequences using piecewise linear approximation. In *Proc. British Machine Vision Conference*, volume 2, pages 398–408, September 1998.

[68] D. R. Magee and R. D. Boyle. Building class sensitive models for tracking applications. In *Proc. British Machine Vision Conference*, pages 594–603, September 1999.

[69] D. R. Magee and R. D. Boyle. Detecting lameness using ‘re-sampling condensation’ and ‘multi-stream cyclic hidden markov models’. *Image and Vision Computing*, 20(8):581–594, 2002.

[70] F. Marzani, E. Calais, and L. Legrand. A 3-D marker-free system for the analysis of movement disabilities - an application to the legs. *IEEE Trans. on Information Technology in Biomedicine*, 5(1):18–26, 2001.

[71] S. J. McKenna, S. Jabri, Z. Duric, and H. Wechsler. Tracking interacting people. In *Proc. Fourth IEEE Intl. Conf. Automatic Face and Gesture Recognition*, pages 348–353, 2000.

[72] R. J. Morris and D. C. Hogg. Statistical models of object interaction. *Intl. Journal of Computer Vision*, 37(2):209–215, 2000.

[73] K. P. Murphy. Dynamic Bayesian networks. *To appear in Probabilistic Graphical Models, M. Jordan*, 2002.

[74] IEEE Intl. Series of Workshops on Performance Evaluation of Tracking and Surveillance (PETS). `http://visualsurveillance.org/` Last accessed: 1/10/03.

[75] Y. Ohno, J. Miurs, and Y. Sharai. Tracking players and a ball in soccer games. In *Proc. IEEE Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 147,152, 1999.

[76] H.-W. Ok, Y. Seo, and K.-S. Hong. Multiple soccer players tracking by condensation with occlusion alarm probability. In *Proc. of the Statistical Methods in Video Processing Workshop*, pages 1–6, Copenhagen, Denmark, 2002.

[77] N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. In *Proc. Intl. Conference on Computer Vision Systems*, number 1542 in LNCS, pages 255–272, Gran Canaria, Spain, 1999. Springer-Verlag.

[78] E-J. Ong and S. Gong. Tracking hybrid 2D-3D human models from multiple views. In *Proc. IEEE Workshop on Modelling People*, Corfu, Greece, 1999.

[79] J. Orwell, P. Remagnino, and G. A. Jones. Multi-camera colour tracking. In *Proc. IEEE Intl. Workshop on Visual Surveillance*, pages 53–68, Fort Collins, Colorado, 1999.

[80] N. Paragios and R. Deriche. A PDE-based level set approach for detection and tracking of moving objects. In *Proc. Sixth IEEE Intl. Conf. on Computer Vision*, pages 1139–1145, Bombay, India, 1998.

[81] J. Perš and S. Kovačič. Computer vision system for tracking players in sports games. In *First Intl. Workshop on Image and Signal Processing Analysis*, pages 81–86, Pula, Croatia, 2000.

[82] G. Pingali, A. Opalach, and Y. Jean. Ball tracking and virtual replays for innovative tennis broadcasts. In *Proc. Intl. Conf. on Pattern Recognition*, volume 4, pages 152–156, Barcelona, Spain, 2000.

[83] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[84] Y. Raja, S. J. McKenna, and S. Gong. Colour model selection and adaption in dynamic scenes. In *Proc. European Conf. on Computer Vision*, Freiberg, Germany, 1998.

[85] R. P. N. Rao. Robust Kalman filters for prediction, recognition, and learning. Technical report, The University of Rochester, December 1996.

[86] P. Remagnino, T. Tan, and K. Baker. Agent orientated annotation in model based visual surveillance. In *Intl. Conf. on Computer Vision*, pages 857–862, Bombay, India, 1998.

[87] G. Retz-Schmidt. A REPLAI of SOCCER: Recognizing intentions in the domain of soccer games. In *Proc. European Conf. on Artifical Intelligence*, pages 455–457, 1988.

[88] C. W. Reynolds. Flocks, herds, and schools: a distributed behaviour model. *Computer Graphics*, 21(4):25–34, 1987.

[89] G. Rigoll, S. Eickler, and S. Müller. Person tracking in real-world scenarios using statistical methods. In *IEEE Intl. Conf. on Automatic Face and Gesture Recognition*, pages 342–347, Grenoble, France, 2000.

[90] S. T. Rowels and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[91] H. Saito, S. Baba, M. Kimura, S. Vedula, and T. Kanade. Appearance-based virtual view generation of temporally-varying events from multi-camera images in the 3D room. In *Proc. Second Intl. Conf. on 3-D Digital Imaging and Modeling*, pages 516–525, Ottawa, Canada, 1999.

[92] H. Sakoe and S. Chiba. Dynamic Programming optimization for spoken word recognition. *IEEE Trans. Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

[93] Y. Seo, S. Choi, H. Kim, and K.S. Hong. Where are the ball and players? Soccer game analysis with color-based tracking and image mosaick. In *Intl. Conf. on Image Analysis and Processing*, Florence, Italy, 1997.

[94] J. Sherrah and S. Gong. VIGOUR: A system for tracking and recognition of multiple people and their activities. In *Proc. Intl. Conf. on Pattern Recognition*, volume 1, pages 1179–1182, Barcelona, Spain, 2000.

[95] J. Sherrah and S. Gong. Continuous global evidence-based Bayesian modality fusion for simultaneous tracking of multiple objects. In *Proc. Eighth IEEE Intl. Conf. on Computer Vision*, volume 2, pages 42–49, Vancouver, Canada, 2001.

[96] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conf. on Computer Vision*, pages 702–718, 2000.

[97] N. T. Siebel and S. Maybank. Fusion of multiple tracking algorithms for robust people tracking. In *Proc. 7th European Conf. on Computer Vision*, volume IV, pages 373–387, København, Denmark, 2002.

[98] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, second edition, 1999.

[99] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 246–252, Fort Collins, Colorado, 1999.

[100] G. P. Stein. Tracking from multiple view points: Self-calibration of space and time. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1521–1527, Fort Collins, Colorado, 1999.

[101] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Object localization by bayesian correlation. In *Intl. Conf. on Computer Vision*, volume 2, pages 1068–1075, 1999.

[102] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Bayesian object localisation in images. *Intl. Journal of Computer Vision*, 44(2):111–135, 2001.

[103] T. Taki, J. Hasegawa, and T. Fukumura. Development of motion analysis system for quantitative evaluation of teamwork in soccer games. In *Proc. Intl. Conf. Image Processing*, pages 815–818, 1996.

[104] K. Tanaka, I. Noda, H. Nakashima, and H. Matsubara. MIKE: An automatic commentary system for soccer. In *Intl. Conf. on Multi-Agent Systems*, pages 285–292, Paris, France, 1998.

[105] H. Tao, H. Sawhney, and R. Kumar. A sampling algorithm for tracking multiple objects. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 53–68, Corfu, Greece, September 1999. Springer-Verlag.

[106] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[107] K. Toyama. Probablistic tracking in a metric space. In *Proc. Eighth IEEE Intl. Conf. on Computer Vision*, volume 2, pages 50–57, Vancouver, Canada, 2001.

[108] J. Triesch and C. von der Malsburg. Self-organised integration of adaptive visual cues for face tracking. In *Proc. Automatic Face and Gesture Recognition*, pages 102–107, Grenoble, France, 2000.

[109] R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, FL, 1986.

[110] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 586–591, Maui, Hawaii, 1991.

[111] N. Vandenbroucke, L. Macaire, and J.-G. Postaire. Color pixels classification in an hybrid color space. In *Proc. Intl. Conf. on Image Processing*, pages 176–180, 1998.

[112] N. Vandenbroucke, L. Macaire, and J.-G. Postaire. Unsupervised color texture feature extraction and selection for soccer image segementation. In *Proc. IEEE Intl. Conf. on Image Processing*, Thessaloniki, Greece, 2001.

[113] N. Vandenbroucke, L. Macaire, C. Vieren, and J.-G. Postaire. Contribution of a color classification to soccer players tracking with snakes. In *Proc. IEEE Intl. Conf. on Systems, Man, and Cybernetics*, 1997.

[114] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical Report TR 95-041, University of North Carolina at Chapel Hill, 1995.

[115] A. Willis, R. Kukla, J. Hilne, and J. Kerridge. Developing the behavioural rules for an agent-based model of pedestrian movement. In *Proc. 25th European Transport Congress*, Cambridge, UK, 2000.

[116] A. Willis, R. Kukla, J. Kerridge, and J. Hilne. Laying the foundations: the use of video footage to explore pedestrian dynamics in PEDFLOW. In *Pedestrian Evacuation Dynamics*, pages 181–186, Duisburg, Germany, 2001.

[117] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[118] L. Xie, S.-F. Chang, A. Divakaran, and H. Sun. Structure analysis of soccer video with hidden Markov models. In *Proc. Intl. Conf. on Acoustic, Speech and Signal Processing*, volume 4, pages 4096–4099, Orlando, FL, 2002.

[119] C. Xu and J. L. Prince. Gradient vector flow: A new external force for snakes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 66–71, 1997.

[120] D. Yow, B. Yeo, M. Yeng, and B. Liu. Analysis and presentation of soccer highlights from digital video. In *Proc. Second Asian Conf. on Computer Vision*, pages 499–503, 1995.