Segmentation in image sequences: tracking human figures in motion

by

Frances Robertson

Submitted to the Department of Electrical Engineering
in fulfillment of the requirements for the degree of

Master of Science in Engineering

at the

UNIVERSITY OF CAPE TOWN

June 2001

Declaration

I declare that this thesis is my own work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination at this or any other university.

F.C. Robertson

Signature of Candidate

Cape Town

June 2001

Acknowledgements

I would like to thank the following people and institutions for their involvement with this project:

- My supervisor Professor Gerhard de Jager, for his enthusiastic support and helpful suggestions.
- The members of the Digital Image Processing Group at UCT for entertaining me when necessary.
- DebTech, a division of De Beers Consolidated Mines for their financial suppport, and for providing some of the image sequences.
- The National Research Foundation for financial support.

Abstract

This thesis addresses the issue of segmentation and tracking of human figures in video sequences. Various issues affecting segmentation are investigated, including selection of a colour space representation. Different methods for segmentation are implemented and their respective merits and disadvantages discovered. These are largely various forms of colour and motion segmentation. The different forms are then combined in order to produce a segmentation that is more effective than any of the component techniques on its own.

The tracking problem is addressed separately through the use of a *Smart Room*, where the segmentation can be easily performed. The tracking involves estimating a trajectory for a given person over time in the image frame co-ordinate system. This also includes distinguishing one person from another using low-level image features which have been extracted from each person over time.

Suggestions are made for simple modifications which will allow the segmentation and tracking to be performed simultaneously, and these modifications are shown to be an improvement on the original tracking algorithm.

Results indicate that the segmentation algorithm produces segmented human figures which are reliable enough for low-level image features to be extracted from them. The centroid position over time is used as an estimate of segmentation reliability. The tracking results show that the tracking algorithm is effective as long as people's clothing colours are sufficiently varied and that every expected person in the scene can be matched to a unique segmented blob. Complicating factors in tracking which are evident from the results are: attempting to match many people to one segmented blob and segmentation inaccuracies which lead to ambiguities in the occlusion reasoning process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Automatic segmentation and tracking of people in video sequences is an important task in computer vision which has been receiving a lot of attention especially in recent years. The applications of both of these include:

- Video conferencing.
- Automated and semi-automated surveillance for security purposes.
- Gait analysis for medical purposes.
- Analysis of sporting activities: for individuals' motions, for example, analysis of a cricketer's bowling technique, and for team game strategies such as soccer or handball [57] [56].
- Other automated video analysis, for instance, to compile statistics about people's activities and movements.
- Gesture recognition systems, for automatic sign language recognition, for example.
- Video compression.
- Intelligent environments which use vision to sense what the user is doing and react to these actions [81] [41].

1.2 Problem formulation

Segmentation in this instance is merely the task of classifying picture elements (pixels) in an image as person or background, with the ultimate aim of extracting silhouette images of human shape which are as accurate and smooth as possible. This is something a human can do easily, although to formulate a set of rules by which this classification can be made automatically by computer proves a lot more difficult. Tracking involves keeping a record over time of the location and path of motion of the connected objects which have been identified as people through the segmentation process.

Segmentation is thus a different problem from tracking, although the two are closely linked. Tracking can be performed without explicitly performing segmentation, if curves or templates in the image are tracked and an approximate location for an object is found. Likewise, segmentation can be performed without tracking. For segmentation it is sufficient to identify all pixels that correspond to humans without being able to provide any information about their identities or trajectories in time and space.

In this thesis, the approach is taken of segmenting the person or people present in the scene as accurately as possible using a combination of image information. Spatial, colour and motion models are kept of the background and of the foreground objects or people for this purpose. Once people have been identified as such, features are extracted from the segmented people which allow them to be tracked in subsequent frames and information to be made available about the paths they follow and changes they undergo. This approach makes the tracking completely reliant on the efficiency of the segmentation. Other approaches, such as tracking using eigenspace decomposition of high dimensional features [49] are not considered.

The problem of segmentation is not yet reliably solved for the general case, and different algorithms are used for different purposes and in different environments. Problems facing segmentation are shadowing of both foreground and background objects, camera motion, moving objects in the background, occlusion and lighting variations: both spatially within an image, and temporally within a sequence.

In order to make the problem solvable, certain constraints may be applied to restrict the problem domain, as certain complicating elements may be removed almost entirely. The most extreme example of this is to control the background completely so as to make it as simple as possible to remove. A method of doing this is discussed in chapter 7.

For this thesis a single stationary camera is assumed. In this way camera motion does not need to be taken into consideration and no use need be made of stereo images. It is also assumed that only slowly moving background objects are present, and that we have some

prior knowledge of the initial appearance of the background[1]. Problems that remain are: to compensate for lighting changes over time, and to deal with shadowing and occlusion.

1.3 Outline of thesis

In **chapter 2** a broad survey of the literature of the last decade is presented. Various methods of attacking the segmentation problem are outlined, ranging from colour-based segmentation, through stereo and gradient-based methods to motion-based methods. Some segmentation techniques which have specifically been applied in the context of tracking are also mentioned and described.

Chapter 3 presents the issue of colour vision with particular regard to the human visual system. In this chapter various colour spaces used in computer vision are also introduced, and a colour space in which to perform the segmentation and tracking for this project is selected. An experimental evaluation of certain colour spaces is also performed, by applying a simple segmentation algorithm for skin colour detection in the selected colour spaces and ranking the results.

Colour segmentation is addressed in **chapter 4** and Gaussian mixture modelling is introduced as a method of estimating colour probability distributions to model the foreground and background. An algorithm for selecting the number of components for a Gaussian mixture is described and a method for adapting the model to compensate for changes in background and foreground appearance over time is discussed. The colour segmentation module of the developed algorithm is described.

Chapter 5 discusses briefly motion estimation and more specifically change detection in order to locate and segment objects in a scene. A Kalman filtering method for updating a reference background is developed, and used to locate a candidate region in an image to be tested for the presence of a person.

In **chapter 6** two different methods of combining the outputs from a change detection module and a colour segmentation module are proposed and evaluated.

Chapter 7 raises the question of tracking, once segmentation has been performed. The *Smart Room* concept is introduced and a prototype system is described which performs the tracking once segmentation has been isolated through the use of a controlled environment. Some results from this system are presented, which show situations under which the tracking is largely successful as well as conditions which contribute to a low success rate, and drawbacks of the system are discussed.

[1]For this it is sufficient to assume that there will not always be people in the room, but that we are able to obtain frames consisting of only the background image. This seems a reasonable assumption in most situations.

In **chapter 8** further work is described which involves modifications to the existing tracking algorithm which will allow the segmentation and tracking algorithms to be combined.

In **chapter 9** results from the segmentation and tracking are presented separately. The segmentation accuracy is estimated indirectly by measuring the departure from smoothness of the trajectory of the centroid of the segmented person over time. This method would lead one to believe that the segmentation algorithm is mostly effective, although the segmentation results are difficult to quantify. Tracking results are divided into two sub-categories and show a high degree of accuracy for the simpler category of possible tracking error and poor accuracy for the more complicated situation. In **chapter 10** conclusions are drawn and some recommendations for improving the system are made.

The hardware used is described in Appendix B.

Chapter 2

Literature review: tracking and segmenting humans

This chapter is a broad review of some of the current literature concerning the problem of segmentation of humans in video sequences. Some of the publications discussed in the chapter also concern the tracking of people, although some form of segmentation is usually required before the tracking can take place. If the segmentation is performed in order to perform tracking this is made explicit in the description of the system. This is not intended to be a complete review of all the work done in the field, as this would be lengthy and cover a large amount of material which is irrelevant to this thesis, so instead mention is made of the various different approaches that have been taken, together with the most important examples of each approach.

The relevant literature is split into categories, according to the primary information used to perform the segmentation. These can each be split into several sub-categories. Section 2.1 describes some colour-based techniques for both segmentation and tracking. Section 2.2 describes methods which use stereo vision to perform an initial segmentation and various feature extraction methods for tracking. Gradient-based methods for segmentation are discussed in section 2.3. In section 2.4 segmentation from motion is discussed and finally in section 2.5 methods of segmentation combining motion with other information, such as shape and colour, are described.

2.1 Colour-based techniques for segmentation and tracking

Most segmentation algorithms which rely heavily on colour information use a model of the colour distributions of either the body, the background or both. These are usually updated

with time in order to try to compensate for changes in lighting. The discussion of colour-based techniques that follows is split into two: simpler colour based techniques which assume single Gaussian distributions for colour, and techniques using Gaussian mixtures in various ways. The latter, of which there seems to be a plethora in the current literature, is particularly relevant to this thesis, given the method used for colour tracking which shall be described in chapter 3.

2.1.1 General methods using colour

Pentland [81] and his colleagues at MIT have developed a system called *Pfinder* for tracking the human body in real-time.[1] The system segments and tracks a single person at a time using a single stationary camera. The initialisation process consists of building a model of the scene by examining frames known to contain no people and representing the colour distribution at every pixel as a Gaussian with mean μ and full covariance matrix. Pixels in the background which are not occluded by the foreground are updated at every frame t to compensate for small lighting changes, as shown in equation 2.1 below, where y is the vector containing the position and colour information of the point and α a weighting factor .

$$\mu_t = \alpha y + (1 - \alpha)\mu_{t-1} \tag{2.1}$$

A person entering the scene is detected when the Mahalanobis distance (formally defined in equation 8.1 on page 96) in a three-dimensional YUV colour space exceeds some threshold for a large region of pixels. A blob model of the person is built up by identifying the head, hands and feet of a person when he or she enters the scene. Feature vectors are formed using the spatial and colour co-ordinates clustered to form a collection of blobs. The person is modelled as a Gaussian distribution:

$$P(O) = \frac{exp[-\frac{1}{2}(O-\mu)^T\Sigma^{-1}(O-\mu)]}{(2\pi)^{\frac{m}{2}}\Sigma^{\frac{1}{2}}} \tag{2.2}$$

where μ is the mean value (x, y, Y, U, V) of each blob and Σ the covariance matrix.

To analyse each image, the spatial model for each blob is updated to predict the spatial distribution for the current image, using a state vector of position and velocity and assuming Newtonian laws of motion. Secondly the log-likelihood of membership to the blob model or the background is calculated for each pixel and the pixels are assigned to the most likely class. A binary support map is kept to show which pixels belong to each blob in the blob model and which to the background map. Finally the statistical models for both the blob and background

[1] Pfinder operates at 10 frames a second.

models are updated.

Fieguth and Terzopoulos [17] use only colour for the tracking problem, without explicitly performing a segmentation. Regions are matched to target regions using colour ratios in each colour band. If the ratios in the three colour bands are roughly equal, the fit is a good one. Occlusion with objects is explicitly addressed instead of relying purely on the level of detail provided by the object models.

2.1.2 Methods using Gaussian mixture models and colour

Raja and McKenna [62] use Gaussian mixture models, as is also done in [52], to estimate probability distributions of the person and the background. These models are then used for the segmentation and tracking. The models are dynamically updated to account for lighting changes, and the adaptation process is made selective by calculating the log-likelihood of the data, so that the tracked person is not lost by the tracking system.

The person is modelled using a Gaussian mixture model, the order of which is selected automatically to appropriately represent the training data. The mean, variance and prior probabilities for each Gaussian component are selected using the expectation maximisation algorithm (hereafter referred to as the EM algorithm).

Korhonen et al. [40] model the background using a non-adaptive mixture of Gaussians. Candidate foreground regions are identified and then tracked using colour, position and velocity features and a Kalman filter to estimate the next state of the object. The tracking is done by assigning each object to the hypothesis which is closest in a Mahalanobis sense. The measurement error for the filter is determined using the variance of the measured values. Again, no use is made of spatial information.

In work by Grimson and Stauffer [75], an adaptive Gaussian mixture model is also used, this time to model the background only. Each pixel belonging to the background is modelled as a mixture of Gaussians. They argue that it is easier to model a background pixel than foreground, as being static it produces less variance in colour. K (a number between 3 and 5) Gaussians are used to model the most recently observed pixel values at a point. Each time a new pixel value is observed, it is matched to one of the Gaussian components if it is within 2.5 standard deviations of the distribution. The matched distribution then has its parameters updated in favour of the newly observed data point. If no match is made, the Gaussian with the lowest prior is replaced by a Gaussian centred at the new data point, with a high variance and a low prior. The Gaussian distributions in the model are ordered according to highest prior and lowest variances: these are the distributions most likely to be responsible for background colours. Anything below a certain threshold is considered foreground. In this way the background model is continuously updated and foreground classification occurs in each

frame.

Khan and Shah [37] use a background model consisting of the mean and covariance of colour values observed at each pixel, and construct a Gaussian mixture to model the colour classes of a person entering the scene. The mixture models corresponding to the peoples' colour distributions are updated when people are not involved in an occlusion and are used to segment each person from the background as well as to decide which person is which during an occlusion. This is done by assigning each pixel to the class which produces the maximum *a posteriori* probability for the models.

Friedman and Russell [18] use a probabilistic approach to model a single pixel's appearance when it forms part of different classes, as an attempt to solve the shadowing problem encountered when modelling the background simply with its mean and covariance at each pixel. The segmentation is applied to sequences of moving vehicles for traffic surveillance purposes. Each pixel in this model is background for some of the time, in shadow for some of the time, and for the remaining time forms part of a moving foreground object. The pixel's appearance over time is therefore a weighted sum of three distributions and can be modelled as a mixture of Gaussians with three components, the components being the foreground, shadow and background classes. The mixture model is initialised with weak priors and an incremental version of the EM algorithm is used to update it once new data becomes available. The components of the model are labelled with the classes to which the Gaussians are assumed to correspond: the darkest being the shadow component, the one with the largest variance the vehicle class and the remaining one the background. The pixels are then classified according to the current model. This is an example of learning a Gaussian mixture model from incomplete data. Gaussian mixture modelling is discussed in greater detail in chapter 4.

2.2 Segmentation techniques which make use of stereo

Another real-time system[2] developed at the University of Maryland, W^{4}S [23] makes no use of colour either for segmentation or for tracking. Instead, it makes use of intensity images and disparity images obtained through a stereo vision system. The system is intended for use with a stationary camera, but is able to track multiple people through occlusions and other interactions through the use of stereo analysis. The background is modelled in the same way in the intensity image and in the disparity image obtained through the stereo analysis module. Each pixel is modelled by the maximum and minimum values appearing over a period of time, as well as the maximum change observed between successive frames. These values are updated for pixels which are identified as background.

[2] W^{4}S operates at between 5 and 20 frames a second depending upon how many people are present in the scene.

Foreground regions are obtained by thresholding both the intensity and disparity images where they differ from the background model. This is followed by noise cleaning and morphological filtering before the objects detected in the two images are resolved. Detected objects are matched to existing objects using shape analysis, template matching and spatial occupancy by testing the overlap between the predicted positions and the locations of the objects. Motion models are used to predict the location of objects in successive frames.

The use of stereo as well as intensity helps to eliminate problems which occur in only the one or the other image type. The segmentation of the disparity image is insensitive to short term illumination changes and shadowing and keeps foreground regions intact, whereas the intensity segmentation is more efficient if there is not enough texture in the background to produce a reliable disparity image and provides a more accurate silhouette outline of the foreground object.

Darrell et al. [11] (Interval Research Corporation) combine stereo, colour and face detection. They also make the distinction between short term, medium term and long term tracking, using more persistent features such as face pattern for long term tracking and position and velocity for short term tracking. Depth estimation is used for segmentation, colour segmentation to detect skin in the segmented regions and to model clothing colours and intensity pattern classification to detect faces.

Researchers at Microsoft [41] likewise use stereo to locate people and colour to identify them. The system works at 3.5Hz. A person's colour identity is maintained through use of a histogram and histogram intersection [5] is used to calculate the similarity between histograms obtained from the segmented image and stored histograms. A person creation/deletion zone is constructed, representing valid routes of entering and leaving the room.

2.3 Gradient-based methods

The methods discussed in this section are generally based only on spatial gradient information. Although methods based on motion estimation make use of spatial gradient information they are discussed under motion segmentation in section 2.4.

Segmentation

Bichsel [4] [3] uses simply-connectedness of an object and the approximation that local image derivatives within textured regions and along object contours show a Laplacian distribution, to segment moving objects. The spatial derivative of the brightness of the image is calculated for each point in the image. The difference image between two successive gradient images

is formed and the distribution should remain Laplacian, especially along the contours of the moving object. The logarithmic local probability that a pixel belongs to the background, given that at least one neighbouring pixel belongs to the background, is calculated for every pixel. The local background probability is then optimised using global information, incorporating the simply-connectedness of the background.

Westberg [80] also uses edge-based segmentation. A single simply connected moving object without holes is also assumed to be present in the scene in this case. He performs a hierarchical segmentation based on the refinement of object boundaries. The difference image is used in this method, and divided into blocks which are classified as inside the object, outside the object or object boundaries. Each block is then split into smaller child blocks. The children of a block that has been classified as inside or outside are automatically classified the same as the parent, but the child blocks of boundary blocks are reclassified.

This process applied iteratively breaks the image down to blocks which are pixel-sized and every pixel is classified as inside, outside or border, where between an innermost outside pixel and an outermost inside pixel there is one pixel that has been classed as a border pixel. This is obviously used for a segmentation whereby the object consists of the border pixels and all the pixels classed as inside pixels.

Jabri, Duric and Wechsler in [34] present good segmentation results using a combination of background differencing and edge information. A background model is built for each colour channel and for the horizontal and vertical edges of the background image. Subtraction is performed for each of these models and a confidence normalisation procedure is used to create a confidence map between two thresholds for each changed region. The maximum for each pixel between the edge difference confidence map and the colour difference confidence map is then chosen and thresholded to segment the foreground object. The use of the edge difference image makes this a robust algorithm in clutter, although its efficacy seems to rather depend upon edges being present in either the foreground, background or both. However since it is combined with a colour difference image, presumably in the absence of clutter the colour difference image would perform reliably enough to compensate for this.

Tracking

Huttenlocher et al. [29] use a two dimensional model to track. The models are shape and motion, so an important constraint is that the object's shape not change radically between frames. The location of the object is not constrained. The object model and image in which it is searched for are binary images (which are edges detected using the Canny edge detector). First stationary background edges are removed from the image, then the best matching position is found for the model from the previous frame using a partial directed Hausdorff

distance measure. Ambiguities are resolved by using the direction of motion. The new model is selected by using the pixels of the frame that are within a threshold distance, δ of the translated model of the previous frame.

The method used in [5] for tracking heads in a video sequence combines the outputs of two different and complementary modules. The gradient of the intensity along the border of a region and the colour histogram of the interior of the object are used to this end. The head shape is modelled by an ellipse with a fixed aspect ratio and tracking is done by finding the region where the values of the image best match the model values. The search for the head is done within range of the predicted location calculated using a velocity estimate. The normalised sum of the perpendicular gradient magnitudes around the perimeter of the ellipse is calculated for each predicted possible ellipse position and axis length. The colour histogram tracking is initialised by the user presenting his/her face to be modelled before the tracking begins, and the intersection between the histogram at the hypothesised head position and the model histogram is calculated for tracking. The goodness of match of the combination of the two criteria: interior colour and border gradient, is calculated by converting the two into percentages and seeing at which predicted head location the sum of these is a maximum.

2.4 Motion segmentation

The simplest form of motion segmentation is change detection, which is performed using two images. This can be either an image and an empty background image, which should reveal the entire object, or the difference between two consecutive images, which reveals the small area of change, space that the object did not occupy in the previous frame and background which was covered by the object in the previous frame and has become uncovered. Several algorithms that use image differencing construct an adaptive reference image [36] [63] or median [66] or mode [71] filtering in the time domain, so that lighting changes are modelled and the background image is updated as the foreground image changes. Others use statistical models and Markov Random fields [1] [77] to refine the image differencing operation. Differencing operations which are performed on a pixel-wise basis are subject to noise and ideally some of the global or local spatial data in the image should be taken into account [74] [45] [33].

Motion estimation can be performed either using spatial and temporal gradient-based methods or feature-based methods, such as points, lines and edges. Gradient methods need additional constraints and work better for smooth objects as the constraints impose uniform conditions which increase the error at the boundaries of moving objects. Features are difficult to extract robustly for feature correspondence establishing techniques, so both methods have disadvantages. Most forms of motion segmentation depend upon the computation of optical flow and motion estimation between frames of a sequence. These techniques are explained in chapter

5.

2.4.1 Image differencing

The simplest form of change detection is a threshold on the difference image between a frame and the previous frame, or between the frame and an empty background reference image.

Leung and Yang [44] use the difference of two consecutive images and an adaptive thresholding technique based on the histogram. Yalamanchili et al. [82] describe a difference picture system which uses the grey level difference between two consecutive images to extract descriptions of moving polygonal objects. In [82] the difference image obtained is used as a basis for a region-growing operation, constrained by geometrical observations, within the region of difference.

Skifstadt and Jain [74] propose two techniques to better previously existing techniques for image differencing. The three existing techniques they describe are first, the "Geo-pixel method" where regions, rather than pixels, are compared according to the likelihood ratio shown in equation 2.3. If L exceeds some threshold the two corresponding regions in the successive images are not considered regions of change.

$$L = \frac{[(\sigma_1^2 + \sigma_2^2)/2 + ((\mu_1 + \mu_2)/2)^2]^2}{\sigma_1 \sigma_2} \tag{2.3}$$

Secondly, the "Quadratic Picture Function Model" models the grey level distribution in each region as a function in order to obtain a difference measure [35]. Thirdly, they describe a grey scale normalisation technique which normalises the grey values G of a corresponding region i in images A and B so that the normalised values can be subtracted to allow for lighting change.

$$G_{Norm} = \frac{\sigma_A(i)}{\sigma_B(i)}[G_B(x, y) - \mu_B(i)] + \mu_A(i) \tag{2.4}$$

Their improvement on the "Quadratic Picture Function method" is to use the partial derivatives with respect to x and y of the quadratic function model for the same region in consecutive images and subtract these from each other.

The other approach they describe is, like the grey level normalisation, an attempt to account for illumination changes to extract only moving objects. Each image is split into many sub-regions and the variance of the ratio of the two intensities is calculated. If this is near zero the region has not changed, and if it is much greater than zero the region is assumed to have undergone change. This method is effective as it relies upon illumination ratio information and not absolute luminance difference values.

Lim et al. [45] use the simple image differencing method and follow this by fitting ellipses to the contours in order to classify objects. Kuno [42] uses an extraction function of image and background shown in equation 2.5, which shows remarkable improvement on the simple background subtraction in background/foreground separation.

$$f(A,B) = 1 - \frac{2\sqrt{(A+1)(B+1)}}{(A+1)+(B+1)} \times \frac{2\sqrt{(256-A)(256-B)}}{(256-A)+(256-B)} \tag{2.5}$$

Here A is the image with the object present, and B the background image. The grey-scale values of the images range from 0-255.

Ivanov, Bobick and Liu [33] use a multiple camera method of eliminating the effect of shadows when the background is subtracted. The current image is geometrically warped on to the corresponding pixel of a reference image, which is taken by a different camera at a different view. If the colour and luminance of the two pixels are similar the pixel is classified as background; if not, it is either an object pixel or an occlusion shadow pixel. The shadowed pixels are eliminated by noting that object pixels will appear different from the background in views from all cameras, whereas shadowed pixels are only likely to appear different in one view. This rather relies on there not being many objects casting different shadows in the scene, which would make it difficult to resolve shadow ambiguities.

Intille, Davis and Bobick [31] use background subtraction in the chrominance bands only of a colour image to extract blobs which are matched from frame to frame using various statistics drawn from the blob. This is done for the purpose of tracking the blobs in future frames. In [71] a background is recovered from a scene by mode filtering in the time domain: each background pixel is set to the most frequently occurring value over a period of time.

In [8] an image differencing method is used to locate the background reference image. A forward and a backward difference image is taken and the regions which are common to both is where the moving objects lie. The background is estimated over a few frames by including pixels which do not lie within the estimated regions. The background is used for locating moving objects. This is done both for moving and stationary viewing systems; in the moving case the motion of the viewing system is estimated and compensated for to find the background image. Tracking is done much like [31], by matching tracked objects with similar size, velocity and colour attributes.

Cai, Mitiche and Aggarwal [8]; Intille, Davis and Bobick [31] and Rosin and Ellis [66] find blobs using image differencing and use these to estimate locations of people and to match to models.

Paragios and Tziritas [54] detect change from two consecutive images in a sequence. They state that the inter-frame difference alone is not sufficient to locate precisely the boundaries

of a moving object, and propose to model this as a mixture of Laplacian distributions and use a maximum a posteriori probability criterion to adaptively determine the threshold.

2.4.2 Simultaneous motion estimation and segmentation

An important criterion in motion estimation and segmentation techniques is the estimation of motion boundaries. This leads to a fundamental problem in motion estimation and segmentation: a good estimation provides a good segmentation and a good segmentation produces an accurate motion estimation. This fact is used in different ways for motion estimation. Some researchers take into account knowledge of the structure of an image and use prior information such as regions [71] or lines [77]. Others create an initial estimation to produce a segmentation and then use this segmentation to better the initial estimation [10].

Shio and Sklansky [71] concentrate on trying to analyse and model the average motion of a human body. Regions should be grouped according to how they move. A region that includes different velocities should be split. An object model is needed to distinguish between objects that are different but move in the same way. They present an algorithm for first motion estimation and then person segmentation. The motion is estimated from pairs of images in the sequence. First regions with similar grey levels are extracted using a difference image (this is a feature-based motion estimation). The motion field is obtained around the edges using a correlation method. The flow field thus obtained is spatially smoothed within the object boundaries, and temporally smoothed so that all moving parts converge to a global value. The segmentation stage consists of region splitting and grouping based on the direction of motion.

Tian and Shah [77] use mean field technique to determine boundaries and optical flow. A flow field is calculated in their approach and horizontal and vertical line fields are computed to help the location of discontinuities in the flow field. They use a Markov random field representation to deal directly with discontinuities in the line fields.

Cloutier, Mitiche and Bouthemy [10] use an affine motion model shown in equation 2.7, together with the gradient optical flow equation shown in equation 2.6, to produce a linear system of equations which constrain the optical flow to local rigid motion. A least median of squares method is used to find outliers that lie on different sides of motion boundaries.

$$f_x u + f_y v + f_t = 0 \tag{2.6}$$

$$\begin{cases} u = a + \alpha x + \gamma y \\ v = b + \beta x + \delta y \end{cases} \tag{2.7}$$

Shi and Malik [70] use "normalised cuts" on a graph constructed by treating the pixels in an image as nodes, and connecting ones which show some spatio-temporal similarity. The edges are weighted according to their motion profiles. The graph is then recursively partitioned using their normalised cut measure which minimises the sum of edges that need to be disconnected to partition the graph. This technique is very useful in the case of a non-stationary camera.

2.5 Segmentation combining motion and other information

Mitiche and Aggarwal [48] emphasise the need to integrate different types of image information to produce better and more meaningful segmentation methods. The importance of making assumptions which fit the data, and of keeping in mind the goal of the final segmentation is stressed. Motion information can either be used to localise an area of interest or to make a segmentation more robust. Motion can be combined with intensity or colour information, or with shape and edge information.

2.5.1 Motion and intensity or colour

Thompson [76] uses a method which combines the use of difference measures and grey scale, rather than colour, information. Only translational motion is accounted for in this model, which helps to distinguish the boundaries of moving objects. For each point, rather than each feature, in the image the velocity is estimated and this information is then combined with grey scale boundary information, so that the image is separated into regions with different motion characteristics. Only two frames are used to find object boundaries in this manner.

The velocity information is the relation of the time variation of intensity at a point (because of motion) and the spatial variation over a surface. The spatial variation is given by a gradient, G, so a change in intensity due to motion is given as

$$\frac{di}{dt} = -(G_x v_x + G_y v_y) \tag{2.8}$$

where $\frac{di}{dt}$ is change in intensity with time and v_x and v_y are the x and y components of the velocity respectively. Two consecutive frames are used to calculate the velocity map values and from one of these frames the static boundaries are located. Regions of pixels with the same grey scale and velocity map values are combined and then these regions are merged according to proximity and similarity. This method only deals with translational motion and although it could be extended to deal with rotational motion, it is inappropriate for segmenting people in motion.

Dubuisson and Jain [14] combine image subtraction and a colour segmentation obtained from

a split-and-merge algorithm and the Canny edge-detector, to segment the contours of moving objects. An object mask is obtained using an image differencing technique, and a colour segmentation is obtained by splitting and merging square regions according to their means and variances and using edge information from the Canny edge detection algorithm. The collection of regions produced by the colour segmentation is merged according to the proportion of pixels from each region which also form part of the object's motion mask.

Moscheni et al. [50] propose a region-merging segmentation technique for video sequences. The input to the algorithm is an over-segmentation of the scene, which can be obtained using any conventional segmentation technique. The segmentation uses both the motion and brightness information to split the scene into regions which represent moving objects. The spatial similarity between two regions is calculated using the medians of the luminances of two regions along their common border, for every combination of regions in the image.

The motion similarity, for simplicity, is calculated only for adjacent regions, using two frames of a sequence. Region merging is performed according to a graph-based strategy where each node in the graph represents a region to be merged and every edge represents the spatiotemporal similarity between the two regions. The graph is thresholded so that similarities below a certain threshold are judged insignificant and ignored. By looking for cycles in the graph, regions which are more similar to each other, but less similar to other regions are merged together. A second merging strategy is subsequently performed to merge small or badly-defined regions with larger, already merged regions.

Healey [24] uses a hierarchical segmentation of two images and matches regions from one to the other using a segment tree, so that regions at different scales can be matched by matching different levels in the tree. The segments are characterised by size, mean, variance and centre. The motion of matched regions is estimated and target regions that show different motion from the background are identified. This is in effect a sort of region merging.

Lin et al. [46] combine an initial motion segmentation with statistical spatial segmentation. A hierarchical principal component split algorithm is implemented for the motion segmentation and a voting region is established by region-growing from the centre of a clustered region until an edge is located. Likewise [44] uses a voting scheme based on co-incidence regions from a static and a differencing segmentation scheme and a history of recorded parts.

Pers and Kovacic [57] use motion detection in the form of background subtraction and template matching and colour for tracking. For the background subtraction each field in the RGB image is subtracted from a reference frame. The differences are added together then thresholded.

2.5.2 Motion and shape

Huttenlocher et al. [29] use motion and shape in order to track. Their algorithm has been discussed under gradient-based methods in section 2.3.

As a closing to this chapter, mention must be made of the CONDENSATION (Conditional Density Propagation) algorithm proposed by Isard and Blake [32], which uses shape and motion to track curves in substantially cluttered backgrounds. This method has proved more effective for tracking than Kalman filters, which apply only to Gaussian densities and thus do not work well in clutter. This algorithm combines a statistical factored sampling technique, in which a random set represents the distribution of possibilities, with a stochastic differential equation for object motion. The probability distribution for shape and position is propagated over time in this way. The tracking resulting from the use of this algorithm is extremely robust even in very cluttered backgrounds.

In the following chapter the use of colour in image segmentation is discussed; including descriptions of human colour vision, colour representation in computer vision and the importance of colour space selection in image segmentation.

Chapter 3

Colour

3.1 A physiological approach to colour vision

The science of colour vision forms part of the study of physics, physiology, psychology, and philosophy [27]. Colour vision is studied with two main aims: to understand how human vision works and to enable machines to see in colour. This is important in segmentation and recognition of objects such as people in a room.

In order to understand the criteria that affect the choice of a colour representation in which to perform the task of segmentation it is necessary to discuss briefly some physical and physiological aspects of colour vision. This, together with a discussion of various computer vision colour spaces available for use later on in the chapter, will motivate the selection of the colour space used in this project.

3.1.1 The trichromatic nature of colour

Colour is both a psychological and physical experience caused by reflected light from an object hitting the retina. The perceived colour of the light is primarily a consequence of its wavelength, which if it falls between 400 and 700 nm [59], is within the spectrum of visible light, as illustrated in figure 3.1. The retina of the eye has three[1] types of colour receptor cells, called cones, which each have a typical response curve to a range of light wavelengths, each peaking at a different wavelength.

Thus, each cone is more responsive to either red (long wavelength receptive cone), green (medium wavelength) or blue (short wavelength) light, although the responses of each cone

[1] A fourth receptor, the rod is activated only at low light levels and does not play an important role in colour vision [59][27].

type do overlap to a large degree. The CIE[2] has defined peak wavelengths for the three primary colours which are shown in table 3.1.

Because of the existence of these three receptor types, a colour can be described as a three-component quantity: a triplet in a colour space. This *trichromatic* theory, was first developed by Helmholtz and Young. Also because of this trichromatic nature of colour vision, many different spectral distributions can produce the same viewed colour. For instance, the cones cannot distinguish between a pure yellow light at a certain wavelength and a mixture of red and green lights at different wavelengths. Light colours that have different spectral compositions but appear identical are called metamers. Ideally then a colour space would have a unique set of three co-ordinates to describe every colour and every possible visible colour would be able to be described in this way.

Pure Colour	Wavelength
Red	700nm
Green	546nm
Blue	435.8nm

Table 3.1: Wavelengths corresponding to primary colours as defined by the CIE

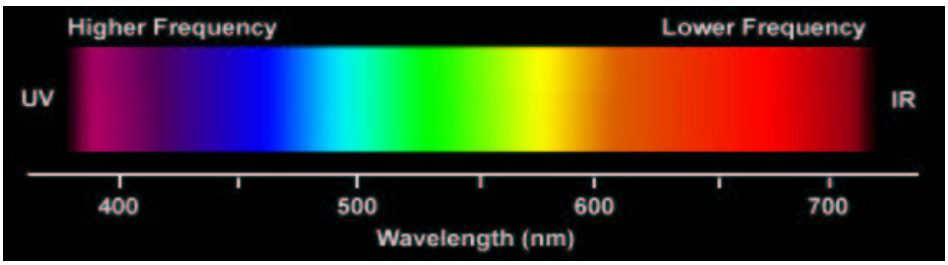


Figure 3.1: The spectrum of visible light [83]

3.1.2 The opponent colour theory

The opponent theory of colour vision was first proposed by Ewald Hering in 1878 and was later revised by Hurwitz and Jameson [28] as an alternative to the classical trichromatic theory. While the trichromatic theory supports the fact that any colour that can be seen is composed of the three primaries, red green and blue, there are at least four other aspects of colour vision which are not accounted for by this model [22].

First the fact that although a colour or mixture of colours can be described as greenish blue or reddish yellow there is no such thing as a reddish green, a bluish yellow or a blackish white. Second, mixing two complementary colours, for example, red and green, produces a result

[2]Commission Internationale de l'Eclairage - International Lighting Commission, based in Vienna.

which is neutral, which suggests that these colours can "cancel out" each other's chromaticity. Third, after-images of a complementary colour are not explained by the trichromatic theory, but according to opponent process theory sustained viewing of colour, for example, red, fatigues the red process and when a neutral colour is viewed immediately afterward both processes should be equally stimulated, but instead a green after-image is seen because of the red's previous over-use. Finally, aspects of colour-blindness are explained, because if all processes are at their balance point a neural grey is observed.

The opponent colour model proposes that opponency and trichromacy correspond to different levels of neural activity. Instead of merely three primaries, the three primary colours received on the cones of the eye are further processed in the lateral geniculate nuclei, which are receptive to three opponent colour pairs: black-white, blue-yellow and red-green, as shown in figure 3.2. These second stages of colour vision processing are stimulated or inhibited by inputs from the cones. For instance, the blue-yellow system is stimulated by input from blue cones and inhibited by input from red and green. If the stimulation is greater than inhibition we see blue, otherwise we see yellow.

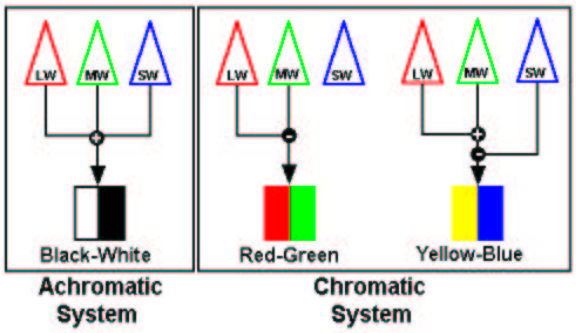


Figure 3.2: The recombination of primary stimuli into opponent colour pairs [22]

3.1.3 Factors affecting colour perception

Perceived colour can be qualitatively described by humans in terms of a hue, saturation and lightness components [22]. Hue is the nature of the colour itself: more accurately the dominant wavelength of a spectral power distribution [58], whether red, green or blue, or a mixture of two of these. The lightness/brightness perception is the quantity of light which seems to be coming from the colour, to which the human visual response is approximately logarithmic. Saturation/chroma is the purity of a colour or the amount of white it appears to have in it. The more the light distribution is concentrated at one wavelength, the more saturated the colour.

Both chromatic (hue and saturation) and achromatic (lightness) contrasts make objects visu-

ally separable from their backgrounds. The achromatic system has far better visual sharpness, as spatially people respond more accurately to lightness changes than to colour differences. The higher the brightness contrast the better the spatial resolution, and for fine detail to be resolved a high brightness contrast is essential as a colour contrast is often not sufficient [22]. People also respond more sensitively to temporal brightness change than to colour change, as is evidenced in the human sensitivity to flicker on a television screen or computer monitor.

There are, however, many factors apart from dominant wavelength which affect the perceived hue of a colour. Hue discrimination for humans becomes worse as colours become less saturated and less bright. Surface colour appearance is also affected by background colour (simultaneous colour contrast), chromatic adaptation (temporal colour contrast), colour constancy (global variations within an image), and size apart from brightness and saturation.

To humans, objects retain their apparent colour when viewed in a different light. Since the light reaching the retina is a product of the object's surface reflectance properties and the illumination of the scene, this suggests that the visual system is somehow able to compensate for these lighting changes. The colour of an object appears to rely more on its inherent surface reflectance properties than the light under which it is viewed. This is wavelength selective adaptation and is the same as what happens when we walk from bright sunlight into a dark house. On film, however, these changes are very evident. This creates the problem of computing colour constancy when processing images, which is something that humans appear to do automatically. Constancy is a problem which affects the scene viewed on a global scale, whereas contrast is a local phenomenon: the colour of an object might appear to change when the background colour changes. When the illumination of a region changes, however, the light that it reflects changes in the same way as the light that its surrounds reflect [27].

Colour cannot be the only cue used by the human visual system to segment objects of interest from background regions. Evidence suggests that colour, intensity, change and motion all contribute somehow to the human visual system's ability to separate foreground objects from their backgrounds.

3.2 Colour space considerations

3.2.1 Colour models

Colour models are ways of representing colour. There are four primary categories : physiologically inspired, colourimetric, opponent and psychological models [43].

Physiological

These are based on the Young-Helmholtz theory of colour vision. They use three primaries (thus taking into account the existence of three types of cone in the human retina), for example, the RGB model used in computer graphics.

Opponent

Opponent models are based on Hering's theory using opponent colour pairs. There can be considered four perceptual primary colours, which do not appear to be made up of any other colours; these are red, green, blue and yellow [22] which are also the opponent colour pairs.

Psychological

Psychologically inspired models are based on the appearance of colour to observers, derived either impressionistically (Munsell and Ostwald) or experimentally (HSV). Munsell and Ostwald are comparative references for artists. The Munsell system is still used industrially. Both are based on subtractive colour so are not generally used in computer vision, which is additive. Both use cylindrical co-ordinates with hue, saturation and brightness components.

Colourimetric

These models are based on measurement of spectral reflectance, for instance the CIE chromaticity diagram which is shown in figure 3.3. Light of any spectral composition can be exactly matched using wavelengths of just three primaries.

3.2.2 The CIE chromaticity diagram and the XYZ colour space

According to the trichromatic theory, light of any wavelength can be matched by an additive mixture of three primaries. Thus, any colour can be specified by the relative amounts of the three primaries needed to create that colour. The CIE has defined colour matching functions for the three primaries red, green and blue, with respect to a standard observer. These colour matching functions are transforms of the spectral sensitivity functions of the human cones, determined experimentally [12].

As can be seen from the CIE's spectral response curves for the cones responsive to red, green and blue, shown in figure 3.4, the entire spectrum of visible colour cannot be represented by positive RGB values. Some colours require a negative red stimulus. The CIE addressed this

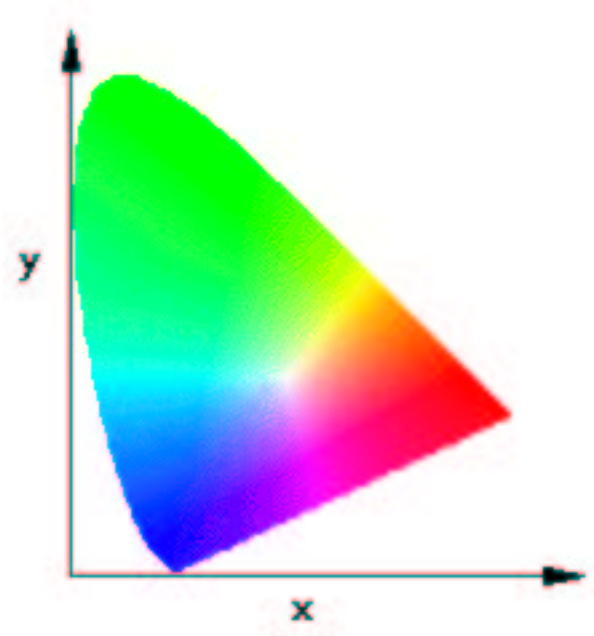


Figure 3.3: CIE chromaticity diagram

problem by linearly transforming the RGB primary stimuli to create a colour space in an XYZ co-ordinate system, such that every visible colour can be represented as a triplet of positive co-ordinates in the space. The primaries thus obtained are called virtual primaries [12]. The tristimulus values XYZ, which are the amounts of the virtual primaries which combined will reproduce a colour spectral distribution, can be calculated given a colour spectral composition, the standard observer colour matching functions in figure 3.5 and a set of three primaries which can be seen in figure 3.4.

The design of the XYZ system is also such that all luminance information is contained in the Y channel. The CIE's XYZ tristimulus response curves are shown in figure 3.5. From the XYZ co-ordinates the chromaticity co-ordinates are calculated, by normalising to disregard intensity as in equations 3.1 and 3.2.

$$x = \frac{X}{X + Y + Z} \tag{3.1}$$

$$y = \frac{Y}{X + Y + Z} \tag{3.2}$$

z can be obtained using $1 - x - y$.

The CIE chromaticity diagram in figure 3.3 represents every colour as a point within a boundary defined by the spectral colours on this x y co-ordinate frame. The white point is at the

centre. Primaries are chosen to maximise the area of the chromaticity diagram covered, as each primary is a vertex of the colour gamut.

Apart from being able to represent all visible colours, the XYZ space is device independent and can be used to compare different colour devices. Another useful aspect of the space is that it is linear: all colours made by combinations of two colours lie on a line joining those two colours, and all colours composed of three colours lie within a triangle whose vertices are the three colours. The equation giving the transformation from standard RGB to XYZ values is shown in equation 3.3.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.3}$$

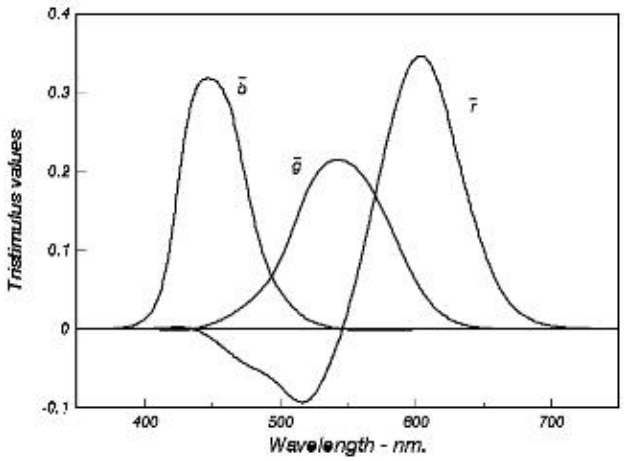


Figure 3.4: Spectral response curves for the three cone types [30]

The CIE have defined two uniform[3] colour spaces based on the chromaticity diagram, which are the L*a*b* (for reflective light) space and the L*u*v* space (for additive light).

3.2.3 Some colour spaces

An important criterion in colour segmentation is the choice of colour space in which to do the segmentation. The RGB colour space, which is the default used to represent digital images, is inadequate for this purpose. This is because it seems logical that to teach a machine to automatically detect an object according to colour, something humans can do with no trouble at all, a colour space should be closely resemblant to the way in which humans perceive colour.

[3]The meaning of a "perceptually uniform" colour space will be made clear later in this chapter.

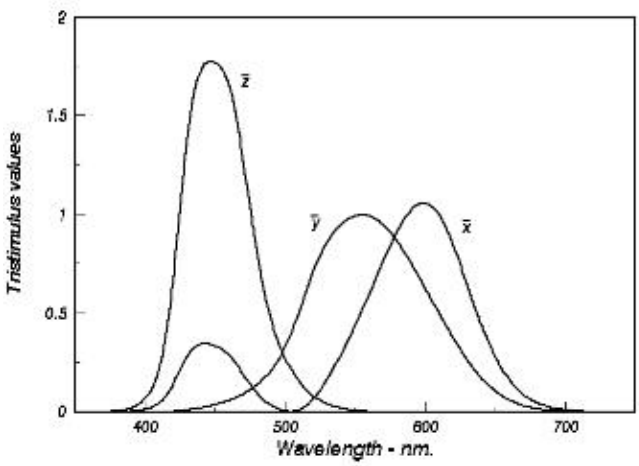


Figure 3.5: CIE spectral response curves [30]

Colour Space	**Components**	**Device Dependent**	**Purpose**
RGB	Red, Green, Blue	Yes	CRT
CMY	Cyan, Magenta, Yellow	Yes	Printing
CMYK	C, M, Y, Black	Yes	Printing
HSB	Hue, Saturation, Brightness	Yes	Perceptual
HSV	Hue, Saturation, Value	Yes	Perceptual
XYZ	Tristimulus values	No	CIE tristimulus
xyY	Normalised tristimulus	No	CIE tristimulus
L*a*b*	Luminance	No	Uniform, Orthogonal
L*u*v*	Luminance	No	Uniform
Munsell	Hue, Value, Chroma	No	Artist's guide
YIQ	Luminance, In-phase, Quadrature	Yes	Television
YCbCr	Luma, chrominances	Yes	Video
sRGB	R G B	No	Proposed for Internet

Table 3.2: Some commonly used colour spaces

In addition, a known problem with segmentation is lighting change. It is therefore reasonable to conclude that a colour space model is required in which the illumination of the scene can be separated as distinctly as possible from the hue information. To this end several colour spaces have been suggested: the YUV [81], HSI [62], and L*a*b* [55] [13] [40] colour spaces have all been used with some success.

It thus seems possible that a system that mimicked this opponent colour model would assist in the automatic classification of different hues. In [7] this technique, which imitates the neuro-biological structure of the human visual system, was seen to aid in segmentation of coloured areas and to reduce the unwanted effects of illumination changes.

An experimental comparison of RGB, YIQ, LAB, HSV and opponent colour models [69] with the aim of determining in which colour space a user was able to reproduce a given colour most quickly and easily, found that the Opponent model and the RGB provided the quickest matches with L*a*b* third. For the closest match, the L*a*b* colour space proved most accurate, followed by the HSV and Opponent colour model. The conclusion reached in this study which may be of importance was that subjects matched lightness better if the colour model had a lightness axis, whereas the advantage of having a hue axis was not obvious.

The colour space which most closely resembles the opponent colour model is the CIE L*a*b* colour space in which the L* component is the luminance value and the a* and b* components are the red-greenness and the blue-yellowness respectively. This space was created for its perceptual uniformity [69], which is also an advantage in segmentation. In a uniform colour space a colour difference perceived by an observer is approximately the same as the Euclidean distance between two points in the colour space.

3.2.4 The L*a*b* colour space

The CIE recommends the use of the L*a*b* colour space for measuring colour differences [12] under illumination conditions which resemble daylight. The transformation from XYZ tristimulus values to the L*a*b* co-ordinates are as shown in equations 3.4, 3.5 and 3.6. Adaptation to different light sources is built into this equation by normalising the tristimulus values by the reference white point, X_n, Y_n, Z_n. Often the D65 white point defined by the CIE as reference daylight is used in this normalisation.

$$L* = \begin{cases} 116(\frac{Y}{Y_n})^{\frac{1}{3}} - 16 & if \quad \frac{Y}{Y_n} > 0.008856 \\ 903.3(\frac{Y}{Y_n}) - 16 & if \quad \frac{Y}{Y_n} \leq 0.008856 \end{cases} \tag{3.4}$$

$$a* = 500[f(\frac{X}{X_n}) - f(\frac{Y}{Y_n})] \tag{3.5}$$

$$b* = 200[f(\frac{Y}{Y_n}) - f(\frac{Z}{Z_n})] \tag{3.6}$$

where

$$f(t) = \begin{cases} t^{\frac{1}{3}} & if \quad t > 0.008856 \\ 7.787 \times t + \frac{16}{116} & if \quad t \leq 0.008856 \end{cases} \tag{3.7}$$

The L*a*b* space, being designed to be perceptually uniform, is intended to be an accurate space in which to measure colour differences. Colour differences in this space correspond approximately to a Euclidean distance measure between two points. A colour difference metric, ΔE is defined in the L*a*b* space as:

$$(\Delta E)^2 = (\Delta L*)^2 + (\Delta a*)^2 + (\Delta b*)^2 \tag{3.8}$$

This means that this colour space is a likely choice for tasks such as colour segmentation and content-based image retrieval, where differences between colours should be measured as accurately as possible [12]. It has been claimed to be less than uniform in the blue and magenta regions [78].

Most of the literature regarding the segmenting of people in motion uses colour information, not necessarily as a criterion on its own, but often combined with motion and situation information. In these, the use of certain colour spaces prevails, particularly those which, unlike the RGB colour space, closely resemble the way humans perceive colour, and importantly, separate the hue information from the lightness information. Some colour spaces used are the RGB [37], HSV [62] [52] and YUV [81], as well as L*a*b* [40] more recently. In [40] a comparison is made between logarithmic L*a*b*, HSI, RGB and normalised RGB. Logarithmic L*a*b* is found to be marginally better for tracking.

3.3 Experimental comparison of some colour spaces

A preliminary investigation into the suitability of various colour spaces for segmentation was made. The same segmentation technique was used on a set of images in four different colour spaces. Since, as we have seen, it is often desirable to omit the luminance information for colour-based segmentation, the experiment was carried out using only the chrominance bands of the colour spaces. The colour spaces used for the purpose of this experiment were: the normalised rg space, h-s space taking only the hue and saturation components of HSV, a*b* - the chromaticity components of L*a*b, and the Cb and Cr components of the YCbCr colour space.

The colour spaces chosen are examples of differing colour models: RGB, the conventional computer graphics space; HSV, a perceptual and non-uniform colour space; YCbCr, belonging

to the YUV family of colour spaces for television and L*a*b*, designed by the CIE to be perceptually uniform.

3.3.1 Colour space conversions

The conversions from RGB to the colour spaces used in the experiment are given below [58].

Normalised rg space

$$r = \frac{R}{R+G+B} \tag{3.9}$$

$$g = \frac{G}{R+G+B} \tag{3.10}$$

HSV space

$$H = \begin{cases} \frac{(G-B)}{max(R,G,B)-min(R,G,B)} & if \quad max(R,G,B) = R \\ 2 + \frac{(B-R)}{max(R,G,B)-min(R,G,B)} & if \quad max(R,G,B) = G \\ 4 + \frac{(R-G)}{max(R,G,B)-min(R,G,B)} & if \quad max(R,G,B) = B \end{cases} \tag{3.11}$$

$$S = \frac{max(R,G,B) - min(R,G,B)}{max(R,G,B)} \tag{3.12}$$

$$V = max(R,G,B) \tag{3.13}$$

YCbCr space

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -93.986 & -18.214 \end{bmatrix} * \begin{bmatrix} R/255 \\ G/255 \\ B/255 \end{bmatrix} \tag{3.14}$$

L*a*b* space

The conversion from RGB to XYZ tristimulus values has already been described in equation 3.3 and the conversion from XYZ to L*a*b* in equations 3.4, 3.5 and 3.6. For ease of reference the conversions to L*a*b* are repeated here.

$$L* = \begin{cases} 116(\frac{Y}{Y_n})^{\frac{1}{3}} - 16 & if \quad \frac{Y}{Y_n} > 0.008856 \\ 903.3(\frac{Y}{Y_n}) - 16 & if \quad \frac{Y}{Y_n} \leq 0.008856 \end{cases} \tag{3.15}$$

$$a* = 500[f(\frac{X}{X_n}) - f(\frac{Y}{Y_n})] \tag{3.16}$$

$$b* = 200[f(\frac{Y}{Y_n}) - f(\frac{Z}{Z_n})] \tag{3.17}$$

where

$$f(t) = \begin{cases} t^{\frac{1}{3}} & if \quad t > 0.008856 \\ 7.787 \times t + \frac{16}{116} & if \quad t \leq 0.008856 \end{cases} \tag{3.18}$$

3.3.2 Skin colour detection experiment

The object of the experiment was to detect skin colour in the set of test images, given an example data set under different colour space transformations, and to determine which colour space produced the best[4] segmentation. The example or training set of skin coloured pixels were taken from nine different images, taken under varying lighting conditions (assuming roughly daylight conditions) and incorporating different individual skin colours. These were transformed into the four two-dimensional colour spaces. The two test images, which are shown in figure 3.6 are also taken under unknown lighting conditions and contain people with different skin colours. The transformation to L*a*b* was done assuming a reference daylight

(a) Test image 1 (b) Test image 2

Figure 3.6: Test images for skin colour detection

white point as it was considered unfair to take advantage of the inherent normalisation factor built into this transformation. However, using the white point of a reference image for the L*a*b* transformation would be expected to increase the accuracy of the segmentation.

In order to classify the skin coloured regions in the test images the Mahalanobis distance[5] of

[4]"Best" means lowest false skin detection rate AND skin omission error as determined from a hand-segmented version of the same image.

[5]The Mahalanobis distance measure is discussed in chapter 8.

every pixel in the colour-transformed test images from the example pixel set is taken. The Mahalanobis distance from the test set x to an example set μ is given by equation 3.19:

$$\Delta^2 = (x - \mu)^T \Sigma^{-1} (x - \mu) \tag{3.19}$$

The resulting image is then thresholded to produce a segmentation of the skin-coloured regions. The pixels with the smallest Mahalanobis distance from the training set are the ones which can be most confidently classified as skin colour. Therefore it is easy to see that as the threshold placed upon this distance from the training set is increased, more pixels will be classified as skin, until the point where an unacceptably large number of pixels which are not skin will mistakenly be classified as such. The optimum segmentation is then produced by the threshold value which classifies the most skin-coloured pixels correctly as skin (fewest false negative classifications) and the fewest non-skin pixels as skin (fewest false positive classifications). The choice of threshold is a crucial factor influencing the segmentation, but there exists a threshold for the distance measure in each colour space which yields an optimum segmentation, minimising both the false positives and the false negatives. To find this optimum the Mahalanobis distances in each colour space are thresholded at increasing values and the segmentation at every threshold is evaluated.

A hand-segmented image is used to evaluate the quality of the segmentation at each threshold. Four error metrics extracted from the comparison with the hand-segmented image are plotted in figures 3.7 and 3.8. The non-skin pixels error is the proportion of non-skin pixels mistakenly classified as skin, the skin error is the proportion of skin-coloured pixels which are misclassified, the percentage correct is the proportion of pixels in the entire image which have been correctly classified, and the percentage of skin correct is the proportion of skin-coloured pixels which have been correctly classified.

3.3.3 Segmentation results

As can be seen from the error plots in figures 3.7 and 3.8, at a low threshold the number of misclassified skin pixels is high, and this number falls as the threshold increases. The misclassified non-skin pixels, however, increase as the threshold increases. It can be seen that there is some point at which these two lines intersect and at which both misclassifications are at a minimum. This also corresponds roughly with the peak in the plot of the overall correctly classified pixels. The threshold at this point is the optimum threshold, as any increase in threshold will lead to an increase in non-skin pixels classified as skin and any decrease will lead to more skin pixels being classified as non-skin, both of which will degrade the quality of the segmentation.

The results for the segmentation of the first test image at the optimum threshold are shown

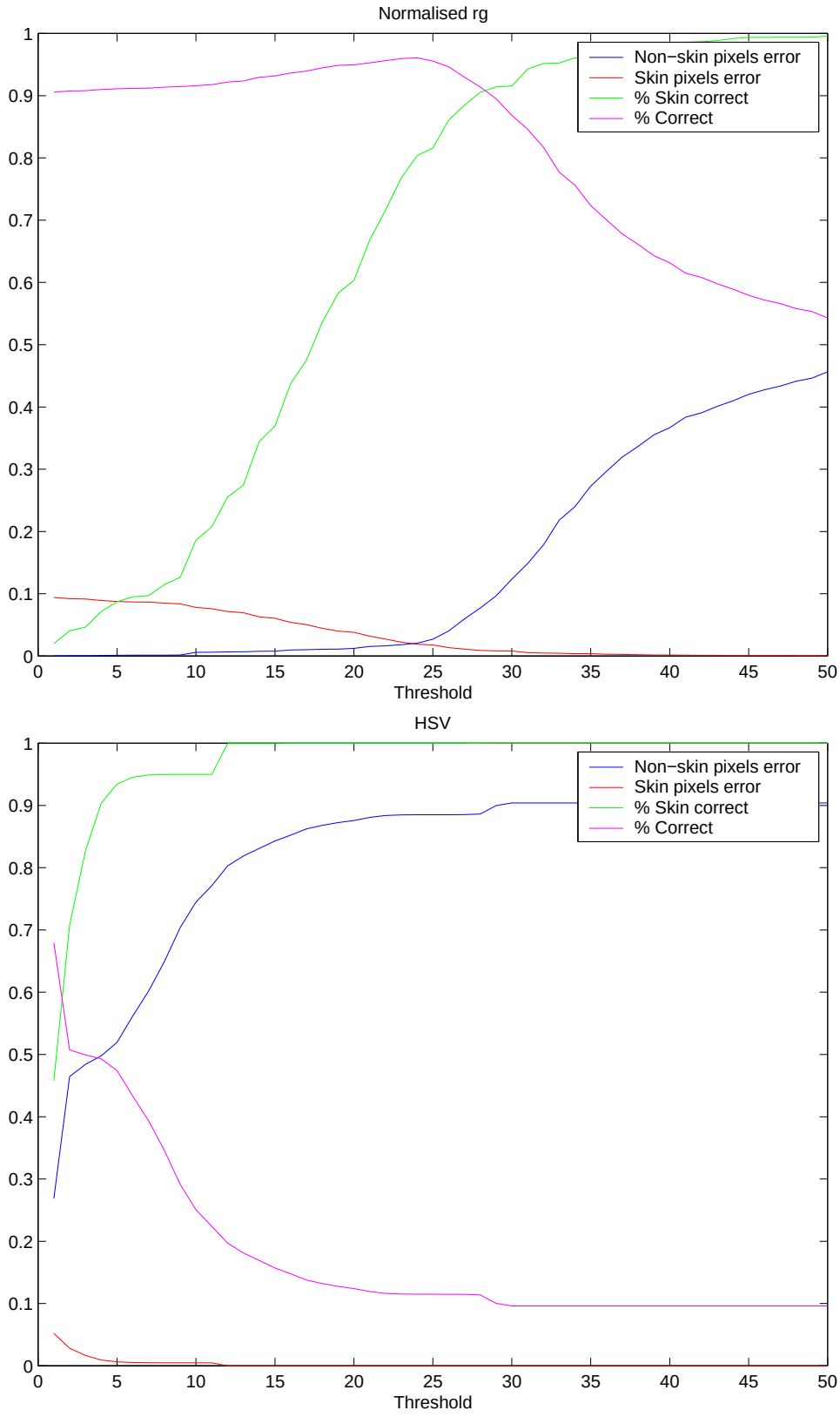


Figure 3.7: Error values for increasing threshold for normalised RGB and HSV colour segmentation

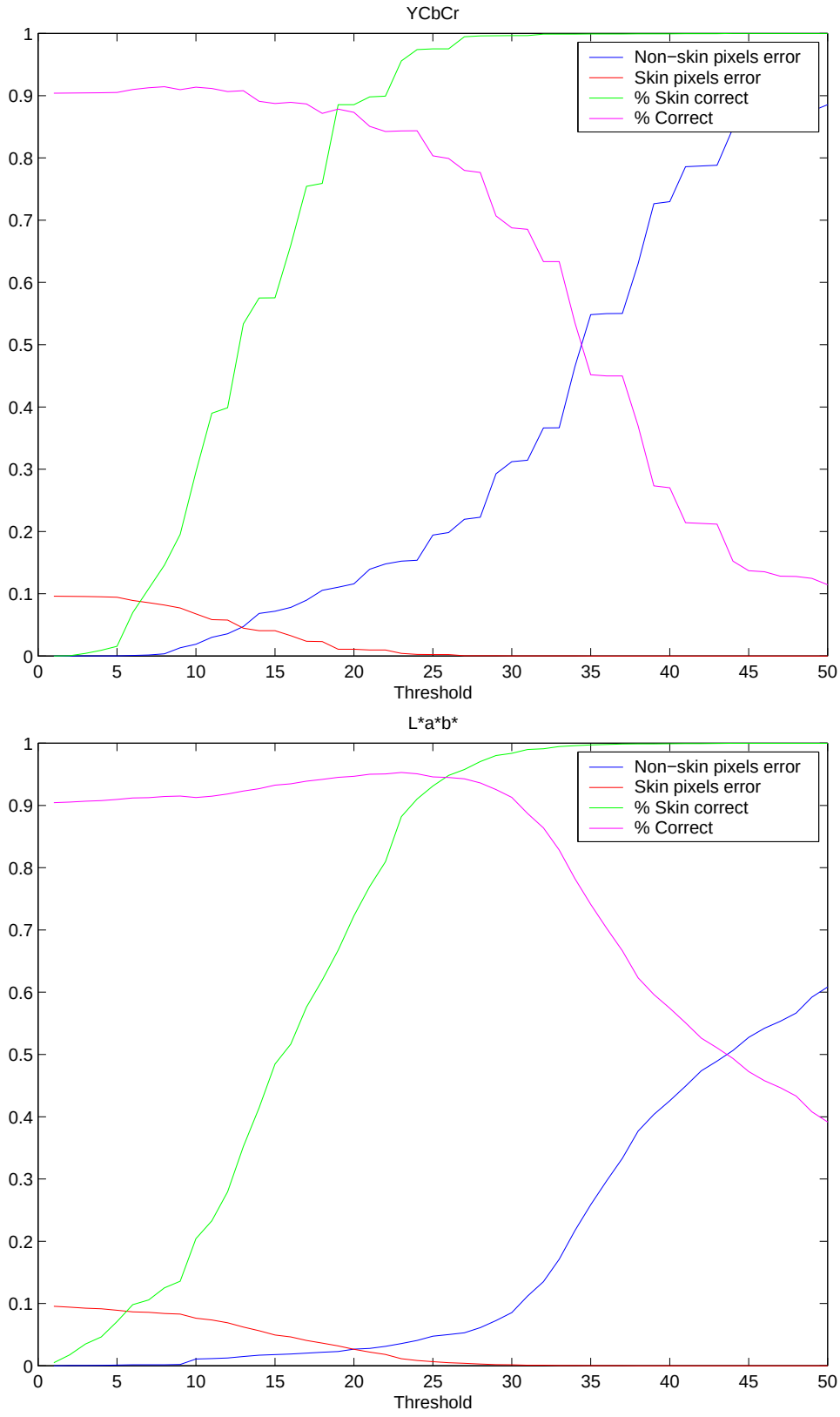
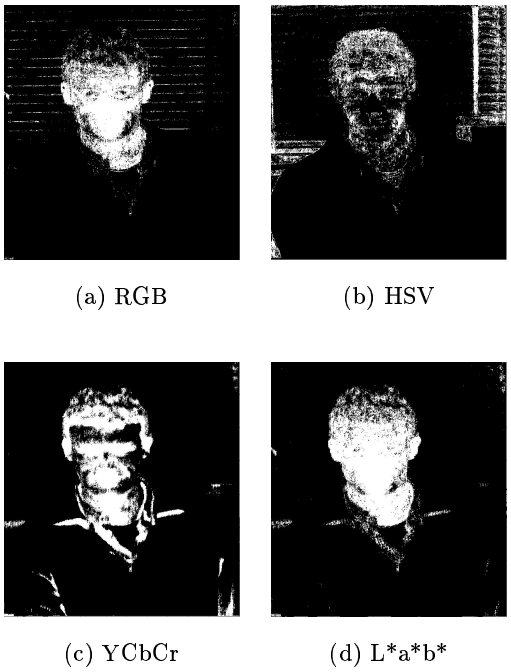


Figure 3.8: Error values for increasing threshold for YCbCr and L*a*b colour segmentation

in figure 3.9. It seems clear that the segmentation with the most correctly classified skin colour (and least non-skin classified as skin) is the a*b* space segmentation, followed by the normalised rg, then Cb-Cr and lastly, HS. For the second test image, which contains two people with very obviously differing skin colours, the segmentation results appear to be equally as good in all colour spaces except HSV.



(a) RGB (b) HSV

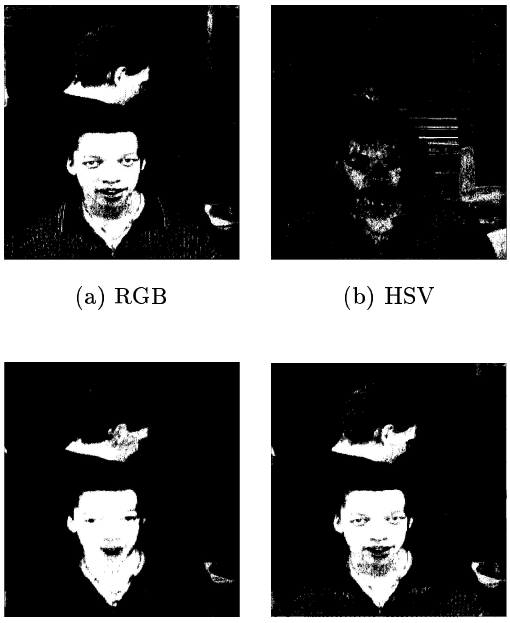
(c) YCbCr (d) L*a*b*

Figure 3.9: Skin colour segmentation results on test image 1

3.4 Colour constancy

There exist several algorithms for estimating illuminant chromaticity: that is to say the colour of the light that is falling on a given object or scene. The same object may look very different in terms of colour appearance if it is illuminated by two differently coloured lights. Techniques which estimate the colour of this illuminant are useful in order to be able to colour-correct an image for object colour properties which appear different from image to image. This phenomenon occurs mainly locally within image regions, for example, as a shadow falls on a previously brightly lit part of the background.

Most of these algorithms for estimating illuminant colour use assumptions either about the range of possible illuminant colours or the range of surface reflectances in the image. A neural network method does not rely on any assumptions about the conditions causing a certain colour effect and was used with seeming success by Funt, Barnard et al. [20].



(a) RGB (b) HSV

(c) YCbCr (d) L*a*b*

Figure 3.10: Skin colour segmentation results on test image 2

Some of the algorithms which do rely on prior assumptions are the grey-world algorithm, the white point algorithm and the 2D convex hull gamut mapping algorithm. Brief descriptions of each of these algorithms follow.

- The **grey-world algorithm** makes the assumption (usually incorrect) that the average of all the colour in an image, or set of images, is grey. Any departure from grey is assumed to be the colour of the light illuminating the scene [20].
- The **white point algorithm** assumes that somewhere in the image are points that maximally reflect each of the colour bands. This maximum is taken to be the chromaticity of the illuminant. This of course does not work as well for real images, as there is the possibility of the response from one or more channels (R,G or B) being saturated, thus making the maximum of a channel the maximum possible digital grey-scale value [19] [20].
- The **2D convex hull gamut mapping algorithm** considers the set of possible illuminants that could map the observed gamut of image colours to a known gamut of expected colours under the standard known illuminant.

Funt et al. [20] [21] use a multi-layer perceptron to estimate illuminant chromaticity. They use a neural network trained on images under known illuminants and then one trained on images

of which the illuminant is estimated using the grey-world algorithm. The neural network eventually outperformed the grey-world algorithm on which it was trained.

An algorithm of the same sort as these is necessary in order to be able to segment reliably under different conditions. The problem in the context of tracking a person over time is not extreme in the sense that the background or foreground as a whole can only change slowly with time, and resampling and adapting a model should be able to cope with this slow change. It is especially locally within each frame that the problem occurs, as a shadow cast by the person may alter the background colour distribution for a small region in such a way as to make it be misclassified.

3.5 Colour space selection

There are several reasons why the L*a*b* colour space seems superior to all the other colour spaces considered thus far, and only one reason why it should possibly not be used. The only disadvantage of using this space is the relative computational expense of the nonlinear transformation. Given that there is some computational overhead involved in any colour transformation and that the purpose of this thesis is not to construct a real-time application, this reason bears little weight in the selection process.

The benefits of the L*a*b* space are outlined as follows:

1. The luminance information is separable from the chrominance, which is desirable in an application in which lighting changes could occur.
2. The transformation itself takes into account the illuminant of the imaged scene, by using its white point in the transformation. This is effectively incorporating colour constancy in the form of the white point algorithm described in the previous section.
3. The space is perceptually uniform and orthogonal; thus perceived colour differences are similar to measured colour differences which is ideal for segmentation.

In addition, the results of the experiment performed using four colour spaces revealed the L*a*b* space to be as good if not better than any other space. For these reasons it was decided to attempt the segmentation and tracking task in the L*a*b colour space.

Chapter 4

Colour segmentation

Colour segmentation is the process of selecting regions of interest in an image based on features in colour space. The aim is to extract meaningful objects which correspond to human perception. There exist supervised and unsupervised methods of assigning data to classes in the colour space. Supervised methods involve fitting a function to training data which is known to belong to one of a certain number of classes and then applying this function to unclassified data to find the class it is most likely to belong to. However it is not always possible to tell a system what classes the data forms,[1] and in the absence of training data a classifier that can learn distinguishable categories can be very useful. This is done in an unsupervised manner, by finding clusters in the data, and assigning each data point to the most appropriate cluster.

4.1 Unsupervised learning and clustering

Unsupervised learning approaches attempt to assign unlabelled data to classes based on clusters found within the data. Clustering is a process in which points belonging to each class must be connected with respect to each other and distinct from points belonging to all other clusters [55]. Most clustering methods require some degree of supervision, as the number of clusters needs to be specified in advance. The K-means algorithm, a common clustering method used for colour segmentation, is one of these.

One method for estimating the optimum number of clusters to represent an unlabelled set of data is given in [55]. Every data point is convolved with a kernel. The width of the kernel (and thus the number of clusters chosen) affects the estimate of the density. In [55] a very small value for the kernel width is chosen and the clusters are located by finding the local

[1] This can happen if it is possible to obtain data but not to label them, or if the features specific to each class change over time.

maxima of the convolved dataset. This gives an overestimation of the number of clusters. Then a hierarchical system for merging is constructed which merges clumps in an order which is determined by comparing the values at neighbouring maxima with the density value at the minimum between the two clusters.

In the particular context dealt with by this thesis, unsupervised learning based upon colour features is not applicable. It is known that the number of classes ultimately required is two: one for the background and one for the foreground, but the number of distinct colour clusters formed within these two desired classes is almost guaranteed to be more than one each. It seems plausible that some kind of supervised learning method will be required although no training data is available as yet.

If it is assumed that an image of the empty background scene is available before any people have entered, some training data for the distribution of background colours, and indeed spatial location of these colours is obtained. All that remains is to find an independent method of locating training data for the people as they enter the room. A method for doing this is discussed in chapter 5, but for now we will just assume that as each person enters the scene there are some pixels which are known to belong to the person. Given these two sets of training data we are able to build statistical estimates of the colour distributions of the people in the room and the background. In order to do this we need to determine a suitable method of estimating densities of colour distributions using a training set of which it is known whether it forms part of the background or foreground class.

4.2 Density estimation using a labelled training set

Three possible forms of density estimation are described in the sections that follow. The first is parametric density estimation, in which the form of the distribution is assumed known and parameters are fitted to some data set. The second form is non-parametric estimation in which the data is allowed to determine the form of the density function, which can lead to a large number of parameters having to be set. Lastly semi-parametric density estimation constitutes a compromise between the parametric and non-parametric forms.

4.2.1 Parametric

For the subset of learning approaches using parametric estimation of density a Gaussian distribution is assumed. The form assumed for the distribution of the data might, however, be very different from its true distribution. On the other hand, the evaluation of new data points is very quick as it merely involves the evaluation of a function at that point. The two principal methods of estimating the parameters are maximum likelihood and Bayesian inference.

Maximum likelihood

The problem is to fit appropriate parameters to the model so that it best fits the available data [2]. The "maximum likelihood model" is the one most likely to have generated the observed data. This likelihood can be calculated and in many cases a unique maximum exists.

Maximum likelihood maximises a likelihood function obtained from the training data, to obtain the optimum parameters. Using these parameters the maximum probability of generating the training set is obtained.

Bayesian inference

In Bayesian inferencing the parameters are described by a probability distribution, initially set to a prior distribution and then a posterior distribution is found once the data have been observed. This technique uses the training set to update the density function of the parameters, conditioned on the training set.

4.2.2 Non-parametric

Non-parametric density estimation techniques are slow to evaluate a new data point, since the number of variables to be evaluated grows as the data set becomes larger. The form of the probability distribution is not specified, but depends on the data.

One form of non-parametric density estimation is to estimate the density function directly from the data. A histogram is possibly the simplest density estimation technique. The input space is divided into a number of bins and the density is estimated for each bin by finding the proportion of the data that falls into that bin. The number of bins, or equivalently the width of the bins, greatly affects the resulting distribution: a small number of bins over-smooths the density function, concealing the shape of the distribution, and a large number produces a distribution which is sensitive to individual bins and appears spiky. The distribution is also not continuous and does not generalise well to high dimensions, where an increasing amount of data is needed to obtain a good density measure. The histogram technique has, however, been used successfully for tracking [5].

The second possibility is to use the data directly and assign each new data point to the class to which its nearest neighbour belongs. This is a distance-based classification method and a special case of the K-nearest neighbour rule. In K-nearest neighbours, instead of being assigned to the class of the nearest neighbour, each data point is assigned to the class c for which $\frac{k_c}{k}$ is a maximum, ie. the class to which the highest proportion of the k nearest neighbours belong.

4.2.3 Semi-parametric

Mixture modelling is a form of semi-parametric density estimation. Training methods for these models are based on maximum likelihood. The number of basis functions is much fewer than the data points, so there are fewer parameters to be determined and yet no strict form is assumed for the data distribution. Instead the distribution is represented as a linear combination of component densities .

4.3 Bayesian decisions

In order to explain the notation used in the following section a brief description of Bayesian decision theory is given here.

Bayesian methods of inference use information about the prior knowledge of the likelihood of a particular probability density function. This of course assumes that one can know something about the prior distribution, this being the source of some controversy. The *prior* distribution reflects an initial estimate of the range of values of x, before any data have been observed. This is typically very broad as usually one has little idea of the prior distribution of the variable. Once the data have been observed Bayes' theorem can be used to calculate the *posterior* distribution.

If there are N classes, C_i for $i = 1, ..., N$, and data x are observed, the conditional probability of C_i occurring, given that data x have occurred; or stated otherwise, the posterior probability that x belongs to C_i is:

$$p(C_i|x) = \frac{p(x|C_i)P(C_i)}{p(x)} \tag{4.1}$$

where

$P(C_i)$ is the *a priori* probability of any data element belonging to class C_i

$p(x|C_i)$ is the conditional probability of observing data x, given that the class is C_i

and $p(x) = \sum_{i=1}^{N} p(x|C_i)P(C_i)$ is a normalisation factor, being the probability of observing x over all classes.

4.4 Gaussian mixture models

Mixture models are a semi-parametric form of density estimation. A mixture model can be constructed from a limited number of Gaussians, each with different parameters which best describe the data. In this case the number of Gaussians fitted to the data may be far smaller than the number of data points (which is not the case with non-parametric density estimation), so the number of components M is treated as a parameter of the model. In addition, no form is assumed for the distribution (as in parametric density estimation methods) but rather the model is allowed to find a form, which is a sum of different Gaussian components, which best fits the data. The density $p(x)$ can thus be represented as the sum of component densities $p(x|j)$, where $P(j)$ is the prior probability of the data having been drawn from the jth component.

$$p(x) = \sum_{j=1}^{M} p(x|j)P(j) \tag{4.2}$$

Each component $p(x|j)$ in equation 4.2 is a Gaussian of the following form.

$$p(x|j) = \frac{1}{(2\pi\sigma_j^2)^{\frac{d}{2}}} exp\left\{-\frac{|x-\mu_j|^2}{2\sigma_j^2}\right\} \tag{4.3}$$

The priors $P(j)$ are normalised so that

$$\sum_{j=1}^{M} P(j) = 1. \tag{4.4}$$

This form of density estimation can be accurate if the correct parameters and correct number of components are chosen for the data [6]. The training of a mixture model is done with incomplete data as it is not known beforehand to which component of the distribution each data point belongs.

The posterior probabilities conditioned on this model can be represented in Bayesian terms as:

$$P(j|x) = \frac{p(x|j)P(j)}{p(x)} \tag{4.5}$$

$P(j|x)$ is the probability that component j generated the data point x.

Some methods for determining the parameters for a Gaussian mixture model from a set of data are maximum likelihood, expectation maximisation and stochastic estimation of parameters.

For descriptions of these methods see [6]. Expectation maximisation (EM) is used for the purpose of this thesis and is described in Appendix A.

4.5 Gaussian mixture modelling for foreground segmentation

In this segmentation method a number of Gaussian functions represents the colour distribution of an object in the two dimensional colour space, as a semi-parametric alternative to a histogram. The probability of every colour pixel in the test image belonging to this distribution is calculated.

It is assumed that the first image of a sequence is an empty frame, which can be used as a background image. The empty scene is then modelled as a mixture of Gaussians [60][61][62] representing the two-dimensional distribution of the colours in the *a*b* colour space.

The probability of a pixel x belonging to an object O (which in this case is the person in the scene), can be represented as the sum of M Gaussian components :

$$p(x|O) = \sum_{j=1}^{M} p(x|j)P(j) \tag{4.6}$$

where $P(j)$ is the prior probability of the pixel colour having been generated by the jth Gaussian component of the mixture. Each component j is a Gaussian with mean μ_j and covariance matrix, Σ_j as shown in equation 4.7.

$$p(x|j) = \frac{1}{2\pi|\Sigma_j|^{\frac{1}{2}}} exp^{-\frac{1}{2}(x-\mu_j)^T\Sigma_j^{-1}(x-\mu_j)} \tag{4.7}$$

A model is initialised arbitrarily with a small number M of Gaussian components, and the EM algorithm described in Appendix A is used to find the most appropriate centres (mean μ), widths (covariances Σ) and sizes (priors $P(j)$) for each Gaussian, in order to fit the Gaussian mixture to the training data. Initially two Gaussians are used to provide an estimated model for the two-dimensional distribution of the colours.

The training data for constructing the mixture model of the person is obtained using a change detection algorithm (detailed in chapter 5) using the empty background which is assumed in the first frame. Any significant departure from the known background is considered part of an object whose colours will be modelled. What remains to be done is to find the optimum number of components for the Gaussian mixture models representing the foreground and the background. In other words, the number of distinct colour clusters which make up each of the foreground and background classes must somehow be found so that each of these classes can

be modelled optimally.

4.5.1 Minimum description length models

The choice of a good model to describe the data involves a trade-off between closely fitting the model to the observed data and limiting the number of parameters used to describe the model. As the number of components of a Gaussian mixture increases, the model fits the data better and better until the case where there is one Gaussian centred on each data point. Clearly this is neither an efficient nor general way of modelling the data.

Factors which need to be taken into consideration when selecting the number of components for a model are: one wants to choose the model which assigns the higher probability to the data set, and to minimise the model complexity. A model consisting of raw data is thus perfectly accurate in that a high probability is assigned to each data point, but a large number of parameters (the entire data set) needs to be stored to describe this data. A single Gaussian distribution, however, might be chosen that describes the data adequately and yet can be described simply with a mean and covariance matrix.

The log-likelihood of the model on some data, which can be regarded as an error function (see Appendix A) will increase with respect to the data whenever the number of Gaussians is increased. It is desirable to maximise the log-likelihood and minimise the number of parameters in the model simultaneously.

A model is best in a Rissanen sense if it compresses the data maximally [2]. The idea is to transmit a set of data D using the shortest possible length message. This can be done by specifying a model M, which captures trends in the data and then an error indicating how much the model differs from the data. The length of the message required to send this information is the length of the model $L(M)$ added to the length of the error $L(D|M)$.

$$description length = L(M) + L(D|M) \tag{4.8}$$

Thus, a simple model will generalise badly and have a large error and a longer error description length and a complex model will have lots of information to transmit, so an optimum for both these criteria must be found. The Minimum Description Length (MDL) principle proposed by Rissanen [64] in 1978 to minimise this description length is given by equation 4.9.

$$MDL(k) = -2\sum_{i=1}^{N} \log p(x_i|\widehat{\theta}_k) + k \log N \tag{4.9}$$

where k is the number of components in the model, the data sample is given by $x_1.....x_N$

and $\widehat{\theta}_k$ is the maximum-likelihood estimate of the parameter θ_k based on the data [51]. This technique is used in [40] to construct Gaussian mixture models for tracking.

Another way to determine the rate of exchange between parameter number and log-likelihood if more data is going to become available is by testing the model on the data when it becomes available later to see how well it performs. In instances where more data is available, such as the modelling of a background image, it is more convenient to test the log-likelihood of the model on the new data.

4.5.2 Model order selection based on additional data

The algorithm used in this thesis to select the number M of Gaussians that best describes the training data, is the one used by Raja, McKenna and Gong in [60], [61] and [62]. The number of Gaussians is selected automatically by testing the responsibility of each component for a validation set. The validation set comprises half the pixels which have been selected through the change detection mechanism. The training data obtained through change detection is split into two sets by randomly sampling half the pixels. One of these sets is used for training and the other for validation.

A mixture model is initialised with two components. K-means clustering is used to find the initial centres of the Gaussians from the training data. Expectation maximisation is then applied to find appropriate parameters so that the model best fits the training data. The log-likelihood, which can be regarded as an error function for the mixture model parameters (see Appendix A) is then calculated on the validation set, which has not thus far taken part in the training process. An additional Gaussian is added to the model by selecting the component j with the least responsibility r_j (defined in [61]) for having generated the data in the validation set. This responsibility is defined in equation 4.10.

$$r_j = \sum_x p(j|x) = \sum_x \frac{p(x|j)P(j)}{\sum_{i=1}^{M} p(x|i)P(i)} \tag{4.10}$$

The selected component is then split into two Gaussians with the same covariance matrix and new means which are $\mu_{new} = \mu \pm \frac{\lambda}{2}u$ where u is the eigenvector corresponding to the largest eigenvalue λ of the covariance matrix. The priors for the two new components are both half the prior probability of the component to be split. The EM algorithm is then run again on this new model and the process iterates. The log-likelihood (equation A.1 in Appendix A) for the new model on the validation set is then calculated at each iteration and compared to that of the previous mixture model. The algorithm terminates when a peak is located in the plot of the log-likelihood for a number, K of components. Plots of the log-likelihood for increasing numbers of mixture components are shown in figure 4.1.

The same procedure is run to create a foreground model and a background model.

4.5.3 Obtaining the foreground probability image

Once a density estimate has been obtained for both the foreground $P(x|O)$ and background $P(x|B)$, the posterior probability of a pixel in the image belonging to the foreground can be obtained by:

$$P(O|x) = \frac{P(O)p(x|O)}{P(B)p(x|O) + P(B)p(x|B)} \tag{4.11}$$

This minimises the possibility of misclassification in Bayesian sense, and also gives a confidence value for the classification: if the probability of both background and foreground is low or high then confidence is low. The prior probabilities $P(O)$ and $P(B)$ of a pixel belonging to the foreground or background respectively, are set to the fraction of pixels in the bounding box expected to be classified as foreground and background. Thus $P(B) = 1 - P(O)$. This fraction is obtained through the use of another form of segmentation which does not rely on the colour mixture model, which will be detailed in chapter 5.

4.6 An adaptive colour model

4.6.1 The need for an adaptive model

The exclusion of the luminance band of a colour space [40], such as the HSV or L*a*b* as opposed to using all three bands of RGB, aids to some degree in the insensitivity of a model to lightness changes. However, since the colour of an object in an image is the product of the object's spectral reflectance properties and the chrominance of the light illuminating it [20], a change in the chrominance or intensity of the lighting in a scene can greatly affect the actual hues represented in an image.

This of course is a huge problem for model-based segmentation, such as content-based image retrieval based on colour histogram representation of the object and for tracking in a sequence where colours may change due to lighting changes in the scene or even shadows cast by the object being tracked.

Many attempts have been made to alleviate this problem. One possible solution is to attempt to model the lighting affecting an image and to normalise the image in such a way as to compensate for the lighting, so that the colours are the same as if viewed under some known canonical lighting condition. This is done to an extent by conversion to the L*a*b* colour space using a reference white point which is visible in both the background and foreground images. A partial solution to the problem of tracking is to adapt the colour model of the

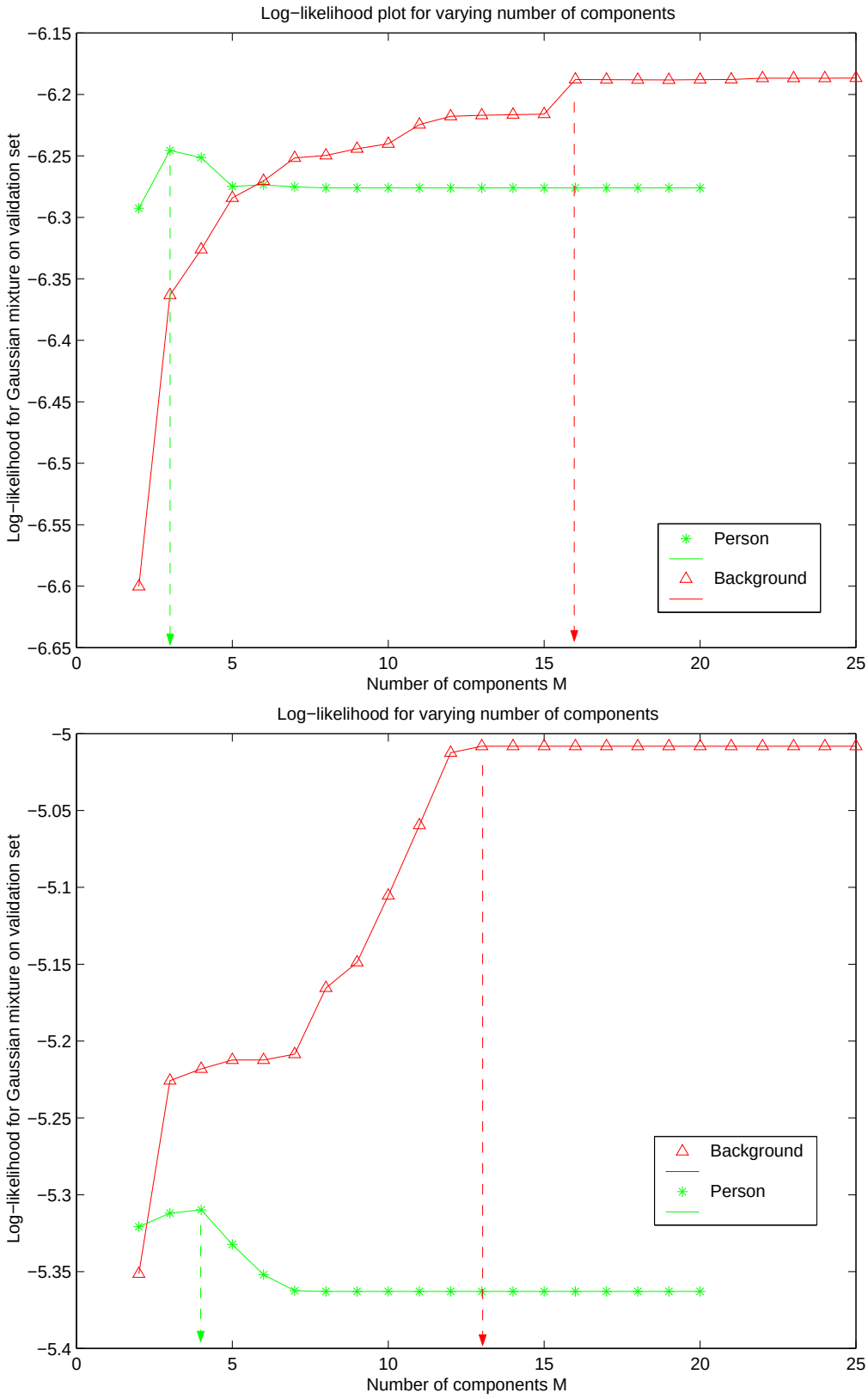


Figure 4.1: Illustration of the choice of number of mixture components for two different datasets consisting of background and person pixels

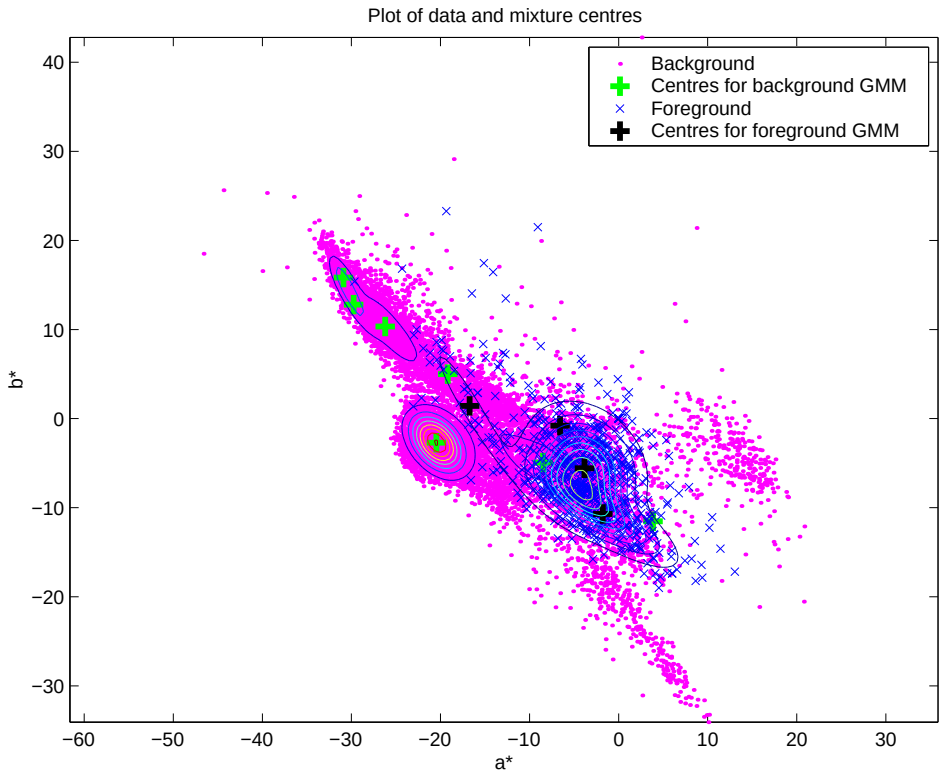
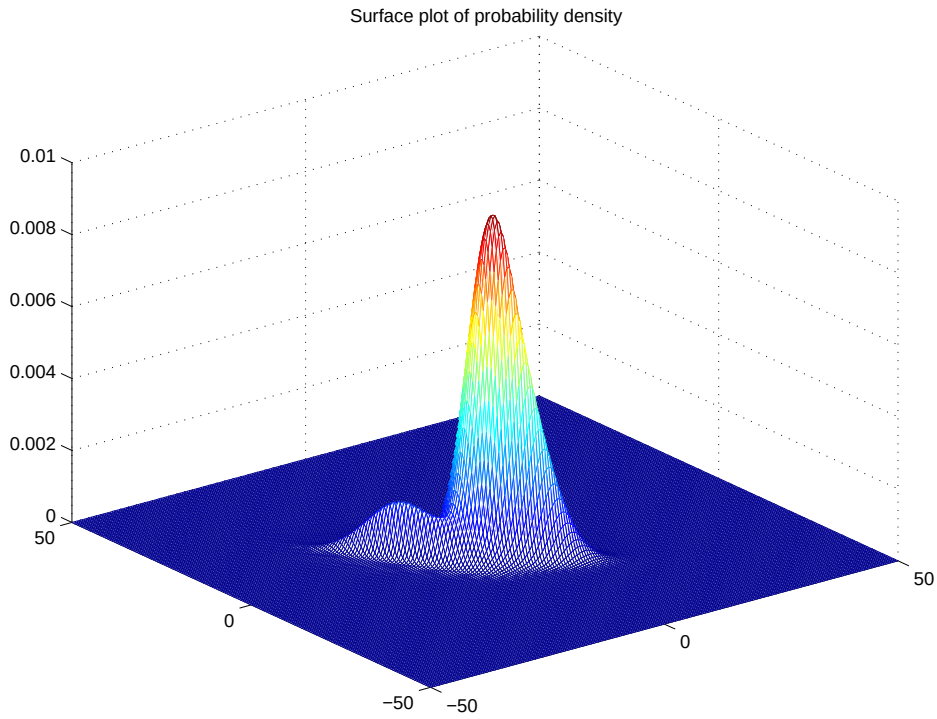
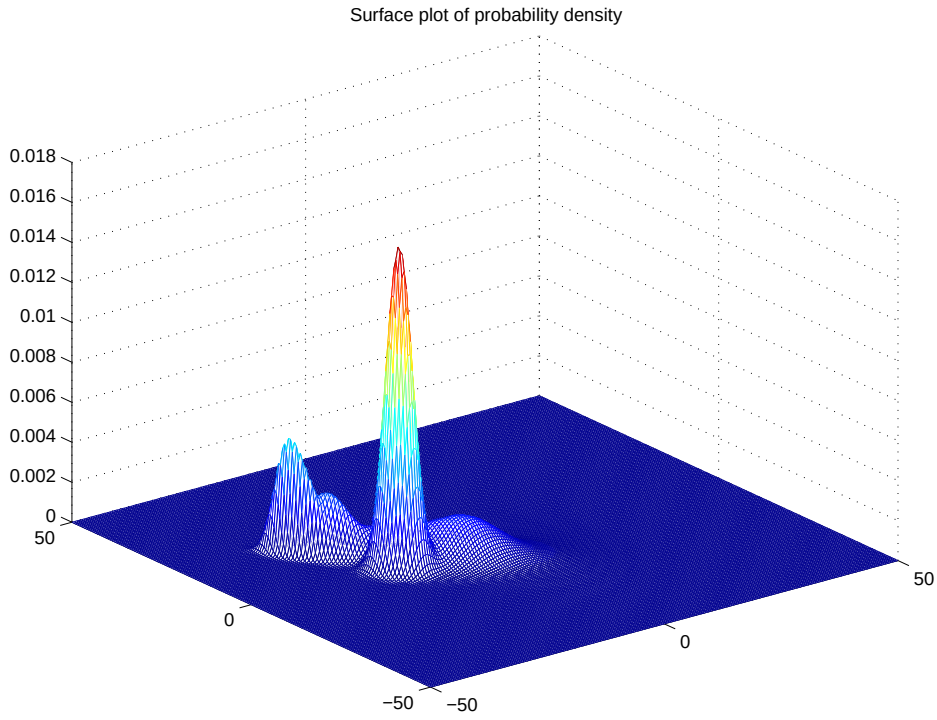


Figure 4.2: Background and foreground data points with mixture model centres superimposed



(a) Foreground pdf



(b) Background pdf

Figure 4.3: Plots of foreground (a) and background (b) membership likelihood

person slowly over time, so that any gradual change in the colours of a person due to a slow lighting change or an effective lighting change caused by varying the distance from the camera or light source to the tracked object can be accommodated into the model.

4.6.2 Adapting the foreground model

Resampling the image

The segmented image from each frame is resampled in order to provide the parameters with which to update the model for the segmentation of the following frame. An approximate centroid for the segmented person is found: taking the median of the segmented pixel positions in the x and y directions.

An "internal bounding box" is then constructed by growing a region outwards from the centroid until a pixel is reached which belongs to the background. This internal bounding box is forced to have the same aspect ratio as the segmented image, so that as many representative colours from the image can be obtained in the sample. Evidently this is not always possible, particularly in the case of the feet which are difficult to include in the sample. It is partly for this reason that the model must be forced to adapt slowly, since with a quickly adapting model different coloured regions of the person can be eliminated very rapidly because of inadequate sampling. A segmented image with its internal bounding box for sampling superimposed is shown in figure 4.4.

Updating the model

The new sampled data from the segmented frame are used to update the existing Gaussian mixture model. The parameters describing each Gaussian component are estimated from this data. The responsibility $r^{(t)}$ at time frame t is defined as in [60] as the sum over all sampled pixels $S^{(t)}$ of the posterior probabilities of that data having been drawn from each Gaussian distribution.

$$r^{(t)} = \sum_{x \in S^{(t)}} p(j|x) \tag{4.12}$$

The mean for each of the Gaussian components j, then becomes

$$\mu^{(t)} = \frac{\sum_{x \in S^{(t)}} p(j|x)x}{r^{(t)}} \tag{4.13}$$

Figure 4.4: Sample box for adjusting the mixture model

the priors for each component become:

$$P(t) = \frac{r^{(t)}}{N^{(t)}} \tag{4.14}$$

and the covariances:

$$\Sigma^{(t)} = \frac{\sum_{x \in S^{(t)}} p(j|x)(x - \mu_{t-1})^T (x - \mu_{t-1})}{r^{(t)}} \tag{4.15}$$

These new parameters are then used to update the current model. The new parameters for the model are a linear combination of the parameters from the previous frame (which in turn are based on those of the frame before that), the parameters of the sample drawn from that frame, and the old model (which could be the original model or the last known good model, a predefined number of stages, L back). The equations for updating the mixture are:

$$\mu_t = \mu_{t-1} + \frac{r^{(t)}}{D_t}\left[\mu^{(t)} - \mu_{t-1}\right] - \frac{r^{(t-L-1)}}{D_t}\left[\mu^{(t-L-1)} - \mu_{t-1}\right] \tag{4.16}$$

$$\Sigma_t = \Sigma_{t-1} + \frac{r^{(t)}}{D_t}\left[\Sigma^{(t)} - \Sigma_{t-1}\right] - \frac{r^{(t-L-1)}}{D_t}\left[\Sigma^{(t-L-1)} - \Sigma_{t-1}\right] \tag{4.17}$$

$$P_t = P_{t-1} + \frac{N^{(t)}}{\sum_{\tau=t-L}^{t} N^{(\tau)}} \left[P^{(t)} - P_{t-1}\right] - \frac{N^{(t-L-1)}}{\sum_{\tau=t-L}^{t} N^{(\tau)}} \left[P^{(t-L-1)} - P_{t-1}\right] \tag{4.18}$$

where $N^{(t)}$ is the number of pixels in the sample. The scaling factors affect how much the model changes in favour of the new or the old model. D_t is the sum of $r^{(t)}$ over the last L frames which is approximated as:

$$D_t \approx \left[1 - \frac{1}{L+1}\right] D_{t-1} + r^{(t)} \tag{4.19}$$

and

$$r^{(t-L-1)} \approx \frac{D_{t-1}}{L+1} \tag{4.20}$$

Evidently L is an adaptivity controlling measure: the greater L, the slower the model will adapt and the faster it will "forget" its original state. For efficiency it is easier to use a smaller value for L and if the sample drawn from the previous segmented frame is accurate, the faster adaptation can be an advantage in the case of sudden lighting changes. The value of 3 was settled upon for L, although this is easily altered on the fly if, for instance, it is known that there are similarly coloured objects in the room which could be false targets for adaptation.

4.6.3 Bootstrapping the model

The problem with an adaptive tracking model is that it may adapt in an undesirable way. For example, if a sample is drawn from an incorrectly segmented image, the segmentation is likely to become worse in subsequent frames and not better. A method is needed of bootstrapping the model to some reliable frame of reference if there is a doubt as to its current efficiency. On the premise that a previous model should perform better than a model which has adapted to the wrong data, the model L frames ago is used as a frame of reference. Even if the object has undergone change in colour because of sudden changes in lighting conditions, the previous model is likely to be at least a better estimate of the colour distribution of the correct object.

It can be established when the mixture has adapted to the wrong data. If the model changes too much and the segmentation is likely to be affected, the model is erased and reverts to a previous model, which is presumed to have performed better because no failure was detected at that point. In order to determine the model's current accuracy, the negative log-likelihood λ of the sampled data can be observed over time. A sudden increase in the negative log-likelihood (or decrease in the log-likelihood) may indicate that adaptation to another object has occurred. The log-likelihood calculation based on the data sample $S^{(t)}$ is shown in equation 4.21.

$$\lambda = \frac{1}{N^{(t)}} \sum_{x \in S^{(t)}} \log p(x|O) \tag{4.21}$$

The threshold T for the allowed limit of variance of all the above criteria which could indicate an inaccurate sampling, is decided by observing the values of the log-likelihood over the last L frames and calculating the median m and standard deviation σ of these. The threshold T, for the log-likelihood is then defined to be $T = m - 2\sigma$, and for any negative log-likelihood greater than T, the model reverts to the one used L frames ago.

4.6.4 Adapting the background model

The background is equally susceptible to lighting changes and is thus an equally likely candidate for an adaptive colour model. The difficulty with this is the large number of pixels which constitute the background. To create a new model using all of the known background pixels would be very time consuming. To draw a small subsample might be hopelessly inadequate if only a small range of the colours in the background were sampled.

A possible solution to this, if the camera is assumed to be static, is to draw a small sample from across the image, and to change the area from which the sample is drawn in every frame. In this way over a period of time every pixel from the background should have participated in updating the background model. This of course assumes a slowly varying background, with a limited amount of clutter.

Another possibility is to take an average colour value over every 10 by 10 pixel block in order to update the model. The method selected by virtue of being computationally feasible and yet not seeming to bias the adaptation unduly in any direction, is to sample 200 pixels at random over the entire background and to do a fast update in a similar manner to the foreground update.

In chapter 5 the measurement of motion is described, and motion information is used for segmentation to supplement the colour information already obtained. The combination of these two pieces of information will eventually lead to a segmentation which is better than that obtained using only one of these forms of image information.

Chapter 5

Motion

Some form of motion estimation is needed in order to:

1. Initially provide a preliminary segmentation of foreground objects so that their colours can be "learnt" by the adaptive Gaussian mixture model.

2. Limit the search space of the scene when testing for correspondence with the colour model. In other words to find the approximate location of a rectangular *bounding box*, which will define the upper, lower, leftmost and rightmost limits of the area of occupation of each person.

3. Combine a motion segmentation estimate with a colour segmentation to produce a more effective segmentation

4. Provide an estimate of the actual motion of the segmented person in order to be able to track him or her.

In section 5.1 the measurement of motion in image sequences using gradient-based methods is discussed. In section 5.2 image differencing techniques are described which will be used to refine the colour segmentation. In section 5.3 a method of updating the background reference image using a Kalman filtering approach is described, and a threshold is automatically generated so that a segmentation can be obtained which is useful both for initialising the colour model and for locating an approximate bounding box. Lastly in section 5.4 the motion estimation for tracking is discussed.

5.1 Measuring visual motion

Motion is important because one uses motion information when colour and edge information is either unavailable or unreliable. Animals make use of this principle in camouflage: predators are less likely to see their prey when it is not moving if it is of a similar colour, texture or shape to its background.

5.1.1 Motion estimation versus optical flow

A motion field assigns a velocity vector to each point in an image. Optical flow is the apparent motion within an image. The two do not always correspond. Two specific examples of this difference are those of a sphere rotating under uniform lighting: there is motion present, but the flow field is zero [25]; and a rotating barber's pole, illustrated in figure 5.1. The optical flow can be defined as "the instantaneous velocity of a brightness pattern at a point".

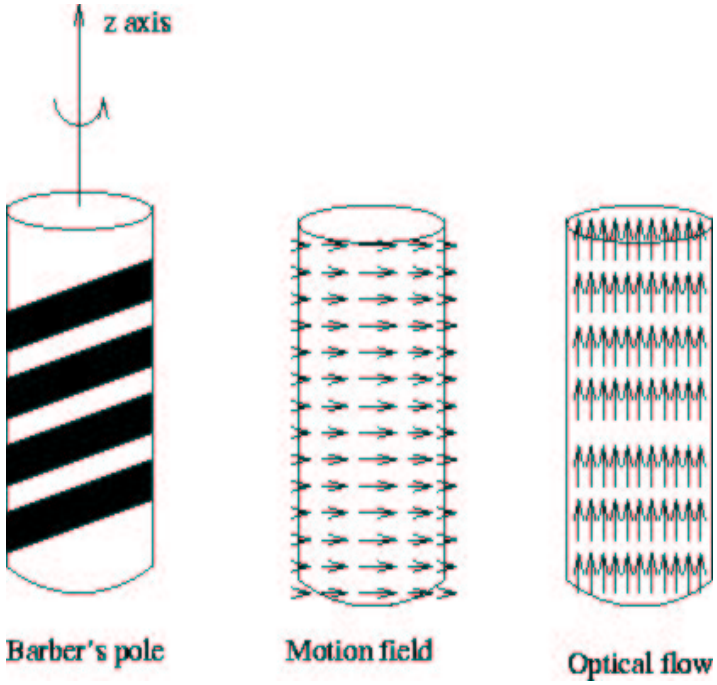


Figure 5.1: Illustration of the difference between optical flow and motion field [53]

Information about the brightness gradient over the image and the brightness gradient in time provides a constraint on the components of optic flow. The flow vector can only be determined in the direction of the brightness gradient. The component perpendicular to this cannot be determined (tangential to the brightness gradient). This is known as *the aperture problem.*

5.1.2 Optical flow measurement

Optical flow can be measured using differential methods, region-based matching, energy-based (filter in Fourier domain) and phase-based (phase behaviour of band-pass outputs) [9]. Optical flow is subject to the aperture problem (only flow parallel to the gradient can be calculated), boundary over-smoothing and multiple moving objects transparency.

Three principal methods for measuring optical flow are:

1. Gradient-based methods
2. Block-based methods
3. Correlation-based methods

The technique used here is the gradient-based method of Horn [25]. The illumination I of a point (x, y) in a brightness image at a time t is a function of the spatial and temporal co-ordinates x, y and t. Then at a a time ∂t later, the illumination will be the same at a point $(x + \partial x, y + \partial y)$, where $\partial x = u\partial t$ and $\partial y = v\partial t$ and u and v are the components of the optical flow in the x and y directions.

$$I(x + u\partial t, y + \partial t, t + \partial t) = I(x, y, t) \tag{5.1}$$

From this the optical flow constraint equation can be obtained after Taylor expansion:

$$I_x + I_y + I_t = 0$$

This is still under-determined as it only defines a plane in velocity-space upon which the solutions must lie. There are three principal methods of constraining this equation. These are: using local optimisation, assuming neighbourhoods have similar velocities; using smoothness constraints[26]; or clustering using Hough techniques [16].

In the present case the smoothness constraints are imposed. To maximise smoothness the integral of the square of the magnitude of the gradient of the optical flow is minimised according to equation 5.2.

$$e_s = \int\int ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))dxdy \tag{5.2}$$

The optical flow constraint equation error must also be minimised.

$$e_c = \int\int (E_x u + E_y v + E_t)^2 dxdy \tag{5.3}$$

Thus, a weighted combination of equations 5.2 and 5.3 must be minimised and a weighting parameter λ must be chosen to weight the error in the optic flow constraints relative to the departure from smoothness.

The optical flow measurement is most accurate at regions where the brightness differs from the surrounds. In areas where the brightness is constant along a direction it cannot be found in that particular direction, and where the brightness does not vary the flow is unable to be found. The values for regions such as these must be deduced from the flow values for the surrounding areas.

For the discrete case an iterative solution can be reached for u and v [25]. The new value of (u, v) at a point is the average of the surrounding values minus an adjustment in the direction of the brightness gradient.

$$u^{n+1} = u^n - I_x[\frac{I_x u^n + I_y v^n + I_t}{1 + \lambda(I_x^2 + I_y^2)}] \tag{5.4}$$

$$v^{n+1} = v^n - I_y[\frac{I_x u^n + I_y v^n + I_t}{1 + \lambda(I_x^2 + I_y^2)}] \tag{5.5}$$

5.1.3 Obtaining spatial and temporal derivatives of a brightness image

In order to perform the iterative calculation for the optical flow field, the spatial and temporal derivatives of the brightness image must be known. For this purpose the derivatives are calculated using masks proposed by Simoncelli in [72]. A matched set of a low-pass pre-filter and a derivative filter is used to obtain a good gradient estimate. Simoncelli [72] shows that the gradient accuracy improves with the size of the filter used. The 3 tap filters he describes in [72] are used: for the pre-filter, [0.2242 0.5515 0.2242] and for the derivative filter, [0.4552 0 −0.4552].

A sequence of three consecutive images in time is therefore necessary for the gradient calculation. This number seems the best compromise between gradient accuracy and number of images which need to be stored for the calculation. The brightness image is the first band of the colour image which has been transformed to the L*a*b colour space (which seems economical, since only the other two bands are taken into account when building the colour model). First this sequence of three images is pre-filtered in time, and then this signal is convolved with the pre-filter in the vertical direction and then the derivative filter in the horizontal direction to obtain I_x, and then with the pre-filter in the horizontal direction and the derivative filter in the vertical direction for I_y. Finally to obtain I_t the sequence is filtered in the time direction using the derivative filter.

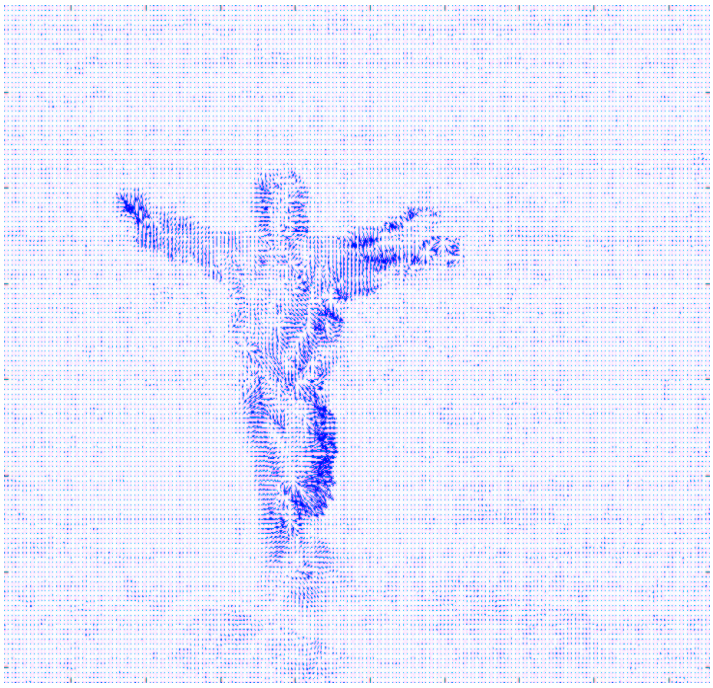


Figure 5.2: Motion vector plot

5.2 Difference image generation

Image subtraction is a crude form of motion segmentation. Subtraction can be done between two consecutive frames or between a frame and a reference image containing only static objects. Since a reference image cannot be depended upon to be unchanging, it is usually desirable to update it in some way in order to account for changing lighting or other moving objects.

The difficulty with using two consecutive images for subtraction is that some pixels that belong to the moving object in both frames will not be classified as changed, and some pixels belonging to the previously occluded background will be classified as changed. A solution which has been suggested is to take two difference images consecutively and combine them using a logical OR operation. This can be done by multiplying the absolute value of the two difference images to obtain one difference image in three colour bands [14] as indicated by equation 5.6. The succeeding step is shown in equation 5.7, where the maximum value in all the colour bands is chosen for each pixel.

$$d(R,G,B) = |im1(R,G,B) - im2(R,G,B)| \times |im2(R,G,B) - im3(R,G,B)| \tag{5.6}$$

$$D = max\{d(R), d(G), d(B)\} \tag{5.7}$$

5.2.1 ΔE measure

A better approach if an empty background image can be obtained is to look for differences between each image and the known background. As mentioned previously, one cannot rely on the background being completely static so a method of updating it over time is necessary. The simplest manner in which this can be done is by observing a large number of reference frames over time and computing the mean and standard deviation. Pixels which are more than a constant number of standard deviations from the mean can then be classified as foreground. This technique can be made more robust by using all three bands of the colour space of the empty background image. If this operation is performed in the L*a*b* colour space the ΔE metric given by equation 3.8 in chapter 3 can be used for estimating this change, using information contained in all three colour bands. The change-detected image thus obtained should correspond more closely with the amount of perceived colour change. This difference image is used in conjunction with the colour probability image to obtain a segmentation.

5.2.2 Variance of ratio measurement

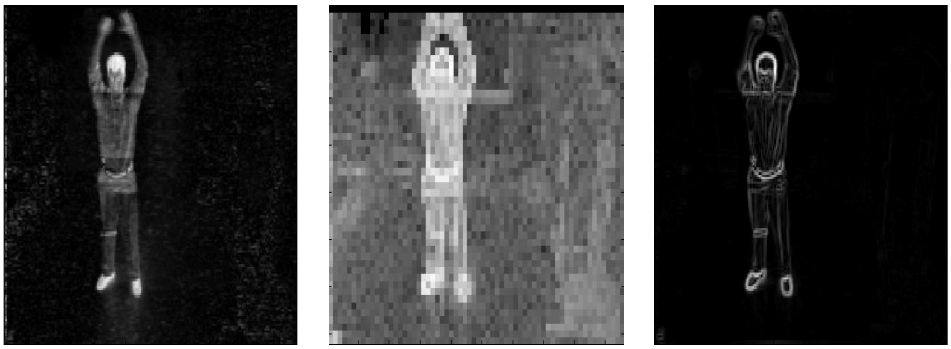
A second change-detection approach used is the one used in [74]. Instead of image differencing the ratio between the current image and the background is taken. This image is subdivided into 20 by 20 blocks, and the variance within each of these blocks is calculated. If this is near zero the region has not changed, and if it is much greater than zero the region is assumed to have undergone change.

5.2.3 Gradient of luminance difference

Shadow boundaries in difference images are frequently less sharp than boundaries between objects. An edge-detected version of the luminance band of the difference image can also be obtained, which can be used to differentiate between shadows and valid foreground difference.

5.3 Kalman filtering

Any image differencing method also requires some compensation for lighting changes, in much the same way as a colour probability density estimation problem does. A common way of doing this is to estimate the changes the scene undergoes over time and update the static background image accordingly. The method used to update the background image, which is subtracted from the foreground in order to obtain the mask of the moving regions, is based



(a) Delta E (b) Variance of ratio (c) Sobel

Figure 5.3: Various change-detection methods

on the method used by [63]. The approach makes use of Kalman filtering in order to model any changes to the background.

5.3.1 The Kalman filter

A simple Kalman filter

If one has two observations about the measurement of a variable (one-dimensional in this case), z_1 and z_2, with associated variances, σ_1 and σ_2, indicating the reliability of the observations, one can combine the information about these two observations to produce a "better" estimate of the measurement and its variance as shown in figure 5.4. The combined estimate is a Gaussian probability density, centred about the new estimate z, with an error variance of s [47]. The values for z and s based on the previous observations are:

$$z = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} z_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} z_2 \tag{5.8}$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \tag{5.9}$$

Hence an estimation $\widehat{x}(t_i)$ based on the two previous observations can be given by:

$$\widehat{x}(t_i) = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} z_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} z_2$$

$$= \widehat{x}(t_{i-1}) + K(t_i).[z_2 - \widehat{x}(t_{i-1})] \tag{5.10}$$

where

$$K(t_i) = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{5.11}$$

is the Kalman gain and $\widehat{x}(t_{i-1})$ is z_1. This means that the best estimate of the system state at t_i is the previous best estimate $\widehat{x}(t_{i-1})$, before the measurement z_2 was taken, plus a weighting factor $K(t_i)$ multiplied by the error between the observation z_2 and the best prediction of its value $\widehat{x}(t_{i-1})$. This leads to the predictor-corrector structure of the Kalman filter, whereby a recursive estimate is made, using all previous information, of the value a measurement will take the next time the measurement is actually made and of the error associated with that measurement. When the measurement is made, the error between it and the estimate is used to predict the next measurement [47].

The Kalman filter thus uses feedback to estimate the state at some time. An estimate is made for the state, and a measurement with associated measurement noise, which is also modelled is subsequently obtained and used to obtain a better estimate of the next state [79].

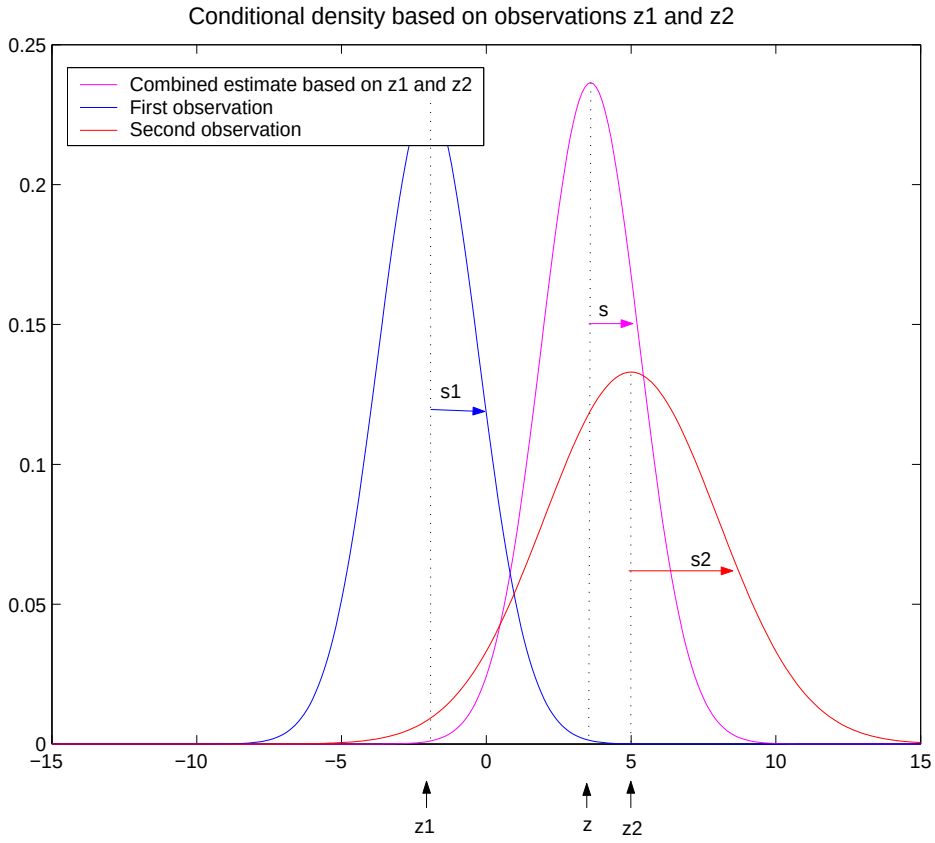


Figure 5.4: Conditional density with variance s of a one-dimensional measurement based on observations z_1 and z_2 with variances $s_1$and s_2.

The Kalman filter approximates the state of a system by assuming some noise to be present

on the measurement of the system values. At time t_i the estimation of the system state is given by:

$$\widehat{x}(t_i) = \widetilde{x}(t_i) + K(t_i).[z(t_i) - H(t_i).\widetilde{x}(t_i)] \tag{5.12}$$

where $\widetilde{x}(t_i)$ is estimated from the previous measurement $\widehat{x}(t_{i-1})$ by

$$\widetilde{x}(t_i) = A(t_i).\widehat{x}(t_{i-1}) \tag{5.13}$$

where $A(t_i)$ is the system matrix which models the relation of the state $\widehat{x}(t_i)$ to the previous state $\widehat{x}(t_{i-1})$.

In equation 5.12 $z(t_i)$ is the measurement of the system state at t_i and $H(t_i)$ is the measurement matrix which relates the state $\widehat{x}(t_{i-1})$ to the measurement $z(t_i)$, where each measurement can be modelled as $H(t_i).\widehat{x}(t_{i-1})$ plus some measurement error with covariance σ_2. $K(t_i)$ is the Kalman gain matrix, which can be seen from equation 5.11 to be a function of the estimated error covariance and the measurement error covariance.

5.3.2 Adaptive background estimation

Kalman filter equations for background estimation

If the intensity values for pixels in an image can be regarded as a function s of x and y at time frame t_i , the system state, which is the estimated intensity at that point is $\widehat{s}(x, y, t_i)$ and the estimated variance of this pixel value in time is $\widehat{sv}(x, y, t_i)$. According to the Kalman filter equation obtained above, the estimation for the pixel intensity and variance becomes

$$\begin{bmatrix} \widehat{s}(x,y,t_i) \\ \widehat{sv}(x,y,t_i) \end{bmatrix} = \begin{bmatrix} \widetilde{s}(x,y,t_i) \\ \widetilde{sv}(x,y,t_i) \end{bmatrix} + K(x,y,t_i).\left\{ s(x,y,t_i) - H(x,y,t_i) \begin{bmatrix} \widetilde{s}(x,y,t_i) \\ \widetilde{sv}(x,y,t_i) \end{bmatrix} \right\} \tag{5.14}$$

with the predictive term

$$\begin{bmatrix} \widetilde{s}(x,y,t_i) \\ \widetilde{sv}(x,y,t_i) \end{bmatrix} = A. \begin{bmatrix} \widehat{s}(x,y,t_{i-1}) \\ \widehat{s}(x,y,t_{i-1}) \end{bmatrix} \tag{5.15}$$

where the system matrix A is given by

$$A = \begin{bmatrix} 1 & 0.7 \\ 0 & 0.7 \end{bmatrix} \tag{5.16}$$

and the measurement matrix H in equation 5.14 is constant, because the measurement is a digital grey-level value.

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{5.17}$$

The Kalman gain $K(x, y, t_i)$, which is dependent on the segmentation, $seg(x, y)$ of the previous frame at time t_{i-1} is given by

$$K(x, y, t_i) = \begin{bmatrix} k_1(x, y, t_i) \\ k_2(x, y, t_i) \end{bmatrix} \tag{5.18}$$

where

$$k_1(x, y, t_i) = k_2(x, y, t_i) = \alpha.seg(x, y, t_{i-1}) + \beta.[1 - seg(x, y, t_{i-1})] \tag{5.19}$$

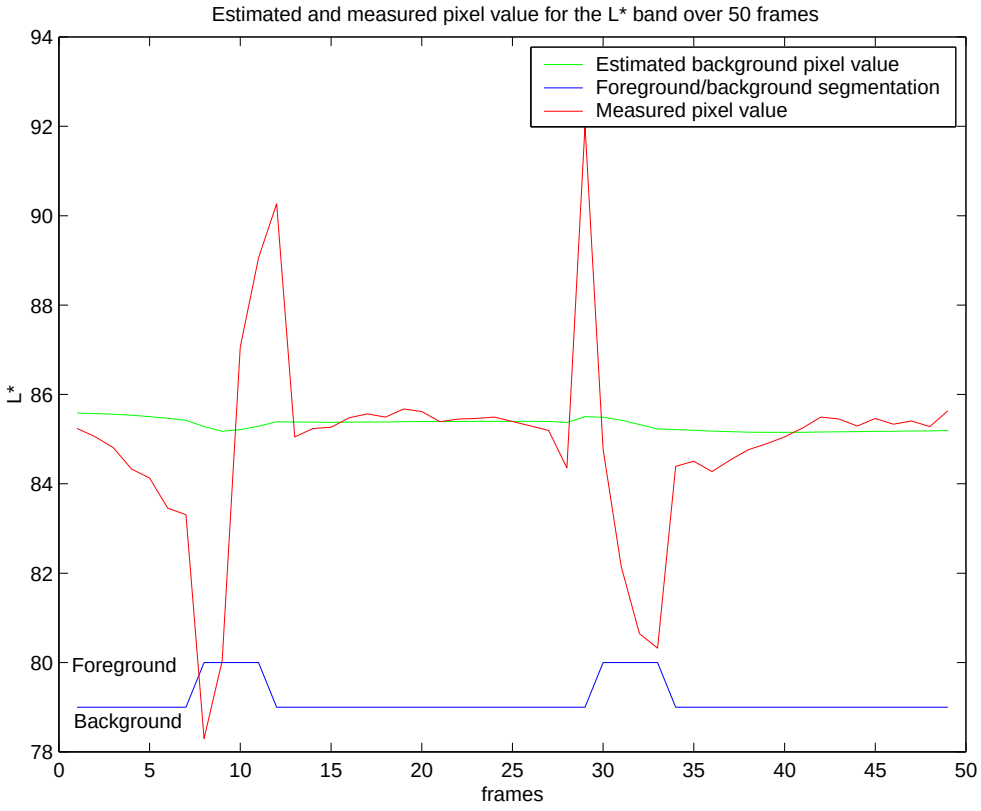


Figure 5.5: Background estimates and actual values of a pixel with its foreground/background classification

Choosing a gain

The gains used in this update method are not truly Kalman gains as they do not depend upon the estimated and measured error covariance. The variables α and β in equation 5.19

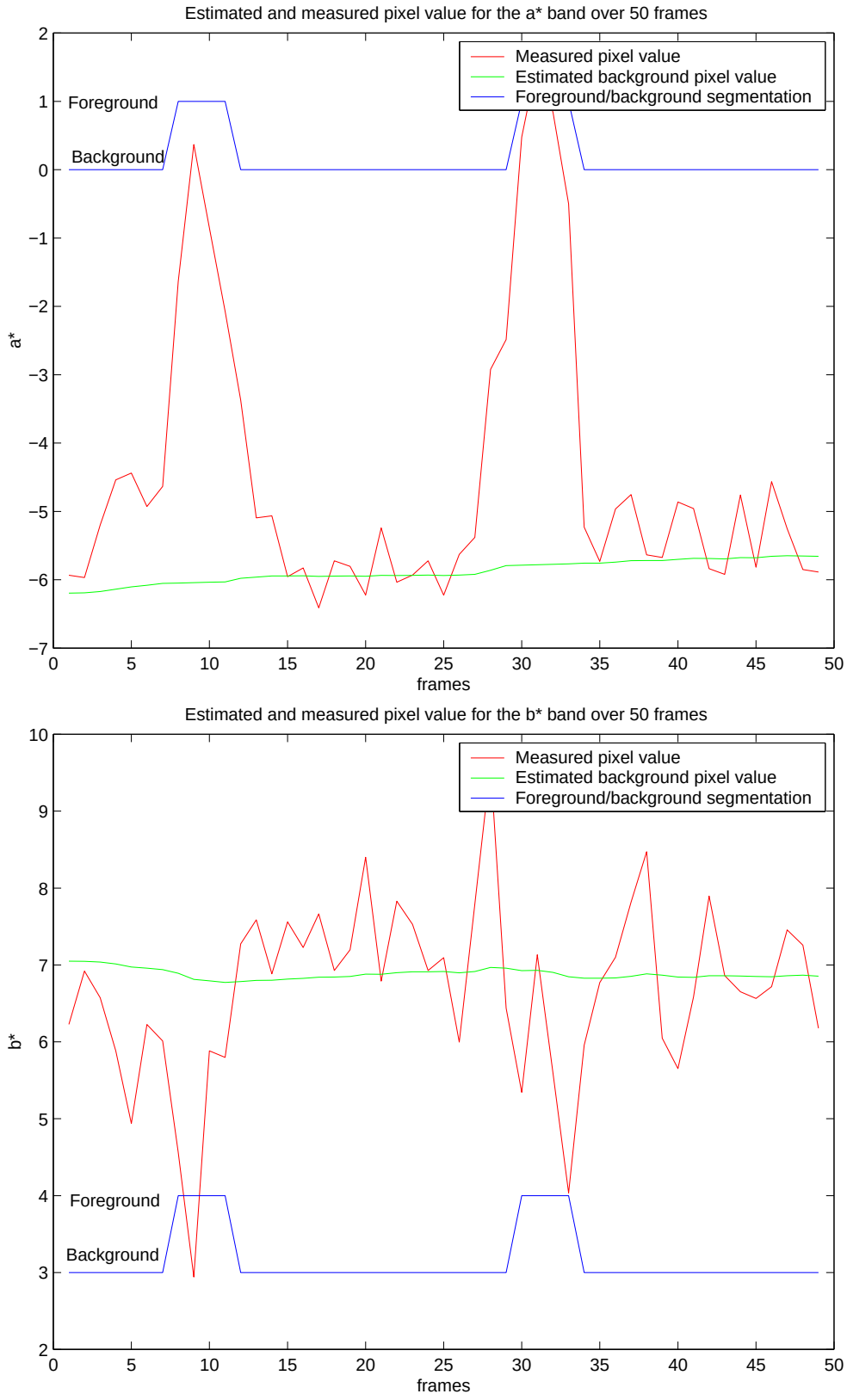


Figure 5.6: Background estimates and actual values of a pixel with its foreground/background classification

control how quickly the filter adapts to changes in the foreground and background. Evidently it is desirable that the system respond as quickly as possible to changes in the background, which could be due to illumination changes, and as slowly as possible to areas which have been classified as foreground. If the foreground is adapted too quickly, stationary parts of the foreground will become incorporated into the background estimation and the resulting segmentation will be full of holes. The manner in which to choose the gain is proposed in [63], which involves some modification of equation 5.19. A pre-estimation is defined using β as follows:

$$\widehat{s\prime}(x,y,t_i) = \widetilde{s}(x,y,t_i) + \beta.[s(x,y,t_i) - \widetilde{s}(x,y,t_i)] \tag{5.20}$$

Then equation 5.19 becomes (remembering, although this is not explicit in the notation, that $\widehat{s\prime}$, $\widetilde{s}$, s and th are all functions of x, y and t_i)

$$k_1(x,y,t_i) = k_2(x,y,t_i)$$

$$= \begin{cases} \alpha & if \quad |s-\widetilde{s}| \geq th \quad OR \quad \{\, |s-\widetilde{s}| < th \quad AND \quad |s-\widehat{s\prime}| \geq th \,\} \\ \beta & if \quad |s-\widetilde{s}| < th \quad AND \quad |s-\widehat{s\prime}| < th \end{cases} \tag{5.21}$$

This means that even if the difference between the image and the estimation is less than the threshold, if the difference between the image and the pre-estimation is greater than the threshold the pixel is still considered foreground and the estimation is performed using the foreground gain α. Otherwise the pixel is considered background and the background gain β is used.

Obtaining a segmentation

The segmentation $seg(x,y,t_i)$ is a binary mask obtained by subtracting the background estimate at time t_i from the actual measurement of the image intensities at t_i, which may of course also contain foreground pixels. Any pixels whose measurement estimation difference is greater than a certain threshold $th(x,y,t_i)$ is classified as foreground and the remainder are classified as background and included in the re-estimation for the following frame. Thus, the segmentation at frame t_i is given by

$$seg(x,y,t_i) = \begin{cases} 1 & if \quad |s(x,y,t_i) - \widehat{s}(x,y,t_i)| \geq th(x,y,t_i) \\ 0 & otherwise \end{cases} \tag{5.22}$$

A more robust segmentation can be obtained if the information from all three colour bands

is used. Each band is filtered separately and has an independent threshold, which adapts as described in the following subsection. The advantage of using all three bands is that one is then able to set the initial threshold for the segmentation fairly high, so that as little noise as possible is obtained in the segmentation. As can be seen in figures 5.7 and 5.8, even sharp peaks in the plot of the difference between the background estimation and the image are below the threshold line in most cases. However, noting that some of the peaks in the difference plot do not actually correspond to the detection of foreground at that pixel, it is as well to have a high threshold for the colour bands in which this happens, which prevents false detection of foreground. If each band is thresholded separately (at a high threshold) and the outputs are combined using a logical OR operation, the resulting combined segmentation contains far fewer false positive pixels than if any one of the bands is used on its own with a lower threshold. Thus in the three plots shown in figures 5.7 and 5.8 the eventual classification of the pixel is obtained only from thresholding the a* band as can be seen in the second plot in figure 5.7. In the other plots the pixels values at the same points fall below the threshold line, but as a result of the logical OR operation, a pixel value higher than the threshold in only one of the three bands is sufficient for a positive classification. In the L*a*b* space image differencing operations on the components bands bring out different regions of interest. This segmentation is used primarily to initialise the colour mixture model and to locate a bounding box for each foreground object in every frame.

Threshold adaption

The threshold used to determine whether a pixel belongs to the foreground or background is also dependent on x, y and t_i. In order to increase the tolerance of the threshold to shadowing, the threshold is made dependent on the variance of the estimated values at that pixel over time. The threshold is thus composed of a fixed part, which is an empirically determined function of the mean difference, added to the variance σ over the last n frames of that pixel, which is calculated according to equation 5.24.

$$th(x, y, t_i) = th_{fixed} + \sigma(x, y, t_{i-1}) \tag{5.23}$$

$$\sigma^2(x, y, t_i) = \frac{1}{n-1}\left[\sum_{t=t_{i-n+1}}^{t_i} s(x, y, t)^2 - n.\overline{s}(x, y, t_i)^2\right] \tag{5.24}$$

where

$$\overline{s}(x, y, t_i) = \frac{1}{n}\sum_{t=t_{i-n+1}}^{t_i} s(x, y, t_i) \tag{5.25}$$

For the pixels where the background is covered for a frame ie. $seg(x, y, t_i) = 1$, the variance $\sigma(x, y, t_{lb})$ at the last time that pixel was classified as background is used, multiplied by an exponential function which is a constant multiple k of the time since that pixel was last background, $t_i - t_{lb}$ as shown in equation 5.26. This means that the variance for a pixel is not biased by frames where the background at that pixel is occluded.

$$\sigma(x, y, t_i) = \sigma(x, y, t_{lb}). \exp k(t_i - t_{lb}) \tag{5.26}$$

5.4 Using motion information to locate a bounding box

5.4.1 Bounding box prediction for segmentation

Since the position of the person in the current frame is known, it seems redundant to test the entire image for colour correspondence with the model when the area to which the person could have moved is in fact extremely limited. If a bounding box for the estimated position in the next frame could be calculated and this area only was tested for colour correspondence, this would lessen the probability of losing the object being tracked, which was the problem with the non-adaptive colour model presented in [62] and [60]. For this reason, a bounding box in each frame is extracted through use of the Kalman-filtering segmentation technique described above. Simply connected objects larger than a certain size have bounding boxes associated with them and the testing for colour correspondence with the foreground model is done only within this bounding box. The accuracy of this segmentation is not critical, as long as the bounding box is bigger, rather than smaller than the actual size of the object it is meant to enclose. For this reason, a morphological closing is performed before the connected components are extracted. It is better to have a bounding box that is too large than an estimate which is too small since the costs are in the first case, a slightly longer computation time, and in the second, an imperfect segmentation because of inaccurate bounding box estimation. The bounding box obtained in this way is the assumed to be the largest estimate of the size of the person in the frame.

The estimated size of the person in the bounding box is used to obtain the priors $P(O)$ and $P(B)$ in equation 4.11 for determining the membership likelihoods of a pixel to the foreground or background classes. $P(O)$ is the ratio of the size of the segmented object obtained in the manner described by section 5.4 to the total size of the bounding box, which corresponds roughly to the proportion of pixels which are not expected to be classified as background.

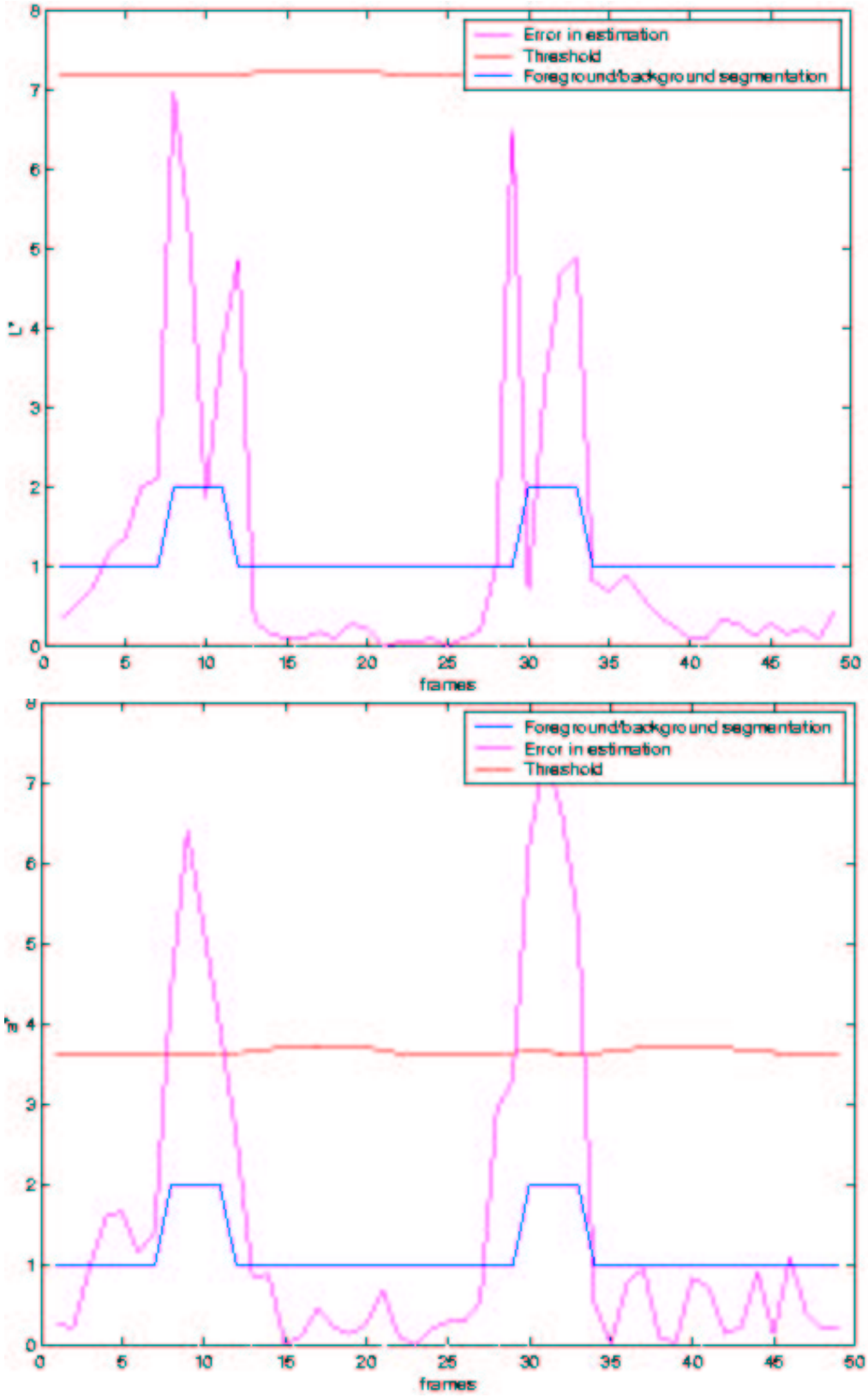


Figure 5.7: Plots showing the estimation error and the threshold for one pixel in different colour bands

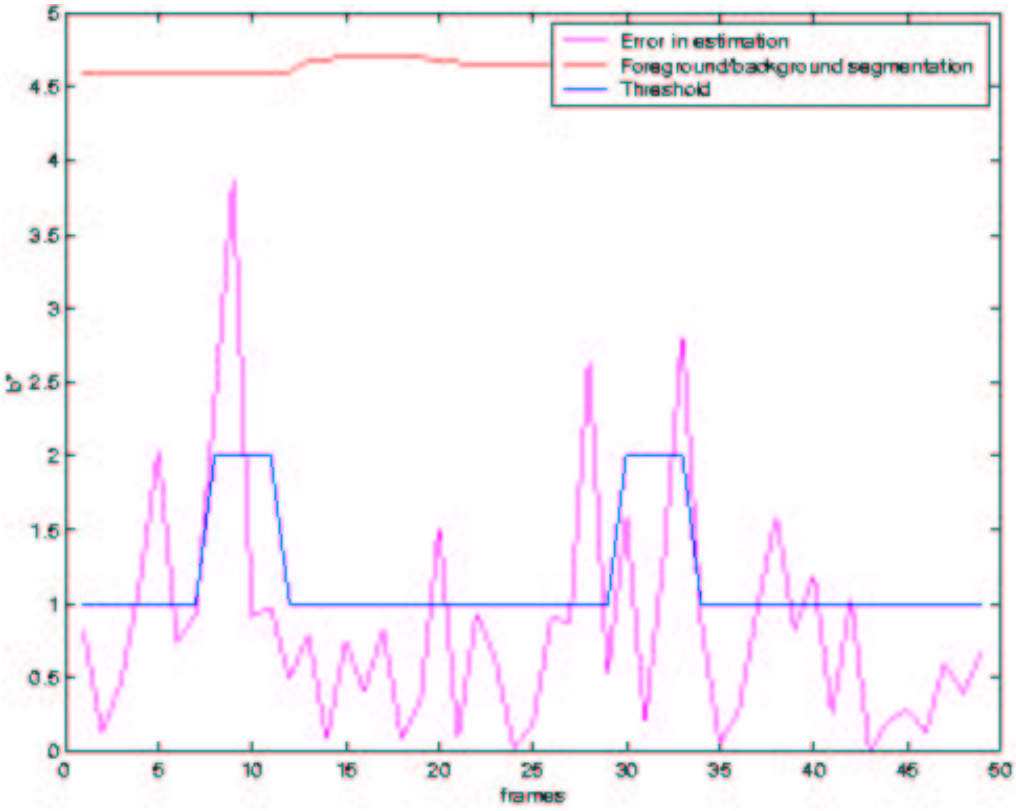


Figure 5.8: Estimation error and threshold for one pixel

5.4.2 Bounding box prediction for tracking

The predicted bounding box for the succeeding image can also be calculated, without having any knowledge of the segmentation of that frame, using the co-ordinates of the bounding box for the current frame. The bounding box position for the current frame is recalculated for every image *after* the segmentation has taken place, and can thus be considered an accurate estimation of the position of the person in the frame.

The gradient-based optical flow estimation algorithm described in section 5.1 can be used to estimate the optic flow parameters (u, v) of the image. Since this is computationally expensive because it involves an examination of the surrounding pixels, and it is unnecessary to have a very precise measure of the motion just for the bounding box localisation, the optic flow is only calculated at every 20th pixel. The values of the motion parameters (u, v), which correspond to flow in the x and y directions are rounded to the closest integer value [73]. This means that flow values which are below 0.5 become 0 and are considered effectively stationary regions.

The maximum and minimum of each of the u and v components of the flow field, which correspond to the four maximum possible directions of motion of the bounding box, ie. the x, $-x$, y, and $-y$ directions, are extracted in each frame. These values are added to the co-ordinates of the bounding box in the current frame to obtain an estimate of the position of the bounding box in the next frame. The flow field, however, is only an approximation of the actual motion and is based on the motion that has already occurred: the gradient estimation includes the current image and the two preceeding it. Therefore there is scope for error, particularly at image frames where a change of direction takes place, which cannot be foreseen.

The segmentation then only takes place within the bounding box which has been predicted for any particular frame. If a large discrepancy exists between the largest bounding box estimation and the predicted bounding box using motion estimation, an area outside the predicted bounding box is also tested for correspondence with the colour model.

For tracking a simpler form of motion estimation can be used by estimating the velocity from the difference between the bounding box positions between frames. This similarly gives a net motion estimate in each of the four directions: x, $-x$, y, and $-y$, which can be used to predict a bounding box in the following frame and is a lot less computationally intensive. To make the velocity estimates less sensitive to outliers, the velocity is computed recursively, so that previous velocity estimates become part of the current estimate.

Chapter 6

Combining Outputs

6.1 Reasons for requiring additional classification constraints

The ultimate goal is the best possible classification method for the current application, which requires different methods depending on the task. Experimentally the best performing classifier should be chosen to solve the problem. The trouble is that it is not always obvious which classifier this is. The data misclassified by each of the different classifiers might not, however, overlap and thus combining the classifiers in some way could lead to an even better correct classification rate [38]. This is certainly the case with the various segmentation techniques discussed thus far.

Even if the Gaussian mixture classifier based on colour distributions were assumed to produce a perfect classification of foreground and background every time, there is at least one set of circumstances under which it can only perform less than perfectly. If some of the colours of the foreground which have been modelled by the mixture of Gaussians appear elsewhere in the image it seems obvious that these colours will be mistakenly classified as foreground. By only requiring this classification to be successful locally in the image, through use of the bounding box, some of these extraneous misclassifications will be eliminated. However when the person moves in front of a similarly coloured object it is likely that the classification within the bounding box will fail.

Likewise, segmentation using only a difference image is likely to fail. Ambiguity can occur in several cases: namely misclassification of background as foreground due to shadowing of the background, and misclassification of foreground as background. The latter manifests itself as holes within the foreground and depends on the colour of the background object *behind* the foreground. In a background difference image the foreground appears "transparent" and objects behind it which are subtracted from it appear in the segmentation.

For this reason, neither the colour classification nor image differencing should be used on its own for segmentation, but both should rather be combined, possibly with additional foreground/background classification methods which provide different information. Each technique by itself has many weaknesses, but in combination with another method of classification the segmentation results improve dramatically.

6.2 Possible methods of combination

It seems clear that the combination of two different type of foreground/background classification must perform better than one alone. Using two methods usually helps to eliminate ambiguities which present themselves by the use of one classification process.

Much information can be lost by thresholding each output before combining it, and by requiring strict constraints on the combination. Some possible methods of combining these various segmentation methods making use of the degree of class membership they provide, are given by [38]:

- Maximum rule - The data point is assigned to the class that has the maximum *a posteriori* probability.
- Median rule - The median (rather than the mean) of the class membership probabilities is used to classify the data. This is less sensitive to outliers than the mean.
- Majority vote rule - Hard class membership values are assigned and the data point is assigned to the class which receives the largest number of votes from the independent classifiers.
- Sum rule - The output according to this rule is the sum of the *a posteriori* probabilities produced by all of the classifiers.
- Product rule - The output according to this rule is the product of the *a posteriori* probabilities produced by all of the classifiers. This has the unfortunate effect that any one classifier that has a low confidence will affect the overall output.

Kittler [38] found the sum rule to perform the best, followed by the median rule, followed by the majority vote rule.

A further method of combining classification methods is to use the classification outputs as inputs to a neural network. An important issue is that the input classifiers be different. In the case of neural networks as inputs, they could be trained on different data. Otherwise classifiers which operate on the same space and can be considered two estimates of the *a priori* class

probability can be combined, or classifiers which correspond to different measurements, so the classifiers operate in different measurement spaces. The colour classification and difference image combination falls into the latter category.

Methods of classification depend also on the kind of output data supplied. Sometime a voting scheme is sufficient if the classes are labels. If continuous outputs such as *a posteriori* probabilities are supplied, the output could be an average or linear combination of the inputs. Otherwise fuzzy rules can be constructed and applied and sometimes it is helpful to use the outputs of the classifiers to be combined as inputs to train yet another classifier.

6.3 Combining colour probabilities and difference images

Assuming that the changes between the background and foreground and the similarities between the colour distributions of the background and foreground fall in different regions in the image, the two images can be simply combined to produce a better one by thresholding each and performing a logical AND operation.

First, if the assumption that any lighting change in the image and colour similarity between object and background are independent of each other is violated this algorithm will fail. That is to say if the person moves so that his shadow falls on a static object which has the same chromaticity as an item of his clothing, that object will be mistakenly classified as part of the person.

Secondly, a drawback rather than a reason for failure, is that this method requires absolutely that both methods classify the pixel as foreground. The possibility that a very high value of colour probability and low change may also constitute a pixel that belongs to the foreground is not considered. Likewise the reverse case, a high difference image value and a low colour probability, although this is more likely to be caused by a shadow, which of course should not be considered in the classification process.

The weakness of this way of combining the two methods seems to work against the very reason for using a colour model at all, since it means that even if the foreground's colours are correctly classified, if the background subtraction does not agree, the correct classification of certain pixels is rejected.

The different forms of background/foreground classification which we have available thus far are: the ΔE difference image measure in L*a*b* space, the hard segmentation obtained through the Kalman filtering method with adaptive threshold discussed in the previous chapter, the variance of the background image ratio and the gradient of the luminance difference also described in the previous chapter, and lastly the colour probability image obtained

through the Gaussian mixture model classification process. These five different forms of information about the location of a foreground object: each with their strengths and weaknesses, can be combined in order to produce a segmentation of the object which is better than any of the individual thresholded components on its own.

Two different methods of combination were attempted: first the sum rule: adding the class membership values and thresholding at a certain limit and second, a neural network, where each output is presented as an input to a neural network together with a target segmentation for those pixels.

All inputs to the combination algorithms are scaled to confidence values between 0 and 1. Each of these can be seen as continuous probabilities of membership to the foreground class. Only the Kalman segmentation has a hard decision limit (1 and 0) enforced. The colour probability image, difference image and Kalman segmentation are all low-pass filtered to slightly reduce the influence of any misclassifications in the initial outputs.

6.3.1 Method 1: the sum rule

An obvious strategy is to combine the unthresholded images which result from the colour and difference images and use a weighted combination of these to obtain an image S which can then be thresholded. Equation 6.1 shows how this can be done.

$$S(i,j) = \frac{w_1 D(i,j) + w_2 C(i,j) + w_3 R(i,j) + w_4 K(i,j) + w_5 S(i,j)}{\sum w} \tag{6.1}$$

D is the unthresholded absolute difference image, C the unthresholded colour probability image, R the variance of the ratio between the foreground and background images, K, the the segmentation obtained through the Kalman filter process, S the edge-detected difference image and w is a weight value.

Weighting all inputs equally works as well as other possible weightings, so $w_1 = w_2 = w_3 = w_4 = w_5 = 1$. The sum obtained in this way is thresholded to produce the segmentation.

6.3.2 Method 2: the neural network approach

The second approach was to train a multi-layer perceptron with 15 hidden layers, using a hand-segmented target image. The confusion matrix for the data obtained after training for 100 iterations is shown in figure 6.1. It can be seen from this that the network created, if it were to be used to classify the same data, would classify 7703 pixels in the training set correctly as not belonging to the foreground class and 1620 pixels correctly as belonging to

the foreground class. On the misclassification side, 47 pixels would be incorrectly classified as coming from the foreground class when in fact they did not and 35 would be incorrectly classified as not belonging to the foreground class.

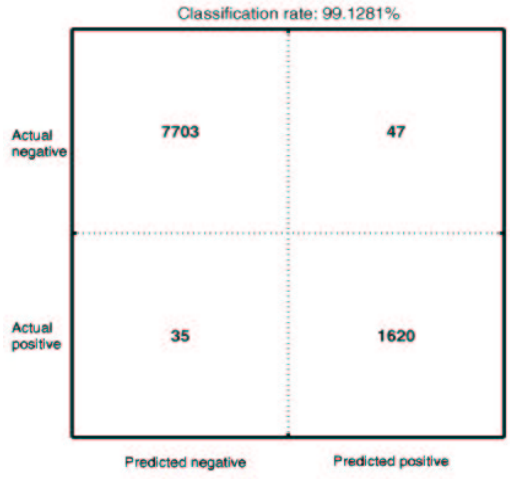
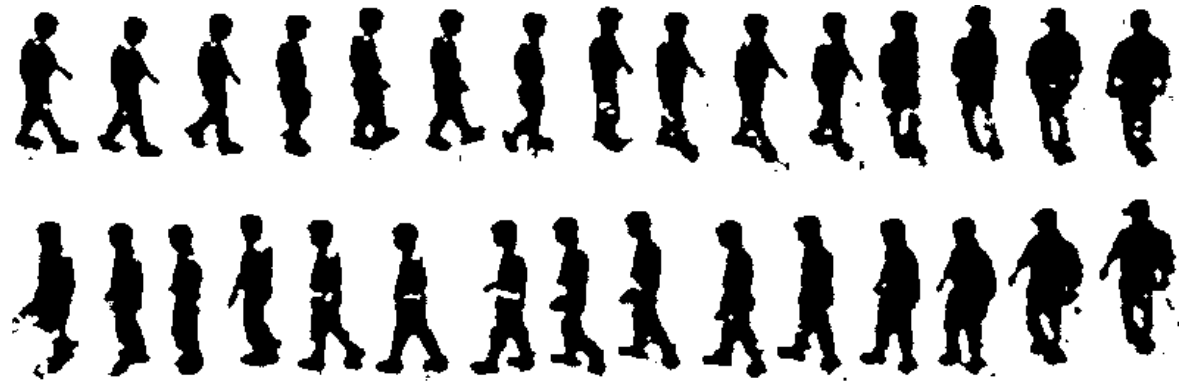


Figure 6.1: Confusion matrix

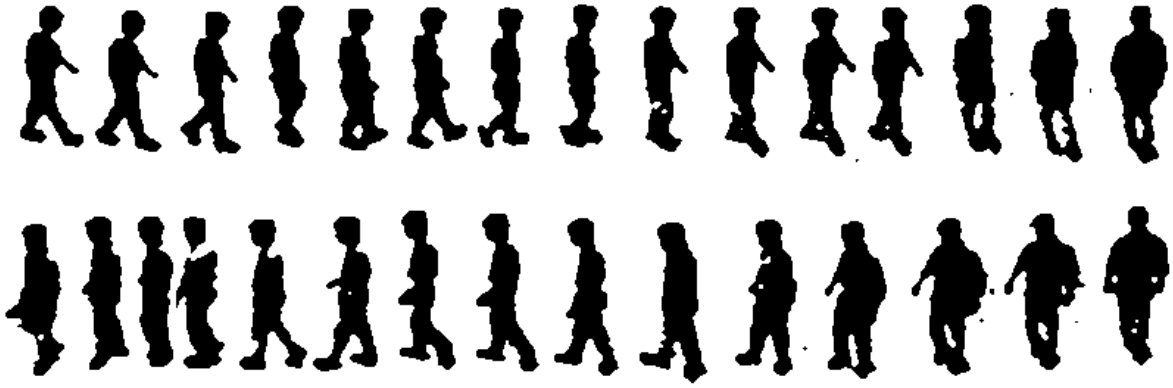
6.4 Evaluation of the two different approaches

Segmentation results for a sequence are shown in figure 6.2. Qualitatively the weighted addition seems to perform better then the multilayer perceptron, as the silhouette boundaries are slightly smoother. The results of each segmentation method for this sequence of 50 images were compared to a hand-segmented version of the sequence. Percentage errors were calculated for the false positive classifications, false negative classifications, correct classifications overall and percentage of foreground correctly classified. Since the segmentation technique in each case involves estimating a bounding box and performing the segmentation only within the bounding box, it is not clear whether the percentages should be calculated over the entire image, or just within the bounding box. There is actually some justification for computing the error over the entire image, as it is possible to obtain an inaccurate bounding box, which might lead to some segmentation error which falls outside the bounding box. For the purpose of comparison it is not really crucial which results are shown, but for completeness percentages over the entire image are shown in tables 6.1 and 6.2 and percentage errors within the bounding box only are shown in tables 6.3 and 6.4.

From the tables it can be noted that the multi-layer perceptron method has fewer false positive classifications and the sum rule method fewer false negatives. Each method produces roughly the same number of pixels correctly classified overall, although the multi-layer per-



(a) Multilayer Perceptron combination scheme



(b) Combination by weighted addition

Figure 6.2: Segmentation results for a sequence using two different combination techniques

	% False Positive	% False Negative	% Pixels correct	% Foreground Pixels correct
Mean	0.81	0.42	98.74	92.98
Median	0.77	0.46	98.80	93.33
Max	1.6	1.02	99.23	97.09
Min	0.44	0.18	98.03	86.42

Table 6.1: Error rates over the entire image for 50 frames for a neural network

	% False Positive	% False Negative	% Pixels correct	% Foreground Pixels correct
Mean	0.92	0.4	98.69	93.86
Median	0.84	0.37	98.76	94.22
Max	2.47	0.89	99.13	97.63
Min	0.58	0.17	97.32	85.63

Table 6.2: Error rates over the entire image for 50 frames for sum rule

ceptron seems to have a slightly higher correct classification rate. Likewise, the percentage of foreground correctly classified is almost the same for each method, though the sum rule seems to have a slightly higher success rate. There does not appear to be much to choose between the two from these data. It must be noted, however, that in the case of bad performance of one classifier, namely the colour mixture model when it has adapted to the wrong data, the multi-layer perceptron method is far less affected than the weighted sum method.

Thus, if different weights are placed on priorities, different combination methods should be chosen. If it is a matter of extreme importance to have as few as possible misclassifications of foreground, the sum rule should be chosen; however, if it is more important to have no background mistakenly classified as foreground the neural network method should be chosen. Moreover, if the system is to be robust against the temporary failure of one component classifier, the multi-layer perceptron method is the better approach.

This said, the difference in classification rates can for the most part be overcome by median filtering the unthresholded outputs of both the sum rule and the multi-layer perceptron and performing morphological operations on the thresholded results.

	% False Positive	% False Negative	% Pixels correct
Mean	5.63	3.45	90.92
Median	5.43	3.27	91.05
Max	8.81	6.72	93.53
Min	2.49	1.61	85.65

Table 6.3: Error rates within the bounding box for 50 frames for neural network

	% False Positive	% False Negative	% Pixels correct
Mean	6.53	3.02	90.44
Median	6.27	2.76	90.57
Max	10.58	6.65	93.40
Min	2.86	1.08	84.80

Table 6.4: Error rates within the bounding box for 50 frames for sum rule

Chapter 7 now introduces the problem of tracking the various people which have been identified through the segmentation algorithm that has been developed up to this point. This involves

re-identifying the segmented blobs in subsequent frames using image features extracted from the blobs.

Chapter 7

Tracking

The first section in this chapter describes the problem of tracking and introduces the low-level image features which are commonly extracted from objects in order to be able to track them. Section 7.2 describes the development of a simplified tracking system in which the segmentation is made robust and fast through use of a blue screen.

7.1 Tracking humans in motion

Tracking and identifying multiple persons in a sequence of images is an essential task for applications which involve computer surveillance of any kind. Much work of this sort has been done in the last few years particularly [31] [37] [81]. Tracking does not invariably rely on segmentation, but sometimes templates are matched across an image instead, in order to find the location of an object whose physical description is known. Tracking involves mainly identifying objects in a sequence and being able to gather information about its past, present and short-term future position. If tracking is done using segmented objects or "blobs" [81], as it is most frequently, and as is the case here, certain features need to be extracted from these "blobs" in order to establish a correspondence between the objects located by each segmentation, and the objects that are known to exist.

Bobick et al. [31] define contextual information as "knowledge about the objects being tracked and their current relationships to one another". They use the contextual information to adaptively weight the image features used for matching objects to identities. They also make use of the "closed-world" assumption - the assumption of regions of space and time where the context is assumed to be known. This closed world assumption also lets the tracker know which image features should be selected for tracking and which are unreliable at any point in time or space. These assumptions serve to define the knowledge that is available about the

scene and its contents and to exploit this in order to make certain predictions and assumptions.

Features commonly used to match segmented objects with an identity are estimated size, colour, velocity and current and past position [31]. These are used to match each object in the previous frame to an image region in the following frame, thus following its progress over time. Shape is another possible feature to use, but in general it is subject to change to such an extent that it is not even reliable between two consecutive frames . Velocity information is more frequently used in conjunction with a predictive filter to estimate an object's future position, instead of explicitly being used in the matching stages [67] [40]. Other useful information is available such as the length of time an object has been in the same place and whether it is adjacent to (and likely to be confused with) another object. This information can be used to adapt the weighting of the features, as they become less reliable [65] [31].

Several sources of error exist for tracking. These are primarily due to segmentation unreliability and complicated occlusion situations. Some of these problems are as follows.

1. Segmentation errors propagate to all features.
2. No reliable features can be extracted from multi-object regions, unless people can be tracked during occlusion [37].
3. Properties of multi-object regions may change, while the identities are not being updated.
4. Matching evaluation is difficult, taking into account global and local optima, particularly after an occlusion.
5. If a definitive match is made every frame, a match cannot be made or altered over a few frames.

The basic principle behind the tracking algorithm used in this thesis can be summarised as follows. In every frame all unoccupied regions are matched to a known object using a feature vector consisting of colour, distance, size and velocity, with appropriate weighting. In similar work by Bobick [31] 0.4 is used for colour, 0.5 for distance and 0.1 for velocity. Then all unmatched objects are compared to already occupied blobs, if their size is great enough, to check for occluded people. Colour is not reliable in this instance, so in [31] distance, weighted 0.75 and velocity weighted 0.25 are used. Different weightings are used in this thesis from those used in [31].

The other important differences between the tracking algorithm described in this chapter and that of [31] are the use of "chroma-keying" for segmentation described in section 7.2.1; the incorporation of a velocity estimate into predicted position instead of the use of an explicit velocity feature, and the use of this predicted position to anticipate occlusions, both of which

are detailed in section 7.2.2; the matching procedure proposed for occluded objects in section 7.2.3; and the confidence values associated with the possible matches, which are also described in section 7.2.3.

7.2 A prototype system - tracking in a *Smart Room* environment

7.2.1 Decoupling segmentation and tracking

It is impossible for a tracking algorithm based on segmentation to perform well unless the segmentation is efficient, and yet segmentation is notoriously unreliable. As a preliminary investigation intended to establish the accuracy with which it is possible to track people in a controlled environment, tracking was initially performed in a *Smart Room* environment. This isolates the segmentation from the tracking module and removes possible sources of error due to segmentation. In addition it makes possible further work involving segmentation and tracking in various more complex environments.

The *Smart Room* setup consists of a uniform blue backdrop, which is evenly lit from the front and from above. This is similar to what is commonly done in the film industry for special effects, when an actor is "chroma-keyed" from the background so as to be superimposed on another background. For this purpose green or blue backgrounds are commonly used, as these hues appear the least in human skin tone; but any uniform colour may be used.

This chroma-keying process makes the segmentation of regions of interest far less time consuming. The segmentation is easily performed by eliminating pixels which are close to pure blue: by taking a threshold of the ratio of blue to another colour (red or green). This makes segmentation more reliable than a simple background difference method which provide approximate locations of moving blobs as is used in [31].

The binary mask thus obtained consists of several connected regions of various sizes. Only connected regions with a pixel count of greater than a certain proportion of the image size are considered to be regions of interest for the purpose of tracking. The remaining areas are eliminated and the regions of interest are each given a label.

7.2.2 Describing and tracking moving regions

The algorithm used is a logic-based occlusion reasoning framework, which identifies people after occlusion has occurred and not while occlusion is taking place. Similar occlusion reasoning processes are used in [31] and [39].

The algorithm relies on colour and position information, extracted from objects over time, in order to locate and label the same objects in the frames that follow. It is able to anticipate occlusions using position and estimated velocity and re-identify objects once they have re-appeared, or become separated from all other objects.

Extracting features from objects

Each object that has been identified as a separate connected region in the frame, has several quantities associated with it. The position (x and y co-ordinates), size (length and breadth of the bounding box), velocity (net motion in the x and y directions) and mean colour (R, G and B values) are all calculated and stored in every frame t. The velocity and mean colour, $\mu_c(t)$ are low-pass filtered over time in order to obtain a reliable estimate of these values without sudden large changes due to quick movements, dropped frames or transient colour changes. Equation 7.1 illustrates how this is done using α as a weighting factor.

$$\mu_c(t) = \alpha\mu(t) + (1 - \alpha)\mu_c(t - 1) \tag{7.1}$$

A feature vector, v, consisting of the position, colour and size estimates is created in order to perform the matching. In every frame each object in the new frame is matched to one of the unique object descriptions which have been obtained from the previous frame(s), using a minimum Euclidean distance measure between the 7-element feature vectors. Objects for which no match can be found are explained using the contextual information about the objects and the scene.

Tracking objects from frame to frame

As is the case in [31], a list of closed-world regions is also kept in which information about the objects composing that image region. Assumptions are made about the composition of the scene and the physical rules affecting the people within it. First, it is assumed that a person leaving the image, will do so to the left or the right of the frame. In other words the camera is set up facing a wall (without a door), such that it is not possible for someone to leave the area without moving towards the right or left edges of the frame. This is the "closed-world " assumption used in [31].

Secondly it is assumed that if two people enter the scene in such a way that they cannot be visually separated from one another from the camera's point of view, they will be tracked as one person until such time as they move apart from the camera's point of view. In addition, once a person has left the scene, his/her recorded description is cleared as he/she will be

classified as a new person upon re-entering. No distinction is made between whether the scene objects are in fact people or inanimate objects, as the segmentation by exclusion of blue does not reveal any information about the objects detected apart from position, size and colour.

A range of logical decisions are made in the respective cases where one of the objects present in the previous frame does not find a match in the current frame, or an object in the current frame does not match an object description from the previous frames. This is similar to the closed world assumption used in [31] and the *a priori* knowledge of the scene assumed in [39]. The simplified flow diagram for the reasoned decision process is shown in figure 7.1. If the number of objects changes in the next frame, the recorded object descriptions are added or deleted once it has been established which object has recently appeared in or left the scene.

For matching between frames when there is no occlusion or no change in the number of objects detected, size and position in combination with velocity (that is to say *predicted* position) are used to perform the matching. In the absence of occlusion a definitive decision can generally be made based only upon an object's position and expected position in the following frame. A threshold is set on the maximum allowable distance between an object's expected position and a possible match, in case one of the objects has left and a new one arrived. If this threshold is exceeded, a procedure is followed which tries to locate and explain the new arrival and the disappearance of one of the previous objects, to avoid mistakenly matching a person who has disappeared to the wrong image object.

If a new image object is detected, all the recorded objects are matched to the closest segmented object in predicted position and size and the remained segmented object(s) are assigned new identities. This means that a new person may appear anywhere in the image area, which allows the system to recover from a failure such as an occlusion that has been missed. Likewise, if fewer segmented objects than recorded objects appear, each segmented object is matched to the closest recorded object. The decision is made whether the object which has no match in the current frame, has left the scene or passed into an occlusion with another object, based on its distance from an exit point and its likelihood of occlusion.

Predicting occlusions

To cater for the eventuality that two people will pass each other and their bounding boxes appear to coincide, the likelihood of occlusion in the next frame is calculated. Once positions, velocities and colours for each person have been obtained, the velocity vector is added to the position vector to create new predicted bounding box co-ordinates and a small margin of pixels is added around these predicted positions. This is a very simple motion model, making no use of predictive filters, such as the Kalman filter. The intersections of all the predicted bounding boxes for a frame are calculated in order to find out which objects' bounding boxes are likely

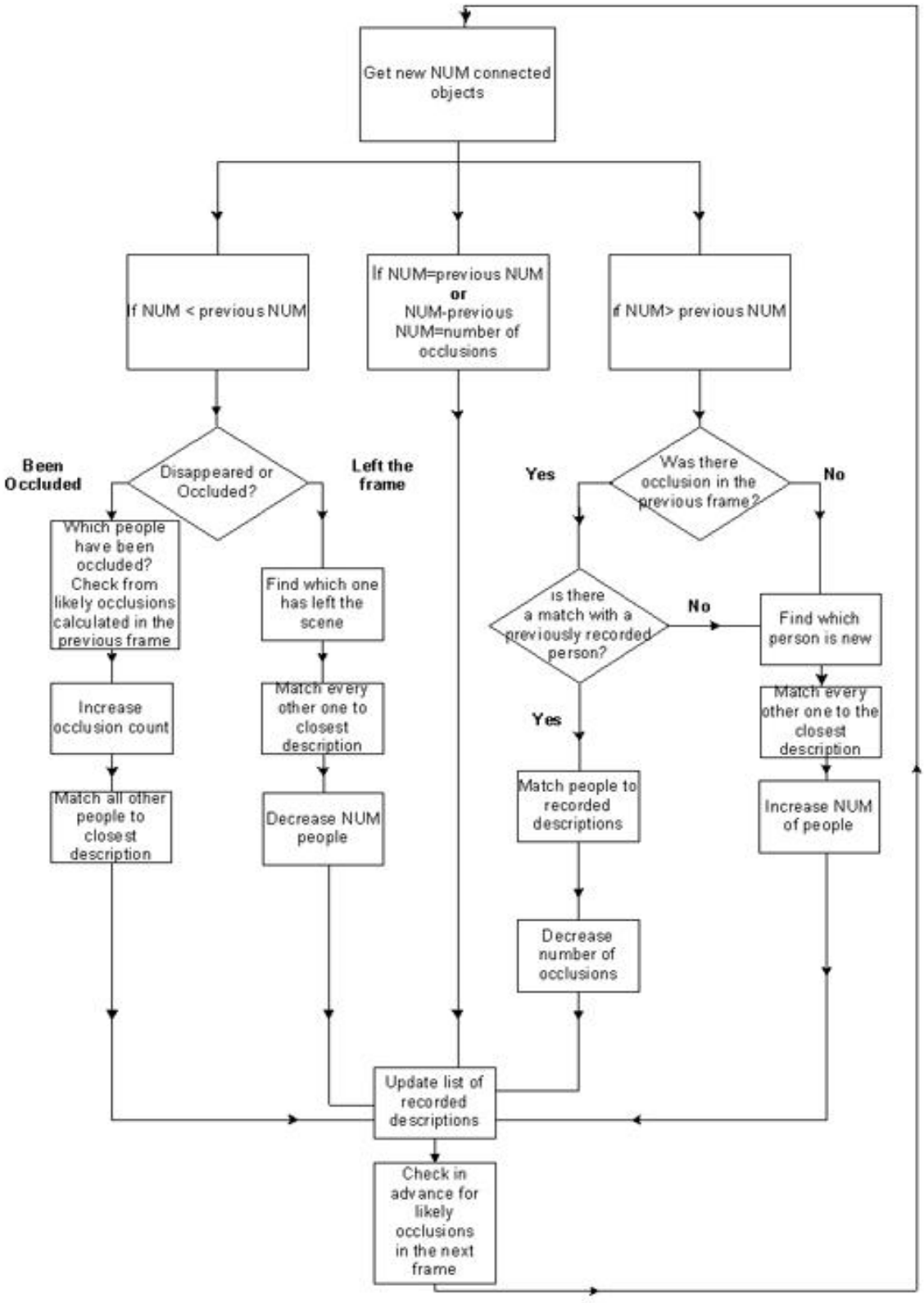


Figure 7.1: Flow diagram for tracking algorithm

to coincide. These intersections correspond to objects which pass behind or in front of another object, or merely touch each other from the camera's viewpoint. The possible intersections are ranked in order of area from largest to smallest, the two bounding boxes with the largest intersecting area being the two people most likely to occlude each other in the next frame.

Once detected, such possible occlusions are confirmed when there appear to be fewer objects in the segmented image. When this occurs, each object description is matched to the segmented object with the smallest Euclidean distance from its feature vector, as in every other frame. The object description which remains once all the other objects have been matched is assumed to have either left the camera's field of vision or passed into the bounding box of another object. At this point, the list of possible occlusions is consulted to determine whether an occlusion had been anticipated for that object, or if the object had been moving towards an "exit". The latter being the case, the object is deleted from the current record. In the case of the former, the two occluding objects share a bounding box and their colour estimates are no longer updated until they separate.

Multiple occlusions are predicted as easily using this method, since, if an object is sharing a bounding box with any other two objects, all three are likely to occlude each other and can be detected when they do. Multiple separate occlusions, when more than one set of people pass each other, also fit neatly into this algorithm. Changes within the occlusion setup are slightly more difficult to detect. For instance, one person involved in an occlusion can leave that group and enter the bounding box of another person or group in the next frame without becoming a separate object in between. This occurrence is detected by looking for a sudden large change in the sizes of the bounding boxes.

7.2.3 Resolving ambiguities: an object matching algorithm

Matching objects after occlusion

The greatest chance of error naturally occurs when two or more objects pass each other. Ambiguities in such a case are resolved by matching the colour and the bounding box position and size, appropriately weighted, of each object in the new frame to each object whose description has previously been recorded.

When a bounding box is known to contain two or more objects, or equivalently when two or more objects have the same bounding box, the colour is not re-sampled but the previous estimated mean colour of the object is kept unchanged. Likewise, a foot position (the lowest edge of the bounding box) is recorded for each person only when objects are not seen to be occluded. Once two objects have re-appeared each of their colours is compared to the previous colour estimates for every object and a match is made such that the sum of the

distances between each new object and its matching object description is minimised, and every new object is matched to one unique object description. The foot position is used to check that this is indeed a likely match, since it is assumed that although people may have swapped places, so that the bounding box co-ordinates are not a reliable estimate of who is who, the foot position should not change much between frames.

The distance is calculated for every feature between each object and every recorded description. A separate matrix of distances is calculated for every feature and these are then added together using an importance weighting, which is empirically determined to provide the most reliable match under the widest variety of conditions, in order to obtain total distances between the feature vectors of every segmented object and every recorded person.

For matching after occlusions, where all objects have been occluded, the features are weighted so as to favour the colour information, which is now the most reliable descriptor. The colour is weighted 0.4, the position 0.1, and size and previous foot position 0.25 each. If not all people present have been occluded, position is more important a feature, since the people whose positions are unambiguously known can be easily matched. In this case the colour is weighted 0.4, position 0.3 and size and foot position 0.15 each.

A global best match is then sought, incorporating knowledge of all the possible matches, since simply matching each object to its closest record might lead to conflict and a local best match might not be optimal. In figure 7.2 each person, which is in fact a point in a 7-dimensional feature space, is represented by a star on a two-dimensional plane and is matched to a unique object, represented by a triangle in this space of reduced dimensionality. As can be seen from the diagram the best local match might not lead to the best overall decision based on proximity in this space. The closest match to the red star is clearly the green triangle, but if the other points are taken into consideration it is clear that once the red star and the green triangle have been matched there is no close match for the green star. A better match would be to match the red star to the red triangle and the green star to the green triangle as has been done in the figure. Likewise the blue triangle and the yellow star are the best local match, but in order to minimise the total sum of distances between matches the blue star should be matched to the blue triangle and the yellow to the yellow.

A distance matrix is constructed which contains the weighted Euclidean distance in the feature space from every new object to every stored object description. The distance between object i 's feature vector v_i and the feature vector belonging to object description j (equation 7.2) is inserted in position (i, j) in the distance matrix D, shown in equation 7.3.

$$d_{ij}^2 = \sqrt{v_i^2 - v_j^2} \tag{7.2}$$

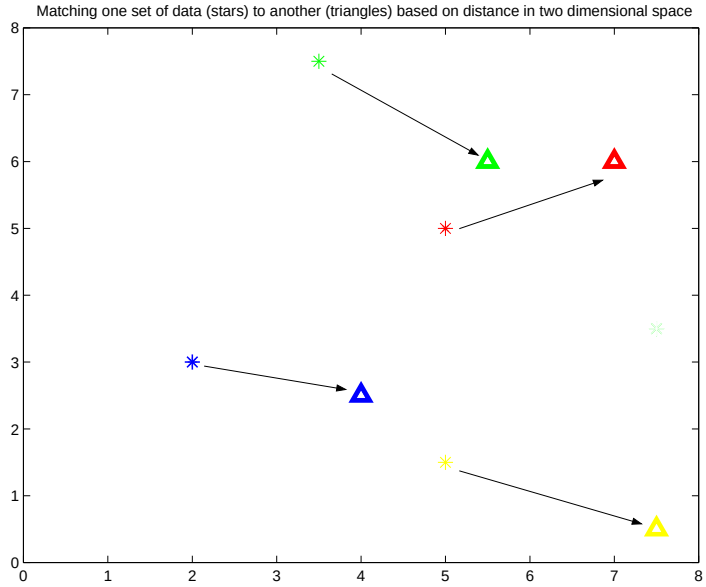


Figure 7.2: Matching segmented objects to recorded objects using a global minimum distance measure in feature space

$$D = \begin{matrix} d_{11} & d_{12} & d_{1...} & d_{1n} \\ d_{21} & d_{22} & d_{2...} & d_{2n} \\ d_{...} & d_{...} & d_{...} & d_{...} \\ d_{n1} & d_{n2} & d_{n...} & d_{nn} \end{matrix} \tag{7.3}$$

The sum of every possible unique combination of i and j in the distance matrix is taken, such that no person i is mapped to the same box j as any other person. This is the equivalent of summing the matrix along all diagonals and combinations of diagonal elements. The minimum such sum is deemed the most likely match for each person and description, as it is the best global match given *all* the information in the scene.

Evaluating the confidence of a match

Two possible metrics are proposed to evaluate the confidence of the match. The first involves calculating a percentage confidence in the match, based on all other possible matches. From the distance matrix of possible matches, two "normalised" matrices are created which indicate the relative confidence of each match when compared only to the other possible matches. Each row represents the distance matrix the distance of every bounding box from a particular person. In the same way in every column is the distance of every person from a particular bounding box. Therefore one normalised distance matrix, D_{nr}, can be created by dividing every element along a row by the sum of all elements in that row as shown in equation 7.4,

and another, D_{nc}, by dividing every element in each column by the sum of elements along that column as in equation 7.5.

$$D_{nr}(i,j) = \frac{D(i,j)}{\sum_{i=1}^{n} D(i,j)} \tag{7.4}$$

$$D_{nc}(i,j) = \frac{D(i,j)}{\sum_{j=1}^{n} D(i,j)} \tag{7.5}$$

When multiplied by 100 these two matrices now give percentage scores for each possible match with regard to every other possible match, where a good match is the lowest percentage in a row or column respectively. Once a match has been chosen, according to the constraints of one and only one bounding box per person and at least one person for every bounding box, the confidence value for that match is given by $100 - D(p,b)$ where p and b are the indices of the match of person to bounding box. Using the two normalised matrices, two confidence values are obtained, which can be used to verify that the best bounding box match for each person is also the best person match to that bounding box. If they disagree, the match with the highest combined confidence value is chosen as the correct one.

This method is, however, sensitive to outliers in that a single large distance in a row or column which is more than two elements in length will affect each percentage, although it is clearly not to be considered in the matching process. For this reason, another confidence metric is proposed in which each match is compared only to the next best match in that row or column. This confidence value is the signed difference between the person-to-bounding box distance of the selected match, and the distance of the next best possible match. If the best local match is also the best global match, in other words, if, given the global constraints of one bounding box per person and at least one person per bounding box, the match with the smallest overall sum of distances is selected, the confidence value: difference between this match and the next best, will be negative. If the chosen match is not the smallest distance in the matrix, because of the global constraints, the difference between the selected match value and the next best (which is actually the global "best") will be positive. Clearly it is desirable to have as large a negative gap as possible, to ensure that the best global match is as far separated as possible from other possible matches, and as small a positive gap as possible so that a match selected to satisfy constraints is as close as possible to the best global match. Thus, the smaller and more negative the sum of the confidence values for this match, the higher our confidence that this is the correct match.

Introducing the ambiguous case

Ambiguities in the matching procedure occur when the match is one to many or many to one. Matching when every detected foreground object in the scene can be accounted for by a unique recorded person is relatively easy, as it merely involves looking for a global optimum given the distances between feature vectors and the constraints of the composition of the scene. The procedure for matching many people to one object has been discussed previously and is effective if people's movements can be anticipated with reasonable accuracy.

Similarly, matching many people to fewer objects and many objects to fewer people can easily be done using a reasoning process given the history of movement and positions of the people, and the composition of the scene. The greatest matching difficulty occurs when a match needs to be performed after an occlusion when not all of the occluded objects have yet become separate. This is explained in more detail in the following subsection.

Matching in ambiguous cases

The drawback of this simple method of resolving occlusions is that it might be necessary at some stage, given more than three people, to perform a match without having enough information about the objects present in the scene. This occurs when n people need to be matched to $n-x$ objects in the scene in the case of x occlusions where $n-x > 1$. For example, if four people are occluding one another and share a bounding box location, if one person leaves the group there is not enough reliable information to determine whether the two connected objects now present are two sets of two people or one set of three and one individual. The information available is unreliable, because for the people that are still in a group, the mean colour is affected, the position is ambiguous and the size is subject to change. The greater the number of people known to be present in the scene, the more possible solutions there are to this problem.

One solution which is only computationally feasible with fewer than 5 people present is to test each possible hypothesis, since the number of occlusions x is known. For instance, in the example mentioned previously, if we name the people persons 1 to 4 and the connected objects objects A and B, once it has been established that there are 4 people ($n = 4$) but only 2 connected objects to match them to, thus 2 occlusions ($x = 2$), the combinations which are possible are shown in 7.1. With 5 people ($n = 5$) and 2 objects, thus 3 occlusions ($x = 3$), the possible combinations are shown in table 7.2. If there are 5 people and 3 objects, thus two occlusions ($x = 2$), there are even more possible combinations to consider, and this number continues to increase as the number of people increases.

Thus, although there are a limited number of solutions possible for every case, it is clear

	Object A	Object B
Possibility 1	3 people	1 person
Possibility 2	1 persons	3 people
Possibility 3	2 people	2 people

Table 7.1: Possible combinations of people given n=4 and x=2

	Object A	Object B
Possibility 1	1	4
Possibility 2	4	1
Possibility 3	2	3
Possibility 4	3	2

Table 7.2: Possible combinations given n=5 and x=3

that the more people there are, the more hypotheses there are to consider for each number of occlusions. In fact, for n people and x occlusions there are $p(n, x)$ possible combinations where $p(n, x)$ is given by equation 7.6 below.

$$p(n, x) = \frac{(n-1)!}{x!(n-x-1)!} \tag{7.6}$$

The number of possibilities to consider for each case can more easily be seen by examining Pascal's triangle in figure 7.3, where $p(n, x)$ is the number in the $(n-x)$th place in the nth row.

					1							
				1		1						
			1		2		1					
		1		3		3		1				
	1		4		6		4		1			
1		5		10		10		5		1		
1		6		15		20		15		6		1

Figure 7.3: Pascal's triangle

An interim solution to the problem in order to find matches for every person on record, while maintaining use of this simplistic colour measure, is to k-means cluster the colours of each object into x clusters and use the mean of the objects that remain (as is usually done); then k-means cluster each object into $x-1$ and 2 clusters then $x-2$ and 3 and so on and find the best match between the cluster centres and the recorded colours of the people. In this way, several possible matches are obtained. Each of the matches between the possible hypotheses has a confidence value associated with it and the match with the highest confidence value is chosen according to a voting scheme, which is discussed in the following subsection.

This solution, proposed in this thesis, is very crude as it assumes a unimodal colour distribution for each person and also that one person is never completely occluded to the extent that none of his/her colours are visible. It is, however, possible to use it with some success with scenes containing few people, where the number of possible combinations of people per object is still low. Some results using this method, which reveal some conditions under which it succeeds and fails, are shown in chapter 9.

Voting scheme

Once the possible matches for all combinations of people have been made, and the confidence values for all of these matches have been obtained, the voting system is invoked to determine which of the possible matches is the correct one. It should be pointed out that none of the matches which have been obtained need be the correct one as the correct hypothesis might still have found a unique solution which is incorrect.

First, the matches are ranked from worst to best according to the average percentage similarity criterion and then according to the total distance from nearest-neighbour criterion. Each match obtains votes based on its positions in the two rankings. The votes are weighted 0.6 for the percentage similarity ranking and 0.4 for the distance from nearest neighbour ranking.

Next an ordering is created based on the size of the segmented blobs in the image such that the largest blob must contain the most people, the second largest the second most and so on down to the smallest. Any match whose numbers agree with this size ordering is given an extra vote. Likewise any matches which occur more than once are given an extra vote weighted 1 and a vote weighted 0.3 is given to any best person-to-bounding box match which agrees with its best bounding-box-to person match. The match which has the highest number of votes then "wins" and is chosen as the most likely match.

7.2.4 Preliminary results and conclusions

Although the algorithm is extremely simple, it is effective in tracking multiple people under these controlled conditions. With a maximum of two people present in the frame the mismatch rate after occlusion is close to zero.

Initially, out of 30 occlusions tested where all the people are visible directly after the occlusion, all 30 were re-identified correctly. Out of 4 occlusions where the clustering algorithm was necessary, 3 were identified correctly and one incorrectly.

The algorithm with simple segmentation runs at 3 frames a second on a Pentium 3 500MHz machine and is effective in tracking up to four people under the conditions described. There

is no reason why it could not be extended for more complex environments with a more refined segmentation algorithm. An example sequence with coloured bounding boxes showing occlusion and a correct match is shown in figure 7.4.

Furthermore, the algorithm does not track people *while* occlusion is actually taking place [37], but rather matches them after the occlusion has occurred. For the time that the objects are hiding one another they are not distinguished from each other. However, the low mismatch rate is very encouraging given the simplicity of the algorithm and relative lack of detailed information.

Figure 7.4: Tracking sequence showing different coloured bounding boxes for different people (read from top to bottom)

Chapter 8

Tracking: extensions and improvements

8.1 Combining segmentation and tracking

An obvious extension to the algorithm presented is to allow it to include both the segmentation and tracking of multiple people in a sequence. As a product of the segmentation itself, a large amount of information is made available about each person, namely:

1. Position - known from the centroid calculation.
2. Size - the bounding box dimensions give the size of the person, which is related to his distance from the camera.
3. Colour - the colour distribution in a*b* space is recorded in the Gaussian mixture, both the updated version and the original version which was initialised as the person entered the scene.
4. Motion - a motion vector description of the largest movements present in the frame is available.
5. History - some of these variables can be recorded as they change over time.

One problem with this extension is that it must be known beforehand how many people we expect in a given image, which can be determined through a thresholded change detection mechanism. Large regions of change which persist through a couple of images in a sequence are assumed to be explained by a new person entering the scene. This unfortunately introduces

the need for an initial threshold which is determined empirically. Initialisation is thus critical to the success of the algorithm.

Inaccuracy in the statistics of the segmented objects are compounded by segmentation error, which also complicates the tracking procedure. Several necessary modifications to the existing tracking algorithm are described in the following section. These improve the tracking accuracy, although they are not expected to do so if the segmentation were to fail.

8.2 Modifications to the existing tracking algorithm

8.2.1 Modifications to colour representation and distance measurement

Colour representation

Since the segmentation method presented in chapter 8 involves keeping a record of the colour distribution of a person's clothing in the form of a Gaussian mixture model in order to perform the segmentation, this knowledge of the colour distribution can be used to aid the tracking algorithm. In contrast to the crude colour descriptor used previously of the colour distribution: the mean, the Gaussian mixture model gives a representation of the number of clusters which best represents the colour distribution and the approximate location and spread of this distribution. This enables a more accurate distance measurement to be performed between a segmented object and a recorded person. Instead of taking the Euclidean distance between the means of the two samples, a Mahalanobis distance between the two samples can be taken, the advantages of which are described in the following section.

Distance measurement: Mahalanobis distance

The Mahalanobis distance metric measures the similarity of a set of values from an unknown sample to a set from a known sample. It gives a statistical measure of how well the unknown sample matches a known sample, measured in terms of the standard deviation from the mean of the training samples. The Euclidean metric, on the other hand, measures the distance from the mean of the group, but does not take into account the distribution of the group.

The Mahalanobis distance Δ from a vector x to a mean vector μ is given by:

$$\Delta^2 = (x - \mu)^T \Sigma^{-1} (x - \mu) \tag{8.1}$$

where Σ is the covariance matrix of the known sample to which the unknown is being compared.

In this case, a known sample is obtained for each person by sampling the Gaussian mixture which represents that person's colour distribution. A sample of 50 points is obtained from the Gaussian mixture model and the Mahalanobis distance is taken from each pixel in the segmented object to this sample. The median Mahalanobis distance is chosen and the minimum of the medians of each sample to each Gaussian mixture model is considered the best match.

Since the Mahalanobis distance from x to μ is a vector distance, normalised by the covariance matrix Σ of the sample [68], some disadvantages of the Euclidean distance measure are overcome by using this metric. The normalisation accounts for poor scaling of the co-ordinate axes of the space from which the feature vectors are drawn, and corrects for correlation between these features. Curved as well as linear decision boundaries can also be produced using this metric [15]. In the special case where the covariance matrix for the known sample is the identity matrix I, the Mahalanobis distance is equivalent to the Euclidean distance.

8.2.2 Possible modifications to matching procedure

In addition to modifying the distance measure from a Euclidean distance between the means of two distributions to the median Mahalanobis distance between two distributions, the method for matching multiple identities to one object can also be modified. The first manner in which this could be done involves attempting to identify pixels that belong to every occluded object even during an occlusion, in a similar manner to [71]. This enables spatial information about each person to be maintained and simplifies the matching procedure. The second possible modification to the matching procedure requires making a "soft", provisional decision about identities until all people have become unoccluded, whereupon the global matching procedure can be used with modified weights, favouring colour as the most reliable feature.

Matching multiple identities to one object

Once occlusion has occurred, identities can still be conferred on the occluded objects by computing the Mahalanobis distance between each pixel and each Gaussian component of every person. Each pixel is then assigned to the class for whom its Mahalanobis distance is a minimum. A connected component analysis on the object should then reveal which collections of component pixels belong to each person. This would enable a continuous record of the approximate position of each person within the occlusion blob to be maintained.

Conserving possibilities

A further modification which can be applied either to the tracking algorithm presented in chapter 7 or on top of the modifications proposed in this chapter, is to reserve judgement on the identities of all object blobs until all people have reappeared from occlusions. This might entail making a provisional decision using the methods already discussed and using these to weight a second, confirming decision once all the objects had re-appeared. This might not of course always be possible, for instance when one occlusion is immediately replaced by another, or when individuals leave the room before all occlusions have been resolved.

8.2.3 Coping with segmentation errors

Initialisation

The number of people in the frame and the rectangular bounding boxes which approximate their space of occupation are estimated through the initial background subtraction method using an adaptive threshold. Naturally, errors in the initial segmentation will lead to an incorrect estimate for the number of people and bounding boxes which may be too small to fully contain each person. For this reason, if the number of connected objects obtained by the initial segmentation technique is more than the expected number, a binary closing is performed upon the entire segmented image in order to try to merge bounding boxes which are adjacent to one another. If two people are mistakenly merged, the consequences are not serious: they will be treated as if they are occluding one another; but if one person is split into two, more people will be expected than are actually present and the bounding box estimation for segmentation will be incorrect.

Because the combined segmentation is only performed within the bounding box located by the initial segmentation, it is desirable that the initial segmentation err on the side of classifying background as person rather than omitting portions of the person. A possibly more effective way of merging the regions produced by the initial segmentations would rely not only on proximity, but also on similarity in terms of expected position, colour and size of the known objects. A probably more reliable method of estimating the number of people one expects in a scene initially would involve the use of a face detection module.

The circular problem of segmentation and tracking

The Gaussian mixture model cannot be used within an estimated bounding box until it is known to which person that bounding box corresponds. This does not pose a problem if each person can be uniquely identified by position and size, but if an occlusion has occurred, the

Gaussian mixture is used to identify which object corresponds to which person, which leads to a circular problem: a good segmentation requires the use of the correct colour information and to obtain the correct colour information requires a good segmentation. The existing algorithm employs an estimate segmentation using no colour which it is hoped will provide a sufficiently good estimate of the true colour distribution of the object. A decision is then made as to which person this object corresponds and the appropriate Gaussian mixture model is then used to improve the existing segmentation.

The Gaussian mixture model could still be used with the modified approach for matching described in the previous section, assigning each pixel to the class for which the posterior probability was a maximum and using that maximum value for the pixel in that location to perform the segmentation.

Chapter 9

Results

9.1 Segmentation evaluation

Evaluation of the segmentation algorithm is intrinsically difficult without a benchmark segmentation for comparison. Hand-segmentation is the obvious choice for comparison, but it is not feasible since it requires a large amount of time to process such large amounts of data by hand. A comparison of the segmentation technique with a hand-segmented sequence, which was already available, has already been made and the results shown in chapter 6. To evaluate the segmentation quality on image sequences which had not been previously hand-segmented a more indirect method of evaluation was used.

Since a centroid calculation is a by-product of the segmentation (it needs to be calculated to sample pixels from the centre of the object to update the Gaussian mixture model), it is informative to plot the position of the centroid as it varies over time. This illustration can be seen in figure 9.1. The centroid position in the x (horizontal) and y (vertical) dimensions of the image plane should vary smoothly from frame to frame as a person moves according to laws of physics. Thus a deviation from smoothness in the path traced out by the centroid over time should indicate an error in the segmentation.

9.1.1 Identification of inaccurate segmentation

If the position of the centroid of the human figure is assumed to move smoothly, the path it traces out over a short period of time can be assumed to be roughly linear. Piecewise linearity of the curve is assumed instead of attempting a polynomial curve fitting as it is not possible to know the form of the curve traced out by the centroid before the activity of the segmented person is known.

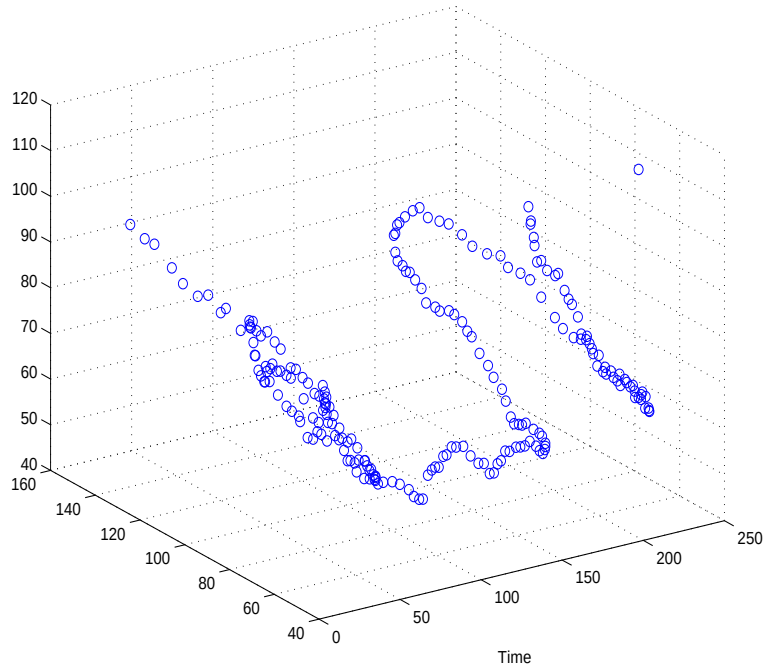


Figure 9.1: Plot of centroid position over time

A linear least-squares regression technique is used to fit a straight line through $n = 3$ consecutive centroid positions in each of the x and y dimensions. This gives a line equation of the form

$$y = ax + b \tag{9.1}$$

where the slope of the best-fit line is

$$a = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_i - \overline{x})^2} \tag{9.2}$$

and the intercept is

$$b = \overline{y} - a\overline{x} \tag{9.3}$$

The distance δ each side of the best-fit line at each x^* within which 95% of the data can be expected to fall is given by[1]

$$\delta = 12.71 \times \sigma \sqrt{1 + \frac{1}{n} + \frac{(x^* - \overline{x})^2}{\sum(x_i - \overline{x})^2}} \tag{9.4}$$

[1]The number 12.71 in equation 9.4 is Student's t-distribution for a 95% confidence interval and 1 degree of freedom. As the number of degrees of freedom increases, this number tends to 2.

where $\sigma = \sqrt{\frac{\sum_{i=1}^{n} e_i}{n-2}}$ is the mean square error of the line. The errors e_i are the differences between the observed and predicted values for the data and $n-2$ is the number of degrees of freedom.

For every four consecutive points on the centroid curve, the first three are used to generate a line equation, which is used to predict the fourth point y_{i+4}. The uncertainty δ_{i+4} for the prediction of the fourth point is also calculated and if the measured value for that point does not fall within $y_{i+4} \pm \delta_{i+4}$ an error in the segmentation at that frame is assumed.

Curves traced out by the x position of the centroid for various sequences are shown in figure 9.2. Centroid positions that do not correspond to the linear predictions are marked in red. Table 9.1 shows the percentage of centroid points over 10 different image sequences that do not obey a linear local variation in the x or y directions and can therefore be classified as unreliable segmentations.

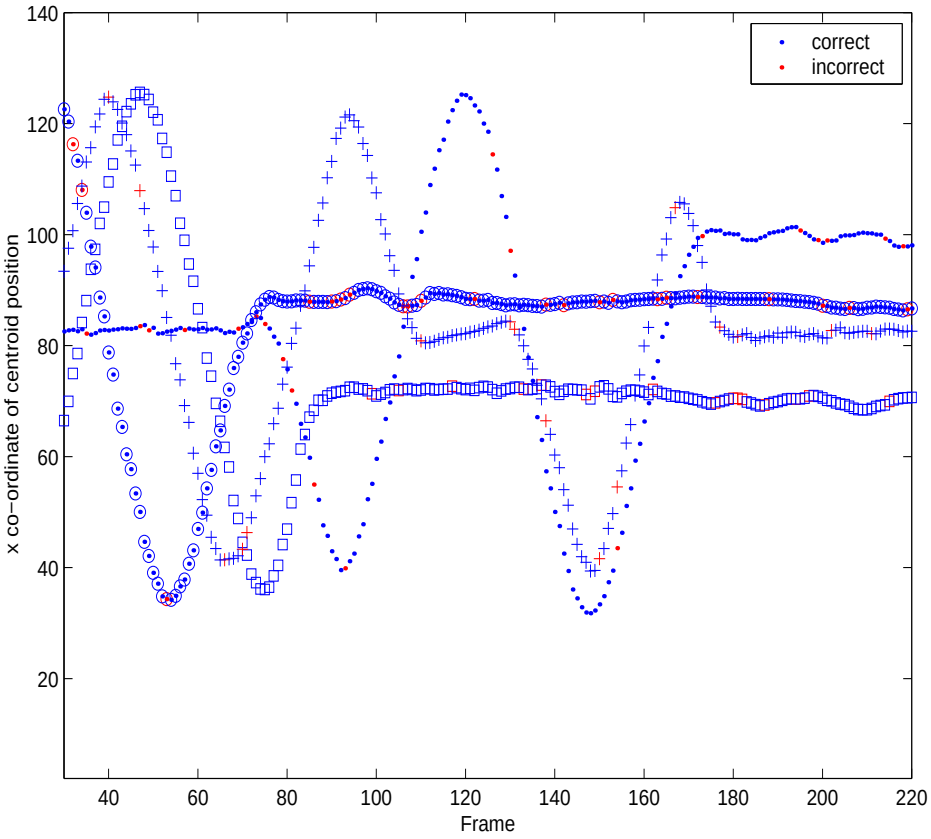


Figure 9.2: x co-ordinates of a single segmented object's centroid position over time for four different sequences. Points that do not correspond to a predicted position are marked in red.

9.1.2 Shortcomings of the method

The method of segmentation evaluation, although it yields good results, is a very indirect method of measuring segmentation accuracy and as such is not necessarily the most reliable measure. Although errors in segmentation which leave out portions of an object, or which include portions of another object, or segment another object entirely, are picked up as an irregular movement of the centroid position; in reality, errors in this adaptive segmentation algorithm often occur gradually and likewise recover gradually, and thus might not be picked up as an outlier point on a smooth curve.

In addition, the assumption that a local linear approximation to the movement of the centroid will accurately reflect the motion of a person is not always valid. Curves traced out by the centroid in the x and y directions for a person performing various activities are shown in figure 9.3. (These are curves for the same sequences as shown in figure 9.2 and the line types marking the corresponding traces in that image are shown in the legend for ease of comparison.) The centroid's movement along the x-axis varies smoothly and outliers in the curves are easier to spot (cf. figure 9.2: which traces correspond to each sequence in figure 9.2 is shown in the legend). The trace of centroid movement along the y-axis for these sequences appears less smoothly-varying, primarily because of more rapid motions along the y-axis, such as the person putting one foot in front of the other, or waving his arms above his head. This, combined with the elongation of the segmented object in the y-direction, makes the number of centroid points that do not correspond to their predictions slightly higher than those along the x-axis.

Turning-points are also mistaken for errors in this linear assumption.

	predictions correct	**predictions incorrect**	**% correct**	**% incorrect**
x co-ord	1358	153	89.87%	10.13%
y co-ord	1304	207	86.30%	13.70%
total	2662	360	88.09%	11.91%

Table 9.1: Percentage of correct and incorrect predictions for the x and y positions of the centroid

9.2 Evaluation of tracking algorithm

9.2.1 Sources of error

Errors in this particular tracking algorithm can be introduced in a variety of ways. Some are implied by the assumptions defined within the structure of the algorithm. When the assumption that every person will at some point appear as a single segmented object is violated, errors

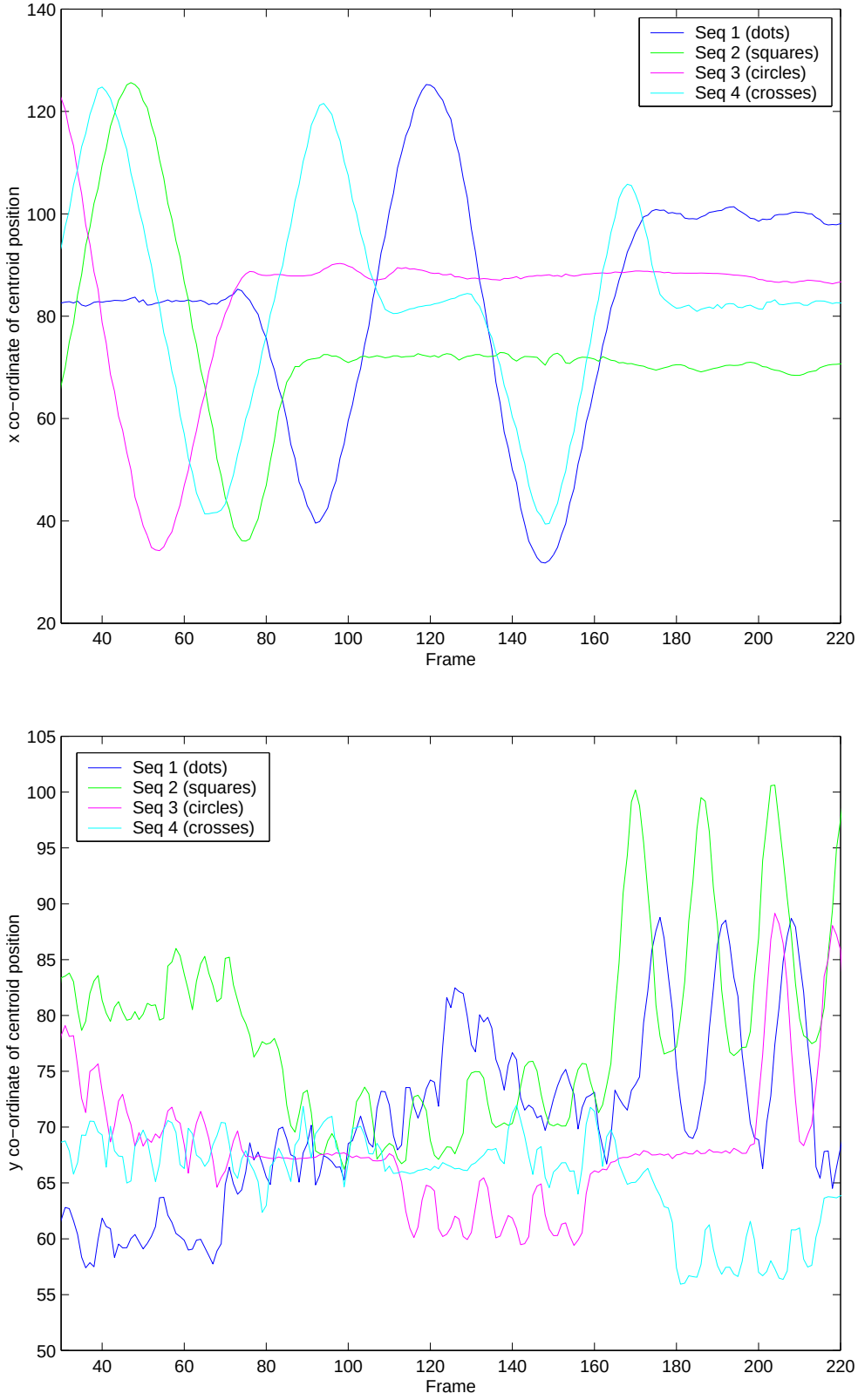


Figure 9.3: Traces of x and y centroid positions of a segmented person for four different sequences

can occur in the occlusion reasoning. Also it will go unnoticed if one person is substituted for another in the entrance/exit zone without first appearing as a single object. Other errors are not introduced by the violation of an assumption. Instances of both kinds of error, and their significance are discussed below.

Segmentation error

Errors in segmentation lead to two possible situations: regions can be mistakenly identified as people, and people who are present in the image can be mistakenly omitted from the segmentation. Since these segmentation errors are transient and occur usually in one or two consecutive frames only, the occlusion reasoning and person creation/deletion framework does not readily extend to extraneous "objects" or "disappearances" which are difficult to explain. In an uncontrolled background, parts of background which are mistakenly segmented as part of a person will affect the apparent colour distribution and size of that person.

For these reasons, in testing the tracking algorithm, segmentation is completely eliminated as a source of error through use of the blue background and, where necessary, manually when segmentation errors persist despite the controlled environment.

Matching after occlusion

Errors naturally occur in the matching procedure, therefore the matching procedure is the principal source of error for the system and testing is focused on this aspect. For simple matches where every person known to be present can be explained by a segmented object in the scene, errors occur if the features are similar to one another, for example, if two people are wearing the same coloured clothing.

For matches where several identities must be matched to one segmented object, features are naturally ambiguous so a greater chance of error is present. Also, the possibility of assigning many identities to an object which corresponds to one person is introduced, which leads to further complications.

Errors in exit zone

Errors occur in the exit zone where people are allowed to enter or leave. If one person leaves and another enters simultaneously from the same entry/exit point they will be classified as the same person. This is implicit in the reasoning framework and will not be counted as an error.

A person leaving the scene may also be classified as an occlusion occurrence if the person in

question is close to another person at the time of departure. This is an error from which it is difficult for the system to recover.

Occlusion undetected

If two people enter together they are tracked as one object until they part. This is also a feature of the algorithm and is not considered an error because once they do part, one of the people will be identified as new to the scene.

If an occlusion is not predicted for a person, that person's disappearance might be explained as a departure from the scene. This is unlikely given a high enough frame rate, although the system has once been tricked by somebody running very fast towards another person.

9.2.2 Results and discussion

The tracking algorithm was tested on sequences containing up to four people at a time. The colour of the clothing worn by the different people was not controlled at all: it is assumed that in a random world, and with a small number of people present in a room at a time, the colour of clothing of people is likely to be sufficiently varied for the majority of people to be distinguished from every other on that basis. Equally, if chance decrees that two people have similarly coloured clothing, it should be possible to distinguish between theses people based on some other visible feature.

Apart from two occasions when the disappearance of a person who has left the scene is mistakenly explained as an occlusion with another person, the only *unexpected* errors occur after occlusions. These errors are measured by manually counting the number of people who are correctly identified after an occlusion. The number of correct matches is measured out of the sum of total number of correct and incorrect matches which are made after every occlusion in a sequence.

The matches are divided into two types: simple matches, where there are as many segmented objects in the scene as people present, and multi-object matches, when there are fewer segmented objects in the scene than people and at least one many-to-one match must be made.

In table 9.2 the percentage of correct matches (out of all possible match situations) is shown for 6 sequences containing 2 to 4 people. The maximum number of people present for a sequence does not appear to affect the number of correct matches, although in general the number of correct matches to multi-object blobs appears to be far lower than the number of simple matches correct. Sequence 5 is the exception to this, with a very low correct match rate for only simple matches.

	Seq 1	Seq 2	Seq 3	Seq 4	Seq 5	Seq 6
Sequence length (no. frames)	785	457	1079	1027	778	771
Maximum number of people	4	4	3	3	2	2
Total no. simple matches	13	-	39	31	20	14
No. simple matches correct	13	-	37	31	12	12
% simple correct	**100**	**-**	**94.87**	**100.00**	**60**	**85.71**
No. multi-object blob matches	15	32	9	6	-	-
No. multi-object matches correct	14	17	4	6	-	-
% multi-object correct	**93.33**	**53.12**	**44.44**	**100.00**	**-**	**-**
Total % correct matches	96.43	53.12	85.42	100.00	60.00	85.71

Table 9.2: Tracking accuracy according to proportion of multi-object blob matches

In order to see the reason for this, table 9.3 shows the same sequences and the two dominant colours of the people in the sequences[2]. The colours for all sequences are reasonably distinct with the exception of sequence 5, which perhaps helps to explain the low correct match rate out of the simple matches. However, distinctly coloured clothing is not sufficient to ensure that multi-object matches will be more accurate. The colours in sequence 2 are the same as in sequence 1, but the number of multi-object matches is higher (table 9.2); thus the correct match rate is a very low 53.12%. These tables suggest that both colour similarity between people and matching people to multi-object blobs are significant sources of error for the matching process.

	no. people	Dominant colours of people's clothing	% correct
Seq 1	4	navy/beige | green/navy | mauve/beige | grey/navy	96.43
Seq 2	4	navy/beige | green/navy | mauve/beige | grey/navy	53.12
Seq 3	3	brown/beige | navy/light blue | light blue/navy	85.42
Seq 4	3	pink/black | beige/dark grey | navy/light blue	100
Seq 5	2	white/medium blue | white/medium blue	60
Seq 6	2	light blue/medium blue | navy only	85.71

Table 9.3: Breakdown of tracking accuracy according to dominant colours

Table 9.5 confirms this inaccuracy in multi-object matches, with relatively low correct match rates compared to the corresponding simple matches. Table 9.4 shows the total number of matches of each kind used to calculate the percentages in table 9.5.

The algorithm was also tested with a person removing a coloured jersey while unoccluded and then passing another person. Because of the slow update method used, both people were correctly re-identified after the occlusion in spite of the fact that one of them had undergone an extreme colour change.

[2]Clearly the naming of the colours is subjective and subject to linguistic restrictions, but it suffices to show the similarity between clothing colours of the individuals in the sequences.

no. people in seq	multi-object matches	simple matches	total matches
4	47	13	60
3	139	267	406

Table 9.4: Number of each type of match in the set of data used for testing

no. people in seq	% multi-object correct	% simple correct	total % correct
4	65.96	100	73.33
3	69.78	88.39	82.01

Table 9.5: Percentage correct of each type of match

9.3 Modified tracking algorithm

With the addition of the Mahalanobis distance measure instead of a Euclidean measure, some improvement in classification has been noted. For example, when the modified algorithm proposed in chapter 8 is run on sequence 5, 16/20 correct matches are obtained instead of 12/20, which is a distinct improvement upon the previous algorithm.

Chapter 10

Conclusions

Algorithms for tracking and segmenting have been presented in this thesis. Both have shown some success in implementation which thus far has been done mostly separately.

Various segmentation algorithms using different techniques have been explored, and their limitations made clear. A segmentation algorithm using a combination of image information has been presented which uses simplifying assumptions, such as a static camera and the availability of an empty background reference frame. Factors contributing to the segmentation problem have been examined: namely colour space selection, colour constancy and motion estimation.

An adaptive Gaussian mixture model, which can be automatically initialised, is used to model the colour distributions of the background and foreground in order to obtain an *a posteriori* estimate of the likelihood of a pixel's belonging to the foreground or background colour class. The model is able to adapt to changing image characteristics and to stabilise itself when an incorrect adaption has occurred or in the event of a bad initialisation. In addition, the slow update procedure makes it possible for people to remove items of clothing (which change their apparent colour entirely) and still be correctly classified after an occlusion.

Motion segmentation is incorporated in the form of background differencing for which a Kalman filtering technique is used to perform an adaptive background estimate to compensate for lighting changes and shadowing. The combination of the differencing technique with the colour model acquisition enables the model to learn the colours of an object automatically without user intervention.

Gradient information is used by detecting sharp changes in the difference image in order to eliminate shadows, which are assumed to have more smoothly varying boundary characteristics.

The use of a combination of image information: namely colour, motion and edge information helps to overcome some of the difficulties often encountered with each type of segmentation when taken on its own, thus contributing to a more robust algorithm at the expense of computation time. The algorithm is implemented in Matlab and primarily for this reason, is slow and does not run in real-time. Successful segmentation throughout a sequence is extremely sensitive to initialisation as it is dependent on a good segmentation for the first frame which is not entirely reliable. The various modules which form part of the segmentation process, however, do help to stabilise each other, and the automatically self-correcting mixture model enables the program to recover slowly from a bad segmentation.

The use of a predictive mechanism in order to estimate the position of objects to be segmented also improves accuracy and reduces computation time. The motion model implemented is very simple and could be extended to use Kalman filtering of the trajectories in order to cater for the presence of noise on the measurement of the position or velocity at every stage.

Tracking has been presented as a separate problem, assuming the segmentation to be correct. An inaccurate segmentation naturally compounds the problem of tracking, as inaccurate colour and position information is obtained. Two cases for tracking in the presence of occlusion have been discussed: an n person to n object match and an n person to $n-x$ object match, the latter presenting a great deal of difficulty for an algorithm that can only resolve objects' identities *after* an occlusion. Several possible solutions to this particular problem have been proposed, although the temporary solution of looking for possible colour clusters in each segmented object has proved less than effective in the sequences used for testing. Factors which aggravate the tracking problem are a large number of people to be tracked simultaneously, and similarity in clothing colours between people which does not enable them to be easily distinguishable from one another.

A modified algorithm combining the segmentation and tracking and using a Mahalanobis distance instead of a Euclidean distance in colour space has proved significantly more accurate in preliminary testing. The segmentation algorithm, although not always successful in producing visually pleasing silhouette images, appears to be sufficiently accurate for the tracking method presented, as it allows sufficiently reliable estimates of size, position, velocity and colour to be extracted from the segmented blobs. The primary difficulty is estimating how many people are present in the scene in the event that an initial segmentation produces more connected blobs than people.

Limitations are slow execution, particularly in Matlab, and extreme sensitivity to initialisation, although the use of combined image information is recommended to simplify the problem of segmentation.

This thesis has explored some of the techniques which have appeared in recent literature on

segmentation and attempted to improve the results by combining them. We have been successful in some instances, but much further work would be needed to assess the generality of the approach. The general solution to what appears to be a rather complex problem might more closely approached using more recent and promising methods such as the CONDENSATION [32] algorithm.

Appendix A

The EM algorithm for Gaussian mixtures

The parameters for a Gaussian mixture model, which must be adapted to fit the data are: $P(j)$, μ_j and σ_j. The negative log-likelihood (error function) is

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

In order to maximise the likelihood L one must minimise the error E. Expectation maximisation is an iterative technique for finding the parameters of the mixture model which minimise this error function. The maximum likelihood solution for this minimisation can be obtained by differentiating A.1 with respect to μ_j, σ_j and $P(j)$ ([6] discusses this in more detail). Setting these derivatives to zero to minimise E, the parameters can be given by the following equations:

$$\widehat{\mu_j} = \frac{\sum_n P(j|x^n)x^n}{\sum_n P(j|x^n)} \tag{A.2}$$

$$\widehat{\sigma_j^2} = \frac{1}{d}\frac{\sum_n P(j|x^n)\,\|x^n - \widehat{\mu_j}\|^2}{\sum_n P(j|x^n)} \tag{A.3}$$

$$\widehat{P}(j) = \frac{1}{N}\sum_{n=1}^{N} P(j|x^n) \tag{A.4}$$

Expectation maximisation consists of initialising the parameters of the model at some arbitrary

value and iteratively calculating new parameters according to equations A.2, A.4 and A.3, replacing the initial parameters with the new estimate at every iteration, in such a way that the error is decreased each time.

The change in error each time the old parameters are replaced by new ones is

$$\Lambda E = -\sum_n \ln\left\{\frac{p^{new}(x^n)}{p^{old}(x^n)}\right\} \tag{A.5}$$

where p^{new} and p^{old} are evaluations of the probability densities using the new and old parameters respectively.

Expanding using the definitions of $p(x)$ in equation 4.2 on page 41, and using Jensen's inequality [6] and the fact that $\sum_{n=1}^{M} P(j) = 1$, an upper limit can be placed on the value of E^{new}, the error value obtained using the new parameters, as shown in [6]. This upper limit is:

$$E^{old} + \sum_n \sum_j P^{old}(j|x^n) \ln\{P^{new}(j)p^{new}(x^n|j)\} \tag{A.6}$$

Thus to minimise the error value the second term in equation A.6 must be minimised. Using the definition of $p(x|j)$ on page 41 and minimising this second term in equation A.6 with respect to the new parameters, update equations can be found for $P(j), \mu_j$ and σ_j, which are similar to those obtained using the maximum likelihood approach.

$$\mu_j^{new} = \frac{\sum_n P^{old}(j|x^n)x^n}{\sum_n P^{old}(j|x^n)} \tag{A.7}$$

$$(\sigma_j^{new})^2 = \frac{1}{d}\frac{\sum_n P^{old}(j|x^n)\left\|x^n - \mu_j^{new}\right\|^2}{\sum_n P^{old}(j|x^n)} \tag{A.8}$$

$$P(j)^{new} = \frac{1}{N}\sum_n P^{old}(j|x^n) \tag{A.9}$$

Appendix B

Hardware

Image sequences are obtained using a *Watec* WAT-202B camera. These images are digitised using a *Matrox Meteor* 2 framegrabber and written as a stream of TIFF images. The image resolution is 144×192 pixels and the frame rate for all sequences varies between 10 and 25 frames per second.

Bibliography

[1] T. Aach and A. Kaup. Statistical model-based change detection in moving video. *Signal Processing*, 31:165–180, 1993.

[2] Michael D. Alder. An introduction to pattern recognition: statistical, neural net and syntactical methods of getting robots to see and hear. Online http://odin.ee.uwa.edu.au/~mike/PatRec, September 1997. (last accessed November 2000).

[3] M Bichsel. Illumination invariant segmentation of simply-connected moving objects. *5th British Machine Vision Conference*, pages 459–468, September 1994.

[4] M. Bichsel. Segmenting simply-connected moving objects in a static scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(11):1138–1142, November 1994.

[5] S Birchfield. Elliptical head tracking using intensity gradients and color histograms. *IEEE Conference on Computer Vision and Pattern Recognition*, June 1998.

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

[7] M. Bollman and B. Mertsching. Opponent color processing based on neural models. Online publication, 1999. http://ima-www.informatik.uni-hamburg.de/.

[8] Q. Cai, M. Mitiche, and J.K. Aggarwal. Tracking human motion in an indoor environment. *Second International Conference on Image Processing, ICIP '95 Washington, USA*, 1995.

[9] Claudette Cedras and Mubarak Shah. Motion-based recognition: a survey. *Image and Vision Computing*, 13(2):129–155, March 1995.

[10] L. Cloutier, A. Mitiche, and P. Bouthemy. Segmentation and estimation of image motion by a robust method. *First IEEE Conference on Image Processing, ICIP '94, Austin, Texas*, November 1994.

[11] T. Darrell, G. Gordon, M. Harville, and J. Woodfill. Integrated person tracking using stereo, color and pattern detection. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 601–609, June 1998.

[12] Alberto Del Bimbo. *Visual Information Retrieval.* Morgan Kaufmann Publishers Inc, 1999.

[13] Claire-Helene Demarty. Color segmentation of still images. Online, June 1999. http://cmm.ensmp.fr/demarty/CTI/segmentation_couleur_ang.html.

[14] Marie-Pierre Dubuisson and Anil K. Jain. Contour extraction of moving objects in complex outdoor scenes. *International Journal of Computer Vision*, 14(1):83–105, 1995.

[15] Richard O. Duda. Pattern recognition for HCI. Online, June 1997. http://www.engr.sjsu.edu/~ knapp/HCIRODPR/PR_home.htm (last accessed March 2001).

[16] Claude L. Fennema and William B. Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9:301–315, 1979.

[17] Paul Fieguth and Demetri Terzopoulos. Color-based tracking of heads and other mobile objects at video frame rates. *Proceedings of the IEEE Computer Vision and Pattern Recognition*, pages 21–27, 1997.

[18] Nir Friedman and Stuart Russell. Image segmentation in video sequences: a probabilistic approach. *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, 1997. Also available at http://www.cs.berkeley.edu/~russell/publications.html.

[19] B. Funt and V. Cardei. Bootstrapping color constancy. In *Proceedings of the SPIE, Electronic Imaging IV*, volume 3644, 1999.

[20] B. Funt, V. Cardei, and K. Barnard. Learning color constancy. In *Proceedings IS and T/SID Fourth Color Imaging Conference*, 1996.

[21] B. Funt, V. Cardei, and K. Barnard. Modeling color constancy with neural networks. In *Proceedings International Conference on Vision, Recognition, Action: Neural Models of Mind and Machine*, 1997.

[22] Marc Green. Basic color and design SBFAQ. ErgoGero human factors science, Online, July 1999. http:/www.ergogero.com/FAQ/cfaqhome.html.

[23] Ismail Haritaoglu, David Harwood, and Larry S. Davis. W4S: A real-time system for detecting and tracking people in 2.5 D. *European Conference on Computer Vision*, 1998.

[24] Glenn Healey. Hierarchical segmentation-based approach to motion analysis. *Image and Vision Computing*, 11(9):570–576, 1993.

[25] B.K.P Horn. *Robot Vision*. MIT Press, 1986.

[26] B.K.P. Horn and B.G. Schunk. Determining optical flow. *Artificial Intelligence*, 17(572):185–203, 1981. Also available as MIT Technical Report AIM 572, 1980.

[27] Anya. C. Hurlbert. *The Computation of Color*. PhD thesis, Massachussetts Institute of Technology, MIT Media Lab, 1989. Also available as MIT Technical Report AITR 1154.

[28] Leo Hurvich and Dorothea Jameson. Opponent response functions related to measured cone photopigments. *Journal of the Optical Society of America*, 50:429–430, 1968.

[29] D.P. Huttenlocher, J.J. Noh, and W.J. Rucklidge. Tracking non-rigid objects in complex scenes. *International Conference on Computer Vision, ICCV93*, pages 93–101, 1993.

[30] IBM. IBM image research: Color science research. Online, June 2000. http://www.research.ibm.com/image_apps/colorsci.html.

[31] S.S. Intille, J.W. Davis, and A.F. Bobick. Real-time closed-world tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, November 1996.

[32] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. *Proceeedings of the European Conference on Computer Vision*, pages 343–356, 1996.

[33] Y. Ivanov, A. Bobick, and J. Liu. Fast lighting-independent background subtraction. In *IEEE Workshop on Visual Surveillance, VS'98*, pages 49–55, Bombay, India, January 1998. Also available as MIT Technical Report AIM 437, 1998.

[34] Sumer Jabri, Zoran Duric, and Harry Wechsler. Detection and location of people in video images using adaptive fusion of color and edge information. *15th International Conference on Pattern Recognition*, 4:627–630, September 2000.

[35] R. Jain, D. Militzer, and H.H. Nagel. Separating non-stationary from stationary scene components in a sequence of real world TV-images. *Proceedings IJCAI*, pages 612–618, 1977.

[36] K-P. Karmann, A. Brandt, and R. Gerl. Moving object segmentation based on adaptive reference images. *Signal Processing*, V: Theories and Applications, 1990.

[37] Sohaib Khan and Mubarak Shah. Tracking people in presence of occlusion. *Asian Conference on Computer Vision*, January 2000. Also available at http://www.cs.ucf.edu/~khan (last accessed November 2000).

[38] Josef Kittler, Mohamad Hatef, Robert Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, March 1998.

[39] Dieter Koller, Joseph Weber, and Jitendra Malik. Robust mutiple car tracking with occlusion reasoning. *Proceedings of the Third European Conference on Computer Vision*, pages 189–196, 1994. Stockholm, Sweden.

[40] Mika Korhonen, Janne Heikkila, and Olli Silven. Intensity independent color models and visual tracking. *15th International Conference on Pattern Recognition*, 3:604–608, September 2000.

[41] John Krumm, Steve Harris, Brian Meyers, Barry Brumitt, Michael Hale, and Steve Shafer. Multi-camera multi-person tracking for EasyLiving. *Third IEEE Workshop on Visual Surveillance*, July 2000. Dublin, Ireland.

[42] Y. Kuno, T. Watanabe, Y. Shimosakoda, and S. Nakagawa. Automated detection of human for visual surveillance sytem. *Proceedings of the International Conference on Pattern Recognition*, pages 865–869, 1996.

[43] Johan M. Lammens. *A Computational Model of Color Perception and Color Naming.* PhD thesis, University of New York at Buffalo, http://www.cs.buffalo.edu/pub/colornaming/diss/diss.html, June 1994. No longer available online.

[44] M.K. Leung and Y-H Yang. Human body motion segmentation in a complex scene. *Pattern Recognition*, 20(1):55–64, 1987.

[45] G. Lim, M.D. Alder, C.J.S. de Silva, and Y. Attikiouzel. Video-based detection and classification of real-world moving objects. *First Australia New Zealand Conference on Intelligent information systems, Perth, Australia*, pages 362–266, December 1993.

[46] Y-T. Lin, Y-K. Chen, and S.Y. Kung. Object-based scene segmentation combining motion and image cues. *Proceedings of the IEEE International Conference on Image Processing, Lausanne*, 1:957–960, 1996.

[47] Peter S. Maybeck. *Stochastic models, estimation and control*, volume 1, chapter 1, pages 1–15. Academic Press, 1979.

[48] A. Mitiche and J. K. Aggarwal. Image segmentation by conventional and information-integrating techniques: a synopsis. *Image and Vision Computing*, 3(2):50–62, 1985.

[49] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. Technical Report 326, MIT, Cambridge, Massachussetts, 1995. Appeared in 5th International Conference on Computer Vision, Cambridge, Mass. June 1995.

[50] F. Moscheni, S. Bhattacharjee, and M. Kunt. Spatiotemporal segmentation based on region merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(9):897–914, September 1998.

[51] C. Olivier, F. Jouzel, and A. El Matouat. Choice of the number of component clusters in mixture models by information criteria. *Vision Interface '99*, May 1999. Trois-Rivieres, Canada.

[52] S. Or, K. Wong, K. Lee, and T. Lao. Panoramic video segmentation using color mixture models. *15th International Conference on Pattern Recognition*, 3:391–394, September 2000.

[53] Robyn Owens. Lecture notes on motion. University of Edinburgh, Online, October 1997. http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/OWENS/LECT12.

[54] N. Paragios and G. Tziritas. Detection and location of moving objects using deterministic relaxation algorithms. *International Conference on Pattern Recognition, Austria*, 1:201–205, 1996. also technical report ICS-FORTH TR-155, Dec 1995.

[55] E.J. Pauwels and G. Frederix. Non-parametric clustering for image segmentation and grouping. *Journal of Computer Vision and Image Understanding CVIU*, May 1999.

[56] Janez Pers, Marta Bon, and Stanislav Kovacic. Errors and mistakes in automated player tracking. *Proceedings of the Sixth Computer Vision Winter Workshop*, pages 25–36, February 2001.

[57] Janez Pers and Stanislav Kovacic. Computer vision system for tracking players in sports games. *First International Workshop on Image and Signal Processing and Analysis*, June 2000. Croatia.

[58] Charles Poynton. Frequently asked questions about color. Online FAQ, 1997. http://www.inforamp.net/~ poynton/Poynton-color.html.

[59] Charles Poynton. A guided tour of color space. Online, 1997. http://www.inforamp.net/~poynton/Poynton-color.html.

[60] Y. Raja, S. McKenna, and S. Gong. Colour model selection and adaptation in dynamic scenes. *European Conference on Computer Vision, Freiburg, Germany*, June 1998.

[61] Y. Raja, S. McKenna, and S. Gong. Segmentation and tracking using colour mixture models. *Asian Conference on Computer Vision*, January 1998.

[62] Y. Raja, S. McKenna, and S. Gong. Tracking and segmenting people in varying lighting condition using colour. *Proceeding of FG98, Nara, Japan*, 1998.

[63] C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using Kalman-filtering. *Proceedings of the International Conference on Recent Advances in Mechatronics ICRAM 95*, pages 193–199, 1995.

[64] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[65] H. Roh, S. Kang, and S-W. Lee. Multiple people tracking using an appearance model based on temporal color. *15th International Conference on Pattern Recognition*, 4:643–646, September 2000.

[66] P. Rosin and T. Ellis. Detecting and classifying intruders in image sequences. *2nd British Machine Vision Conference, Glasgow*, pages 293–300, 1991.

[67] M. Rossi and A. Bozzoli. Tracking and counting moving people. *1st International Conference on Image Processing, Austin Texas*, pages 212–216, November 1994.

[68] Robert Schalkoff. *Pattern Recognition: stastistical, structural and neural approaches.* John Wiley and Sons Inc., 1992.

[69] M. Schwarz, W.B. Cowan, and J.C. Beatty. An experimental comparison of RGB, YIQ, LAB, HSV and Opponent colour models. *ACM Transactions on Graphics*, 6(2):123–157, April 1987.

[70] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), August 2000.

[71] A. Shio and J. Sklansky. Segmentation of people in motion. In *MOTION91*, pages 325–332, 1991.

[72] E.P Simoncelli. Design of multidimensional derivative filters. In *First IEEE International Conference on Image Processing, Austin, Texas*, pages 1–5, November 1994.

[73] A. Singh. *Optic Flow Computation: A unified perspective.* IEEE Computer Society Press, 1991.

[74] K. Skifstadt and R. Jain. Illumination independent change detection for real world image sequences. *Computer Vision, Graphics and Image Processing*, 46:387–399, 1989.

[75] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. *Computer Vision and Pattern Recognition*, 1999.

[76] William B. Thompson. Combining motion and contrast for segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):543–549, 1980.

[77] Tina Yu Tian and Mubarak Shah. Motion estimation and segmentation. *Machine Vision and Applications*, 9(1):32–42, 1996.

[78] Michael J. Vrhel and H. Joel Trussel. Color device calibration: a mathematical formulation. *IEEE Transactions on Image Processing*, 8(12):1796–1805, December 1999.

[79] Greg Welch and Gary Bishop. An introduction to the Kalman filter. Online publication, September 1997. http://www.cs.unc.edu/~welch/kalman (last accessed February 2001).

[80] Lars Westberg. Hierarchical contour-based segmentation of dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):946–952, 1992.

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

[82] S. Yalamanchili, W. N. Martin, and J. K. Aggarwal. Extraction of moving object descriptions via differencing. *Computer Graphics and Image Processing*, 18:188–201, 1982.

[83] U. Zang. Color vision. Online, 2000. http://www.photo.net/photo/edscott/vis00010.htm.