# Detection and Tracking of Humans for Visual Interaction

Antonio S. Micilotta

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Centre for Vision, Speech and Signal Processing
School of Electronics and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

September 2005

# Abstract

This thesis contributes, in essence, four developments to the field of computer vision. The first two present independent methods of locating and tracking body parts of the human body, where the main interest is not 3D biometric accuracy, but rather a sufficient discriminatory representation for visual interaction. Making use of a single uncalibrated camera, the first algorithm employs background suppression and a general approximation to body shape, applied within a particle filter framework. In order to maintain real-time performance, integral images are used for rapid computation of particles. The second method presents a probabilistic framework of assembling detected human body parts into a full 2D human configuration. The face, torso, legs and hands are detected in cluttered scenes using body part detectors trained by AdaBoost. Coarse heuristics are applied to eliminate obvious outliers, and body configurations are assembled from the remaining parts using RANSAC. An *a priori* mixture model of upper-body configurations is used to provide a pose likelihood for each configuration, after which a joint-likelihood model is determined by combining the pose, part detector and corresponding skin model likelihoods; the assembly with the highest likelihood is selected.

The third development is applied in conjunction with either of the aforementioned human body part detection and tracking techniques. Once the respective body parts have been located, the *a priori* mixture model of upper-body configurations is used to disambiguate the hands of the subject. Furthermore, the likely elbow positions are statistically estimated, thereby completing the upper body pose.

A method of estimating the 3D pose of the upper human body from a single camera is presented in the final development. A database consisting of a variety of human movements is constructed from human motion capture data. This motion capture data is then used to animate a generic 3D human model which is rendered to produce a database of frontal view images. From this image database, three subsidiary databases consisting of hand positions, silhouettes and edge maps are extracted. The candidate image is then matched against these databases in real time. The index corresponding to the subsidiary database triplet that yields the highest matching score is used to extract the corresponding 3D configuration from the motion capture data. This motion capture frame is then used to extract the 3D positions of the hands for use in HCI, or to render a 3D model.

# Acknowledgements

Particular appreciation goes to Richard Bowden who has offered much wisdom and guidance over the passed three years. Having a young, energetic supervisor has often filled me with inspiration and enthusiasm. Eng Jon Ong has also offered great input; working with him brought on a sense of team work which motivated me to do more. My parents and sister have offered a world of support, and the encouragement and faith to push forward when times were tough. Lastly, great appreciation goes to the Centre for Vision, Speech and Signal Processing for providing a studentship and the opportunity to continue my education in a world class facility.

# List of Publications

1. A.S. Micilotta, E.J. Ong and R.Bowden. Real-time Upper Body Detection and 3D Pose Estimation in Monoscopic Images. Submitted to ECCV for review in September 2005.

2. A.S. Micilotta, E.J. Ong and R.Bowden. Human Body Part Detection and Tracking. Submitted to PAMI for review in August 2005.

3. E.J. Ong, A. Hilton, A.S. Micilotta. Viewpoint Invariant Exemplar-Based 3D Human Tracking. To appear in Proceedings on Modeling People and Human Interaction Workshop, October, 2005.

4. A.S. Micilotta, E.J. Ong and R.Bowden. Human Body Part Detection and Tracking. In Hans-Helmut Nagel, editor, Cognitive Vision Systems, to be published in 2005.

5. A.S. Micilotta, E.J. Ong and R.Bowden. Detection and Tracking of Humans by Probabilistic Body Part Assembly. In proceedings on British Machine Vision Conference, volume 1, pages 429-438, September 2005.

6. A.S. Micilotta, E.J. Ong and R.Bowden. Real-time Upper Body 3D Pose Estimation from a Single Uncalibrated Camera. In proceedings on Eurographics, Short Presentations, pages 41-44, August 2005.

7. A.S. Micilotta and R.Bowden. View-based Location and Tracking of Body Parts for Visual Interaction. In proceedings on British Machine Vision Conference, volume 2, pages 849-858, September 2004.

8. A.S. Micilotta and R.Bowden. View-based Location and Tracking of Body Parts for Visual Interaction. In BMVA Symposium on Spatio-temporal Image Processing, March 2004.

# Nomenclature

| | |
|---|---|
| GMM | Gaussian Mixture Model |
| HCI | Human Computer Interaction |
| PCA | Principal Components Analysis |
| PDF | Probability Density Function |
| PDM | Point Distribution Model |
| CMYK | The Cyan, Magenta, Yellow and Black colour model used in standard colour printing |
| HSI | Hue, Saturation, Intensity |
| HSV | Hue, Saturation, Value |
| RGB | Red, Green, Blue |
| YCbCr | Luminance and Chrominance |
| Ground truth | The manual marking of interesting features, for example marking the locations of a subjects hands throughout a video sequence |
| *A priori* model | A statistical representation of the ground truth |
| Real time | A software application that processes data greater than 25 frames per second |
| On line | Processes which are actioned while the software application is running |
| Off line | Processes which occur prior to the initialisation of the software application. Typical processes included ground truthing, training and database construction |
| SVM | Support Vector Machine |
| RVM | Relevance Vector Machine |

# Contents

**1 Introduction** **1**

**2 Literature Review** **5**

2.1 Modelling the Background . . . . . 5

2.2 Modelling the Foreground . . . . . 9

2.2.1 View Based 2D Detection and Tracking . . . . . 9

2.2.2 Model Based 3D Human Reconstruction . . . . . 14

2.2.3 Body Part Detection using Specific Detectors . . . . . 17

**3 Tracking Human Body Parts Using Particle Filters** **21**

3.1 Bayes' Rule . . . . . 22

3.2 Particle Filtering - A Graphical Example . . . . . 22

3.3 Tracking Human Body Parts - The System Overview . . . . . 28

3.4 Background Suppression . . . . . 30

3.4.1 Chroma-Keying . . . . . 30

3.4.2 Adaptive Background Suppression . . . . . 30

3.5 Tracking using manually designed body part primitives . . . . . 31

3.5.1 Computation of a particle's fitness . . . . . 33

3.5.2 Tracking the Torso . . . . . 35

3.5.3 Tracking the face and hands . . . . . 38

3.6 Integral Images for Real-Time Performance . . . . . 43

3.6.1 Integral image benefit . . . . . 46

3.7 Results . . . 47
3.7.1 Chroma-keying . . . 47
3.7.2 Adaptive background segmentation . . . 48
3.7.3 Determining the Optimal Population Size . . . 50
3.7.4 Tracking Robustly in Cluttered Scenes . . . 52
3.8 Conclusions . . . 57

**4 Prior Data for Pose Estimation 59**
4.1 Gaussian Mixture Model Construction . . . 59
4.2 Disambiguating the Hands . . . 62
4.3 Estimation of Elbow Positions . . . 64
4.4 Results . . . 65
4.5 Conclusions . . . 66

**5 Detection and Tracking of Humans by Probabilistic Body Part Assembly 69**
5.1 Object Detection . . . 70
5.1.1 Features . . . 70
5.1.2 Training the Classifier . . . 72
5.1.3 Applying the Trained Detector . . . 75
5.2 Boosted Body Parts Detectors . . . 75
5.2.1 Exploiting Colour Cues for Reduced False Detections . . . 76
5.3 Human Body Assembly . . . 78
5.3.1 False Part Elimination using Coarse Heuristics . . . 79
5.3.2 Determining a Pose Likelihood . . . 81
5.3.3 Final Configuration Selection . . . 82
5.3.4 Estimation of Elbow Positions . . . 83
5.3.5 Detection in Sequences with a Static Background . . . 83
5.3.6 Overcoming Occlusions . . . 84
5.4 Results . . . 85

5.4.1 Body Part Detector Performance . . . . . 85
5.4.2 Body Part Detection and Assembly in Images . . . . . 87
5.4.3 Detection and Assembly in Video Sequences . . . . . 89
5.4.4 Detection of Rotated Faces . . . . . 92
5.5 Conclusions . . . . . 95

**6 Real-time Upper Body 3D Pose Estimation 97**
6.1 Data Acquisition . . . . . 98
6.2 Subsidiary Databases . . . . . 101
6.3 Model Matching . . . . . 103
6.3.1 Background Suppression and Tracking . . . . . 104
6.3.2 Scale Adjustment of the Input Image . . . . . 104
6.3.3 Extracting Subsets of the Subsidiary Databases . . . . . 105
6.3.4 Silhouette Matching using Integral Images . . . . . 106
6.3.5 Chamfer Matching and Final Selection . . . . . 109
6.4 Results . . . . . 111
6.5 Conclusions . . . . . 114

**7 Closing Discussion 115**
7.1 Summary . . . . . 115
7.2 Future Work . . . . . 118

**A Adaptive Background Suppression 121**

**B The HSV Colour Model 125**

# List of Figures


2.1 Observation and measurement process of a particle . . . . . 10

3.1 Particles propagating through time . . . . . 25
3.2 Determining the number of newly generated particles of Figure 3.1 26
3.3 System overview of the particle filter tracking systems . . . . . 29
3.4 Diagram of the Renaissance Vitruvian man . . . . . 32
3.5 Design of the torso, face and hand primitives . . . . . 33
3.6 Torso primitive examples producing negative scores . . . . . 34
3.7 Primitive parameters producing extreme fitness scores . . . . . 34
3.8 Computing the generation of new particles . . . . . 36
3.9 The nominal scale of the torso primitive . . . . . 37
3.10 Torso particle filter initialisation and convergence . . . . . 39
3.11 Initialisation locations for the face and hand filters . . . . . 40
3.12 Integral Image Computation . . . . . 43
3.13 Binary images and their corresponding integral images . . . . . 45
3.14 Computation of $\sum A$ using an integral image . . . . . 45
3.15 Number of operations using standard summation versus an integral image . . . . . 47
3.16 The effect of blue spill and motion blur using chroma-keying . . . 49
3.17 The effects of using a high scene learning rate . . . . . 51
3.18 Determining the particle population size . . . . . 52
3.19 Hand tracking error through a video sequence using the hand filters 53
3.20 Tracking a subject in a complex, cluttered scene . . . . . 55

3.21 Tracking multiple subjects in a complex, cluttered scene . . . . . 57

4.1 Ground truth labelling of body parts using a particle filter system 60
4.2 Projection of a dataset onto the first three eigenvectors . . . . . 61
4.3 The cost of a dataset constructed from K-means with an increasing number of components . . . . . 62
4.4 K-means clustering of a dataset using 100 components . . . . . 63
4.5 Error of the left and right elbow estimations . . . . . 65
4.6 Error calculation of estimated elbow positions using elbow markers 66
4.7 Estimation of elbow positions . . . . . 67

5.1 Examples of the rectangle features used for object detection . . . 71
5.2 Example images of the face, torso, leg and hand databases . . . . 76
5.3 Illustration of reduction of false detections using colour cues . . . 77
5.4 Face detector performance using colour. The increased true positive rate is obtained by reducing the number of layers in the cascade. 78
5.5 Estimating the skeletal unit length from a face detection . . . . . 80
5.6 Detector performance on the test databases . . . . . 86
5.7 Hand tracking error through a video sequence using the hand detector 87
5.8 Human assembly with all body parts present . . . . . 88
5.9 Human assembly with occluded body parts . . . . . 89
5.10 Body part assembly and elbow prediction on a video sequence . . 90
5.11 Detector and assembly performance on a video sequence . . . . . 91
5.12 Upper body part detection and assembly on a video sequence . . . 92
5.13 Upper body part detection and assembly on a video sequence from broadcast television . . . . . 93
5.14 Face rotation evaluation of the face detector . . . . . 94
5.15 Upper body part detection with rotated and side profile face . . . 95

6.1 Generic 3D human model . . . . . 99
6.2 Adjusting mesh materials . . . . . 100

6.3 Colouring body parts independently to facilitate accurate edge detection . . . . . 101

6.4 Extraction of frontal view images from motion capture . . . . . 102

6.5 Derivation of the subsidiary databases . . . . . 103

6.6 The segmented, adjusted input image . . . . . 105

6.7 A frontal view silhouette and corresponding integral image extracted from a subject . . . . . 107

6.8 Number of operations using standard summation versus an integral image . . . . . 109

6.9 The distance transform applied to an edge image . . . . . 110

6.10 Chamfer matching . . . . . 111

6.11 Frontal pose with corresponding 3D model . . . . . 112

6.12 Frontal pose with corresponding 3D model . . . . . 113

7.1 Finn the fish, an interactive artificial intelligent agent . . . . . 118

7.2 Facial expressions of Finn the fish . . . . . 119

B.1 The HSV hex cone . . . . . 125

# Chapter 1

# Introduction

Human-computer interaction (HCI) is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use [26]. With the rapid development of fast, inexpensive personal computers, visual based HCI not only has a place in industrial applications, but also in the home where users can control electronic equipment or interact with artificially intelligent constructs. HCI utilising video streams is of great benefit to handicapped or injured people who have limited control of their limbs. In addition, it has the ability to enhance the usability of cumbersome devices such as virtual reality systems that are heavily cabled. Aside from HCI, the detection of humans is an important task in many areas e.g. visual surveillance, motion capture, gait analysis and a prelude to many biometric measures. The task is particularly difficult due to the dynamic poses that humans exhibit, varying scales, self occlusions, and the range of clothing that can be worn. The problem is further compounded by poor lighting conditions, shadows and self-shadowing, and in particular, ambiguous background clutter.

The overall objective of this thesis is to explore visual detection and tracking systems that use an un-calibrated monocular camera system, for example a web

camera, in a cluttered environment. A comparable consumer HCI product is the Eye Toy [13, 69] developed by Sony Computer Entertainment, where the input game controller consists primarily of a web camera. Playing action games where the subject must move around frantically may not be appealing to older generations who tire easily, but the concept is invaluable to parents whose children can enjoy modern technology, while still being active.

This research began at approximately the same time as the announcement of the Eye Toy, however their methodology of tracking had not been disclosed. Recently, this research has sparked the interest of Sony's Eye Toy team as they are having difficulty in detecting hands. Experimentation with their current games suggests that body parts are not actually being tracked; it seems that generic motion detection in certain areas of the image dictate an action.

Many underlying processes are required for the development of a robust, real time HCI application. According to this research, four of the fundamental processes include background segmentation, human tracking, gesture recognition, and lastly, visual interaction and representation. This thesis concentrates on the detection and tracking of key parts of the human body, with a strong focus on the real time aspect; an interactive system that has noticeable latency is not only frustrating, but also unpleasant.

This thesis consists of four core chapters, each dedicated to the developments contributed to the field of computer vision. Chapter 3 employs background suppression and a general approximation to body shape, applied within a particle filter framework, to detect and track the respective body parts of humans. Tracking of the face and hands is primarily achieved using a skin colour model that is constructed from the user's face on the fly. Integral images are used for rapid computation of particles in order to maintain real-time performance, and the final system is also demonstrated on multiple subjects in a cluttered scene.

Once the key locations of the upper body are extracted, the developments pre-

sented in Chapter 4 employ an *a priori* mixture model of upper-body configurations to disambiguate the left and right hands of the subject. Furthermore, the likely position of elbows are statistically estimated, thereby completing the upper body pose.

A second method of detection and tracking is presented in Chapter 5 where detected human body parts are assembled into a full 2D human configuration within a probabilistic framework. The face, torso, legs and hands are detected in cluttered scenes using boosted body part detectors trained by AdaBoost. Due to ambiguities present in natural images, coarse heuristics are applied to eliminate the most obvious outliers, and body configurations are assembled from the remaining parts using RANSAC. An *a priori* mixture model of upper-body configurations is used to provide a pose likelihood for each configuration, after which a joint-likelihood model is determined by combining the pose, part detector and corresponding skin model likelihoods; the assembly with the highest likelihood is selected. This technique is initially applied to high resolution images of people in cluttered scenes, and is then extended to video sequences where a tracking framework is used to improve overall system performance.

The final contribution of Chapter 6 presents a method of estimating the 3D pose of the upper human body from a single camera. The objective application lies in 3D Human Computer Interaction (HCI) where hand depth information offers extended functionality when interacting with a 3D virtual environment. A database encompassing a variety of human movements is constructed from human motion capture data. A generic 3D human model is then animated with the motion capture data, and is rendered to produce a database of frontal view images. From this image database, a structure consisting of three subsidiary databases, namely the frontal-view Hand Position (top-level), Silhouette and Edge Map Databases, is extracted. At run time, these databases are loaded, subsets of which are then matched to the subject in real-time. Matching is facilitated using shape encoding

integral images, and chamfer matching. The index corresponding to the subsidiary database triplet that yields the highest matching score is used to extract the corresponding 3D configuration from the motion capture data. This motion capture frame is then used to extract the 3D positions of the hands for use in HCI. Alternatively, it can be used to render a 3D model to produce character animation directly from video.

Each chapter in turn provides results and conclusions pertaining to the theory presented in that chapter, while general closing comments, recommendations and future work are discussed in Chapter 7. Finally, a work-in-progress development that showcases an interactive cartoon 3D animal, is presented.

# Chapter 2

# Literature Review

As discussed in the introduction, this thesis has explored several areas of research that relate to a number of previous works. Many of the selected publications for this literature review contain multiple contributions, a few of which are irrelevant to this thesis. To this end, rather than discussing each reviewed paper in totality, this chapter has been divided into sections that relate to this thesis. The two chief sections discuss the modeling of the background and foreground respectively, where the latter is further subdivided to address view based, model based and detector based approaches. Furthermore, the reviews are presented in chronological order to offer a narrative that illustrates the progression of those fields of research.

## 2.1 Modelling the Background

With the use of a successful background removal algorithm, researches can focus their energy on the subject and ignore the background clutter that can create many ambiguities. Although this thesis does not progress research in background/foreground segmentation algorithms, it does however make extensive use

of an adaptive background segmentation algorithm. For this purpose, the topic of background removal deserves a brief review of a few of the popular techniques that have been developed.

Publicly, the most commonly known technique which was initially called 'blue-screening' was developed in the late 1980's by the visual effects industry for the blockbuster films Star Wars and Superman. It is currently referred to as chroma-keying [112], where an evenly lit blue *or* green surface represents the background. This technique is also used in broadcast, and can be applied to virtual reality applications [70]. Due to the pioneering use of this technique prior to commercially available low cost PCs, chroma keying was initially performed using dedicated hardware systems. Since then, software-based solutions have been developed that exhibit far more accurate separation than hardware [102].

Frame differencing [88] is another relatively straightforward technique used to identify foreground elements. Here, consecutive images are subtracted, after which a threshold is applied to the resulting difference image to determine the pixels that may correspond to motion. There are two understandable problems with this approach. If regions of the foreground contain similar colours to that of the background, they will be removed. Secondly, shadows cast on the background are often detected as foreground.

A further shortcoming of both chroma-keying and frame differencing is that they can primarily only be used for indoor applications where the background and lighting conditions are controlled. Background identification and removal for outdoor activities are far more complex as the physical environment and lighting conditions can change. The remainder of this section presents algorithms that overcome these changes.

This thesis employs a background segmentation algorithm that makes use of per pixel statistical models to represent the background distribution, which are updated over time to account for slow changes. The background segmentation

developed for Wren's [110] Pfinder is the forefather of this adaptive background segmentation technique, and models a pixel's history using a single Gaussian in the YUV colour space. In each frame, pixels have their statistics updated recursively using a simple adaptive filter. Horprasert et al [43] follow the same basic model, however model a pixel in the RGB colour space, and include the brightness and colour distortion. The brightness distortion is a scalar value that brings the observed colour close to the expected chromaticity line, while the colour distortion is the orthogonal distance between the observed colour and the expected chromaticity line. Grimson and Stauffer [40, 96] extended the statistical approached by modelling the pixel history using a mixture of Gaussians (typically three to five). The weighting of each Gaussian component represents the probability that the colour of that pixel remains the same, i.e. is part of the background. Every new pixel value is compared to existing model components and is updated with the new observation if a match is found. If no match is found, a new Gaussian component is added, and the weighting parameter is amended. KaewTraKulPong and Bowden [54, 55] based their system on this method, however they demonstrate superior performance by deriving the update equations from expected sufficient statistics and L-recent window formula. This provides a system which learns the background scene faster and more accurately. For a more in depth explanation of this adaptive background segmentation technique, refer to Appendix A.

Like chroma-keying and frame differencing, the predominant problem with Gaussian Mixture Model (GMM) background segmentation using colour is that if regions of the foreground contain similar colours as the background, they can be falsely removed. In an attempt to address this issue, Ivanov et al [51] introduce range, the measure of depth obtained using two or more camera views. Making use of both a primary and auxiliary image, image points are classified according to their 'belonging' to a known surface rather than to a group of neighboring points of

similar textures. A disparity map is built by comparing the images taken from each camera of the empty scene. The introduction of a moving subject violates this model, allowing for easy foreground/background segmentation. However, the shortcoming of this method is that invalid results are produced in low contrast scenes, or in regions that are not visible in both camera views. Difficulties are also experienced where the subject moves closely against the background e.g. the walls of a room. Gordon et al [38] therefore presented a method for background estimation and removal that takes advantage of the strengths of both colour and range. They show that this combination produces improved results compared to using either data source individually.

More recently, Kim et al [57] proposed a background modeling and subtraction algorithm by codebook construction. The codebook algorithm clusters pixel samples into a set of codewords on a per pixel basis. The technique offers the same advantages of using a GMM, but also offers unconstrained training that allows foreground objects to move in the scene during the initial training period. 6.5 codewords are used per pixel, and the algorithm runs at 30 frames/second, which is considerably faster compared to the GMM method which processes data at approximately 15 frames/sec.

Background segmentation plays a crucial role in the detection and tracking of body parts in Chapter 3, and again later in Chapter 6 where the silhouette of the subject is to be extracted. It is also used in Chapter 5 to reduce the body part false detection rate, thereby improving system speed and performance. Initially, due to the research of this thesis placing a primary focus on human detection and tracking, chroma-keying was used as an easy and fast method of extracting the foreground subject. However, due to the availability of the source code kindly provided by KaewTraKulPong and Bowden [55], an adaptive background segmentation algorithm was used in latter work as it allowed for research to be conducted in unconstrained cluttered environments. At the cost

of a relatively noisy segmentation, the algorithm is conducted on sub sampled images to improve system speed performance. Although this is not ideal, the detection methods employed in the latter chapters are nonetheless able to cope with this noisy segmentation.

## 2.2 Modelling the Foreground

In order to recognise the actions and behaviour of a person, the ability to track the human body is an important visual task. The techniques discussed in this section have been separated into three main areas. The first covers 2D appearance or view based approaches, where coarse body part descriptions are manually designed to detect body parts using colour, edges, contours and silhouettes etc. Secondly, 3D human reconstruction can also be performed using an appearance based approach, where either single, or multiple camera systems are used. The final method is model based human detection where prior models have been learned from a selection of images using machine learning techniques such as boosting, neural networks and Support Vector Machines etc.

### 2.2.1 View Based 2D Detection and Tracking

Robust detection and tracking of objects in scenes with both cluttered and constant backgrounds is a challenging task. Cluttered scenes obviously pose a greater problem as background objects can often provide similarities to the foreground object of interest. A common approach to tracking objects in cluttered scenes makes use of what has been termed *particle filtering*. There are several variations belonging to the same Bayesian [78] framework, a few of which will be discussed shortly. The basic idea of a particle filter is that the posterior is initially approximated by a set of discrete random samples or *particles* with associated weights

[62]. Particles with small weights are discarded in the subsequent iteration, while those with large weights are duplicated in order to maintain the population size. Provided that the population is adequately sized, the particle filter will converge on a hypothesis after several iterations. Section 3.2 provides a graphical example illustrating the propagation of particle through time as particle filtering forms the basis of the tracking algorithm presented in Chapter 3.

Under this framework, a common technique for object detection and tracking uses a contour, which is a predefined outline description of the object to be tracked, as shown in Figure 2.1. The thick red line represents a head and shoulder shaped contour in a hypothesised configuration, the purpose of which is to correspond to a generic upper human torso. To determine how well the contour fits the object, a one dimensional edge detector is applied along the thin spines or *measurement* lines; the data that is of interest is the black dot showing the strong edges exhibited by the detector. A simple method of measuring the fitness is to calculate the distance of the edges to the contour; the smaller the mean distance, the greater the fitness.

Figure 2.1: Observation and measurement process of a particle

Contours can be applied to the image space in a Kalman filter framework [37, 41], which is a recursive linear estimator [31]. Random variations of the contour are initialised, and improved estimates are calculated as new data is obtained in

subsequent iterations. Using a single generic contour to estimate the shape of an object is not very robust, and deformable contours have been employed [99, 5, 81]. Also referred to as snakes or B-splines, these are beneficial in that they can be deformed to fit the object of interest more closely. The Kalman filter however has primarily been shown to operate effectively in relatively clutter free environments, and it offers a further limitation in that it is based on the assumption that the posterior probability distribution of the moving object is Gaussian. To address the non-Gaussian movement of objects, Isard and Blake presented CONDENSATION (Conditional Density Propagation) [49, 50, 6], which combines factored sampling [39] with a dynamic model to propagate a probability distribution over time. The algorithm is a form of particle filter, and illustrates the use of active, deformable contours to track the outlines of agile moving objects in substantial clutter. A limitation of this algorithm is that a large number of particles is required if the dimensionality of the search space is high. This was addressed by MacCormick et al [66] by introducing Partitioned sampling to reduce the curse of dimensionality. This is achieved by using multiple particle filters to track specific objects, rather than representing the entire complex model as a single entity. This methodology is highly advantageous in tracking multiple [67] or articulated objects [60] and is followed in Chapter 3 where four particle filters are used to track the torso, face and hands rather than a single particle filter. The set of particles is resampled after the evaluation of each partition, contrary to the standard approach where the resampling is performed once at each time step for all model parameters.

Objects like the face and hands that consist of a dominant colour, intuitively suggest that colour should also be used as a cue for tracking. Rasmussen and Hager [86] do just that, and combine the use of edge information with colour. More recently, several particle filter techniques ignore edge cues, and use a coarse shape and a colour model to track the object. Pérez et al [80] use a vertically aligned

rectangular box to describe the shape of the face, and a reference histogram modelling skin tone in the HSV colour space. Once the filter is initialised, histograms are built from the contents of each rectangular particle. To obtain scores of fitness, each candidate histogram is then compared to the reference histogram using a distance metric referred to as the Bhattacharyya similarity coefficient [3, 1]. Nummiaro et al [74] follow a similar procedure with two additions. An ellipse, rather than a rectangle, is used to describe the object shape, where candidate observations contained within the ellipse are weighted according to their distance to the centre of the ellipse i.e. the further away from the centre, the lower the weight. In addition, the colour model of the object is updated slowly over time in a similar fashion to the adaptive background modelling algorithms described earlier in Section 2.1. In this case however, thresholds obtained from the object's predefined colour model are used such that the model is not updated if the tracker has lost the object.

Moving away from particle filters, other methods of human detection and tracking have also been explored. Ju et al [53] define a 'cardboard person model' in which a pair of legs is represented by a set of connected planar patches. A focus is placed on the constant intensity of each limb rather than the edges, and optical flow is used to deal with the articulated motion. The method is demonstrated on cyclic walking motion, where the subjects wear uniformly coloured trousers.

Pfinder [110], mentioned previously for it's background modelling in Section 2.1, tracks humans using a blob representation method. It was initially developed by Pentland [79] to extract compact descriptions of multi-spectral satellite imagery. In Pfinder, structurally meaningful body parts are segmented by forming feature vectors at each pixel by adding spatial coordinates to the colour components of the image. These vectors are then clustered so that similar properties (location and colour) are combined to form coherent connected regions or 'blobs'. A pixel-by-pixel support map is also included that shows the actual occupancy of clusters,

and divides the segmented regions into spatio-colour classes. The hands and head are also divided into blobs and are tracked for the analysis of gestures.

As previously mentioned, a deformable contour representing the generic shape of a relatively simple object can be used for detection and tracking. Although contours can cope with a certain degree of shape variation, they cannot easily model a complex object whose appearance changes greatly and rapidly; for example, the side profile silhouette of a walking human. Gavrila and Philomin [36] employ the Chamfer Matching System [2] to detect pedestrians by comparing the candidate distance transformed image [27] to a database of pedestrian templates. These templates consist of an outlines of a various pedestrians, and are grouped according to similarity in order to establish a hierarchy for efficient matching. This technique of pedestrian detection has recently even been incorporated into a vehicle driver warning system [35] for accident prevention. The procedure of Chamfer matching is discussed in more detail later in Chapter 6, where it assists in determining the 3D pose of the upper human torso.

The last area of research discussed in this section is that of Ioffe and Forsyth [48] who investigate probabilistic methods for finding people, where they assume that an image of a human can be decomposed into a set of distinctive segments. A parallel-edge segment detector is used to locate image regions that could possibly be body parts; as can be expected, many false segments are also detected. Once the segments have been detected, they need to be assembled into groups that could represent a person. These groupings are built incrementally by sequentially considering groups of increasing size, and adding segments to them. Since the body is subject to strong physical joint constraints, many ambiguities can be overcome. The search is also made more manageable by realising impossible groupings, and ignoring them. A pre-defined top level classifier takes a grouping of segments and determines whether the grouping corresponds to a person. If it is decided that no such augmentation is possible, then it, together with all other

groupings containing it are rejected. Those groupings that are not discarded are grown as described previously. Should groupings be discarded at an early stage, the search becomes more efficient. This ideology of searching through the image space for evidence of human parts, and them assembling them into a 'body plan' [33], is adopted in Chapter 5. However, rather than using a simple parallel line detector to identify random parts, specifically trained detectors are employed to identify key body parts.

### 2.2.2 Model Based 3D Human Reconstruction

With an increasing popularity over the past decade, animated 3D human models feature frequently in the film and games industry. Character animation can be done laboriously by hand via key framing, however the results often prove to be unnatural and jerky. Motion capture systems provide a fast method of acquiring 3D motion data, and offer results that are obviously very natural since a human actor is used. The major drawback however is the high combined cost of a studio with controlled lighting, a multi-sensor suit, and multiple cameras. The ability to animate a model directly from video would therefore be an extremely beneficial tool, especially if this can be done using a single camera. The 3D information of a human subject can also benefit applications like HCI, where hand depth information can extend the library of recogniseable gestures, or allow the user of the HCI interface to engage with a 3D virtual environment. This section discusses a variety of 3D human reconstruction and pose estimation techniques that use either single or multiple camera systems.

Relative 3D hand positions can be obtained in a straightforward manner by determining the size of the object i.e. the greater the size of the hand, the closer it is to the camera. However, this will only work if the size of the hands remain constant by keeping them in a fixed pose like a fist. Another method would be

to raise the camera, with it titled down toward the subject. This technique was employed in Brooks' intelligent room project[12] to determine the distance of the subject from the camera. This could also be extended to determine the camera-to-hand distance, however this would only prove useful if the hands moved in a plane parallel to the ground, which is clearly limiting.

Alternatively, since the seminal work of Hogg [42], several researchers use articulated models to track humans and their parts. As discussed earlier, particle filters offer an efficient and robust method of object tracking in 2D; Deutscher et al [23] extended this methodology and presented the Annealed Particle Filter (APF) with an articulated geometric body model to reconstruct a moving human in 3D. The purpose of the adaptation was to reduce the search time as particle filters tend to be slow in implementation due to the complexity of the approach and dimensionally of the search spaces. It also makes use of a likelihood function based on edge and region measurements, therefore requiring prior knowledge of the width of the articulated body segments. The approach does however make use of a calibrated multi-camera system to overcome ambiguities, which limits the application to intensive off-line processing and a multi camera studio environment. The APF was then improved and named 'The Amended APF Algorithm' [24], where it bears a great similarity to genetic algorithms.

Sidenbladh [93, 92] overcomes the constraints of multiple cameras by employing strong motion priors to overcome visual ambiguity. A motion capture system is used to acquire a set of walking cycles, and a model of the mean walking cycle, as well as the most common variations, is learned using principle component analysis. Tracking is conducted using particle filters, where the posterior is estimated using the prior. The main concern with the use of prior data of specific actions is that it is difficult to extend tracking to more general applications where the motion of the subject is less well defined.

Using training data of a single motion is clearly limiting, and an *a priori* Gaussian

Mixture Model (GMM) constructed from several motions is adopted by Howe [44] et al to compute prior probabilities of full body 3D motions. A Point Distribution Model (PDM) [20] is constructed from twenty tracked points of the full human body in a monocular sequence, and is compared to the GMM to determine the best fit. Although the method is shown to operate well on several actions, the sequences on which they are tested are only 11 frames in length.

Methods that track various articulated points of the body are prone to high error, and the accuracy of the 3D reconstruction is dependent on the robustness of the tracking algorithm. Rather than relying on the tracking of multiple key points, Bowden [11] restricts tracking to the face and hands, and constructs a PDM that includes the positions of these tracked parts, and points extracted from the subject's contour.

A different method of acquiring a 3D model of a human is to reconstruct the shape using 2D Silhouettes, taken from multiple synchronised cameras [71]. The 2D silhouettes define a set of rays from the camera to the object, thereby defining a cone containing the object. Integration of these provides a volume in the scene in which the object lies, and the construction of the 3D human shape is determined using two or more contours and prior geometric models of the human body.

In [61], the reconstruction begins with locating the subject's hands and elbows using convex curvature peaks. The knowledge of these positions allows for a prediction of the shoulder position, after which the head is then detected using a head-shoulder contour. Finally, the torso is detected by extracting the medial axis of each 2D silhouette, matching them, and then fitting a line.

In [19], the spine or main axis of the person is detected by extracting the medial axis of each 2D silhouette. Thereafter, the position of the neck and torso are extracted. Using the main axis, the height of the subject can be determined, allowing for a coarse estimation of the neck and torso. The width of the silhouette is analysed, with the most narrow section indicating the neck. Finally, the limbs

are detected using a PCA approach on points lying away from the main axis. Geometric body constraints such as limb ratio are also used. A likelihood function that focuses attention on the similarities between the body median axes and the articulated body model is used, thereby eliminating the need for estimation of body segment widths.

Interestingly, the matching of shape and edge templates has also received attention in hand pose estimation [97] where shape matching follows a cascaded approach to reduce the number of edge template comparisons. This method is extended in Chapter 6 to determine a corresponding 3D model of the upper human body extracted from a frontal view image sequence.

### 2.2.3 Body Part Detection using Specific Detectors

Object detection using simple image cues like boundaries, edges and colour etc. has the shortcoming of being highly susceptible to ambiguities. This is also due to the fact that a coarse depiction of the target is constructed by hand. Machine learning offers an advantage in that an algorithm constructs a description, derived from the analysis of a library of images of the same object. The first disadvantage is that appropriate training images need to be obtained, which can be a long process, especially since cropping of the object is typically required. Furthermore, the detector needs to be trained which can be a slow process. The primary motivation in using such a method is that the positive detection rates offer an improvement over using a few basic cues alone.

The papers reviewed in this section span over a period of approximately six years, and employ a variety of machine learning techniques. Due to the radical improvement of computer hardware during this period, it is difficult to assess which procedure bears the best real time application.

Neural networks were widely used in engineering applications in the 1990's, and

were also applied to face detection [64, 89]. The detection system of Rowley et al [89] consisted of two neural networks, trained to detect frontal, upright faces in gray scale images. The first, faster network, performed an initial sweep to pre-screen candidate regions for the second, slower and more accurate network. To facilitate the reliability of the training images, and hence the detectors, the training images were improved using lighting correction and histogram equalisation.

The face detection system recognised widely for detecting faces in real-time, was proposed by Viola and Jones [104, 105]. The detection procedure classifies images based on the value of simple features, reminiscent of Haar basis functions [68]. Training and feature selection was performed by Schapire's AdaBoost [91, 90], which is a general method for improving the accuracy of any given learning algorithm. The rapid detection speed was aided by the use of integral images to compute the values of the square features. In addition, they extended Rowley's [89] notion of using multiple networks by constructing the detector with a cascade of classifiers with increasing complexity. Appendix 5.1 provides an explanation of object detectors trained by Adaboost.

Section 2.2.1 described the detection of pedestrians by the matching of a contour to template examples. This was also achieved by Papageorgiou et al [75, 77] who used a trained detector, where the full human body was regarded as a single object. Haar wavelets were used to represent the images, and Support Vector Machines (SVM) [103] classified the patterns. The method however does not cope well in situations where body parts are occluded.

An alternative method of detecting humans is to detect their constituent parts, and to assemble them into a human-like configuration. Mohan et al [73] applied Papageorgiou's detector construction method to create distinct body part detectors for the head, legs, left arm and right arm. The detections are then grouped and classified by a second SVM based classifier to determine 'person' or

'non-person' configurations. The results presented do appear promising, however the examples only contain upright humans, with their arms at their sides. In terms of assembling detections into intelligent configurations, Felzenszwalb and Huttenlocher [28] made use of pictorial structures [30], which are collections of parts arranged in deformable configurations. In their work, each part is represented using a simple rectangular appearance model constructed from the mean and variance of the colour estimated from one example of the part to be detected. The deformable configuration is then represented by spring-like connections between pairs of parts.

The detection of humans by components was further extended by Ronfard et al [84] to detect articulated humans. Fifteen rectangular body part detectors were trained using SVMs, where the feature set consisted of a Gaussian filtered image, and its first and second derivatives. All detections were then parsed using a body tree, which was constructed using a Relevance Vector Machine (RVM) [100]. RVMs do not provide significantly lower error rates than SVMs, but they do however provide similar results using far few training examples. Ronfard places a large emphasis on the ability to configure the detections into a articulated models, however the method is only demonstrated on upright humans, again with their hands at their sides.

Mikolajczyk et al [72] model humans as flexible combinations of boosted face, torso and leg detectors. Parts are represented by the co-occurrence of orientation features based on 1st and 2nd derivatives. The procedure is computationally expensive, and 'robust part detection is the key to the approach' [72]. This co-occurrence ideology is also applied to facial features by Cristinacce et al [21] using a Pairwise Reinforcement of Feature Responses.

The aforementioned detectors were constructed to identify objects using particular features in a similar fashion to the way humans do. Rather than using specific features, Roberts et al [87] created probabilistic region templates for the head,

torso and limbs, also determined using image databases. Likelihood ratios for individual parts are learned from the dissimilarity of the foreground and adjacent background distributions. The process is however computationally expensive, and the greatest likelihoods occur where the foreground and background hold similar colour distributions, which is typically not a common property in natural images.

The last object detection methods presented here employ adaptive techniques in a tracking framework such that the detector updates over time. A Relevance Vector Machine (RVM) is employed by Williams et al [109] to train the detector on the fly from a few sample frames. A sparse probabilistic learning algorithm updates the detector as time passes, and the method has been demonstrated to track faces and car number plates. Updating the tracker can obviously pose problems if the incorrect information is contributed to an update cycle. Kang [56] combine a variety of image cues including foreground-background colour similarity, skin colour, head contour and edge density for face/head tracking, applied within a particle filter framework. As can be expected, their results show great improvement over using any single cue alone.

# Chapter 3

# Tracking Human Body Parts Using Particle Filters

This chapter presents the use of particle filters to track moving objects including the face, upper torso and hands of single or multiple subjects. The speed at which the particle filter operates is naturally dependent on the dimensionality of the search space, and the number of cameras. This chapter makes use of a single camera, and the search space dimensionality is kept to a minimum, allowing for the system to operate with above real time performance.

Section 3.1 firstly introduces Bayes' Theorem which is used to estimate the posterior of each iteration of a particle filter. Section 3.2 in turn explains the operation of a particle filter with the aid of a graphical example and corresponding calculations. This methodology is extended in the latter sections to locate and track the aforementioned body parts of a subject moving in a video sequence. The final algorithm employs background suppression to allow for operation in cluttered scenes, and a general approximation to body shape for body part tracking such that integral images can be used to improve speed performance.

## 3.1 Bayes' Rule

Bayes' rule is a mathematical formula used for calculating conditional probabilities. A conditional probability is defined as the probability of event $A$ occurring given that a related event $B$ has occurred. This probability is written as $P(A|B)$, and is read as 'the probability of $A$, given $B$'. It is referred to as the *posterior* as it computed once the information of both events $A$ and $B$ is known.

The definition of conditional probability is given by

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A) \tag{3.1}$$

where $P(A \cap B)$ is the joint probability and $P(A)$ is referred to as the prior probability of $A$. $P(B)$ in turn is the prior probability of $B$, and is also referred to as the normalising constant. Rearranging equation 3.1 yields

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.2}$$

which is conventionally known as Bayes' rule. Computing $P(B)$ is however a difficult task, and since it is a normalising constant, it can be omitted. This form is also referred to as factored sampling, and is presented in equation 3.5 where the posterior is estimated by muliplying the observations of the current iteration by the weights determined from the previous iteration. Section 3.2 illustrates how the use of this methodology upgrades each particle's prior probability to a posterior probability which assists in converging the system more rapidly.

## 3.2 Particle Filtering - A Graphical Example

For the purpose of visual simplicity, the particle filter is explained by making use of Figure 3.1, where $S_{\mathrm{x}}$ represents the parameterisation of an unknown distribu-

tion whose global maximum is to be determined. The table of Figure 3.2 provides the corresponding calculations, and is discussed at the end of this section.

At initialisation, $N$ samples or *particles*, are randomly dispersed across the parameter space $x$, the purpose of which is to find the absolute maximum. The initialisation represents $P$, the prior probability that $S_{\mathrm{x}}$ exists at time $t = 0$, and is represented discretely by

$$\{\hat{S}^{(n)}\,, \pi^{(n)}\,, n = 1 \ldots N\} \tag{3.3}$$

where $\hat{S}$ represents the sample set, with each particle weighted by $\pi$. $S_{\mathrm{x}}$ is sampled accordingly, and the weights are determined by

$$\pi_t^{(n)} = P(\omega_t^{(n)} | \hat{S}^{(n)}) \tag{3.4}$$

Although the posterior is unlikely to stay the same with each iteration, the prior gives some indication as to what the posterior will be. Any inconsistencies between the two are dealt with by the repeated iterations of the particle filter, which gradually filter out any error implicit to the prior. The observations $\omega_t^{(n)}$ are therefore upgraded to obtain $\omega_t^{'(n)}$ by taking the corresponding prior weights into account:

$$\omega_t^{'(n)} = \omega_t^{(n)} \times \frac{\pi_{t-1}^{(n)}}{\sum_{n=1}^{N} \pi_{t-1}^{(n)}} \tag{3.5}$$

At initialisation, a prior set of weights does not exist, and these are therefore set to be equal: $(\forall n)\pi_{t-1}^{(n)} = 1$ at $t = 0$.

A new set of weights $\pi_t^{(n)}$, which form the estimate to the posterior (and therefore the prior for the next iteration), is then derived from the normalised updated observations:

$$\pi_t^{(n)} = \frac{\omega_t^{'(n)}}{\sum_{n=1}^{N} \omega_t^{'(n)}} \tag{3.6}$$

Naturally, the larger the value of the observation, the greater the derived weight. This weight is then used to determine the number of particles $G_t^{(n)}$ that are to

be generated from each parent, for use in the subsequent iteration. This process serves to duplicate high scoring particles and to discard poor scoring particles:

$$G_t^{(n)} = \pi_t^{(n)} \times N \tag{3.7}$$

With the inclusion of dynamics, subsequent iterations disperse the newly generated particles away from the respective parents, which serves to acquire new samples along $S_{\mathrm{x}}$. Using a white Gaussian drift term to represent the system dynamics, a predictive set of particles for the posterior is then generated:

$$\hat{S}_{t+1}^{(n)} = \eta(\hat{S}_t^{(n)}, \Sigma noise) \tag{3.8}$$

where $\Sigma noise$ is a constant.

The remainder of this section provides a walk through of particles $\hat{S}_t^{(n=2)}$ and $\hat{S}_t^{(n=4)}$ in the particle filter diagram of Figure 3.1 and the corresponding table of calculations in Figure 3.2. At initialisation ($t = 0$), the distribution $S_{\mathrm{x}}$ is sampled: $\omega_t^{(n=2)} = 22$ and $\omega_t^{(n=4)} = 96$. The observations are then multiplied by the corresponding weights to produce the updated observations $\omega_t^{'(n=2)} = 4.40$ and $\omega_t^{'(n=4)} = 19.20$. These are then normalised by the sum total of the updated observations to create a new set of weights $\pi_t^{(n=2)} = 0.07$ and $\pi_t^{(n=4)} = 0.31$, which form the estimate to the posterior and therefore the prior for the next iteration. In the diagram, the magnitude of these weights is represented graphically by the size of the 'dot'. The weights are then multiplied by the population size $N$ to determine the number of particles to be generated: $G_t^{(n=2)} = 0.4$ and $G_t^{(n=4)} = 1.5$. Since only an integer number of particles can be created, these are rounded such that $\hat{S}_t^{(n=2)}$ and $\hat{S}_t^{(n=4)}$ generate 0 and 2 particles respectively. In large particle populations, rounding may cause the population size to fluctuate slightly, but typically not more than five percent from the nominal.

In the diagram, iteration $t+1$ demonstrates the dispersion of the new particles of $\hat{S}_{t+1}^{(n)}$ from their parent particles. It is also evident that particle $\hat{S}_t^{(n=2)}$ is discarded,

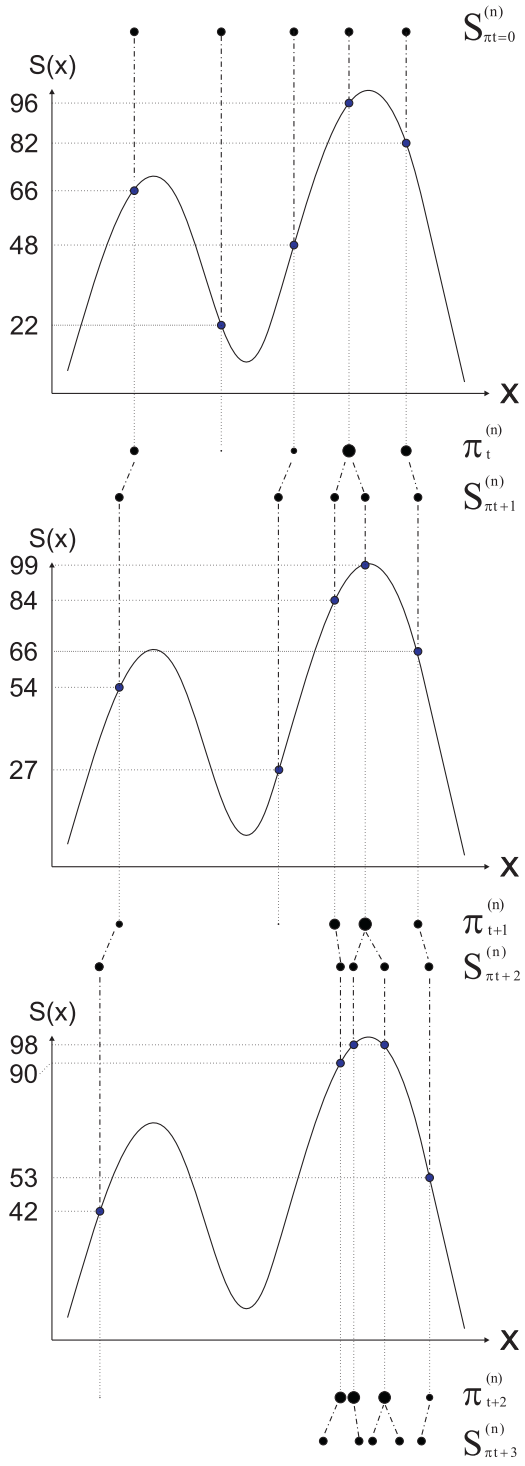



Figure 3.1: Particles propagating through time

| $N = 5$ | $\hat{S}_t^{(n=1)}$ | $\hat{S}_t^{(n=2)}$ | $\hat{S}_t^{(n=3)}$ | $\hat{S}_t^{(n=4)}$ | $\hat{S}_t^{(n=5)}$ | | |
|---|---|---|---|---|---|---|---|
| $\omega$ | 66 | 22 | 48 | 96 | 82 | | |
| $\pi_{t-1}$ | 1 | 1 | 1 | 1 | 1 | $\sum \pi_{t-1} = 5$ | |
| $\omega' = \omega.\left(\frac{\pi_{t-1}}{\sum \pi_{t-1}}\right)$ | 13.20 | 4.40 | 9.60 | 19.20 | 16.40 | $\sum \omega' = 62.80$ | t=0 |
| $\pi_t = \frac{\omega'}{\sum \omega'}$ | 0.21 | 0.07 | 0.15 | 0.31 | 0.26 | | |
| $G = \pi_t \times N$ | 1.05 | 0.35 | 0.76 | 1.53 | 1.31 | | |
| rounded | 1 | 0 | 1 | 2 | 1 | | |
| $\omega$ | 54 | 27 | 84 | 99 | 66 | | |
| $\pi_t$ | 0.21 | 0.15 | 0.31 | 0.31 | 0.26 | $\sum \pi_t = 1.24$ | |
| $\omega' = \omega.\left(\frac{\pi_t}{\sum \pi_t}\right)$ | 9.19 | 3.34 | 20.78 | 24.49 | 13.95 | $\sum \omega' = 71.75$ | t+1 |
| $\pi_{t+1} = \frac{\omega'}{\sum \omega'}$ | 0.13 | 0.06 | 0.29 | 0.34 | 0.19 | | |
| $G = \pi_{t+1} \times N$ | 0.64 | 0.28 | 1.45 | 1.71 | 0.97 | | |
| rounded | 1 | 0 | 1 | 2 | 1 | | |
| $\omega$ | 42 | 90 | 98 | 98 | 53 | | |
| $\pi_{t+1}$ | 0.13 | 0.29 | 0.29 | 0.34 | 0.19 | $\sum \pi_{t+1} = 1.24$ | |
| $\omega' = \omega.\left(\frac{\pi_{t+1}}{\sum \pi_{t+1}}\right)$ | 4.33 | 20.97 | 22.83 | 26.91 | 8.29 | $\sum \omega' = 83.33$ | t+2 |
| $\pi_{t+2} = \frac{\omega'}{\sum \omega'}$ | 0.05 | 0.25 | 0.27 | 0.32 | 0.10 | | |
| $G = \pi_{t+2} \times N$ | 0.26 | 1.26 | 1.37 | 1.61 | 0.55 | | |
| rounded | 0 | 1 | 1 | 2 | 1 | | |

Weighted random sampling implementation of iteration t+2

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\omega$ | 42 | 90 | 98 | 98 | 53 | $\sum \omega = 381$ | |
| $\pi_{t+2} = \frac{\omega}{\sum \omega}$ | 0.13 | 0.27 | 0.30 | 0.30 | 0.16 | | t+2 |
| $G = \pi_{t+2} \times N$ | 0.64 | 1.36 | 1.48 | 1.48 | 0.80 | | |
| rounded | 1 | 1 | 1 | 1 | 1 | | |

Figure 3.2: Determining the number of newly generated particles of Figure 3.1

and $\hat{S}_t^{(n=4)}$ generates two particles. The distribution $S_{\mathrm{x}}$ is sampled with the new set of particles, and the weights and particle generation are computed as before. The only exception is that weights can now be retrieved from the prior iteration. From the table, since particles $\hat{S}_{t+1}^{(n=3)}$ and $\hat{S}_{t+1}^{(n=4)}$ were drawn from $\hat{S}_t^{(n=4)}$, it follows that $\pi_{t+1}^{(n=3)} = \pi_{t+1}^{(n=4)} = \pi_t^{(n=4)}$. The process is repeated again at $t+2$ where it is already apparent how the particles have begun to converge on the global maximum.

For comparative purposes, the last section of Figure 3.2 demonstrates a weighted random sampling implementation of the particle filter for iteration $t+2$ where the prior is not used, ie. the particle observations are *not* multiplied by the weights of their parents. The table indicates that particle $\hat{S}_{t+1}^{(n=1)}$ does not perish, and that particle $\hat{S}_{t+1}^{(n=4)}$ is not duplicated. Should this method be applied in subsequent iterations, it is likely that the particle filter would require many more iterations to converge. Although this may be acceptable in terms of a stationary object, it is unlikely that a particle filter with such an implementation would be sufficiently robust if used to track fast moving objects.

In order for a particle filter to converge correctly, a certain number of particles is required. This number naturally increases according to the dimensionality of the configuration space. Being able to determine the minimum population size is beneficial as a filter with too few particles may converge on a local maximum. MacCormick and Isard [67] present a method of estimating this minimum by making use of two quantities, namely the *survival diagnostic* $\mathcal{D}$ and the *survival rate* $\alpha$. $\mathcal{D}$ is defined for the particle set as

$$\mathcal{D} = \left( \sum_{n=1}^{N} \pi_t^2 \right)^{-1} \tag{3.9}$$

The survival rate $\alpha$ is a property of a given prior $p(\mathrm{x})$ and posterior $p'(\mathrm{x})$, and is given by

$$\alpha = \left( \int p'(\mathrm{x})^2 / p(\mathrm{x}) d\mathrm{x} \right)^{-1} \qquad (\alpha < 1) \tag{3.10}$$

Finally, according to [14, 23], the minimum number of particles required for convergence is determined by

$$N \geq \frac{\mathcal{D}}{\alpha^d} \tag{3.11}$$

where $d$ is the number of dimensions. In contrast to a small population, an overly large population will more than likely converge on the correct solution, however at the cost of expensive computation. In the simple application discussed in this section, where only a single maximum is to be determined, an increasing number of particles could be used without noticeable effect on system performance. However, in the terms of tracking multiple objects which have several varying parameters such as translation and scale, the population size for each particle filter system must be kept at a minimum should real-time performance be required.

## 3.3 Tracking Human Body Parts - The System Overview

This approach to tracking of body parts includes various steps which are interdependent. With reference to Figure 3.3, a brief overview of the steps is listed below:

1. Acquire a colour image of the subject using a single un-calibrated camera.
2. Segment the subject from the background using chroma-keying or adaptive background suppression to create a foreground image and a foreground binary mask.
3. Create an integral image (Section 3.6) from the foreground mask. This serves to improve system performance in terms of speed.
4. Initialise a particle filter system to detect the torso. The posterior is fed back into step 1 to continue tracking the torso.

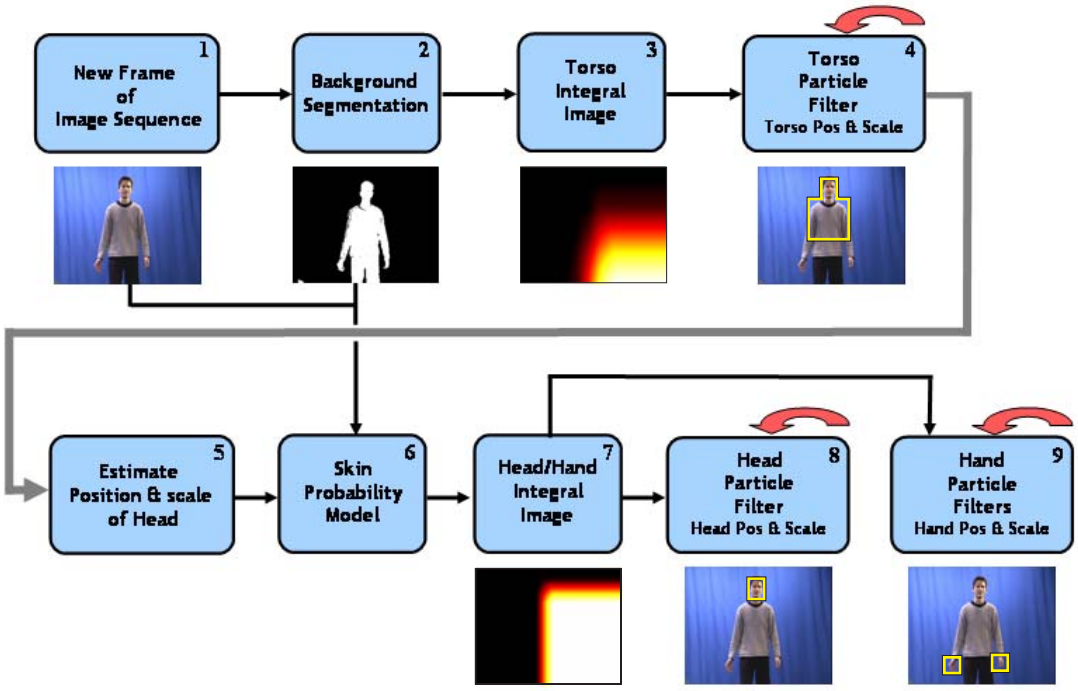



Figure 3.3: System overview

5. Once the torso particle filter has converged, estimate the scale and position of the subject's face and hands (Section 3.5.3) based on the torso parameters of step 4.
6. Build a subject specific skin model from the colour information in the estimated facial region. A generic prior skin model is used to prevent the inclusion of unlikely skin pixels.
7. Using the subject specific skin model, segment regions of skin tone from the foreground image. Create an integral image of this skin-segmented image.
8. Initialise a particle filter system for the face.
9. Initialise 2 particle filter systems for the left and right hands.

## 3.4 Background Suppression

Segmentation of the subject from the background plays a central role in tracking the torso, and a beneficial role in tracking the face and hands. Segmentation does not need to be flawless as the particle filter systems only require a coarse representation of the subject. Two background suppression methods have been explored, namely chroma-keying and adaptive background suppression.

### 3.4.1 Chroma-Keying

Chroma-keying is a straightforward and computationally inexpensive technique that is primarily used in the film industry, where a subject is captured against a solid coloured blue or green screen. When using a blue screen (as in the early research of this thesis), segmentation is achieved by computing the chromaticity of each pixel according to inequality 3.12. Should this value fall below a threshold T, the pixel is set to black, and white otherwise.

$$2B - (R + G) < T \tag{3.12}$$

where R, G and B represent red, green and blue colour channels. This threshold is adjusted manually and is dependent on the scene lighting and the colour intensity of the background screen. Relying on chroma-keying limits the tracking algorithm to controlled indoor studios. A background suppression algorithm that can cope with cluttered dynamic scenes is therefore highly desirable.

### 3.4.2 Adaptive Background Suppression

The background removal algorithm employed here (with the permission of Kaew-TraKulPong and Bowden [55]) was originally developed for exterior visual surveillance and relies upon modelling the colour distribution with a Gaussian mixture

model on a per pixel basis. This model is built in an on-line fashion, and once the background has been learned, each pixel is assigned a foreground likelihood which increases according to sudden intensity variation i.e. moving objects. Appendix A provides a more elaborative explanation of the algorithm. This likelihood image is later converted to an integral image which is passed to the torso particle filter for tracking.

To assist with tracking of the face and hands, a threshold is used to accept foreground pixels to produce a binary mask where white ('1') denotes foreground and black ('0') background. This binary mask is then used to create a *foreground image*, a colour RGB foreground and black background extracted from the natural image. The subject specific skin model is learned using this foreground image, discussed in Section 3.5.3.

## 3.5 Tracking using manually designed body part primitives

In this section, a particle is represented by manually designed *primitives* that are used to detect and track the respective body parts. Figure 3.4 illustrates the Renaissance subdivision of the human figure into eight lengths each of which is equal to the length of the face, measured from the hairline to the chin. For the purpose of this thesis, this will be referred to as a *skeletal unit length*. This length assists in the design of the body part primitives.

Four particle filter systems are used, each dedicated to tracking a specific part, namely the *torso*, *face*, *left hand* and *right hand* filters. Assuming independence reduces the dimensionality of the search space, and therefore the number of particles required to estimate the posterior. However, each system is applied in a hierarchical manner such that the gross location of the torso influences the pre-

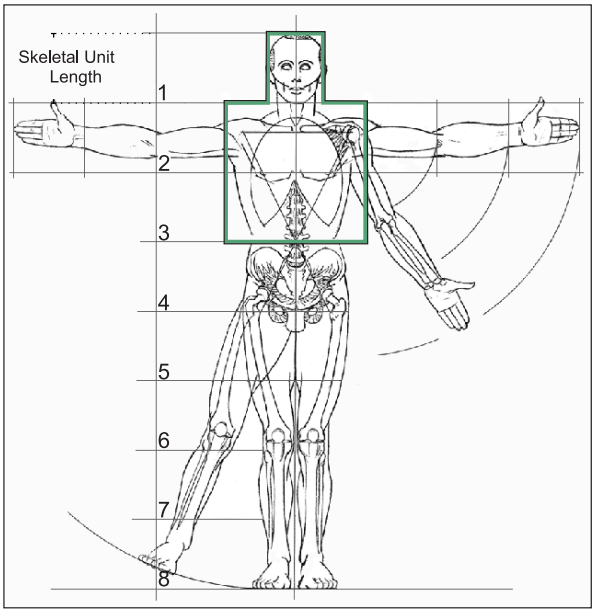



Figure 3.4: The Vitruvian man (*Figure taken from* [76])

dicted priors of the latter filters. In addition to position, the initial scales of the face and hand primitives are also dependent on that of the torso primitive.

Figure 3.5 shows the torso, face and hand primitives that represent the particles in the respective particle filter systems. Each has been constructed according to the proportions of the Vitruvian man. The primitives consist of an inner Region $A$ (gray) and outer Region $B$ (yellow); $Area(A) = Area(B)$, where $A$ holds a positive weighting, and $B$ negative. These positive and negative weightings form the primary step in calculating a fitness score for each particle, shown in Section 3.5.1. With respect to the torso and hand primitives, the bottom section of Region $A$ and Region $B$ coincide as to prevent the lower abdomen and neck from influencing the respective scores. The hands however are self-contained skin coloured objects, and Region $B$ surrounds Region $A$ completely.

At initialisation, the primitives are constructed using the captured image dimensions whereby a maximum scale of '1' produces a primitive whose height equals

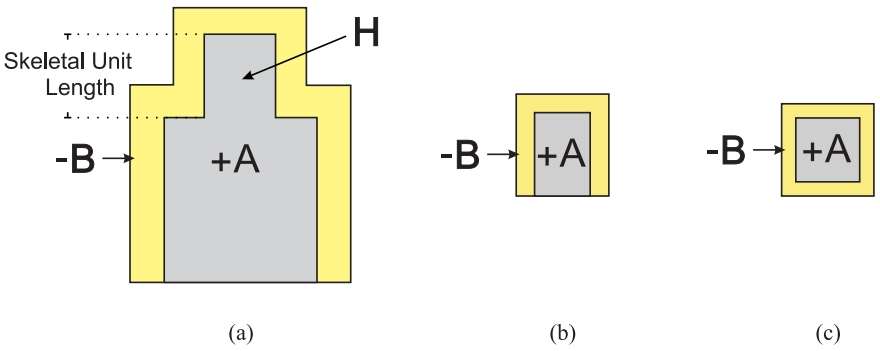



Figure 3.5: (a) Torso primitive (b) Face primitive (c) Hand primitive

that of the image. Once the torso filter has converged, the nominal scales of the face and hand distributions are set to one third of that of the representative torso primitive (the torso primitive spans three skeletal unit lengths). This image size dependency allows the particle filter systems to function in sequences of any capture resolution.

### 3.5.1 Computation of a particle's fitness

The objective is to find the parameterisation such that Region $A$ envelops the subject closely. Once a particle is cast onto the foreground binary image, all non-zero valued pixels are summed for both regions, and with a slight abuse of notation, are denoted as $\sum A$ and $\sum B$. A *net* result is then computed by adding the positively weighted $\sum A$ to the negatively weighted $\sum B$ (see Figure 3.5). Situations as per Figure 3.6, yield negative values and are therefore set to zero to prevent duplication of such unwanted particles.

The *net* is then normalised by $Area(A)$ to give a fitness score $\omega$ in the range [0,1]:

$$\omega = \frac{1}{Area(A)} \times \left\{ \begin{array}{ll} \sum A - \sum B & \text{if} \quad (\sum A > \sum B) \\ 0 & \text{otherwise} \end{array} \right\} \tag{3.13}$$

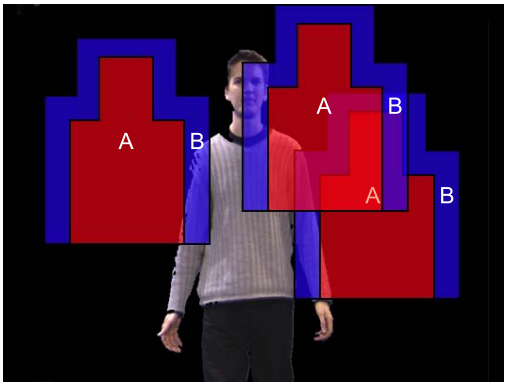
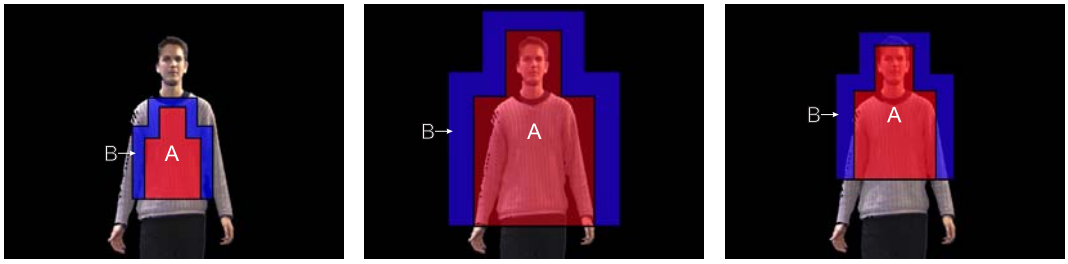



Figure 3.6: Torso primitive examples producing negative scores

This score is computationally inexpensive to calculate, thereby contributing to the real-time aspect of the particle filter tracking system.

Figure 3.7 shows three uniquely scaled torso primitives cast onto a half PAL ($384 \times 288$) foreground image. For the purpose of demonstrating the functionality of a particle filter system in a tracking framework, let Figure 3.7 represent a torso particle filter system whose population consists only of three such particles at initialisation. The scales of these primitives have been chosen to produce extremes of the fitness score $\omega$.




(a) (b) (c)

Figure 3.7: (a) Case 1: Low Score (b) Case 2: Low Score (c) Case 3: High Score

The table of Figure 3.8 provides the corresponding observations and calculations that illustrate the derivation of subsequent particles to estimate the posterior.

Case 1 of Figure 3.7 (a) shows a primitive where both Region $A$ and Region $B$ residing within the subject. Since $Area(A) = Area(B)$, $\sum A \simeq \sum B$, i.e. the number of non-zero valued pixels in Region $A$ approximately equals that of Region $B$. Their summation produces a small *net* value of 1000 ($\sum B$ is weighted negatively), which in turn yields a low score of $\omega \simeq 0$. In contrast, the primitive in Case 2 is over sized, and provides a large value for $\sum A$. However, the resulting $\omega$ is reduced substantially due to the normalisation by the large $Area(A)$. The highest value for $\omega$ arises in the situation demonstrated by Case 3 where the subject occupies the majority of Region $A$, with little overlap into Region $B$. Here $\sum A \simeq Area(A)$ and $\sum B \simeq 0$, which results in the highest score for $\omega$.

Following the same procedure as per Section 3.2, the prior weights $\pi_{t-1}^{(n)} = 1(\forall n)$ at $t = 0$. The scores are then updated using these prior weights, and a set of weights $\pi_t^{(n)}$ for the current iteration is obtained. The number of particles that are to be generated from each particle is then determined by $G_t^{(n)} = \pi_t^{(n)} \times N$. Again, only an integer number of particles may be produced, and this number is therefore rounded. The particle of Case 1 perishes, and Case 2 only generates one particle. The high scoring particle of Case 3 in turn generates two particles, thereby maintaining the population size of $N = 3$.

As per the graphical example of Section 3.2, the newly generated particles are offset from their parent particles by a Gaussian drift term; their scale is altered in a similar manner allowing the primitives to locate the optimal solution, and to shift in accordance with any movement.

### 3.5.2 Tracking the Torso

Following Figure 3.5 (a) and Figure 3.4, Region $A$ of the torso primitive spans $3 \times 2$ skeletal unit lengths enveloping the face (Sub-region $H$) and shoulders, and extending down to the waist where major spinal rotation occurs. The motivation

| N=3 | Case 1 | Case 2 | Case 3 | |
|---|---|---|---|---|
| $Area(A)$ | 30 000 | 120 000 | 53 200 | |
| $\sum A$ | 30 000 | 66 000 | 46 400 | |
| $-\sum B$ | -29 000 | 0 | -7 000 | |
| $net$ | 1 000 | 66 000 | 39400 | |
| $\omega = \frac{net}{Area(A)}$ | 0.03 | 0.55 | 0.74 | |
| $\pi_{t-1}$ | 1 | 1 | 1 | $\sum \pi_{t-1} = 3$ |
| $\omega' = \omega . \left(\frac{\pi_{t-1}}{\sum \pi_{t-1}}\right)$ | 0.01 | 0.18 | 0.25 | $\sum \omega' = 0.44$ |
| $\pi_t = \frac{\omega'}{\sum \omega'}$ | 0.03 | 0.42 | 0.56 | |
| $G = \pi_t \times N$ | 0.08 | 1.25 | 1.68 | |
| Generated particles | 0 | 1 | 2 | |

Figure 3.8: Computing the generation of new particles

behind using a coarse body-shaped primitive is twofold. Firstly, the generic shape is able to cope with human subjects of different height and mass. Secondly, the primitive can be subdivided into rectangular shapes, which is a requirement for the application of integral images (see section 3.6).

To recapitulate, the graphical example of Section 3.2 described a method of determining the maximum value of an unknown distribution where a series of particles was distributed randomly across the entire space. In the case of computer vision, and more specifically, in a HCI application, primary interest lies in locating the upper torso that occupies the central region of the captured image. A Gaussian distribution of $N$ torso primitives is therefore cast onto the foreground likelihood image with its origin at the centre of the image. To account for subjects that differ in size due to physical mass or distance to camera, the population of torso primitives covers a range of scales. The optimal distance from camera to subject requires that the upper torso and free arm movement remain in view – a torso

length of half the image height is adequate. The torso primitive scales are also initialised according to a Gaussian distribution, with a nominal scale of 0.5; this yields a primitive whose length covers half the image height $H$ as per Figure 3.9. That is, $3l_{su} = \frac{h}{2}$, where $l_{su}$ is the skeletal unit length.

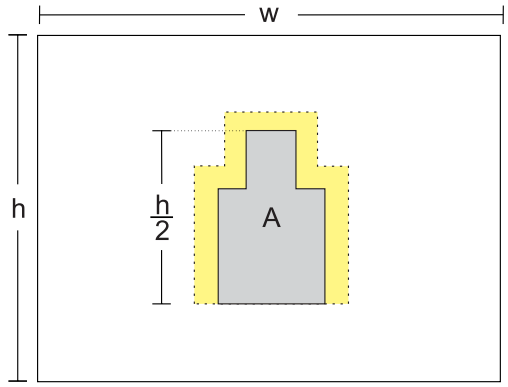



Figure 3.9: The nominal scale of the torso primitive

Figure 3.10 shows the initialisation of a torso particle filter which consists of five hundred particles. The procedure of determining this population size is shown later in the results of Section 3.7.4. The initialisation at Frame 1 shows the normal distribution of the particles across the image, centred at the centre of the image. Once all particle scores have been computed, the parameters of the top ten percent are averaged to create a *representative* particle (highlighted here in red). Since the initialisation nominal scale creates a particle half the length of the image height (see Section 3.9), the representative particle is undersized. The nature of the particle filter is inherently noisy as high scoring particles are duplicated and dispersed. Displaying the mean of the top ten percent therefore serves to smooth the output.

In the second iteration (Frame 2), the outlying particles of Frame 1 have already perished, with the majority of the population clustered around the subject. Particles are however constantly distributed around the perimeter of the subject to account for any significant movement. The larger the variance of the drift term,

the quicker the movement it can overcome. The subsequent iterations illustrate the scale adjustment of the particles, with the representative particle suggesting convergence as soon as Frame 30.

Computing $\sum A$ and $\sum B$ by the sequential summation of pixels is computationally expensive, especially when dealing with high resolution images, and large particle population sizes. Section 3.6 shows how the foreground binary image is converted to an Integral Image, which reduces the computational overhead of $\sum A$ and $\sum B$ dramatically, thereby contributing significantly to the real-time objective.

In addition to computing the fitness score for each particle, from [4], an overall particle filter system efficiency $\alpha$ is also determined for each iteration according to

$$\alpha = \frac{1}{N}\sum_{n=1}^{N}(\omega'^{(n)})^2 \qquad 0 \leq \alpha \leq 1 \tag{3.14}$$

where $\omega'$ is the updated observation. The greater the number of particles that provide strong observations, the greater the filter efficiency. Should this efficiency fall below a pre-defined threshold, the particle filter is re-initialised such that different subjects moving on and off the camera view can be detected. The same applies to the hand filters where the hands often move out of view.

### 3.5.3 Tracking the face and hands

Using the representative particle of the converged torso filter, the position and scale of the subject's upper body are extracted. In Figure 3.11, the face, left hand and right hand filters are initialised according to normal distributions centred at $H$, $LH$ and $RH$ respectively, each with a nominal scale equal to one third of the torso scale. The dashed circles around these centres represent the variance of the particle distribution. Seeing that the hands are likely to move considerably faster than the face, a broader distribution of particles is required to maintain

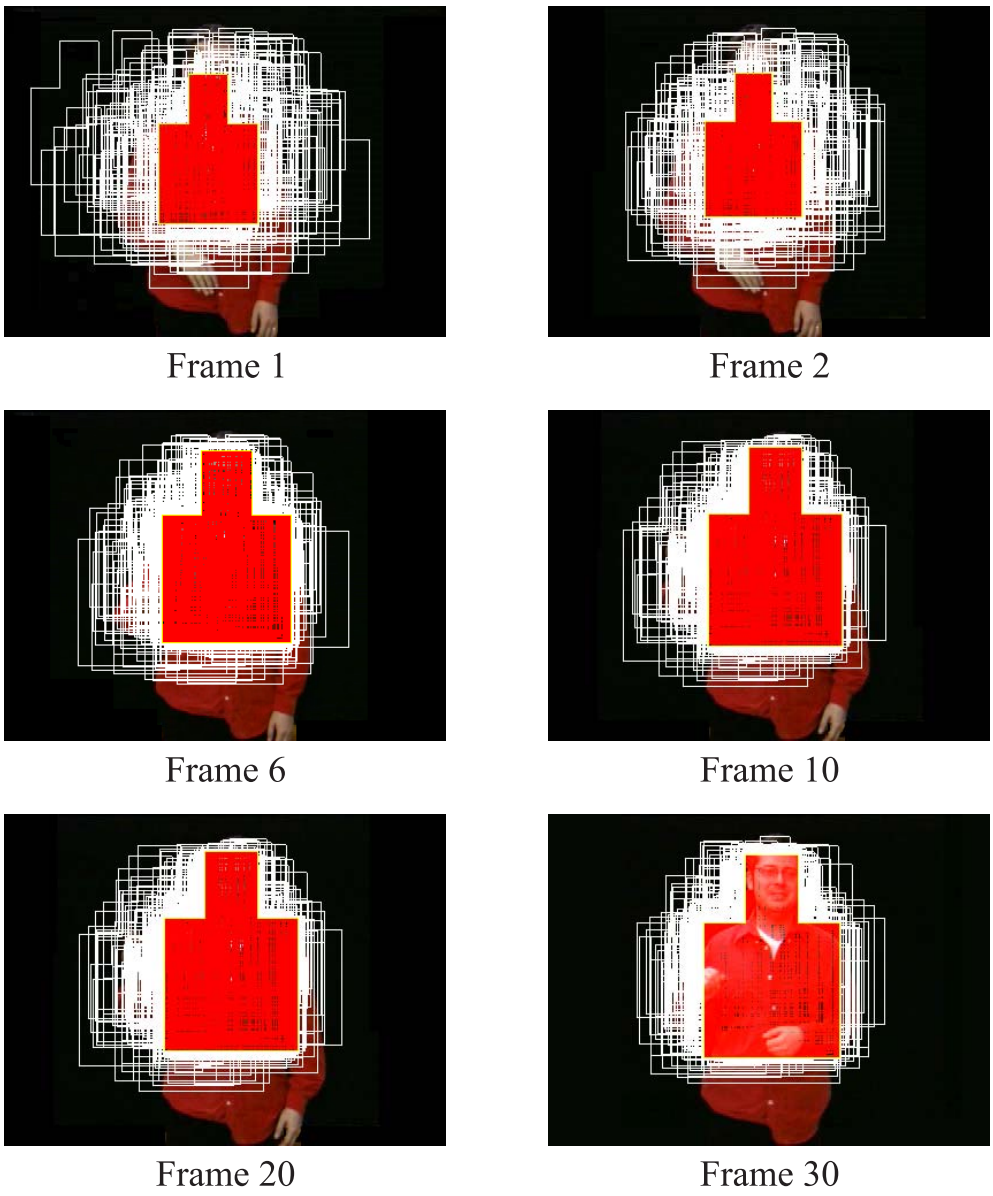



Figure 3.10: Torso particle filter initialisation and convergence

successful tracking of these objects. The locations of Regions $LH$ and $RH$ have been chosen such that hands in a relaxed pose can be detected easily. Following the Renaissance subdivision, these are found to be 1.5 skeletal unit lengths below the representative torso primitive.

Robust tracking of the torso requires that the subject be segmented from the

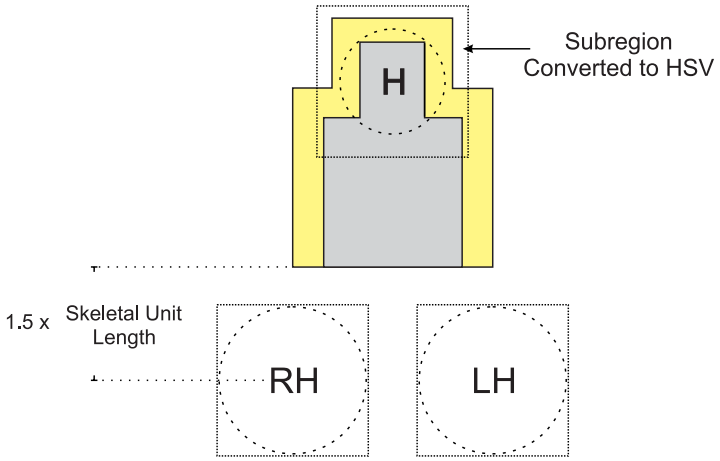



Figure 3.11: Initialisation locations for the face and hand filters

background. Similarly, tracking of the face and hands requires that they be isolated from the rest of the image; this is achieved by segmenting skin tone regions from the foreground image.

Skin colour models have been used in various colour spaces including normalised RGB [94, 86, 18], YCbCr [46, 82], HSV [101, 16, 80] and HSI [45]. However, if Lambertian reflectance [108, 32] is assumed, intensity can be separated from chromaticity as in CbCr [17, 7] and HS [95] such that the observed object in these colour spaces offers a degree of invariance to lighting conditions. All these colour spaces are nonetheless particularly useful as skin colour across different races forms a tight cluster in each of these spaces. A recent colour space study by Gasparini et al [34] offers a noteworthy set of skin detection results using the aforementioned colour spaces.

In this research, the HS (Hue, Saturation) colour space has been selected to represent skin. Appendix B provides a brief explanation of the HSV hex cone, and the RGB to HS conversion algorithm. Following Figure B.1 of Appendix B, Hue and Saturation exist in a polar plane, and according to [9], Hue ranges from approximately 355° to 40°. This 'wrap around' is problematic when training a

statistical skin model, and two solutions are presented. The Hue-Saturation polar coordinates of each pixel can be projected onto the Cartesian plane following $x = r\cos(\theta)$ and $y = r\sin(\theta)$ where $r$ is the Saturation, and $\theta$ the Hue. Alternatively, to conserve computation time, the skin model can be learned from the HS model directly by excluding $H > 50$. Due to the minimal presence of skin pixels with $355° < H \leq 360°$, their exclusion produces a slightly less accurate model that proves to be adequate for skin detection.

With an estimate of the position and size of the face (determined by Region $H$ of Figure 3.11), a set of parameters for the subjects's skin tone can be determined. Statistical methods have been employed to make use of two bivariate Gaussian skin models in the Hue-Saturation colour space: a primary generic model $\mathsf{S}_1$, and a secondary subject specific model $\mathsf{S}_2$. $\mathsf{S}_1$ is created from a selection of various faces from natural images. $N$ pixels are sampled, each represented by $\mathbf{x_i}$, a two-dimensional feature vector consisting of that pixel's corresponding Hue and Saturation value. The mean $\boldsymbol{\mu_1}$ of the sampled skin data is then obtained by:

$$\boldsymbol{\mu_1} = \frac{1}{N}(\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \ldots + \mathbf{x}_N) \tag{3.15}$$

With $\mathbf{x}_i$ and $\boldsymbol{\mu}_1$ in column vector form, the skin model $\mathsf{S}_1$ is determined by the $2 \times 2$ covariance of the samples:

$$\mathsf{S}_1 = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu}_1)(\mathbf{x}_i - \boldsymbol{\mu}_1)^T \tag{3.16}$$

This skin model is obviously far too general for use in robust segmentation, but it does however exclude pixels that are obvious outliers. Once the torso filter converges, a subject specific skin model $\mathsf{S}_2$ is built from Region $H$, using the remaining inliers determined by $\mathsf{S}_1$. This is achieved by firstly determining $P(\mathbf{Y})$, the probability that the feature vector of a newly sampled pixel $\mathbf{Y}$ is skin. To do this, we firstly compute the Mahalanobis distance [85] $md(\mathbf{Y}, \mathsf{S}_1)$ between $\mathbf{Y}$

and the generic skin model $\mathsf{S}_1$. This distance provides the number of standard deviations that the vector $\mathbf{Y}$ lies from the mean of the dataset $\mathsf{S}_1$

$$md(\mathbf{Y}, \mathsf{S}_1) = \sqrt{(\mathbf{Y} - \boldsymbol{\mu}_1)^T \mathsf{S}_1^{-1} (\mathbf{Y} - \boldsymbol{\mu}_1)} \tag{3.17}$$

Again, $\mathbf{Y}$ and $\boldsymbol{\mu}_1$ are in column vector form, and yield a 1 dimensional (scalar) $md(\mathbf{Y}, \mathsf{S}_1)$. Using this Mahalonobis distance, $P(\mathbf{Y})$ can now be determined:

$$P(\mathbf{Y}) = \frac{1}{2\pi |\mathsf{S}_1|^{\frac{1}{2}}} e^{\left(-\frac{1}{2} md^2\right)} \tag{3.18}$$

Should $P(\mathbf{Y})$ provide a high probability, $\mathbf{Y}$ is added to the new skin sample set. Determining the skin probability on a per pixel basis is however computationally expensive, and a similar model can be built using only the Mahalanobis distances. Chebyshev's Theorem [65] states that for $k > 1$, at least $1 - 1/k^2$ of the data set, regardless of shape, will fall within $k$ standard deviations of the mean. For example, at least 88.8% will fall within $\pm 3$ standard deviations. This percentage can be further increased if a symmetrical bell-shaped distribution is assumed, where the Empirical Rule [111] states that 99.7% of the data elements are within $\pm 3$ standard deviations of the mean. Taking advantage of this knowledge, should $md(\mathbf{Y_i}, \mathsf{S}_2)$ be less than three standard deviations, the pixel is considered to be skin. Once all appropriate pixels in Region $H$ have been added to the sample set, the subject specific skin model $\mathsf{S}_2$ is created as per equations 3.15 and 3.16.

The rectangular subregions (indicated in Figure 3.11) of the foreground image are then converted to HSV. Using the subject specific skin model $\mathsf{S}_2$, and assuming that the subject's skin is the same for the face and hands, all pixels in these subregions are tested for skin. Three binary images are then created from the skin pixel tests, where '1' denotes skin. The face and hand filters are finally initialised on these binary images and operate in exactly the same manner as the torso filter. As in the case with the foreground likelihood image of the torso, the face and hand binary images are also converted to Integral Images to improve speed performance.

## 3.6 Integral Images for Real-Time Performance

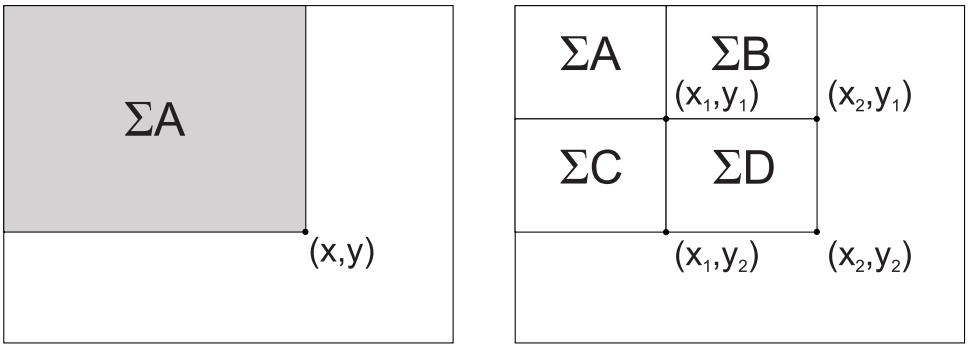
Rectangular features can be computed quickly using an intermediate representation called an *integral image* [104], which is also known as a *summed area table* in texture mapping [22]. Figure 3.12 (a) represents an integral image, where the value of the integral image at point $(x, y)$ is equal to the sum of all the pixels of the original image, above and to the left of $(x, y)$, inclusive:

$$II(x, y) = \int_{i=0}^{x} \int_{j=0}^{y} I(i, j) \mathrm{d}j \, \mathrm{d}i \tag{3.19}$$

where $II$ is the integral image, and $I$ is the original image. In the case of a discrete image, $II(x, y)$ is approximated by

$$II(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i, j) \tag{3.20}$$

With a slight abuse of notation, this is expressed in shorthand as $II(x, y) = \sum A$.




(a) (b)

Figure 3.12: Integral Image Computation

The entire $II$ can be computed in this manner for all(x,y), however it is more efficient to determine the integral image incrementally, rather than repeatedly summing the previous pixels:

$$\forall x, y \; II(x, y) = I(x, y) + II(x, y - 1) + II(x - 1, y) - II(x - 1, y - 1) \tag{3.21}$$

The pixel value sum for Regions $B$ and $C$ are computed as $\sum B = II(x_2, y_1) - II(x_1, y_1)$, and $\sum C = II(x_1, y_2) - II(x_1, y_1)$. $\sum D$ can therefore be computed as

$$\sum D = \int_{i=x_1}^{x_2} \int_{j=y_1}^{y_2} I(i,j)\mathrm{d}j\ \mathrm{d}i \equiv II(x_2, y_2) - II(x_2, y_1) - II(x_1, y_2) + II(x_1, y_1) \tag{3.22}$$

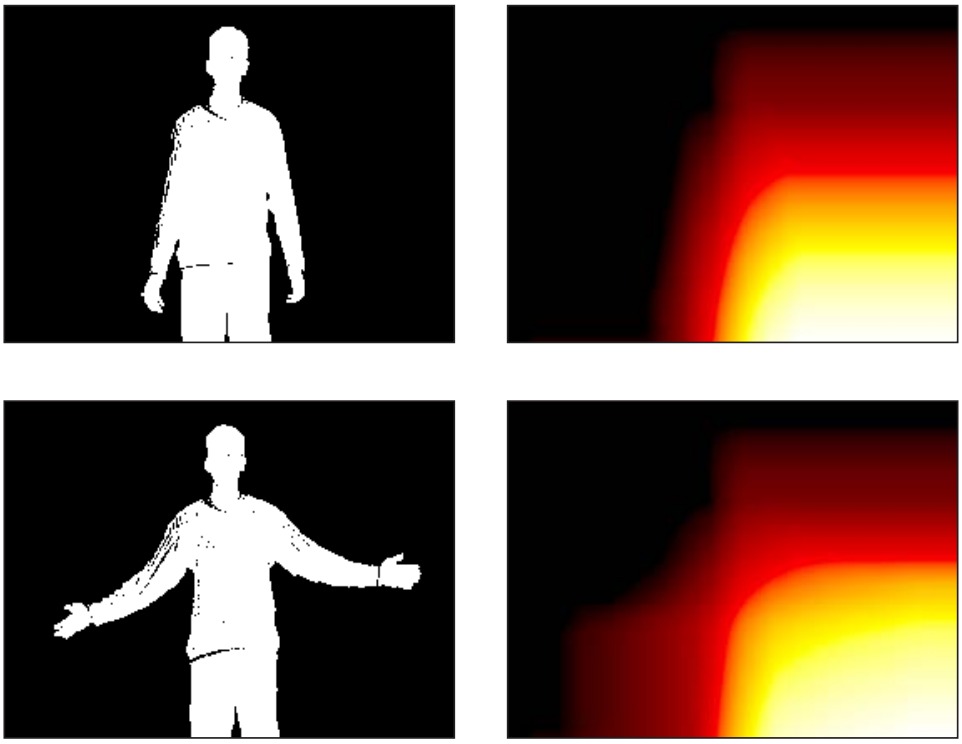
Using an integral image, any rectangular sum can therefore be computed in a maximum of three mathematical operations. However, due to the large overhead of creating the integral image, its use is only beneficial if multiple rectangular features are to be computed. This is clearly the case with the aforementioned particle filter systems which together extract 2 500 rectangular features per frame.

For ease of visualisation, Figure 3.13 (a) provides examples of foreground binary images, with a graphical representation of the corresponding integral images in Figure 3.13 (b).

As previously discussed, the body part primitives of Section 3.5 have been constructed from rectangular shapes such that their sums can be computed using these integral images. Considering the binary image in Figure 3.14(a), $\sum A$ of the torso primitive can be computed by sequentially summing all pixel values in this region. Once this binary image is converted to an integral image however, the same torso primitive is divided into two rectangular regions, with corresponding vertices 1 to 4, and 5 to 8. Following the same process outlined in Equation 3.22 $\sum A$ can now be computed as:

$$\sum A = II(3 - 2 - 4 + 1) + II(7 - 6 - 8 + 5) \tag{3.23}$$

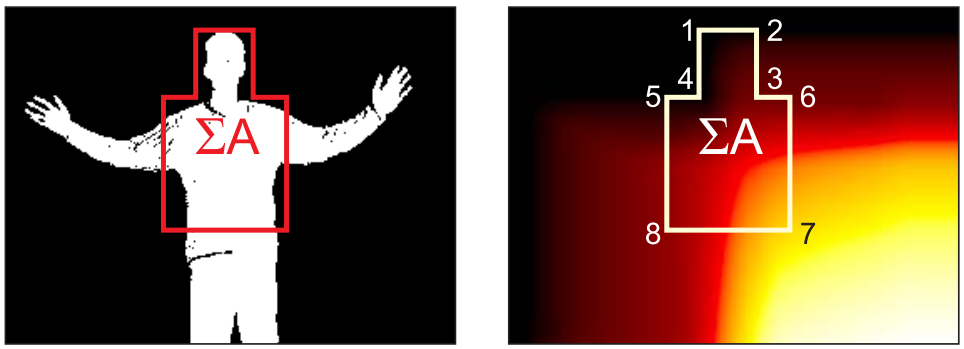
which consists of a mere 7 mathematical operations. The outer Region $B$ is also divided into rectangular regions, and $\sum B$ is calculated in the same manner. This also means that the algorithm is constant time as the scale of a primitive does not change the complexity of the calculations. Section 3.6.1 provides an



(a) (b)

Figure 3.13: (a) Binary image (b) Corresponding integral image

illustrative example showing the benefit of using an integral image with large particle population sizes.




(a) (b)

Figure 3.14: Computation of $\sum A$ using an integral image

### 3.6.1 Integral image benefit

Experimentation has determined that a camera capturing frames sized at half PAL ($384 \times 288$) contain sufficient information for robust tracking of the upper body. The subject of Figure 3.14 is sufficiently close to the camera such that all relevant parts can be tracked. The representative torso particle of this example occupies approximately 17 400 ($120 \times 120 + 50 \times 60$) pixels, a typical size when the full upper body is in camera view.

The computation of $\sum A$ in the standard manner therefore requires 17 400 addition operations. This number is doubled to 34 800 as $\sum B$ is also to be obtained for the computation of the particle fitness. For illustrative purposes, it is assumed that the average particle size is similar to that of the representative particle. The torso filter population size consists of five hundred particles and therefore requires a total of 17.4 million ($34\,800 \times 500$) operations. Should the subject move closer to the camera, thereby occupying more of the image space, the computational cost is further increased.

Making use of the integral image, $\sum A$ and $\sum B$ are determined using 7 operations each. Following Equation 3.21, the binary to integral image conversion process entails approximately 330 000 ($384 \times 288 \times 3$) operations. The initial overhead is large, but only occurs once per frame. In addition, the computation of $\sum A$ and $\sum B$ is unaffected by the size of the subject. For a population of 500 particles, the total number of required operations is approximately 337 000 ($330\,000 + 500 \times 14$), 0.02 percent of that of the standard summation method.

Using both methods, Figure 3.15 shows the total number of operations required to compute the fitness for an increasing number of particles. The plot of the sequential summation method offers a steep ramp, while that of the integral image method is virtually flat. The intersection of the two curves shows the

break-even point where the benefit of the integral image supersedes the standard summation method – processing time is already conserved when the population consists of ten particles.

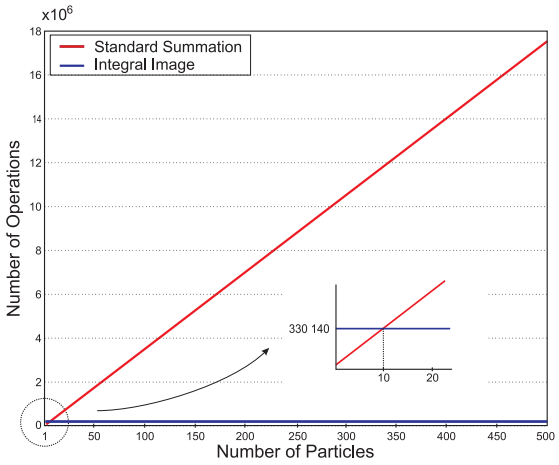



Figure 3.15: Number of operations using standard summation versus an integral image

## 3.7 Results

### 3.7.1 Chroma-keying

The greatest limitation of chroma-keying is that a controlled studio environment with special screening is required. Furthermore, the subject cannot wear clothing of similar colour to the background screen. Doing so will cause those sections of the subject to be labelled as background, making the tracking algorithm useless.

The threshold $T$, as presented previously in equation 3.12, can range from 0 to 512, and varies according to different lighting conditions. Experimentation using the facilities provided at the CVSSP of the University of Surrey has shown that a

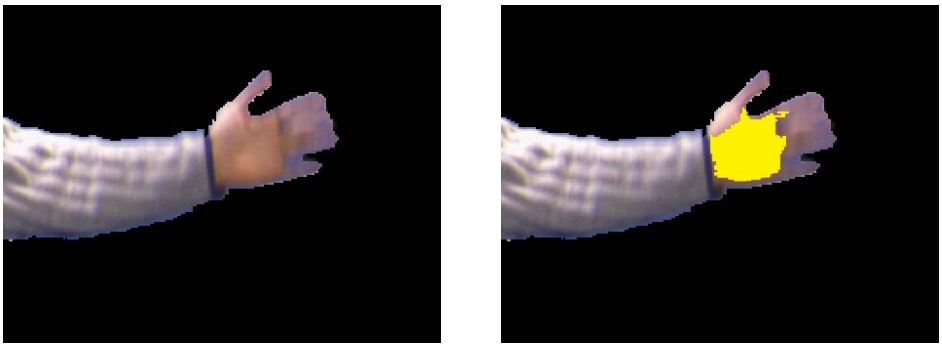
value of $T \approx 50$ is the most successful in determining predominantly blue pixels, i.e. the blue background.

*Blue spill* is a phenomenon whereby the outer regions of the subject reflects blue light due to the surrounding screens. It is particularly noticeable around reflective skin regions due to the oils secreted by the pores. These regions of blue spill are recognised as background and are removed, thereby altering the size and shape of the face and hands. A morphological dilation could be performed to reclaim this information, but skin segmentation still fails due to the influence of the blue spill.

The blue background offers yet another difficulty when the hands become semi-transparent due to motion blur. Blue is added to the skin colour, making segmentation poorer. The tracking algorithm is essentially reduced to identifying skin-like blobs; these detected regions still prove to be of sufficient size for effective tracking. Figure 3.16(a) shows the segmented arm using chroma-keying; the presence of blue in the skin due to motion blur is also subtly present. Figure 3.16(b) highlights the detected skin in yellow, and it is clearly apparent that there is insufficient information to detect hand shape or rotation. Even if the segmentation was more robust, detection of hand rotation is only viable if the subject keeps their hands open which is seldom the case. The square design of the hand primitive is therefore more than adequate to detect and track the subject's hands.

### 3.7.2 Adaptive background segmentation

Using an adaptive background segmentation offers three main advantages. Firstly, use of the tracking algorithm can be extended to a variety of indoor and outdoor environments. Secondly, although the background still affects the colour of the motion blurred hands, the inclusion of natural world colours, as opposed to those



(a) (b)

Figure 3.16: (a) Destruction of blue spill and motion blur (b) Resulting detected skin tone

of a saturated blue screen, have less effect on skin segmentation. This results in larger segmented skin regions, which increases the tracking robustness. A third advantage is that the subject is no longer as restricted in terms of clothing.

If not cautious with the learning rate of the adaptive segmentation, the subject can be included in the background model in a short space of time. Seeing that the objective application is HCI, this is of special concern as the subject is likely to stand in front of the camera for an extended period of time. Figure 3.17 provides a few frames from a sequence with a particularly fast scene learning rate. In frame 120, the black and white image on the left represents the foreground binary image of the image on the right. Since the video is captured at 30 frames/second with a USB2 web camera, this corresponds to a duration of four seconds. Due to the fast learning rate, the scene has been predominantly incorporated into the background model, with the presence of a little noise (mainly due to the poor quality of the web camera). At frame 500, the filter systems are initialised, and detection proves to be successful. Consulting the binary image more closely, it is evident that the segmentation is not very clean due to the lack of contrast between the previously learned background model and the new foreground object. The coarse shaped body primitives are nonetheless able to cope with these 'gaps'

in the foreground model. In frame 831 however, the lack of movement of the subject's torso causes it to be incorporated into the background model due to the fast scene learning rate. Frame 859 illustrates the torso and face filters drifting to the side of the subject, where the arm is still part of the foreground model due to its movement. Tracking of the torso and face in Frame 1471 has completely failed, with the torso filter converging on the arm, and the face filter on the hand. In the final frame, once the subject moves out off camera, the binary image now indicates the presence of a foreground object. However, seeing that the initial background model is never destroyed, this residual foreground is re-incorporated into the background quickly.

The motivation behind learning the scene quickly is that a subject will want to run the application and begin using it immediately. A slow learning rate obviously requires that the algorithm be running for a considerable period of time to learn the scene completely. The obvious compromise makes use of a self-adjusting learning rate – the application initiates with a fast learning rate, which is then slowed dramatically when a subject comes into view.

### 3.7.3 Determining the Optimal Population Size

Determining the optimal particle population size plays an important role in finding a balance between tracking parts robustly/smoothly and tracking in real time. To illustrate how a particle population size affects tracking, a sequence of a subject performing rigorous gestures (Figure 3.20) was used for test purposes. The test focused on the hand filters as they are the most susceptible to the particle population size; they track the fastest moving body parts. Figure 3.18 shows the mean pixel error of the left and right hand particle filters versus an increasing number of particles. From the curve, the system fails to converge when the population size is less than approximately thirty particles, and the respective filters

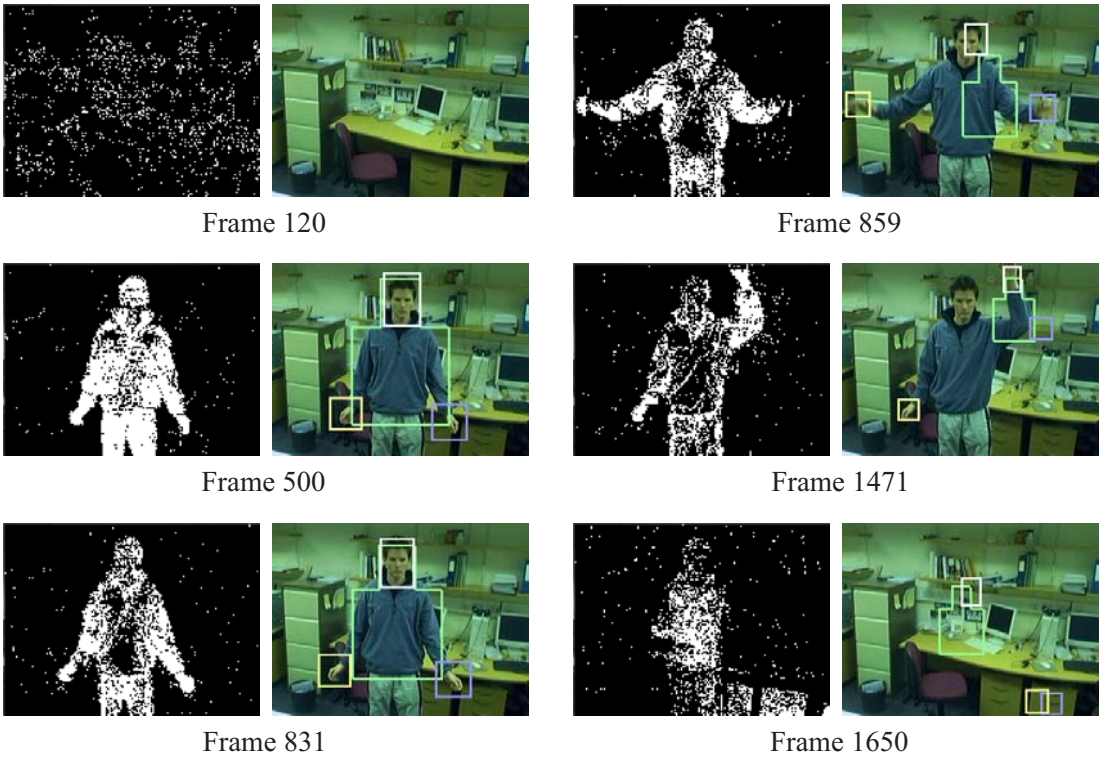



Figure 3.17: The effects of using a high scene learning rate

remain at the hip initialisation point. The error reduces rapidly as the number of particles increases, when at approximately three hundred particles, the hands are tracked throughout the sequence. This also corresponds to the point of inflection of the curve, and has a mean error of 3.17 pixels.

Figures 3.19 (a) to (d) provide the corresponding hand tracking error plots from the sequence, each generated using a different number of particles. Figure 3.19 (e) in turn shows the mean hand velocity determined using the ground truth, and indicates fast chaotic motions. Comparing the error against the velocity validates that high tracking errors co-occur with the faster, frantic actions. The population of three hundred particles provides a robust tracker as the target is lost for a maximum of five frames before recovering. This is not particularly noticeable to the naked eye as this time frame equates to less than one fifth of

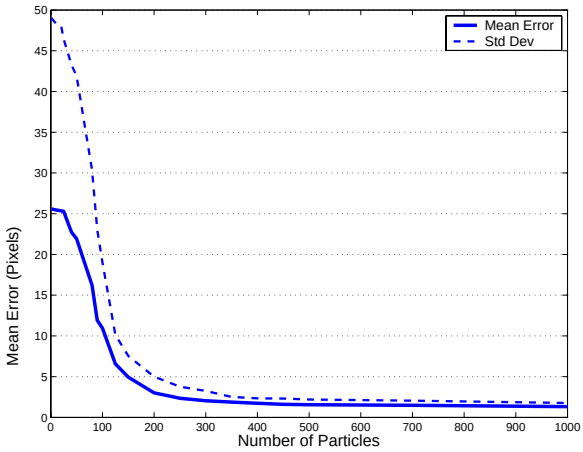


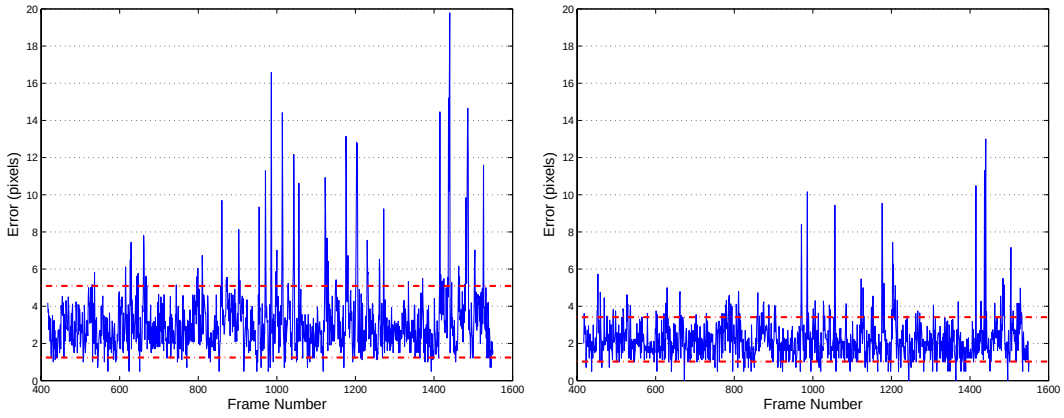

Figure 3.18: Determining the particle population size

a second as the system runs at above real-time. Although the hands are tracked throughout the sequence with a small mean error, the output is however extremely noisy. The jitter around each part is noticeable, and proves to be inadequate for gesture recognition purposes. Four and five hundred particles produce smaller mean errors of 2.2 and 1.8 pixels respectfully, and have significantly smoother outputs, with fewer error spikes.

Although particle computation itself is constant time due to the use of the integral image, the sorting of the lists in which their parameters are stored is not. Further increases in the population size only provide a slightly smoother output and lower mean error. For example, a population of one thousand particles produces a mean error of 1.3 pixels at the cost of a noticeable latency. A population of five hundred particles has therefore been selected for use in each of the particle filters as the tracking is sufficiently robust, and the output adequately smooth.

### 3.7.4 Tracking Robustly in Cluttered Scenes

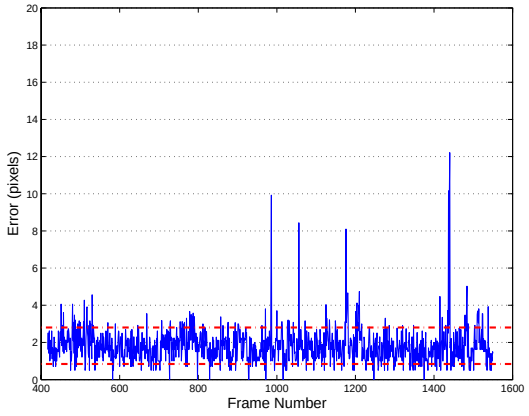
With respect to the live application of the particle filter tracking system, the scene is learned within approximately five seconds after initialisation. It is important



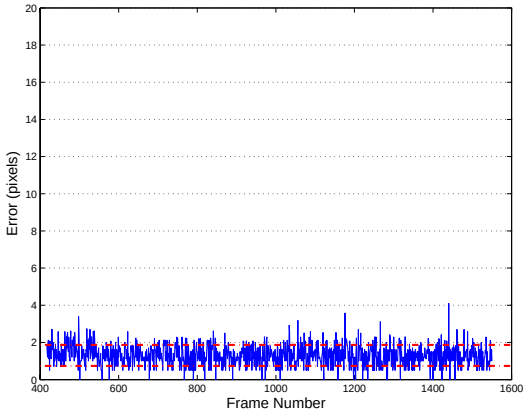

(a) 300 particles



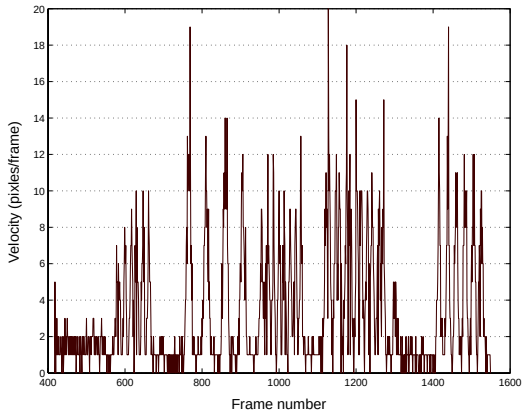

(b) 400 particles




(c) 500 particles




(d) 1000 particles




(e) Mean hand velocity

Figure 3.19: Hand tracking error through a video sequence using the hand filters

that no subjects wishing to use the system are present at initialisation as they will be included in the background model. After the scene has been learned, the torso particle filter is initialised at the centre of the image. Should a subject not be in camera view at any time, the resultant particle population efficiency will be low; should this efficiency fall below a defined threshold, the filter is re-initialised on each frame until a subject moves onto the scene.

The respective filters of the full system each consist of five hundred particles, and include a Gaussian white noise drift term to compensate for movement. The noise variance of the hands is greater than that of the torso and face in order to detect their faster movement. The mean of the top 10 percent of particles from each filter is selected to represent the likely position and scale of each body part.

Figure 3.20 shows the same sequence used in Section 3.7.2 where a fast scene learning rate caused tracking to fail; here however, the tracking is more robust due to the variable learning rate. This sequence offers a good example of the type of data likely to be used for HCI where only the upper body is of concern. In frame 336, there is still no evidence of a representative torso particle, however as the subject moves closer to the centre in frame 378, the filter offers sufficient efficiency to provide the first representative particle. The system then quickly converges onto the subject in less than one second. At this point, the subject specific skin model from the face region is learned, and the face and hand filters are initialised in the locations described in Section 3.5.3. Due to the reasonably accurate initial estimates, the hand filters converge in less than ten frames.

It is evident how tracking of the hands remains robust even when when they are in close proximity (frame 1125), and when distorted with motion blur (frame 1437). From frame 1595, the subject begins to move off scene, and the filters continue to track the subject as far as the image border. If the subject is standing less than half a skeletal unit length from the image border, it is assumed that the corresponding hand is out of view. At this point, the associated hand filter is


Figure 3.20: Tracking a subject in a complex, cluttered scene

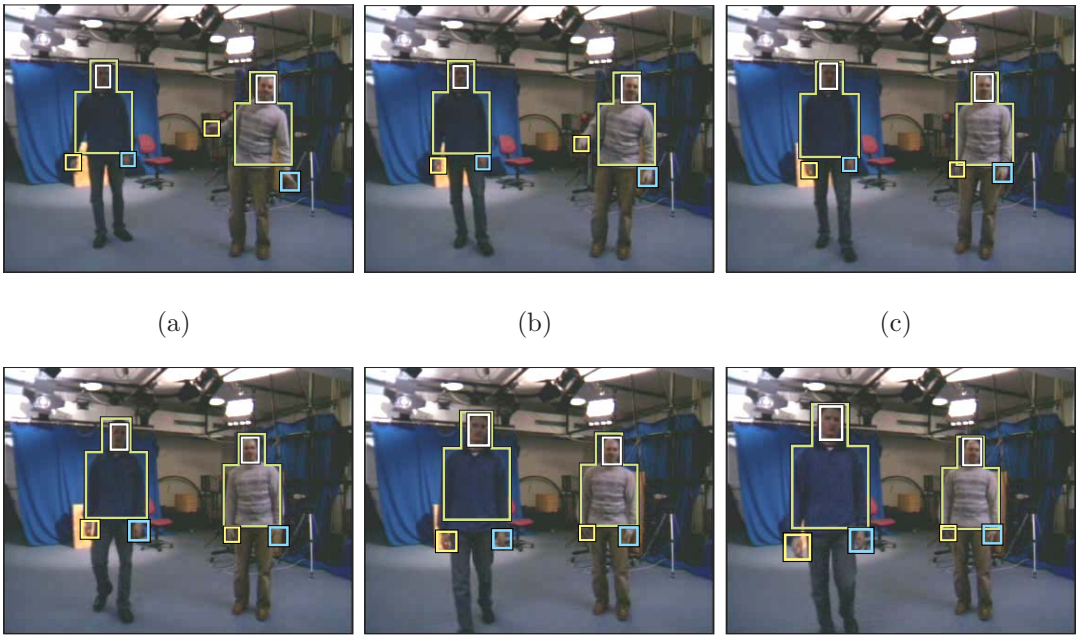
disabled to prevent multiple filters from attempting to track the same object. The same applies when the subject is standing very close to the camera – both

hand filters are disabled. If the subject moves back into view, the respective hand filters are initialised as before.

The hierarchical link between the torso and face filters restricts the distribution of face particles to exist within one skeletal unit length of the head region (Region $H$) of the representative torso particle. This restriction assists in preventing face particles from drifting onto the hands, yet is lenient enough to ensure that face tracking continues when the subject is bending forward as in frame 1603, or when tilting to the side. In frame 1624, the face filter has been re-initialised, while in frame 1630 the torso filter returns to a state of continuous re-initialisation. The subject specific skin model is discarded, and will be re-learned when a new subject moves into the scene.

The same methodology of tracking a single subject can be extended to detect and track multiple subjects. Figure 3.21 shows two subjects moving in a cluttered scene, with separate particle filter systems governing each. Following background segmentation, a k-means clustering of the foreground pixels is performed to extract an initial estimate and size for the position of each subject. In this particular example, $K$ was selected by hand to be the number of subjects to track. Alternatively, a connected component analysis of the binary foreground image can be used to estimate the number of moving objects.

$K$ distinct particle filters are run concurrently, each initialised with Gaussian distributions from the $K$ component estimations. In terms of modelling the posterior, this makes little difference but it provides a convenient partitioning of the population samples for monitoring and re-initialisation. In order to cope with subject occlusions, a minimum separation between targets is enforced [86] by comparing the distributions of each torso particle system to detect when more than one particle filter has converged upon a single solution – at this point, k-means is conducted again, and the filter is re-initialised according to the $K$ estimations. From the image sequence, it is also apparent how the torso filter of



(a) (b) (c)

(d) (e) (f)

Figure 3.21: Tracking multiple subjects in a complex, cluttered scene

each system contends with the two differently sized subjects, and their forward movement with respect to the camera.

## 3.8 Conclusions

Basic knowledge of a human figure schematic has allowed for the design of body part primitives to coarsely estimate body part shape and locations. These estimates, combined with the invaluable speed benefits of integral images, have made a real-time particle filter system achievable. Tracking has proven to be even more robust as the particle population sizes can be increased significantly with minimal influence on system performance. Aside from the initial overhead of the integral image computation, the system is not inhibited by image resolution – a higher quality capture would also allow for the analysis of face and hand shape which

could further the capabilities of the HCI system.

The current implementation of the integral image precludes detection and tracking of a tilted torso or face. This is however acceptable as the objective application lies in gesture recognition, and we assume the subject remains upright. Identifying face rotation however could prove useful in gesture recognition where a tilt of the head may be used to suggest a question. Later sections introduce alternative methods which can overcome this while still employing the benefits of the integral image. This encodes the shape of the object such that a template used to detect the object can be rotated. Determining the fitness of each rotated template is however more expensive as the fitness of each row of the template has to be computed and averaged.

The major shortcoming of the system is that a static background is required for background/foreground segmentation. A need therefore exists for a more robust method of detecting body parts in cluttered scenes. Section 5.2 describes the detection of body parts using trained body part detectors. These are then intelligently assembled into human configurations, and can operate without background/foreground segmentation as the approach focuses on modelling the foreground object rather than the background.

# Chapter 4

# Prior Data for Pose Estimation

This chapter presents the use of an *a priori* Gaussian Mixture Model (GMM) of frontal view upper body configurations to disambiguate the hands of the subject and to predict the likely position of the elbows. The same model is used again later in Chapter 5 to acquire a pose likelihood of an upper body configuration consisting of body part detections.

## 4.1 Gaussian Mixture Model Construction

Although the human body has a high number of degrees of freedom, the relative position of body parts are predictable to some extent. Furthermore, when body parts move, there is some dependency upon this relative position. If a model of this dependency were to be constructed then it could be used to both disambiguate hand detections as well as infer unseen structures from the image. To this end, two prior models $\phi$ ($\in \mathbb{R}16$) and $\psi$ ($\in \mathbb{R}20$) of body configurations have been built from 4600 representative examples of humans in image sequences performing deaf sign language and a variety of upper body movements. An example frame from a training sequence is shown in Figure 4.1. The parts of interest include

the shoulders (labels 2 and 3), hips (1 and 4), chin (5), hairline (6) , left hand (10), right hand (7), left elbow (9) and right elbow (8). Blue and yellow coloured gloves, green elbow markers and a black background were used to facilitate the ground truth labelling. A modified version of the particle filter system of Chapter 3 was used to record the body part positions rather than doing so meticulously by hand. The constituent 16D feature vectors of $\boldsymbol{\phi}_i$ are formed by concatenating the $x, y$ co-ordinates of eight parts, namely the shoulders, hips, chin, hairline and hands. Once constructed, $\phi$ is used to disambiguate the elbows. The 20D feature vectors of $\boldsymbol{\psi}_i$ are formed in the same manner, with the addition of the elbow positions. $\psi$ in turn, assists in estimating the elbow positions.

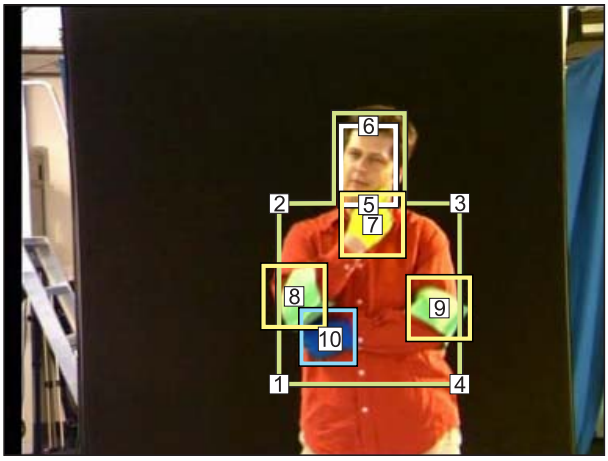



Figure 4.1: Ground truth labelling of body parts using a particle filter system

Assuming that the set of feature vectors can be described using a single Gaussian, $\psi$ is represented by the covariance of its constituent feature vectors:

$$Cov_\psi = \frac{1}{N-1}\sum_{i=1}^{N}(\boldsymbol{\psi}_i - \boldsymbol{\mu}_\psi)(\boldsymbol{\psi}_i - \boldsymbol{\mu}_\psi)^T \tag{4.1}$$

where $N$ is the number of examples, $\boldsymbol{\psi}_i$ is a single feature vector of the set, and $\boldsymbol{\mu}_\psi$ is the mean of that set. Once Principal Component Analysis (PCA) [52] has been performed on $Cov_\psi$, the projection of data onto the primary eigenvectors

(corresponding to the largest eigenvalues) provides a useful visualisation of the dataset, as shown in Figure 4.2. From the complexity of this figure, it is apparent that the manifold on which the data set lies is not represented well by a single Gaussian. It is therefore intuitive to represent the training set using a Gaussian Mixture Model (GMM); the optimal number of components must however be selected.

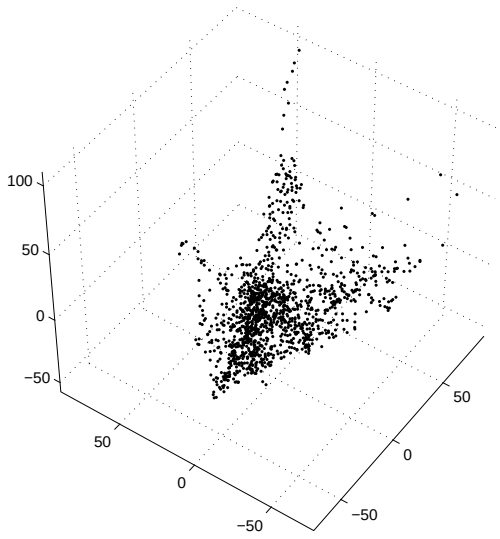



Figure 4.2: Dataset $\psi$ projected onto the first three eigenvectors

Following [9], the cost function from K-means can be used to estimate the optimal number of components, $K$ . The natural number of components to be selected is said to be the number for which further increases do not produce significant benefits in reducing the overall cost. From the corresponding cost function of Figure 4.3, this corresponds to the point of inflection on the curve, and $K$ is estimated to be in the region of 100 components.

Dataset $\psi$ is clustered accordingly, and $K$ $20 \times 20$ covariance matrices $Cov_{\psi,k}$ are formed from the examples in the corresponding clusters:

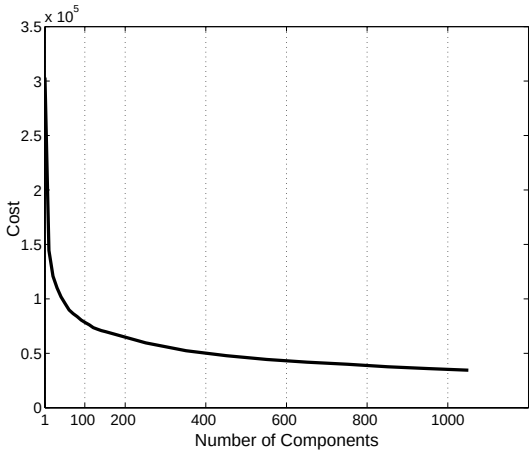



Figure 4.3: Cost versus number of components

$$Cov_{\psi,k} = \frac{1}{N-1}\sum_{i=1}^{N_k}(\boldsymbol{\psi}_{i,k} - \boldsymbol{\mu}_{\psi,k})(\boldsymbol{\psi}_{i,k} - \boldsymbol{\mu}_{\psi,k})^T \qquad (k = 1, \ldots, K) \qquad (4.2)$$

each of which is weighted according to the number of examples $N_k$ in the respective clusters. $\boldsymbol{\psi}_{i,k}$ represents the feature vectors and $\boldsymbol{\mu}_{\psi,k}$ the mean of the $k^{th}$ cluster. Figure 4.4 provides the resultant clustering of data set $\psi$.

Similarly, $\phi$ also proves to be non Gaussian, and is best represented by a GMM. $Cov_{\phi,k}$ is created in the same manner as per $Cov_{\psi,k}$, the main difference being that the constituent feature vectors are 16D as the elbow positions are excluded. With these two *a priori* data sets, the left and right hands can be disambiguated, and the undetermined elbow positions of the tracked subject can be inferred.

## 4.2 Disambiguating the Hands

Using $Cov_{\phi,k} \in \mathbb{R}16$ , the hands of the subject can be disambiguated in the event that the hands overlap temporarily, or if tracking mistakenly switches to the opposing hand.

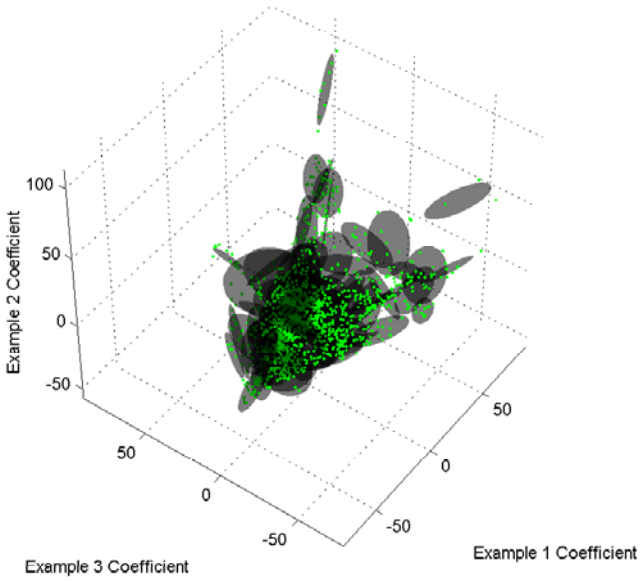



Figure 4.4: K-means clustering of $\psi$ with $K = 100$

Once the eight $(x, y)$ co-ordinates of the subject's body parts are determined, they are concatenated to form a pair of feature vectors ($\mathbf{y}'_j \in \mathbb{R}^{16}, j = \{1, 2\}$), where the co-ordinates of the hands are deliberately swapped in vector $\mathbf{y}'_2$. Here, the Mahalanobis distances, $md_{\phi,k}(\mathbf{y}'_j)$, (initially show in Equation 3.17) are defined as the measurements between vector $\mathbf{y}'_j$ and each component of the GMM. The correct configuration is that which minimises its distance to the model.

$$md_j = \min_{i=1}^{K} \left( md_{\phi,i}(\mathbf{y}'_j) \right) \qquad j = \{1, 2\} \tag{4.3}$$

Awkward poses, for example arms crossed over each other, yield a larger $md$, thereby indicating that the pose is unnatural. Should $\mathbf{y}'_2$ yield the smaller $md$, the hand filters are switched in order to correct the pose.

## 4.3 Estimation of Elbow Positions

Image cues for the detection of elbow positions are not apparent, and predictive methods need to be employed in order to offer a starting point with which to search the image space. Inverse kinematics prove to be cumbersome in 2D applications, and also offer multiple solutions as the arm length 'changes' due to perspective. This approach makes use of the aforementioned $Cov_{\psi,k}$ (created from feature vectors $\in \mathbb{R}20$), and offers a relatively accurate starting point for each elbow.

Eigen decomposition of $Cov_{\psi,k}$ yields eigenvector matrices $\mathsf{P}_{\psi,k}$ and corresponding eigenvalues $\mathbf{b}_{\psi,k}$. With $\mathbf{y}'$ as the hand-disambiguated 16D model of the tracked subject, the objective is to construct the vector $\mathbf{y}^{\mathrm{r}} \in \mathbb{R}^{20}$ which includes estimates for the elbow positions. As per Section 4.2, $md_{\psi,k}(\mathbf{y}')$ is calculated, and the $n^{th}$ component in the GMM that yields the minimum distance is selected. The elbow information is then removed from $\mathsf{P}_{\psi,n}$, giving a 16x20 matrix $\mathsf{P}'_{\psi,n}$. Similarly, the elbow positions are removed from the mean of the $n^{th}$ component $\boldsymbol{\mu}_{\psi,n}$, giving $\boldsymbol{\mu}'_{\psi,n} \in \mathbb{R}^{16}$.

In matrix form, a body configuration $\mathbf{y}^{\mathrm{r}}$ can be reconstructed from the data set mean $\boldsymbol{\mu}$ plus the weighted sum of the eigenvectors:

$$\mathbf{y}^{\mathrm{r}} = \boldsymbol{\mu} + \mathsf{P}\mathbf{b} \quad (\mathbf{y}^{\mathrm{r}}, \mathbf{b} \in \mathbb{R}^{20}) \quad where \quad -3\sqrt{b_i} < b < 3\sqrt{b_i} \tag{4.4}$$

A new set of projection weights $\mathbf{b}_{\mathrm{new}} \in \mathbb{R}^{20}$ can therefore be determined by rearranging Equation 4.4, and making use of the dimensionally reduced $\boldsymbol{\mu}'_{\psi,n}$ ($16 \times 1$ column vector) and $\mathsf{P}'_{\psi,n}$ ($16 \times 20$ matrix):

$$\mathbf{b}_{\mathrm{new}} = \mathsf{P}'^{T}_{\psi,n}(\mathbf{y}' - \boldsymbol{\mu}'_{\psi,n}) \qquad (\mathbf{b}_{\mathrm{new}} \in \mathbb{R}^{20}) \tag{4.5}$$

With this new set of weights, the model $\mathbf{y}^{\mathrm{r}} \in \mathbb{R}^{20}$ is constructed, and therefore provides estimates for the elbow positions.

$$\mathbf{y}^{\mathrm{r}} = \boldsymbol{\mu}_{\psi,n} + \mathsf{P}_{\psi,n}\mathbf{b}_{\mathrm{new}} \tag{4.6}$$

## 4.4 Results

As per the acquisition of ground truth data, colour coded body parts were used to measure the accuracy of the hand disambiguation and elbow estimation methods.

A data set of 7600 representative body configurations was extracted from a video sequence of a deaf signer. This data set was then separated into 4600 training examples and 3000 unseen test examples. K-means was conducted on the training set with $K \in \{1, 2000\}$, and PCA was performed upon corresponding clustered data. For each iteration of $K$, the test data set was then used to estimate the elbow positions as detailed in Section 4.3, and the test data ground truth was used to measure the pixel error between each elbow and its estimate. Figure 4.5 shows the mean and standard deviation error for the estimation of the left and right elbow positions. The higher error of the right elbow is due to the increased movement of the right arm as the signer in the sequences is right hand dominant. Again, the highest benefit in terms of error minimisation is in the region of 100 Gaussians, and corresponds to the cost function of Figure 4.3. Further increases reduce the generalisation of the model as it will eventually regress to a nearest neighbour approach [9].

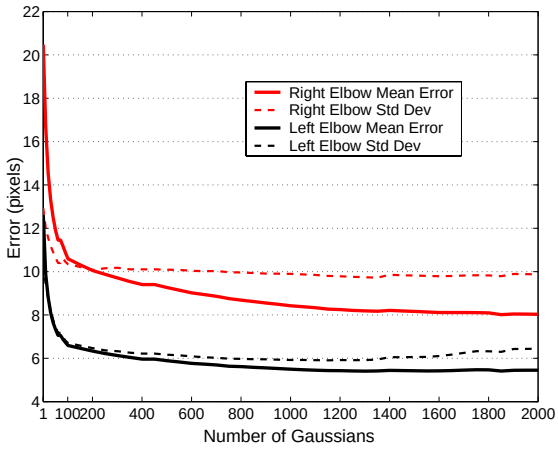
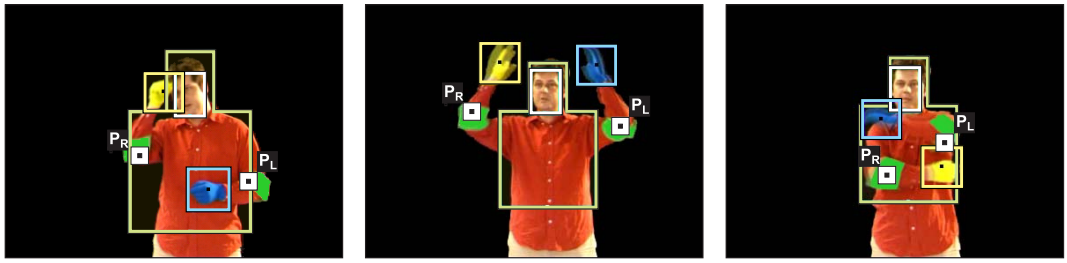



Figure 4.5: Error of the left and right elbow estimations

Figure 4.6 shows a test subject performing deaf sign language, with the torso, head and hands tracked as per Chapter 3. Using a GMM of one hundred components as previously determined, the hands were correctly disambiguated in 98 percent of the test examples using the hand disambiguation method (Section 4.2). The left and right elbows were estimated from the *a priori* model, the positions of which are indicated by $P_L$ and $P_R$ respectively. The elbow markers were used as a reference with which to measure the elbow estimation error; these were not used in the estimation.




(a) (b) (c)

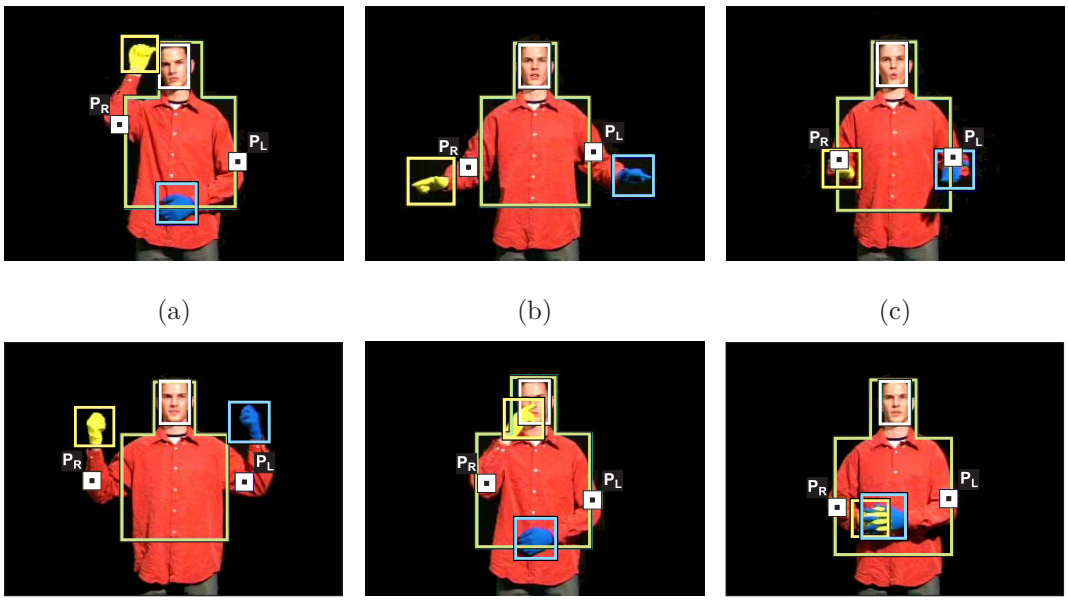
Figure 4.6: Error calculation of estimated elbow positions using elbow markers

Figure 4.7 shows the elbow estimates of another subject that is not wearing elbow markers.

## 4.5 Conclusions

Relatively accurate elbow positions have been estimated using an *a prior* mixture model on body configurations, thereby completing the upper body pose. Furthermore, in terms of gesture recognition and sign language recognition, it offers another reference point in terms of recognising gestures, and also allows for additional gestures in the vocabulary.

This chapter presented subjects wearing coloured gloves and a clean background




(a) (b) (c)

(d) (e) (f)

Figure 4.7: Estimation of elbow positions

to emphasise the elbow estimation. In Chapter 5 however, the elbow positions are estimated from natural images with cluttered scenes.

# Chapter 5

# Detection and Tracking of Humans by Probabilistic Body Part Assembly

The objective of this chapter is to robustly estimate the location and approximate 2D pose of humans in real world cluttered scenes. This is a challenging task as the shape and appearance of the human figure is highly variable. The problem is further compounded as people wear a variety of clothes, and skin tone varies with race.

Using an implementation of the Adaboost face detection method initially proposed by Viola and Jones [104], body part detectors for the face, torso, legs and hands have been created. The solution presented here makes use of a probabilistic framework to assemble the individual body part detections into a full 2D human configuration. A coarse heuristic is applied to eliminate obvious false detections, and body configurations are assembled from the remainder using RANSAC. The *a priori* mixture model of upper-body configurations created in Chapter 4 is used to provide a pose likelihood for each configuration. A joint-likelihood model is then

determined by combining the pose, part detector and corresponding skin model likelihoods. The assembly with the highest likelihood is selected by RANSAC [29], and estimates of the elbow positions are inferred.

## 5.1 Object Detection

Boosting is a general method that can be used for improving the accuracy of a given learning algorithm [91]. More specifically, it is based on the principle that a highly accurate or 'strong' classifier can be produced through the linear combination of many inaccurate or 'weak' classifiers. The efficiency of the final classifier is increased further by organising the weak classifiers into a collection of cascaded layers. This design consists of a set of layers with an increasing number of weak classifiers, where each layer acts as a non-body-part rejector with increasing complexity. A candidate subregion of an image is first passed to the simplest top layer for consideration, and is only moved to the next layer if it is classified as true by the current layer.

The object detector described in this section was initially proposed by Viola and Jones [104] and extended by Lienhart and Maydt [63] for Intel's computer vision library, OpenCV [47]. The reader is directed to Schapire's workshop paper [90] for an overview of the boosting approach to machine learning and to Viola and Jones' journal paper [106] for a full discussion of object detection using trained feature detectors.

### 5.1.1 Features

The object detection methodology employed here classifies images based on simple features. The features were derived from the Haar-like features used by Papageorgiou et al [77] and Mohan et al [73]. Viola made use of three forms of upright

rectangle features, indicated by Figure 5.1 (a), (c) and (e), where white regions have a positive weighting, and the black negative. Lienhart and Maydt extended this set by including ±45° rotated versions of the upright features (Figure 5.1 (b) and (d)), and by adding a 'centre-surround' feature (Figure 5.1 (f)). Lienhart demonstrates that improved detection rates are achieved by including these additional features. For the remainder of this Appendix, a *feature type* refers to any one of the aforementioned unique feature shapes, while a *feature* is a variant of that feature type in terms of location and scale within a bounding box.

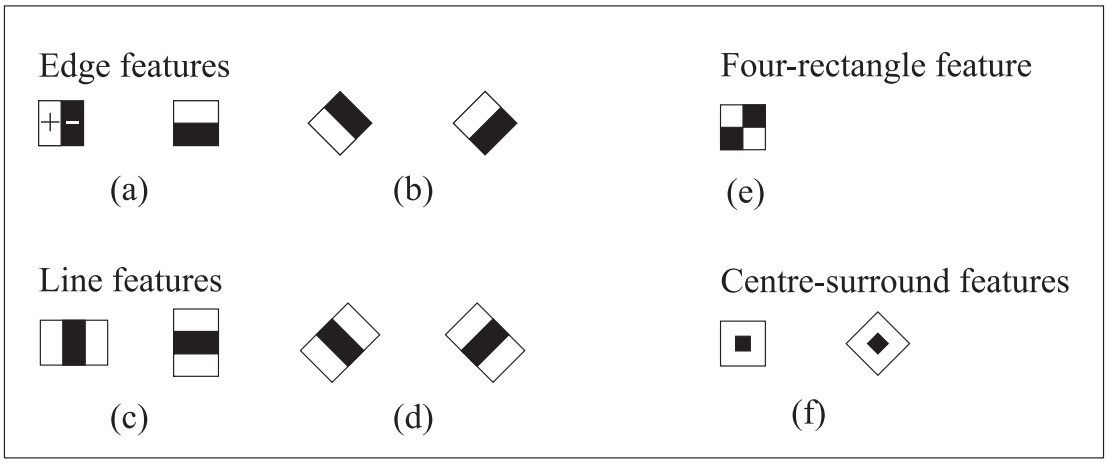



Figure 5.1: Examples of the rectangle features used for object detection

The value of a feature is computed by superimposing it onto an image, summing the pixels in the respective regions, and computing their difference. For example, a two region feature detector placed on the forehead with intense reflection will have a small value. Since the features are comprised of rectangular regions, integral images (see Section 3.6) can be exploited to compute their summations efficiently. In the case of the Lienhart's rotated ±45° features, two rotated integral images are computed by summing the original image diagonally, from top left to bottom right, and top right to bottom left.

With the aforementioned set feature types, a classifier is trained with a set of

sample views of a particular object, referred to as positive examples, each of which have a labeling of '1'. A set of arbitrary images that do not contain the object are also used for training purposes such that non objects can also be identified in order to resolve ambiguities. These training images are referred to as negative examples, and have a labeling of '0'. Generally, there are twice as many positive examples as negative examples, where the number of positive examples is typically between one and two thousand. The objects of the positive examples are cropped such that each example contains a similar view of the object. For example, the frontal view of the face is cropped with a square which extends from the top of the eyebrows to the lower lip. Including images that exhibit a slight rotated view of the object is however advised as this allows for a level of leniency when attempting to detect an object. All positive and negative examples are then scaled to the same size, typically $20 \times 20$ pixels. Larger training images do improve detection performance, however at the cost of hardware resources and time requirements during training.

### 5.1.2 Training the Classifier

The hypothesis of a boosted detector is that a small number of features can be combined to form an effective classifier. Selecting the appropriate features is the main challenge, and AdaBoost is used to select these features, and to train the classifier. The AdaBoost learning algorithm is used to boost the classification of a simple learning algorithm or *weak learner*, and it does this by combining a collection of weak classification functions into a single, stronger classifier. The weak learner is restricted to using a single feature and is designed to select the single rectangle feature that best separates the positive and negative examples.

There are essentially two main variants of the constructed strong classifier. The first form consists of a single or *monolithic* classifier that is highly complex and

consists of a large number of feature detectors. The application of such a classifier will however prove to be computationally expensive as all features must be evaluated for all subregions of the candidate image.

The second form of classifier is comprised of a cascade of simple (top level) to complex (bottom level) classifiers. In terms of training, the error rate of this approach converges less rapidly that of the monolithic approach, and requires a longer training time. However, in terms of application to a candidate image, the cascade approach improves speed performance substantially as the expensive complex classifiers are only invoked if the simpler classifiers provide a positive detection. The disadvantage of this approach is that a simple upper level classifier may incorrectly reject a candidate region, and more complex layers that may have correctly classified that region will not be applied; this leads to a lower detection rate of true occurrences. Since the objective application of this chapter is real-time object detection, this variant was implemented even though it yields a lower detection performance.

Considering the cascaded classifier, each cascade layer consists of a weighted combination of a certain number of weak classification functions, each consisting of a single feature. In this thesis, the number of features per layer ranges from 2 to 150 features from top to bottom in the cascade. The training process is therefore resource intensive and time consuming in that it must repeat the same learning algorithm for all feature variants for each layer of the cascade. For example, the face detector, whose training images are sized at $20 \times 20$ pixels, utilises approximately 1.2GB of RAM and takes approximately 36 hours to train using a server comprised of eight 900MHz processors. The leg detector with its $20 \times 40$ training images requires 3.6GB of RAM, and approximately 72 hours training time.

Considering the training of the first layer in the cascade which consists of *two* features, a single feature type is firstly selected, and is then scaled and super-

imposed onto the training images at a fixed location. The weak learner records the value of the feature for each training image, while taking the labeling of the training image into account. The accuracy of the newly trained weak classification function is then evaluated using the same training set, where a correct detection likelihood is determined for each image. The error is measured by determining the number of test images that are correctly classified as positive or negative examples. Since a detection likelihood is provided, an important design choice includes the selection of a threshold that will best separate positive and false positive detections. Too low a threshold will yield a classifier that produces a large number of false positives, while a high threshold may produce a classifier that fails to detect well defined objects.

The training examples are then re-weighted in order to emphasize those which were incorrectly classified, i.e. the weights of examples that are misclassified are more heavily weighted, while the weights of correctly classified examples are lowered. This process is repeated for all locations and scales of that feature type, which produces hundreds of weak classifiers.

Once training of the first feature type is completed, a second feature type is selected, and weak classifiers are built for all variants of that feature type as before; the procedure continues until all weak classifiers have been built for all feature types and their variants. From the resultant large set of weak classifiers, the two that yielded the lowest error are selected, and a strong classifier is derived from their weighted combination. The remaining layers of the cascade are similarly constructed, however a different set of negative training examples is used in order to increase the generalisation capability of the detector. The weightings of the training images are also re-initialised after the addition of every layer. Layers are continually added to the cascade until a target false positive rate is met.

### 5.1.3 Applying the Trained Detector

When the trained object detector is applied at run time, the top end simplest classifier consisting of two features is first swept (with a range of scales) across the candidate image in order to reject the majority of sub windows that do not even contain the most obvious target features. The second, and slightly more complex classifier is then applied to the sub windows that remain, which rejects outliers further. The detector continues to invoke the more complex classifiers until the end of the cascade is reached, and only sub windows that contain the target remain. As indicated in Section 5.4.1, the greater the number of layers in the cascade, the more stringent the object detection, and the lower the detection rate.

## 5.2 Boosted Body Parts Detectors

Using the object detection method presented in Section 5.1, four different body part detectors were separately trained using their respective image databases. The face, torso and hand training images were sized at 20x20 pixels, and the legs at 20x40 pixels. The face database consists of frontal view and side profile images. The faces used for the frontal view database offer approximately 10 degrees of rotation in multiple directions such that faces looking just off the camera centre can still be detected. Variability is obtained from faces looking left and right, and tilting left, right, forward and back. Example images of the respective databases are shown in Figure 5.2.

Definitions of the different body part detectors are provided, each of which takes the form of a cascaded strong classifier. In order to detect a specific body part in a bounding box, all the weak classifiers belonging to the detector are offset to that location. A positive or negative detection is then computed by combining

Figure 5.2: Example images of the face, torso, leg and hand databases

weak classifier outputs in strong-classifier layers. Each detector returns a score for part detection, which is then normalised to produce a detection likelihood, defined as $L_F$, $L_T$, $L_L$ and $L_H$ respectively. Use of this notation can be found in Section 5.3.3. The performance of the detectors is shown in Figure 5.6.

### 5.2.1 Exploiting Colour Cues for Reduced False Detections

Since detections are performed in grey-scale, it would be advantageous to exploit colour cues to act as *a priori* constraints to initially reduce the most obvious false detections. This is especially useful if the colour is fairly consistent across different instances of the objects. Here, the hands and face benefit from this constraint.

As mentioned previously in Chapter 3.5.3, a weak skin colour model consisting of a single Gaussian in the Hue-Saturation colour space has been created from a large selection of natural images. Given a novel image, a skin colour likelihood map can be generated using this Gaussian model. As previously mentioned, face detection/tracking on colour images has received considerable attention - a large difference however is that they first segment skin colour regions, and conduct face-like feature searches in these segmented regions. The approach presented here is the reverse: The hand and face detectors are applied across the entire image, and provide a bounding box for each detection. The skin probability

map is used to determine the median skin colour likelihood for each bounding box. Should this probability fall below a weak threshold (ie. 3 standard deviations), the detection is rejected. Our motivation is twofold: firstly, skin-colour segmentation can be unreliable, possibly leaving blank areas in the skin-coloured regions. Furthermore, facial cavities are ignored, and fingers are thinned. Such partially segmented body parts would fail to meet the detection requirements of the associated part detectors, as the training images are cropped from larger natural images. Secondly, even if the segmentation were clean, restricting the search space to these regions is naive as a generic skin filter is not guaranteed to segment all skin-like objects.

Figure 5.3 demonstrates the method of exploiting colour cues with the use of the face detector. Figure 5.3 (a) shows all face detections – it is evident that several false detections occur over ambiguous textures. With the aid of the generic skin model, many of these false detections can be eliminated, as shown in Figure 5.3 (b). The resulting detection/skin joint-likelihood image for all detections is provided in Figure 5.3 (c) – these likelihoods come into play when determining the joint-likelihood model in Section 5.3.3.

(a) (b) (c)

Figure 5.3: Illustration of reduction of false detections using colour cues

The improved detection performance due to the inclusion of colour is illustrated in Figure 5.4. The curves are created by repeatedly running the face detector on the test database, while decreasing the number of layers in the cascade. The

inclusion of all layers minimises the false detection of face. However, this also leads to the failure to detect faces that should be detected.

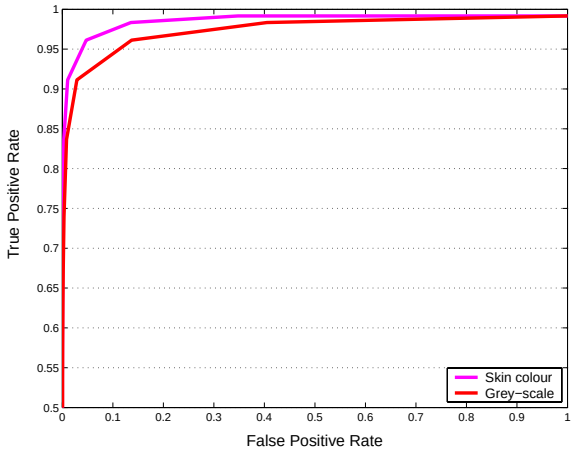



Figure 5.4: Face detector performance using colour. The increased true positive rate is obtained by reducing the number of layers in the cascade.

Later, in Section 5.3.3, in an attempt to associate a pair of hands with a face, a subject specific skin model is built from each face detection in a similar manner described previously in Section 3.5.3.

## 5.3 Human Body Assembly

As mentioned above, the removal of layers in the detector cascade results in an increased positive detection rate. This naturally comes with the trade-off of an increased number of false detections. In order to cope with the false detections, and to determine a final body configuration, a six step process is administered. The details of each process are discussed in the subsequent sections.

1. A weak heuristic is applied to all detections to eliminate obvious outliers (Section 5.3.1).
2. RANSAC is then used to assemble the remaining detections into random body configurations, each consisting of a head, a torso, a pair of legs, and a pair of hands.
3. Each configuration is compared to an *a priori* mixture model of upper-body configurations, yielding a likelihood for the upper body pose (Section 5.3.2).
4. From each configuration, a skin model is learned from the face detector and is used to derive an associated skin colour likelihood for the hands.
5. A resultant joint-likelihood for each configuration is obtained by combining the pose likelihood determined by the prior model, the body part detectors likelihoods and the skin colour likelihood for the hands. The configuration with the highest likelihood is voted for by RANSAC to represent the detected human (Section 5.3.3).
6. With a selected body configuration, the elbow positions estimated and then inferred (Section 4.3) to complete the upper body pose.

### 5.3.1 False Part Elimination using Coarse Heuristics

An image with several human figures and a dense background clutter can easily produce up to a hundred detections for each body part, yielding up to $10^{10}$ possible configurations. Determining a joint-likelihood for each and every possible configuration would clearly be prohibitive, and discarding the most unlikely detections is essential. Several false face and hand detections can be eliminated by exploiting colour as described in 5.2.1. The total number of detections can be reduced further by employing a set of coarse heuristics.

The heuristics have been designed according to a generic human model, and make

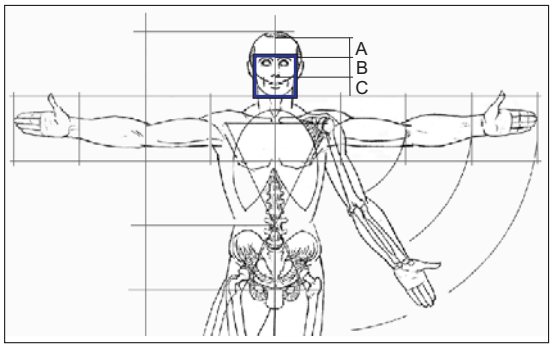



Figure 5.5: Estimating the skeletal unit length from a face detection

reference to the *skeletal unit length*, described previously in Section 3.5, however here this length is estimated in an alternative manner. The detector performance curves shown later in Section 5.4.1 indicate that the face detector is the most robust. The parameters of the face detections are therefore used to determine the skeletal unit length for each body configuration. Referring to the Vitruvian man of Figure 5.5, the head can be subdivided into 3 lengths, A, B and C. A typical face detection occupies B and C, thereby allowing the skeletal unit length to be approximated.

Having selected a face, the skeletal unit length and face centre position are determined, and form the parameters of the heuristics that assist in constructing a body configuration. The heuristic rules are set out in the following order, with $x$ and $y$ referring to horizontal and vertical directions:

1. A torso is retained if its centre $x$ position lies within the face width, *and* if its scale is approximately 3 $\times$ face scale ($\pm$ 0.5).

2. A face is retained if its centre lies within one of the resultant torso detections.

3. A leg detection is retained if its centre $x$ position lies within the face width, *and* if its top $y$ position lies at 4 $\times$ skeletal unit lengths below the face($\pm$

0.5), *and* if the leg scale is approximately 2 × face scale (± 0.5).

4. A pair of hands are retained if both hands are less that 4 × skeletal unit lengths from the face, *and* if the hand scale ≈ face scale (± 0.2).

False hand detections form the bottleneck in the system as a large number are retained by the fourth heuristic. The configurations that pass the set of heuristics are then compared to an *a priori* mixture model of upper-body configurations to obtain a likelihood for the upper body pose (Equation 5.1). This step plays an important role as awkward hand configurations, which are most likely to be false detections, yield a low pose likelihood.

### 5.3.2 Determining a Pose Likelihood

In this step, body configurations are firstly assembled using RANSAC, where supporting evidence is provided by the clustered detections that occur around true occurrences of the object. Should a selected detection not have additional neighbouring detections, it is rejected.

The *a priori* data set of body configurations $\phi$ (without elbow positions) from Chapter 4 is used to determine a pose likelihood for each upper body configuration. An assembled body configuration provides the position of 8 points, namely the four corners of the torso detector, the chin and brow of the face detector, and the hands. After calculating the skeletal unit length, the corner positions of the detected torso are offset accordingly to provide positions for the shoulders and hips, and the brow is offset to provide a position for the top of the head. These 8 $x, y$ coordinates are concatenated to form a feature vector $\mathbf{y}' \in \mathbb{R}^{16}$ as per Chapter 4.

A measure of how well $\mathbf{y}'$ fits the prior data set can now be determined. Firstly, the Mahalanobis distances $md_{\phi,k}$ from $\mathbf{y}'$ to each component of the GMM are

determined. The final pose likelihood $L_P$ is then be obtained from the weighted sum of likelihoods of each component:

$$L_P = \sum_{k=1}^{K} \frac{N_k}{N} \left[ \left( 2\pi^{\frac{d}{2}} |Cov_{\phi,k}|^{\frac{1}{2}} \right)^{-1} exp(-\frac{1}{2} m d^2_{\phi,k}) \right] \tag{5.1}$$

### 5.3.3 Final Configuration Selection

At this point in the algorithm, a coarse skin model with a low threshold was used to reduce obvious false face and hand detections (Section 5.2.1). For each assembled configuration, a new subject specific skin model is learned as per Section 3.5.3 from the skin inliers contained within the bounding box of the selected face as in Section 3.5.3. A skin colour likelihood for the hands is then determined using this model – it is intuitive that a subject's hands will yield a high likelihood when correctly associated with a face. These right and left hand skin likelihoods ($L_{LHS}$and $L_{RHS}$)contribute to the body joint-likelihood model for a body configuration.

The eight determined likelihoods, namely the pose ($L_P$), face ($L_F$), torso ($L_T$), legs ($L_L$), left hand ($L_{LH}$), left hand skin ($L_{LHS}$), right hand ($L_{RH}$) and right hand skin ($L_{RHS}$) are combined to provide an overall body configuration likelihood, $L_{BC}$. Although the assembled configuration is tied together using a set of heuristics, the individual detectors that produced the respective likelihoods are treated as independent. The likelihoods are therefore multiplied together:

$$L_{BCi} = (L_{Pi}).(L_{Fi}).(L_{Ti}).(L_{Li}).(L_{LHi}.L_{LHSi}).(L_{RHi}.L_{RHSi}) \tag{5.2}$$

The configuration that yields the greatest likelihood is selected to represent the final configuration.

### 5.3.4 Estimation of Elbow Positions

Once the final configuration has been selected, its corresponding feature vector $\mathbf{y}' \in \mathbb{R}^{16}$ is used to estimate the elbow positions as previously described in Section 4.3. The human skeleton is then overlaid onto the image to conclude the upper body pose.

### 5.3.5 Detection in Sequences with a Static Background

Extending this work to video sequences with a static background allows for the use of adaptive background segmentation and the application of the detectors in a tracking framework.

Using the background segmentation algorithm of Section 3.4.2, each pixel of the frame is assigned a foreground likelihood. The detectors however, are applied to the full natural frame without segmentation. The mean foreground likelihood $L_{FG}$ of a detection's bounding box is determined by considering the pixel foreground likelihood of the underlying segmented frame. The body configuration likelihood Equation 5.2 is amended to take the foreground likelihood into account:

$$\begin{aligned} L_{BC_i} = (L_{Pi}).(L_{Fi}.L_{FG_{Fi}}).(L_{Ti}.L_{FG_{Ti}}) \\ .(L_{LHi}.L_{LHSi}.L_{FG_{LHi}}).(L_{RHi}.L_{RHSi}.L_{FG_{RHi}}) \end{aligned} \tag{5.3}$$

The chief advantage of detection in a video sequence lies in the tracking framework where the search space is localised in subsequent frames, thereby reducing the number of false detections, the number of hypotheses to be assessed by RANSAC, and therefore improving speed performance. An initial face detection is conducted as before, with consequent body part detections limited by the heuristic proximity rules as defined in Section 5.3.1. Subsequent position and scale variations of each

detector are governed by prior detections. Should a body part fail to be detected, the search region for the corresponding detector is increased and the scale is adjusted by a Gaussian drift term until the detector recovers.

### 5.3.6 Overcoming Occlusions

It is extremely unlikely that images or sequences of people will contain all the constituent body parts at all times. When a body configuration is being assembled, should zero detections of a certain body part be passed by the heuristics, a new body part is synthesised according to the skeletal unit length derived from the selected face. This newly synthesised part is also given a small nominal detection likelihood such that Equation 5.2 does not produce a body configuration likelihood of zero. This is equivalent to adding a wide Gaussian distribution to the detection likelihoods; this is justifiable as in probabilistic terms, a zero detection likelihood should never occur. The nominal likelihood of the synthesised part is set such that any true detection likelihood will surpass it.

As previously mentioned, the hand detector produces a large number of detections, even when no hands are present. The generic skin model and elimination heuristics may discard many false detections, but several always remain. The subject specific skin model learned from the selected face is therefore considered; should the median skin likelihood of a hand detection fall below a low threshold, the hand is re-synthesised. Using the selected skeletal unit length and torso detection as reference, the new hand is positioned at the hips. It is fair to assume that a subject's hands are most likely to be occluded due to them being behind the back or in the trouser pockets. The synthesised hand is also given a nominal skin likelihood.

## 5.4 Results

### 5.4.1 Body Part Detector Performance

Comparison of the different part detectors is a difficult task. The most obvious problem is that each part is of different scale; the number of false detections for the hands can be expected to be larger than that of the torso. In addition, a fair test would make use of an image database in which all the aforementioned body parts exist simultaneously; however, obtaining such a database has not been possible, and three independent test databases have been used. An in-house face database has been assembled, and consists of colour images containing 500 faces. This is of comparable size to the MIT-CMU [98, 89] face database which has 507 faces. The torso and leg detectors were tested on 460 (of 900) images of the MIT pedestrian database [75], while the hand detector was tested on an in-house colour image database containing 400 hands. Figure 5.6 shows the detection performance of the detectors applied to their respective test datasets, where layers from the classifier were removed to increase the number of detections. In this research, a detection is considered true if no less than 75 percent of its bounding box encloses the ground truth body part. In addition, overlapping false detections are not merged as in [89].

From this set of performance curves, the face detector proves to be the most robust This is an intuitive result as the face is a self contained region, while the other body parts are disguised with background clutter and have a greater variability in appearance. In the case of the torso and legs, a colour model can not be used to reduce the false detection rate as there are virtually no similarities in the subjects' clothing. It is evident from the torso and leg detector performance curves that the torso is the poorer detector of the two. Referring back to Figure 5.2, the torso training images contain a larger percentage of background clutter. Furthermore, the texture of the torso can be erratic, as a subject may be wearing a multi-

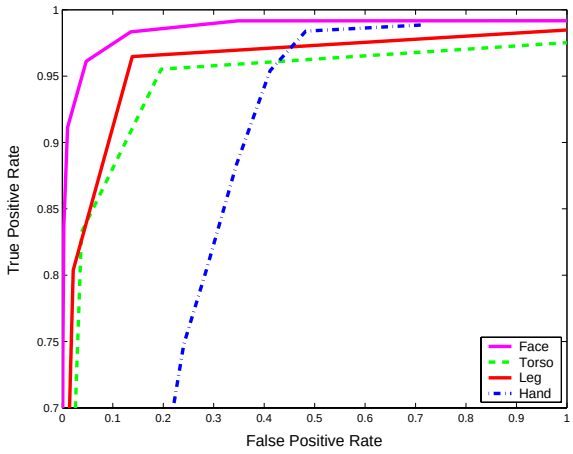



Figure 5.6: Detector performance on the test databases

coloured top, or an opened jacket. Due to the high variability of hand shape, the hand detector can be expected to offer the poorest performance. Owing to the fact that boosting makes use of structured summed area comparisons, it is not surprising that the body part detectors offer a large number of false detections.

The hand detector was put through the same accuracy test as per the hand filter of Section 3.7.3 and the same video sequence of Figure 3.20 was used. Figure 5.7 provides the pixel error of the hand filter (with a population of five hundred particles) and the boosted hand detector.

It is clearly evident that the accuracy of the particle filter is superior. Since the hand detector functions by responding to particular features, motion blur caused by the vigorous movements destroys the necessary features, making the presence of the hand impossible to detect. The particle filter however relies predominantly on colour and is able to cope better with motion blur as the colour information is not entirely destroyed. The mean error of the hand detector is 10.2 pixels, which is comparable to a hand filter with a population of 180 particles. The disadvantage of the particle filter is that it relies heavily on background segmentation to prevent it from converging on random skin coloured objects. In cluttered scenes, once

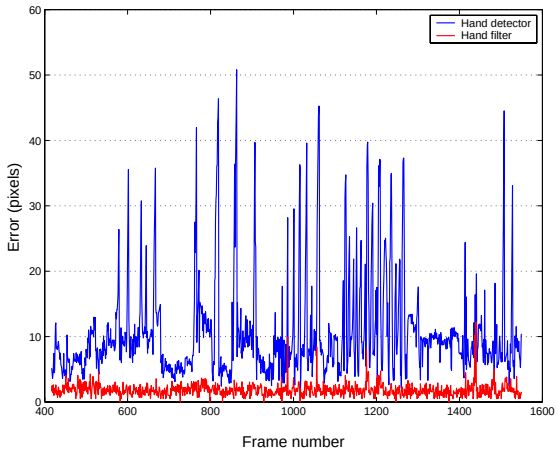
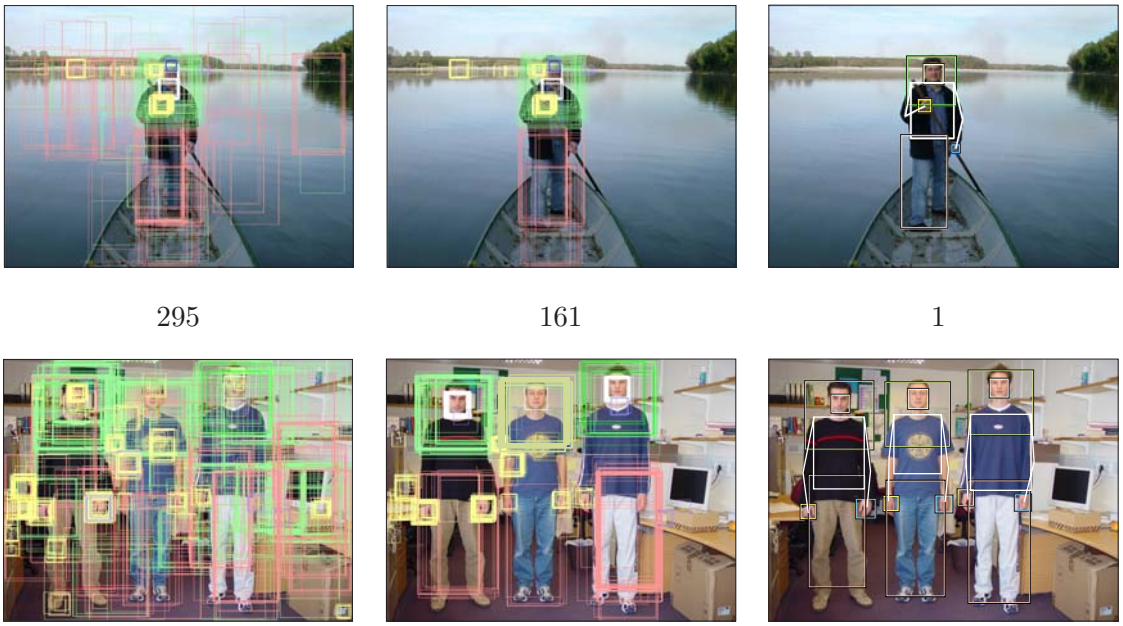



Figure 5.7: Hand tracking error through a video sequence using the hand detector

skin like areas have been identified, the boosted hand detector reduces the more obvious false detections.

### 5.4.2 Body Part Detection and Assembly in Images

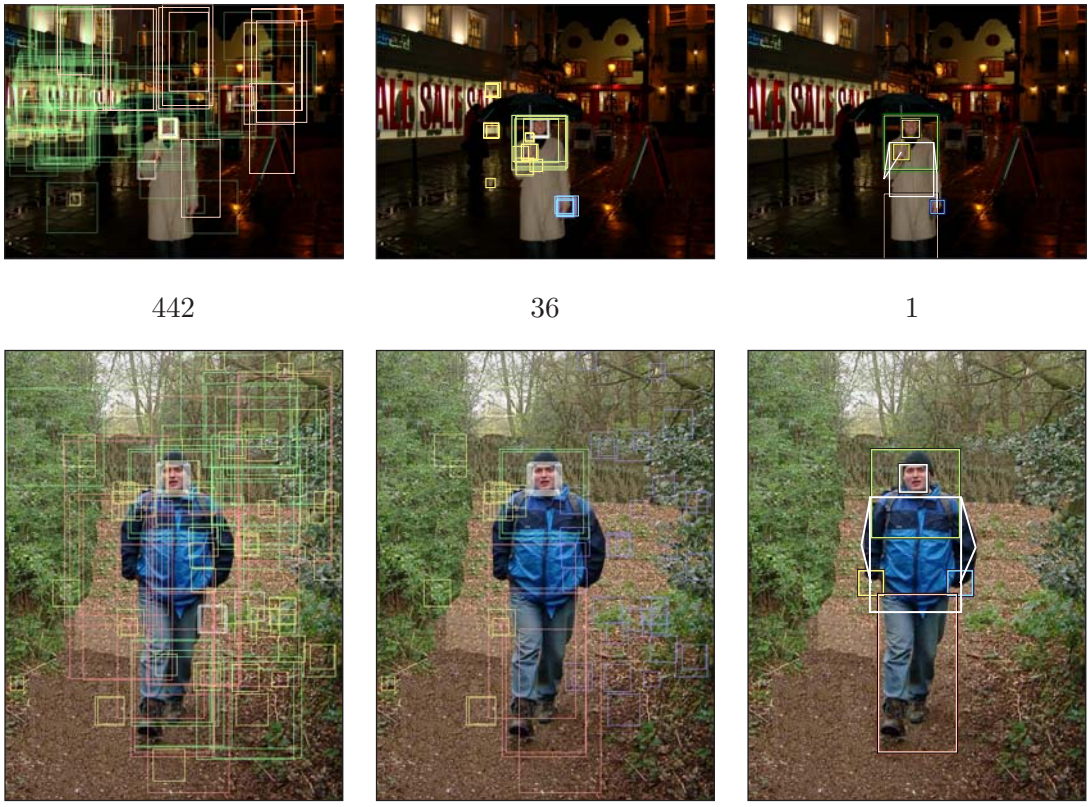
Making use of the performance curves plotted for each detector in Figure 5.6, the desired number of layers was chosen such that the probability of detecting all objects was no less than eighty percent. This decision naturally came with the trade-off of an increased number of false detections. Figure 5.8 (a) shows all detections from the body part detectors applied to a set of images sized at $1024 \times 768$ pixels. In this set of images, all body parts are visible. The false detections are easily eliminated by applying the heuristics, as shown in Figure 5.8 (b). Figure 5.8 (c) highlights the greatest body configuration likelihood as determined by the joint-likelihood model, and also overlays the final body pose with the estimated elbow positions. The total number of detections is indicated beneath each image, and is naturally dependent on the complexity of the image. In the examples shown, the correct body configurations were obtained by including approximately 25% of the total number of possible configurations.




(a) All detections (b) Reduced detections (c) Final assembly

Figure 5.8: Human assembly with all body parts present

Figure 5.9 presents images of people where various body parts are occluded. The legs of the subject in the first image are not present, yet ambiguities in the background lead to multiple leg detections; these false detections are however eliminated by the heuristics, and a leg detection is synthesised. It is interesting to point out the large number of false torso detections clustered around the store's display window located on the left hand side of the image. Upon closer inspection, it is evident that the display window contains suit jackets. In the second image, the subject's hands are occluded. Use of the user specific skin model fails to associate a pair of hands with the face, and the hands are re-synthesised. The entire process from detection to assembly takes approximately 5 seconds on a P4. The detection process is the most expensive, and can be improved upon by using lower resolution images and sampling less frequently. This can be easily justified as the images used to train the detectors have a resolution of $20 \times 20$ pixels.



442 36 1

117 79 1

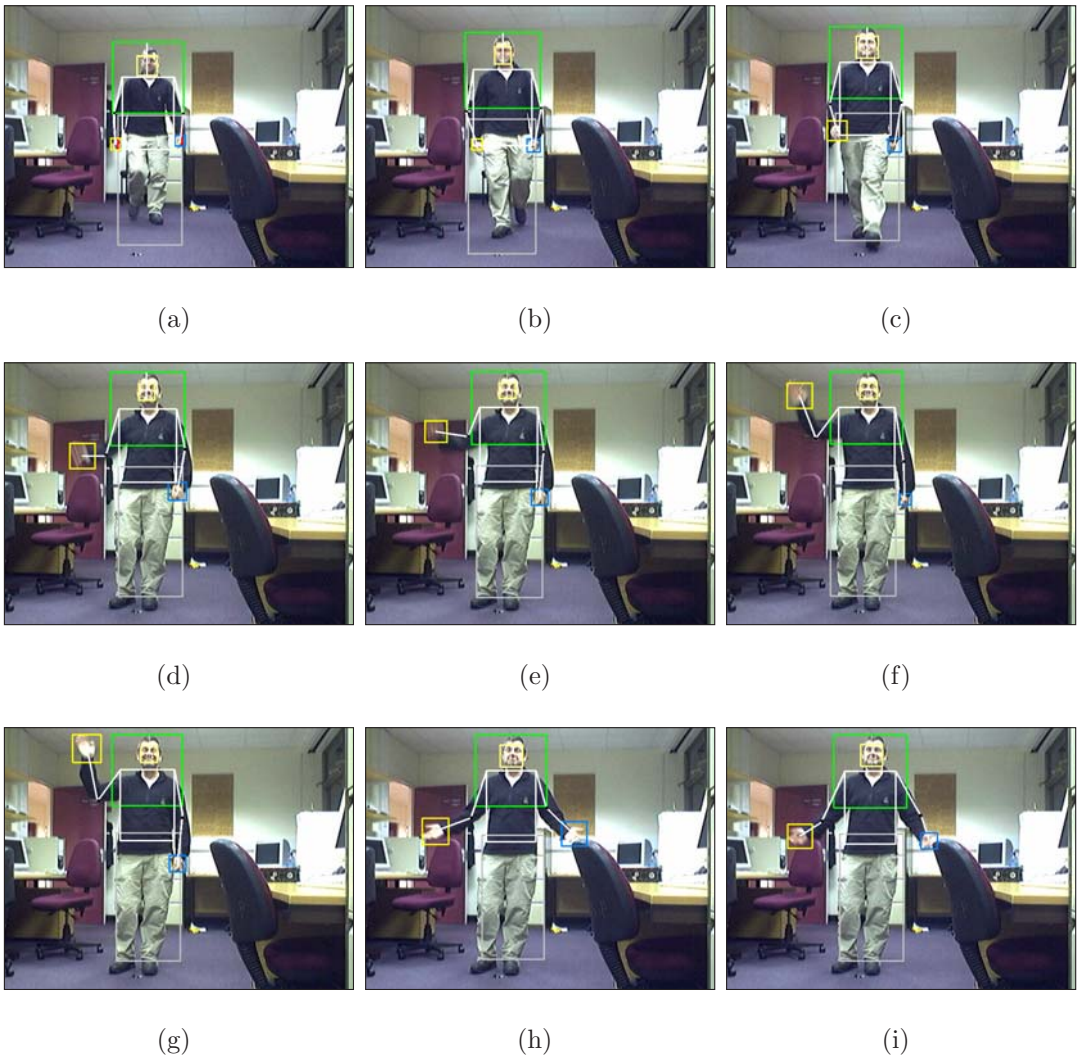
(a) All detections (b) Reduced detections (c) Final assembly

Figure 5.9: Human assembly with occluded body parts

### 5.4.3 Detection and Assembly in Video Sequences

Figure 5.10 illustrates the final body part configuration and elbow estimation of a subject walking into an office and performing hand gestures. Although static, the background is particularly complex, with wood furniture and cork pin boards that have a similar colour to skin. Furthermore, the subject is wearing beige trousers, offering very little contrast to the background of a cream wall and filing cabinets, making background segmentation unfavourable. The assembly system overcomes these difficulties, and with the use of temporal information, operates at 8 frames/sec (frames sized at $640 \times 480$). This is a considerable improvement

compared to the static image case.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5.10: Body part assembly and elbow prediction on a video sequence

A corresponding performance curve for this sequence is provided in Figure 5.11. To maintain consistency with the performance curves of Figure 5.6, each frame of this sequence was treated as a discrete image, with the search space encompassing the entire natural frame. Here, only the hand detector makes use of a foreground likelihood and offers similar performance to the torso and leg detectors. The purpose of using a sequence was to offer a full subject such that the performance

of the assembly method could be evaluated. As expected, the assembly curve supersedes the others, illustrating the robust false part elimination by the assembly methodology.

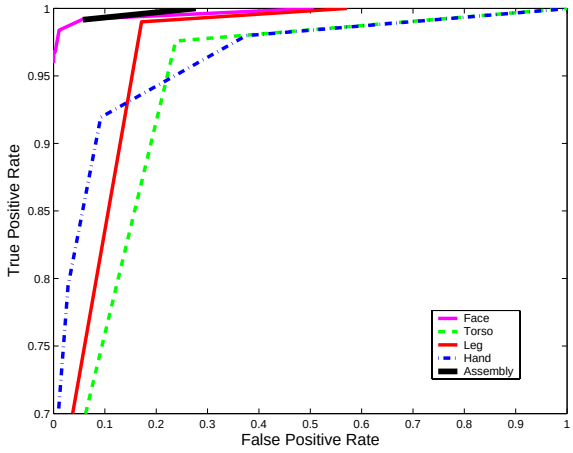



Figure 5.11: Detector and assembly performance on a video sequence

Figure 5.12 makes use of the same sequence of Figure 3.20 that was used previously to demonstrate the particle filter system. Here, the final selected body configuration of the subject presents similar results. Seeing that the sequence is used for HCI purposes, the camera is focused on the upper body, and the leg detector has been omitted from the process. The background is also static, and the foreground likelihood can be included.

Figure 5.13 shows the upper body part detection and assembly of a subject taken from a children's television show. Although the background is not complex, the camera angle changes frequently, as expected in broadcast television. The static camera assumption and motion used in tracking, as per the previous two examples, are therefore no longer valid. Figures 5.13 (b) and (c) offer two consecutive frames where the camera angle changes. This sequence has therefore been treated as a series of still, unrelated images, thereby omitting the foreground likelihood and tracking framework. The reduction in background clutter and the PAL res-

Figure 5.12: Upper body part detection and assembly on a video sequence

olution results in each frame being processed in approximately 2 seconds unlike the 1024x768 photographs in Figures 5.8 and 5.9.

### 5.4.4 Detection of Rotated Faces

As previously mentioned, the face detector has been trained from face images that have approximately 10° of rotation in multiple directions. In a controlled lighting environment, rotations slightly greater than 10° can be detected, as in

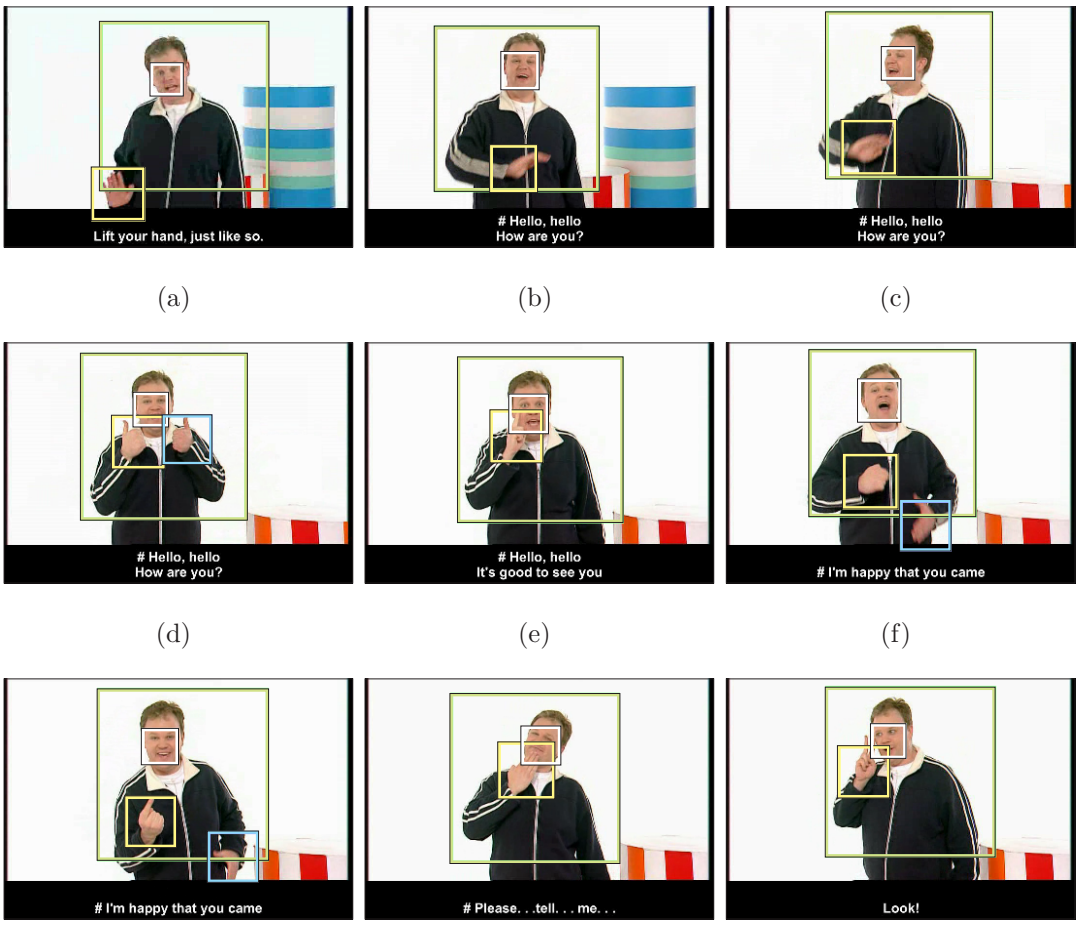



Figure 5.13: Upper body part detection and assembly on a video sequence from broadcast television

the broadcast television sequence of Figure 5.13. Detection of a greater rotation however requires a additional detectors.

The publicly available Intel OpenCV [47] Viola-Jones face detector consists of an interleaved cascade trained from both frontal and side profile images. Images sized at a $24 \times 24$ pixels were used, with the facial region extending from the hairline to the chin. Figure 5.14 evaluates the performance of this rotated face detector on a sequence of a subject looking to the side. The subject starts by facing the camera ($0°$), and moves until a side profile is reached at $90°$. It is

apparent that detection fails between 45° and 60°, which is to be expected as the detector was trained using frontal and side profile images. A third cascade trained with faces angled at approximately 50° should address this problem.

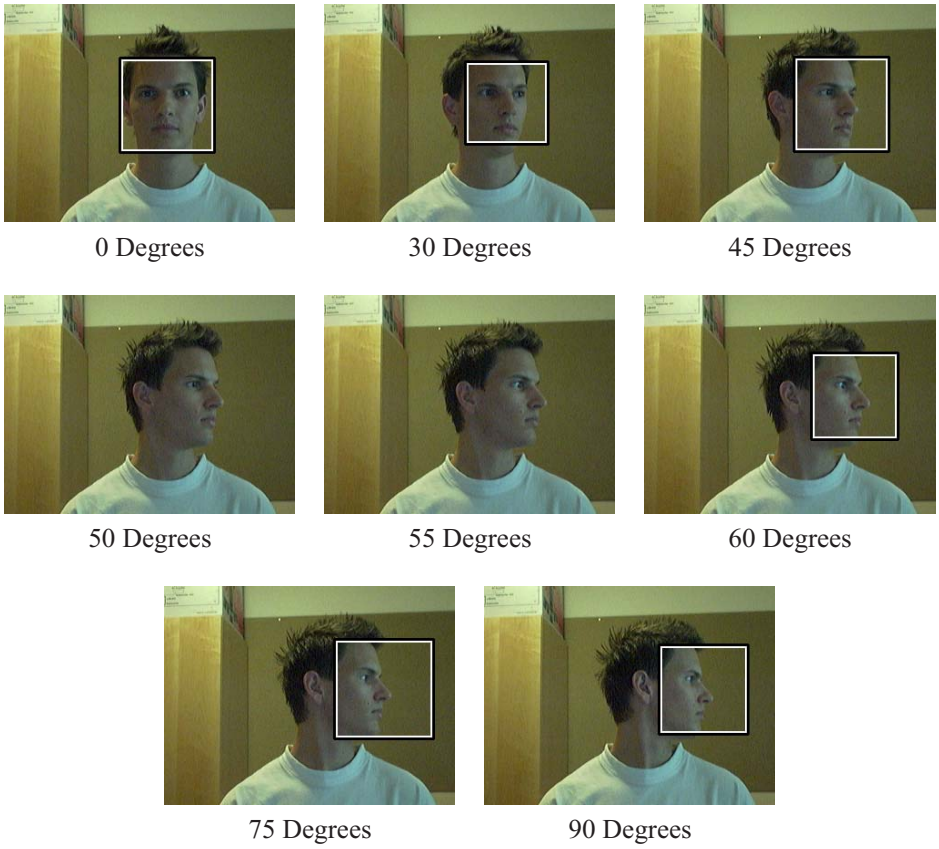
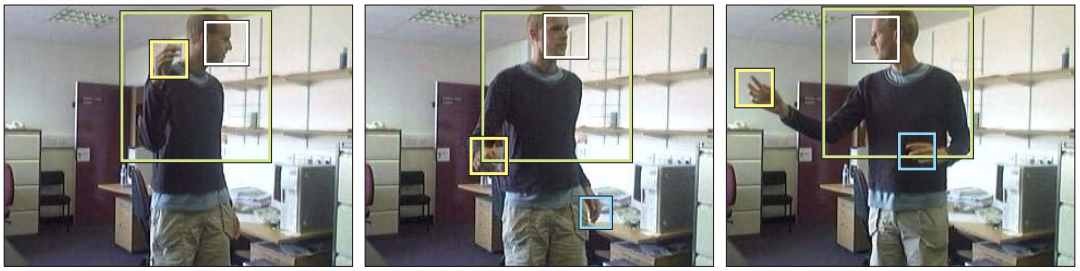



Figure 5.14: Face rotation evaluation of the face detector

Use of the OpenCV face detector now allows for the detection of more natural poses where the subject is not forced to look directly at the camera. Examples of a subject performing random movements in a sequence are shown in Figure 5.15.



(a) (b) (c)

Figure 5.15: Upper body part detection with rotated and side profile face

## 5.5 Conclusions

An existing boosting technique for face detection has been extended to create three additional body part detectors. However, due to the variability of these additional body parts, their corresponding detectors offer a lower performance. Using basic knowledge of the human figure, a set of heuristics was created to eliminate the most obvious false detections. By combining these heuristics with RANSAC and the *a priori* mixture model of upper-body configurations, detections are assembled into accurate human configurations. When this approach is applied to a video sequence, temporal data allows for the use of a foreground likelihood and a tracking framework to further reduce false detections, thereby improving speed performance dramatically.

As previously indicated, the hand detector, trained from a variety of hand poses, limits the speed performance of the assembly method due to it producing a large number of false detections. To address this, Kölsch and Turk [58] propose the use of multiple, independent hand detectors, each trained with a unique hand pose. Eight hand postures were selected, and offer a considerable discriminatory difference in appearance. They also illustrate how training a detector with images of higher resolution can improve the detection rate. This method would clearly not be beneficial in detecting people acting naturally, but is automatically

geared toward a HCI application where the recognition of specific hand shapes is invaluable. Applying all eight detectors on an image simultaneously will most likely prove to be even more prohibitive; a solution to the whole problem would be to apply the 'generic' hand detector, skin colour test and heuristics as before. Subsequently, the pose specific detectors can be applied in the detections that remain.

# Chapter 6

# Real-time Upper Body 3D Pose Estimation

The objective of this Chapter is to estimate the upper body 3D pose of a subject facing a single uncalibrated camera. The intended application lies in 3D Human Computer Interaction where hand depth information offers extended functionality when interacting with a 3D virtual environment. However, it is equally suited to animation and motion capture. To do this, a 3D human model is animated by motion capture data consisting of a variety of upper body movements. A database consisting of three subsidiary databases, namely the frontal-view Hand Position (top-level), Silhouette and Edge Map Databases is then created off-line from the frontal view of each animation frame.

Similarly, at run-time, the frontal view hand positions, silhouette and edge information of the subject are extracted, which are then simultaneously compared to each of the three subsidiary database examples to obtain a matching score. The database example that yields the highest matching score is selected, and the index to the original 3D configuration of the motion capture data is retrieved. From this motion capture frame, the 3D position of each joint can be determined,

or alternatively, the representative 3D model can even be rendered.

## 6.1 Data Acquisition

With the aid of 3D Studio Max [25], a generic human mesh is constructed to resemble a person wearing loose fitting clothing, presented here in Figure 6.1. 3D Studio Max provides a 3D human skeleton or *biped* (pronounced bi-ped), on which the human mesh is to be fitted. The biped is a hierarchical model with physical joint constraints, and allows for realistic human animation. Once the mesh is aligned with the biped, the mesh vertices are assigned to the respective bones of the biped. These assignments are weighted according to the influence a bone should play on that vertex. The weighting can be done on a per vertex basis, or through the use of of *envelopes*. Either method however still requires a visual trial-and-error manipulation to ensure that vertices are correctly assigned. For example, the hands bones should not influence mesh vertices in the vicinity of the upper arm. Once the mesh is correctly fitted and weighted, animating the underlying biped, either by key framing or motion capture, results in a controlled deformation of the mesh, offering a realistic animated human model.

Since only the upper body is of interest, a transparent material is assigned to the mesh from the waist down. Rendering of this model provides an upper human body, against a constant background as in Figure 6.2 (a). Using the Sobel edge detector [83], the edge image corresponding to the rendered model with a standard material is highly cluttered due to shadowing. A model with a uniform material is therefore used to create an accurate edge image without artifacts. This is achieved by employing *cell shading* [107], which provides a flat, cartoon-like effect, where the number of colour levels can be controlled. The remaining images of Figure 6.2 show the same model using a cell shaded material, with a decreasing number of colour levels. This reduces false edges, but at the cost of discarding valuable

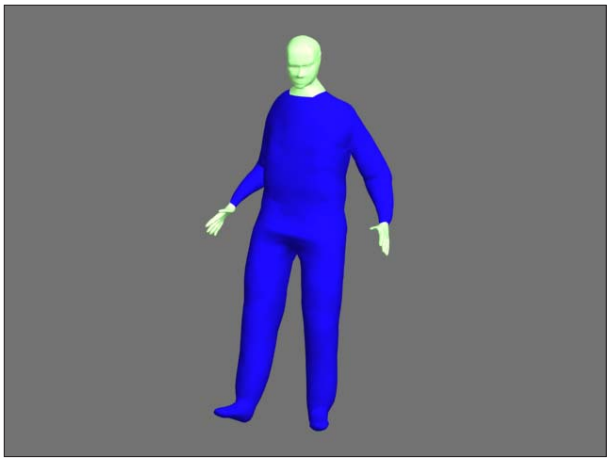
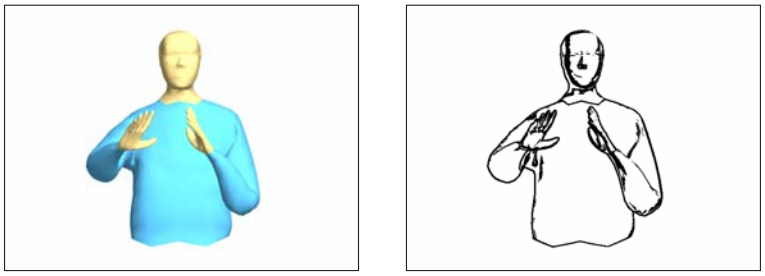


Figure 6.1: Generic 3D human model

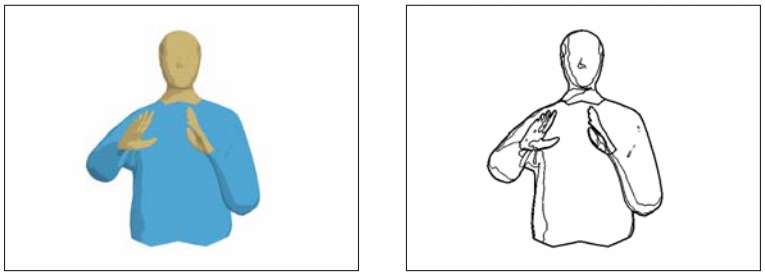
edge information. As seen from the figure, one colour level produces an accurate edge detection, but the edges of the arms are lost when occluding the torso.

In order to preserve this valuable edge information, each body part is assigned a different material colour, as shown in Figure 6.3 (a). The adjacent figure shows the correct, even detection of all the edges. The left and right hands are coloured blue and yellow respectively, thereby resembling a subject wearing gloves. As per Chapter 4, this facilitates the ground truth labelling, and the construction of the hand position database of Section 6.2.

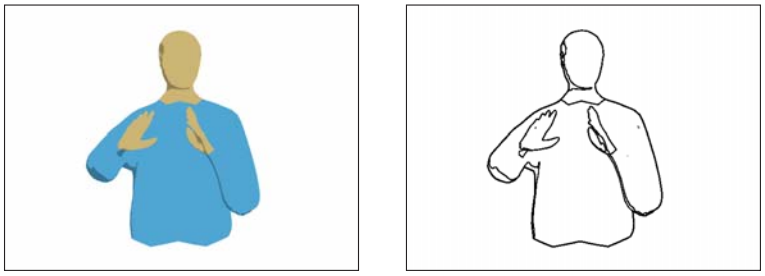
A single *target* camera (a camera whereby the camera-to-target distance remains fixed) is then attached to the top of the biped chest bone. The roll and pitch of the camera is also dependent on that of the chest bone, such that a frontal, upright view of the model is always viewed. A motion capture database of human movement, consisting of a wide variety of activities such as athletics, dancing, and acrobatics, is used to animate the multi-coloured model. Figure 6.4 (a) presents the original perspective view of the 3D model performing a few actions, with the target camera moving in accordance with it. Figure 6.4 (b) in turn shows the corresponding images rendered from the target camera view: an upright upper



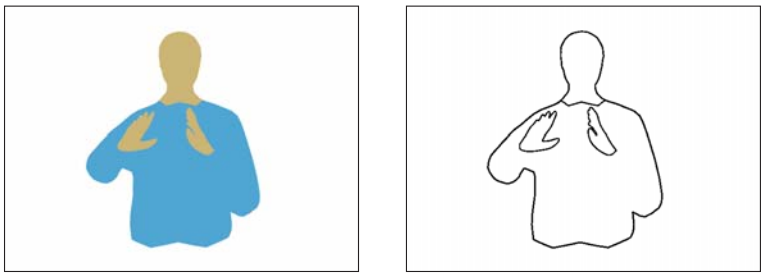
Standard material



Cell shading - 3 colour levels



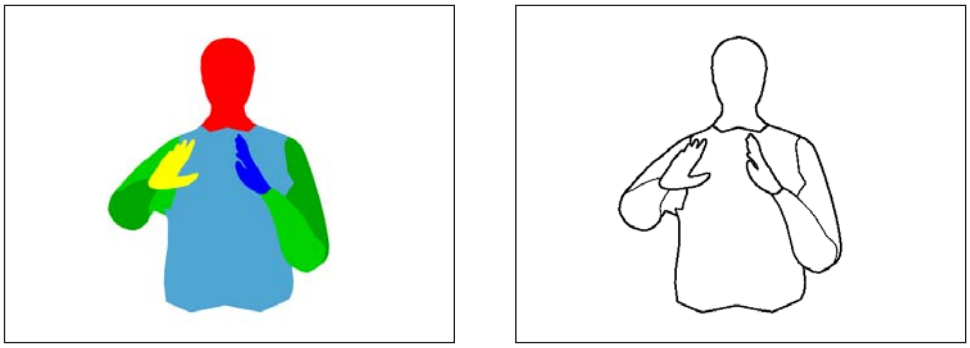
Cell shading - 2 colour levels



Cell shading - 1 colour levels

(a) Rendered model (b) Corresponding edge image

Figure 6.2: Adjusting mesh materials



(a) (b)

Figure 6.3: (a) Independent colouring of body parts (b) Resultant edge image

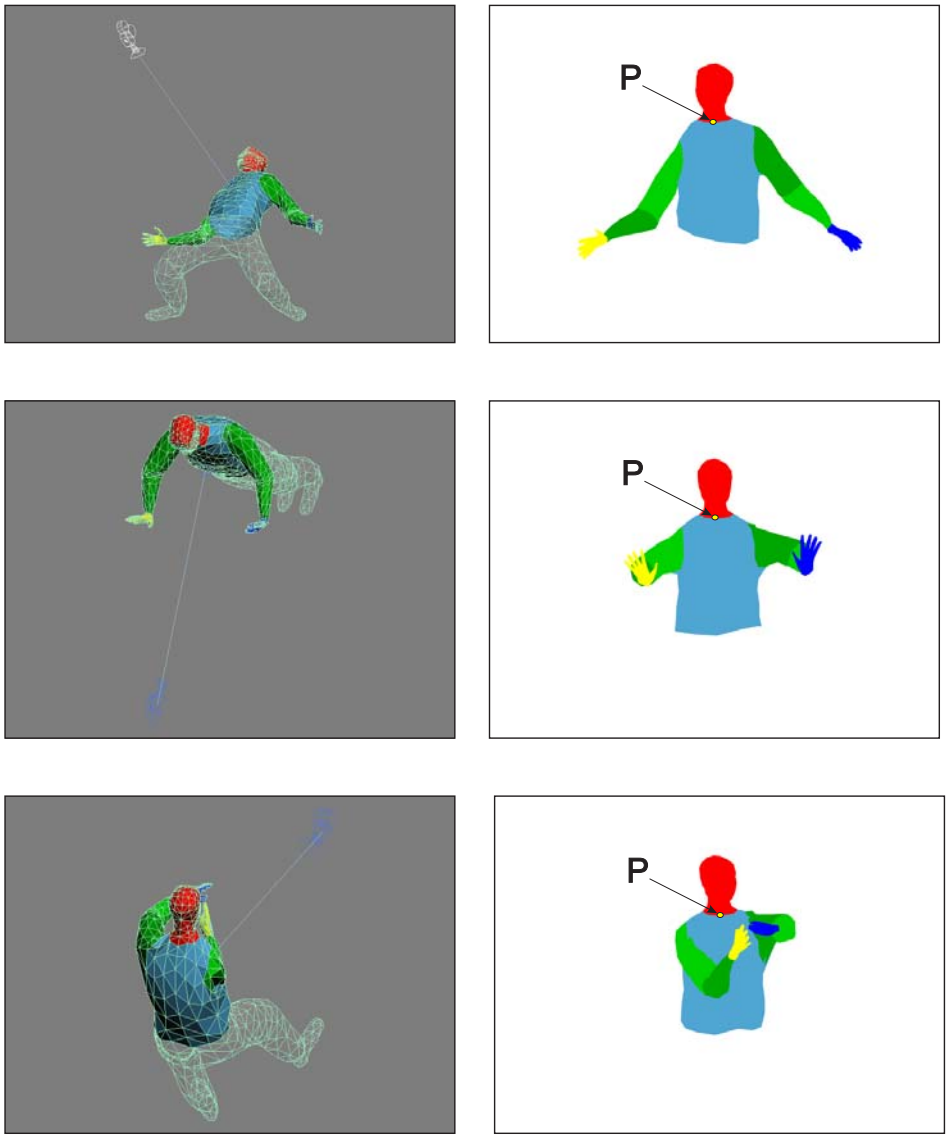
body of fixed scale, centred at position $P$ (the camera target). The actions of the database include a significant amount of upper body movement, thereby generating a large number of 3D upper body configurations. This sequence, consisting of approximately 5000 frames, is rendered from the target camera view to produce the *Frontal View Database*.

## 6.2 Subsidiary Databases

The images of the Frontal View Database are then used to produce three subsidiary databases. These are pre-computed off-line, and are used later in real time matching. From parent down, these are:

1. **Hand Position Database.** This consists of the 2D positions of the left and right hands. As independent primary colours are used, these positions are determined by the respective centroids of the blue and yellow regions of each frame.

2. **Silhouette Database.** Owing to the contrasting, constant background, extraction of the silhouette image is straightforward. However, due to the



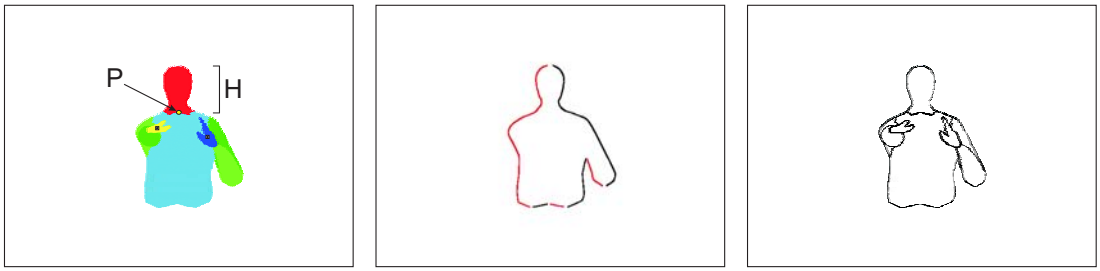

(a) Original 3D model (b) Extracted frontal view

Figure 6.4: Extraction of frontal view images from motion capture

size of the dataset, storing a silhouette image for each frame is unrealistic as the entire dataset would occupy several Gigabytes in raw format. It is therefore more efficient to represent each silhouette image in terms of its boundary, as shown in Figure 6.5 (b). Each row of the silhouette consists of an entry and exit pair along a scan line, indicated here in red and black.

This representation not only minimises storage requirements, but offers a fast and efficient method of comparison to the human subject's silhouette, which is represented as a *shape encoding* integral image (see Section 6.3.4).

3. **Edge Map Database.** As mentioned previously, conducting an edge detection on the rendered cell shaded, multi-coloured model, provides a clean edge image as in Figure 6.5 (c). Again, to reduce storage requirements, only non zero edge locations are stored, not the edge image.




(a) (b) (c)

Figure 6.5: (a) Frontal view of the 3D model (b) Boundary (c) Edge map

For the purpose of this chapter, any three corresponding examples taken from each of the respective databases is referred to as a *triplet*. Each triplet is indexed according to the Frontal View Database, and hence the 3D body configuration database (motion capture data) that generated it.

## 6.3 Model Matching

The sections below discuss the processes that occur at run-time, where the candidate image is compared to the precomputed subsidiary frontal view databases.

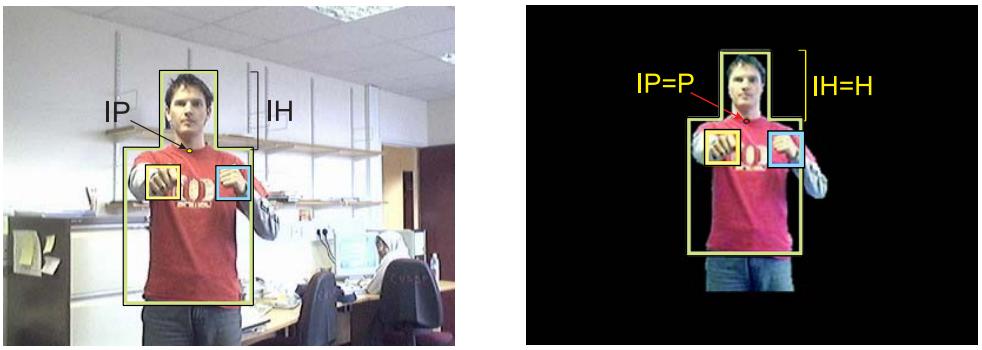
### 6.3.1 Background Suppression and Tracking

In this chapter, the *input image* refers to the image captured at run time, and consists of a cluttered scene and a subject facing the camera. The *input silhouette* and *input edge image* refer to the corresponding silhouette and edge images derived from the input image. Segmenting the subject from the input image plays an important role in determining a matching score of the input silhouette against the Silhouette Database examples. Naturally, background segmentation also assists with tracking the various body parts by discarding ambiguous background information. The background segmentation algorithm of Chapter 3 is employed to extract the silhouette of the subject, and the body parts can be tracked using either the particle filter of Chapter 3, or the body part detection and assembly methodology of Chapter 5.

### 6.3.2 Scale Adjustment of the Input Image

Referring back to the example of the Frontal View Database of Figure 6.5 (a), the 'head' length $H$ is measured from the top of the head to the neckline. $\bar{H}$ is therefore the mean head length of the entire Frontal View Database, and is used as the reference length with which to scale the input image. Position $P$ represents the target of the camera, and remains constant throughout.

Comparing the input image to the subsidiary Frontal View Databases requires that the input image foreground occupies the same spatial domain as that of the examples. To achieve this, $IP$, the top of the chest bone of the subject and the head length $IH$ must be determined (see Figure 6.6 (a)). The particle filter tracking system of Chapter 3 provides the positions and dimensions of the torso and hands. $IP$ is approximated to be the same as the shoulder height, and $IH$ is therefore the length from the top of the head to $IP$.




(a) (b)

Figure 6.6: (a) Input image (b) Segmented, adjusted input image

The input image adjustment scale factor is determined by

$$S = \frac{IH}{\bar{H}} \tag{6.1}$$

and the offset $\delta$ from $P$ to $IP$ is determined by

$$\delta = P - \frac{IP}{S} \tag{6.2}$$

Once the subject is segmented, the input image is scaled and translated in a single pass, creating the *adjusted input image* ($AdjIm$) of Figure 6.6 (b), such that $IP = P$ and $IH = H$:

$$\forall x, y \; AdjIm(x, y) = inputImage(x, y)/S + \delta; \tag{6.3}$$

From this adjusted input image, the input silhouette $IS$ and input edge map are extracted.

### 6.3.3 Extracting Subsets of the Subsidiary Databases

Comparing the input silhouette and input edge map to each and every example of the Silhouette and Edge Map Databases would be exhaustive and unnecessary.

A subset of the Silhouette Database is therefore initially extracted by making use of the Hand Position Database and the subject's hand positions. The bounding box parameters of the subject's left and right hands (provided by the tracking algorithm) are used to search through the Hand Position Database for hand positions that are simultaneously contained by these bounding boxes. The indices of successful examples are then used to extract the corresponding Silhouette Database examples. The input silhouette is then compared to this subset of silhouette examples.

### 6.3.4 Silhouette Matching using Integral Images

Since it is likely that several example silhouettes corresponding to the subject's hand configuration will be identified, a matching score is determined for each comparison.

A set of matching scores for the Silhouette Database subset is determined by computing the percentage pixel overlap between the input silhouette and each silhouette example. A straightforward method would be to reconstruct the silhouette image from the boundary information stored in the Silhouette Database, and to compute the difference between the input and the example on a per pixel basis. However, doing so would be prohibitive as a reconstructed silhouette contains thousands of non-zero valued pixels. The matching procedure is therefore made more efficient by using an intermediate representation of the input silhouette $IS$, the integral image $II$.

The integral image employed here however differs subtly from that discussed in Chapter 3, where only rectangular features could be computed. Here the shape of the object is encoded by computing the summation of pixels on a row by row basis, i.e. the value of the $II(x, y)$ equals the sum of all the non-zero pixels to

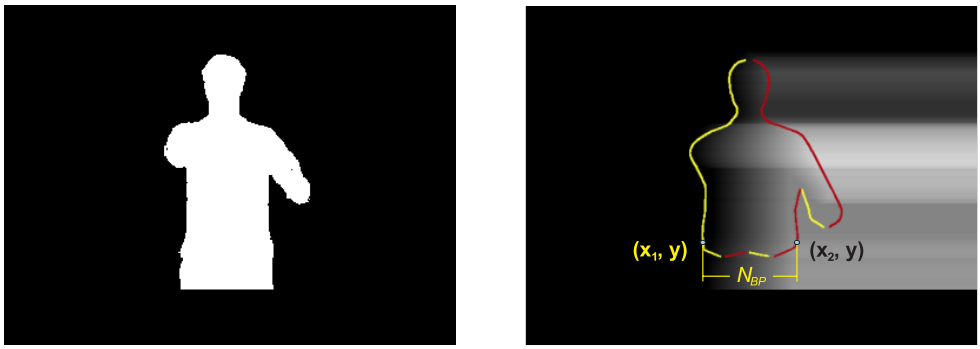
the left of, and including $IS(x, y)$ for that row only:

$$II(x, y) = \int_{i=0}^{x} IS(i, y)\mathrm{d}i \tag{6.4}$$

The entire integral image can be computed in this manner for all $(x, y)$, but this is done incrementally as before for efficiency:

$$\forall x, y \; II(x, y) = IS(x, y) + II(x - 1, y) \tag{6.5}$$

Figure 6.7 (a) shows the input silhouette derived from the segmented input image of Figure 6.6 (b). A visualisation of the corresponding integral image, with a silhouette boundary example of the Silhouette Database superimposed, is presented in Figure 6.7(b).




(a) (b)

Figure 6.7: (a) Input silhouette (b) Corresponding integral image

In Figure 6.7 (b), $N_{BP}(y)$ is the number of pixels between boundary pair $(x_1, y)$ and $(x_2, y)$, and is computed as

$$N_{BP}(y) = x_2 - x_1 + 1 \tag{6.6}$$

Making use of the integral image, $N_{IS}(y)$, the number of pixels of the input silhouette for the corresponding range of $(x_1, y)$ to $(x_2, y)$ is therefore computed

as

$$N_{IS}(y) = II(y, x_2) - II(y, x_1) + 1 \tag{6.7}$$

$N_{BP}(y)$ is computed and summed for all boundary pairs, and is denoted as $\sum N_{BP}$; this is pre-computed off-line for all silhouette examples, and is then loaded at run-time. $N_{IS}(y)$ is computed and summed at run time ($\forall y$), and is denoted $\sum N_{IS}$.

Since $\sum N_{BP} \geq \sum N_{IS}$, the matching score $S$, or percentage overlap between the input silhouette and the silhouette boundary example, is computed as:

$$S = \frac{\sum N_{IS}}{\sum N_{BP}} \qquad 0 \leq S \leq 1 \tag{6.8}$$

Figure 6.8 provides a comparison of the matching score computation using standard per pixel subtraction and an integral image. A typical silhouette reconstructed from the Silhouette Database typically contains approximately 17 000 non-zero valued pixels. Computing the difference between the input silhouette and the reconstructed silhouette example would therefore require 17 000 subtraction operations.

In contrast, the input silhouette can be converted to a shape encoded integral image in 110 000 ($384 \times 288$) operations, one third of that required previously in Section 3.6 which required three operations per pixel. Analysis of the Silhouette Database examples indicates that the mean number of boundary pairs lies at approximately 200. $\sum N_{IS}$ therefore requires approximately 400 operations: the computation of $N_{IS}(y)$ ($y = 1, \ldots, 200$), and 200 operations to sum them.

The matching score is therefore computed in a few hundred operations, which is again considerably less than tens of thousands of pixel-to-pixel comparisons. The intersection of the curves indicates that computation time is already saved when seven examples are compared.

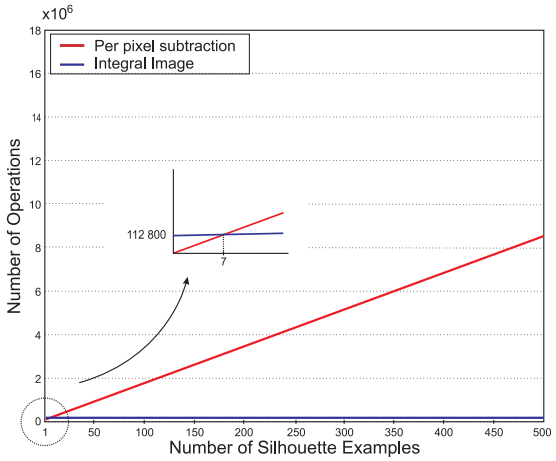



Figure 6.8: Number of operations using standard summation versus an integral image

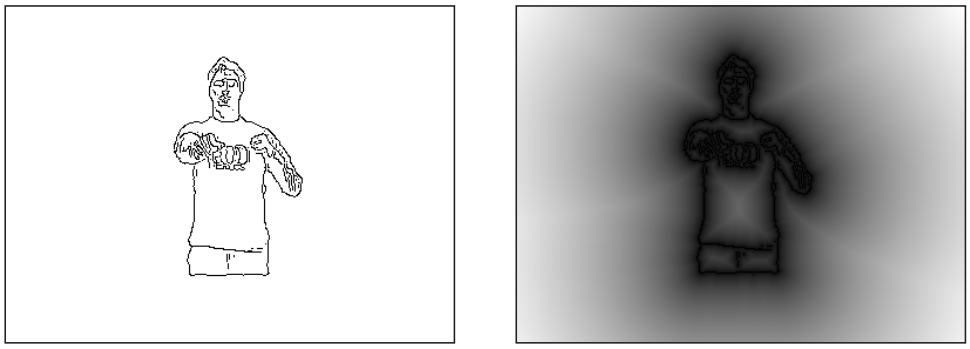
Once matching scores are computed for the examples of the Silhouette Database *subset*, the top 10 percent are then used to extract a subset of the Edge Map Database.

### 6.3.5 Chamfer Matching and Final Selection

Poses with the arms partially occluding the torso may produce similar silhouettes, which suggests that silhouette matching alone is insufficient. The edge information is therefore also considered to offer further support, and to resolve ambiguities. Once a subset of the Edge Map Database is extracted using the results of silhouette matching, each edge map example is compared to the input edge map to compute a second matching score.

Since humans vary in physique, it is unlikely that the edges of the input and the examples will overlap exactly. A distance transform [27] is therefore applied to the input edge image to 'blur' the edges. The distance transform specifies the Euclidean distance of each pixel to the nearest edge i.e. the lower the value of a pixel, the closer it is to an edge. The transform operates by selecting a non-edge

pixel as the centre, and searches with an increasing radius for the nearest edge pixel. This is repeated for all non edge pixels of the candidate image. Figure 6.9 (a) provides an input edge image, with the corresponding distance image in Figure 6.9 (b).



(a) (b)

Figure 6.9: (a) Edge image (b) Distance image

Each example edge map is then superimposed as per Figure 6.10 onto the distance image in order to determine the *edge distance* [8]. This is computed by taking the mean of the distance image pixel values that co-occur with the example edge locations. The example that yields the shortest distance is selected to represent the best match. This method of matching edge images is referred to as Chamfer matching [2].

Using the index of the selected edge map example, the original motion capture frame is retrieved. This data provides all the body part lengths and joint angles, allowing for the 3D position computation of all body parts, including the hands. Alternatively, the motion capture frame can be used to pose the 3D model which can subsequently be rendered to create part of an animation.

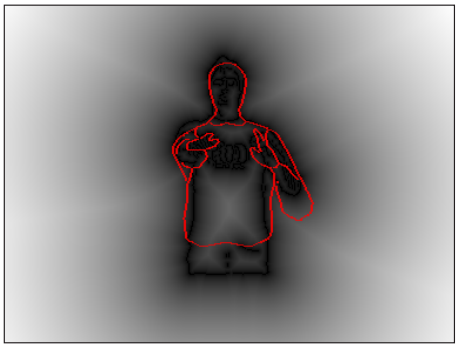
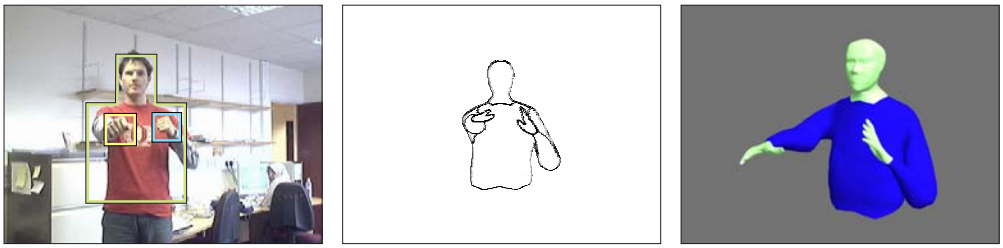


Figure 6.10: Chamfer matching

## 6.4 Results

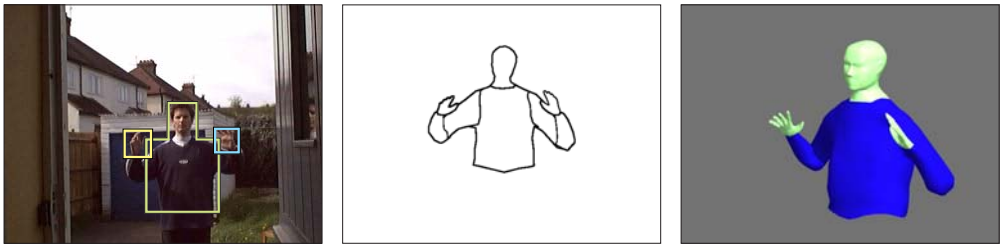
The images of Figure 6.11 (i) and Figure 6.12 (i) present the body part detection and tracking of subjects in various scenes. Following the model matching techniques of Section 6.3, the final selected edge map from the Edge Map Database is shown in Figure (ii). A representative 3D human model, corresponding to the best triplet example match, is shown in Figure (iii). The perspective view of this model indicates that forward movement of the hands has been identified. The model illustrated here is the same as that used to generate the example databases, and can be easily replaced with another 3D human model. The current implementation consisting of particle filter tracking and model matching, operates at 16 frames/second. Since the small motion capture database consists of 5000 examples, a linear search for the corresponding hand positions is conducted; clustering or binary search trees would allow for considerably larger databases.

The body part detection method used is irrelevant to the reconstruction, provided that an estimate of the subject's scale can be determined. This information is used to adjust the captured subject's scale such that it matches that of the example databases. The subjects of Figure 6.12 (c) and (d) have been detected using the detection and assembly methodology of Chapter 5. The edge map selection process is conducted as normal, and the representative 3D model is rendered.



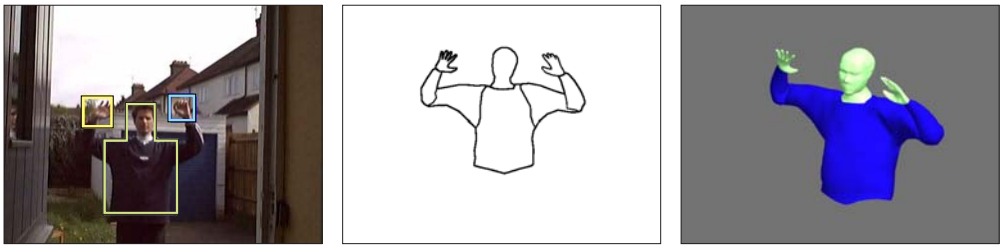
(i) (ii) (iii)

(a)



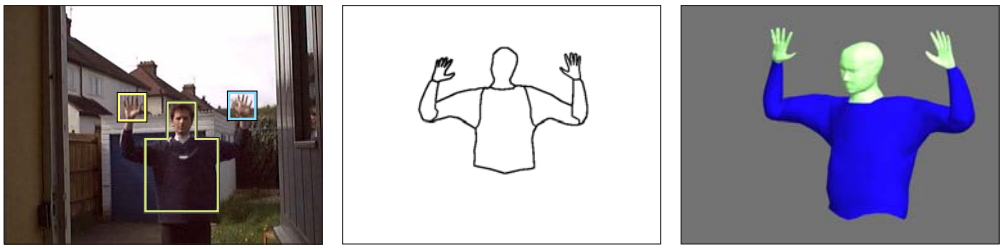
(i) (ii) (iii)

(b)



(i) (ii) (iii)

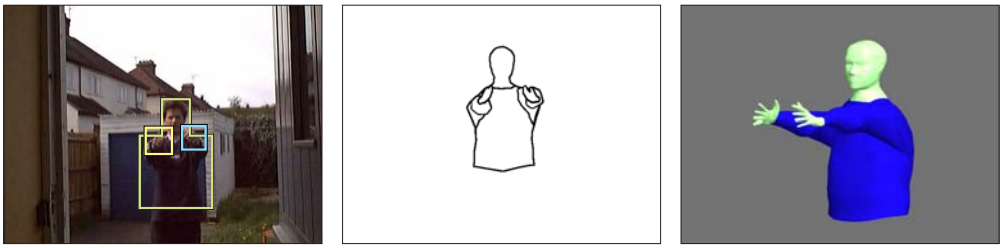
(c)



(i) (ii) (iii)

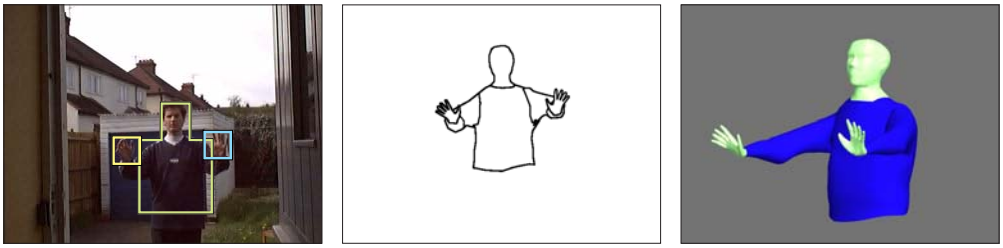
(d)

Figure 6.11: Frontal pose with corresponding 3D model



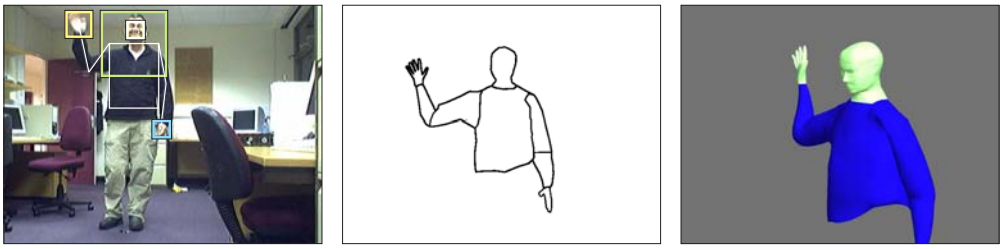
(i) (ii) (iii)

(a)



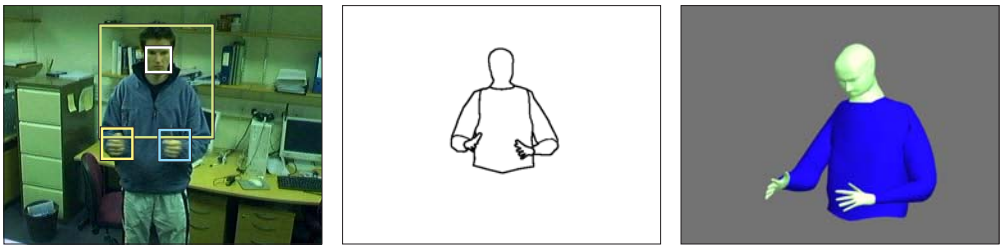
(i) (ii) (iii)

(b)



(i) (ii) (iii)

(c)



(i) (ii) (iii)

(d)

Figure 6.12: Frontal pose with corresponding 3D model

## 6.5 Conclusions

A method of matching a 3D model to the frontal view of a subject in a cluttered scene has been developed. The 3D hand positions can be extracted for HCI, or the computer generated model itself can be used for animation purposes.

Using loose fitting clothing and a generic model naturally inhibits the accuracy of extracting the exact hand positions. For HCI in uncontrolled environments, the relative backward-forward hand movements alone allow for the expansion of the gesture library. Furthermore, the perception of depth allows for interaction with 3D virtual environments. In terms of animation where a higher accuracy is required, a more controlled setting would obviously offer an improvement. A performer with tight fitting attire, and a 3D model that resembles that performer, would facilitate a more accurate matching process. Even with such enhancements, it is unlikely that the technique will ever produce millimetre accuracy like a motion capture system. This should be acceptable when the cost difference is realised: an inexpensive web camera versus a multiple camera motion capture system.

Since the extraction of of the subject's silhouette is required, the technique cannot however be applied to static images with cluttered scenes where subject segmentation is not possible. The use of coloured screens for chroma-keying would obviously circumvent this difficulty.

Finally, matching by example requires a large example dataset, and it is therefore imperative to store the databases in their simplest forms. Not only can these simple representations be accessed quickly, but they also contribute to the fast matching methods employed. Furthermore, the structure restricts analysis to subsets of the subsidiary databases, thereby contributing to the real-time aspect of the approach.

# Chapter 7

# Closing Discussion

## 7.1 Summary

Detection and tracking using the particle filter method of Chapter 3 offers a fast and robust tool for use in Human Computer Interaction (HCI). Its major limitation is that it requires a constant coloured background, or a static background that can exploit a form of background subtraction. With the omission of background segmentation, a generic skin colour model to detect and track the face and hands could be used, but this would naturally be susceptible to background clutter like wood furniture. The torso on the other hand is even more difficult as people wear different styles and colours of clothing. If the texture of the torso was known however, its detection could be included in the fitness computation of the torso particles.

The current implementation of the respective body part filters precludes rotation due to the use of the standard integral image. However, should the shape encoded integral image presented in Chapter 6 be used, rotations could be detected. Tilting of the head and torso would allow for the extension of recogniseable gestures.

With respect to tracking multiple people, a difficulty arises when they are stand-

ing close to each other. In terms of generic tracking where subjects are walking past each other, re-initialisation is acceptable. However, should the application be a HCI system with a single user (and several observers in the background), it is essential that the user's hands be correctly associated. Even if a single particle filter system is used, it is possible for one of the hand filters to converge on an observer's face or hands. The subject specific skin model does offer assistance, however non-studio lighting conditions warrant the need to lower the skin detection sensitivity. The observers could hide their hands behind their backs, however this would clearly be limiting and unreliable. The difficulty of isolating the hands from the face is somewhat easier in that the face detector could be used to identify all faces; knowledge of the face locations could be used to forbid hand particles from entering those regions.

The boosted body part detectors of Chapter 5 prove useful in addressing the issues of tracking a person in a cluttered scene. As previously indicated, applying the detectors independently produces a large number of false detections. The heuristics and prior model assist in extracting a final configuration, but the entire process proves to be a little too slow for real time HCI, even when applied within a tracking framework. However, with the continuous development of faster personal computers, one could argue that this method could operate in real time within a year. The primary difficulty is detecting the required object, and since the torso and leg detectors are trained using low resolution images, the poor detection performance is not surprising since the target resolution is approximately ten times the size of the training images. The hands themselves are the most difficult to detect; the problem is further compounded due to the loss of feature information caused by motion blur. Perhaps a balance could be found by tracking via detection when the movements are small, and tracking via particle filters when the movements are large. Another possibility would be to construct a colour hand detector, however the hardware requirements to train it

would be greater, and the detection process longer. Again, should the application be HCI, associating body parts with the correct subject is still a challenging task worthy of attention.

Extracting the 3D positional information of the subject has also been investigated. The methodology proposed here matches the silhouettes and edge maps derived from an animated 3D human model to those of a subject. Using an exemplar approach naturally presents the dilemma of insufficient data to produce an accurate estimation of the subject. Based on the Vitruvian human model, each hand can occupy approximately one hundred discrete locations (i.e. without overlapping) around the body without crossing over. This produces at least forty thousand possible configurations $(100 + 100)^2$. Should an adequately sized database be constructed, efficient search trees would need to be implemented to extract the appropriate data in real time. It would be intuitive to build the motion capture database from choreographed movements designed according to the type of movement that is to be matched. For example, it is unlikely that the movements of a subject playing a game of virtual chess on a plasma screen are going to make use of poses extracted from aerobics. Another option would be to treat the left and right hand side of the subject independently. This would require a smaller dataset, and would be an acceptable option if only the hand positions were required. If however a 3D model was required, the two selected half meshes could not merely be joined. Each half of the underlying skeletons would have to be extracted and aligned to allow for the 3D model to be regenerated.

Matching the frontal view of a subject to a 3D model not only provides the hand positions, but the elbow positions too; this offers an alternative to the statistical elbow prediction method presented in Chapter 4. In contrast, the animated 3D model that generated the frontal view databases can also be used to generate prior motion models for statistical elbow prediction and pose likelihoods (Chapter 5). Treating the poses obtained from the motion capture data as feature vectors

would also allow the data set to be represented by a GMM. The feature vector extracted from the subject could then be compared to each Gaussian to determine which set of silhouettes and edge maps it should be compared to. This would provide a more sound search method than using the hand positions to crudely search through the hand position database.

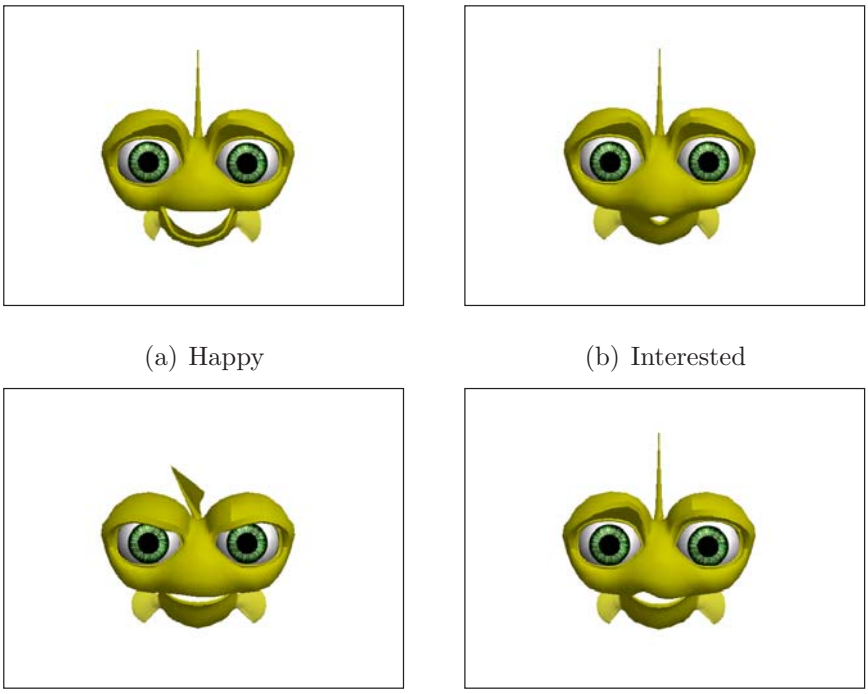
## 7.2 Future Work

The research presented throughout this thesis has focused on detecting and tracking humans for HCI. This section presents *work in progress* that will allow a subject to interact with an intelligent agent rendered on a screen, with a camera mounted above it to represent its vision system. The concept was inspired by *Jeremiah* [10] where a 3D human head is projected on a screen, and follows the most significant object in its field of view. Its vision system uses the same adaptive background segmentation algorithm used throughout this thesis, but the foreground objects are grouped using connected components. The largest blob represents the region of greatest interest, to which Jeremiah is attracted.

Figure 7.1: Finn the fish, an interactive artificial intelligent agent

The artificial agent in the proposed project is a cartoon fish named Finn, illustrated in Figure 7.1. The chosen environment is the ocean, with atmospheric

effects including underwater lighting and bubbles. The inclusion of other environmental objects including vegetation and other animal life is currently under development.



(a) Happy (b) Interested

(c) Angry (d) Confused

Figure 7.2: Finn's facial expressions

Finn's vision system will utilise the particle filter system of Chapter 3 to track the respective body parts of a subject, while his emotions and facial expressions will be determined by the actions performed by the subject. The exhibited expressions illustrated in Figure 7.2 will initially be 'hard coded' according to the performance of specific gestures. This is obviously limiting in that it will not have a long lasting appeal to a new user. The long term objective is to allow *users* to influence how Finn responds to certain actions i.e. the user performing the gesture will specify what the expected response should be. This bears similar light to a game called *Creatures* [59] where the player rewards or punishes his artificial pet if they behaved well or poorly. The more subjects that interact with Finn, the more

varied his responses will be, especially so if expressions are combined to formulate new facial expressions. Another advantage of using unconstrained gestures is that it circumvents the need for associating a pair of hands with a subject. As long Finn is able to identify moving hands, he will be able to respond.

Since Finn is a cartoon-like character, allowing him to speak would also contribute to the entertainment aspect. Again, rather than having fixed responses, self-understanding conversation could provide comical answers like Jabberwacky [15], a text based conversational construct.

The system could be extended in a variety of ways, making it an exciting and rewarding concept to be involved in. The idea has generated significant interest, and market research is currently being conducted to determine where in industry such a novelty could be placed.

# Appendix A

# Adaptive Background Suppression

This appendix offers a brief explanation of the adaptive background suppression algorithm used in Chapters 3, 5 and 6. The reader is directed to the corresponding papers of [96, 55] for a more elaborative guide.

The values of a pixel over time are considered as a time series of vectors (RGB) $\{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$, modelled as a mixture of $K$ Gaussian distributions. A large value of $K$ provides more robust segmentation, but at the cost of slow system performance. Using current-day PCs, 3 to 5 Gaussians have been found to provide sufficiently robust segmentation, while still maintaining real-time performance. At time $t$, $\boldsymbol{\mu}_{k,t}$ is the mean of the $k^{th}$ Gaussian in the GMM, and $\Sigma_{k,t}$ is the covariance. The Gaussian probability density function $\eta$ is given by

$$\eta(\mathbf{x}_t, \boldsymbol{\mu}_t, \Sigma_t) = \frac{1}{2\pi^{\frac{n}{2}}|\Sigma_t|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}_t - \boldsymbol{\mu}_t)^T \Sigma_t^{-1}(\mathbf{x}_t - \boldsymbol{\mu}_t)} \tag{A.1}$$

The probability that a pixel $\mathbf{x}_t$ fits this Gaussian is therefore

$$P(\mathbf{x}_t) = \sum_{k=1}^{K} \omega_{k,t} \times \eta(\mathbf{x}_t, \boldsymbol{\mu}_{k,t}, \Sigma_{k,t}) \tag{A.2}$$

where $\omega_{k,t}$ is the weight estimate, which is set to $1/K$ at initialisation.

An on-line expectation maximisation approximation is then used to update the mixture model as time progresses. A new pixel value $\mathbf{x}_t$ is checked against the existing $K$ Gaussian distributions until a match is found. A match is considered true if the pixel value lies within 2.5 standard deviations of a distribution. If no match is found, the least probable distribution is replaced with a new distribution: the mean as the current pixel value, a high variance, and a low weight.

The mean and variance of the matched Gaussian $j$ are updated as follows:

$$\boldsymbol{\mu}_{j,t} = (1-\alpha)\boldsymbol{\mu}_{j,t-1} + \alpha\mathbf{x}_{j,t} \tag{A.3}$$

$$\sigma^2_{j,t} = (1-\alpha)\sigma^2_{j,t-1} + \alpha(\mathbf{x}_{j,t} - \boldsymbol{\mu}_{j,t})^T(\mathbf{x}_{j,t} - \boldsymbol{\mu}_{j,t}) \tag{A.4}$$

where $\alpha$ is the learning rate, the inverse of which defines the time constant at which the distribution parameters change. The mean and variance of the remaining Gaussians are not changed.

The prior weights for each of the $K$ distributions at time $t$ are then adjusted according to

$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{A.5}$$

where $M_{k,t}$ is 1 for model which matched, and 0 for the remaining models. Once all $K$ weights have been computed, they are re-normalised.

With the newly determined weights and updated Gaussians, the probability that a pixel is background is determined as per Equation A.2. $1 - P(\mathbf{x}_t)$, in turn gives the probability that a pixel is foreground; the foreground probability image is created accordingly and facilitates with the object detection and tracking as discussed in previous chapters.

The significant advantage of this background segmentation method is that when a new object is placed in the scene, it only becomes part of the background if it

is stationary for a long time; it does not however destroy the existing background model. If the object is moved, the distribution describing the previous background still exists (with a lower weighting), and will be quickly re-incorporated into the model.

# Appendix B

# The HSV Colour Model

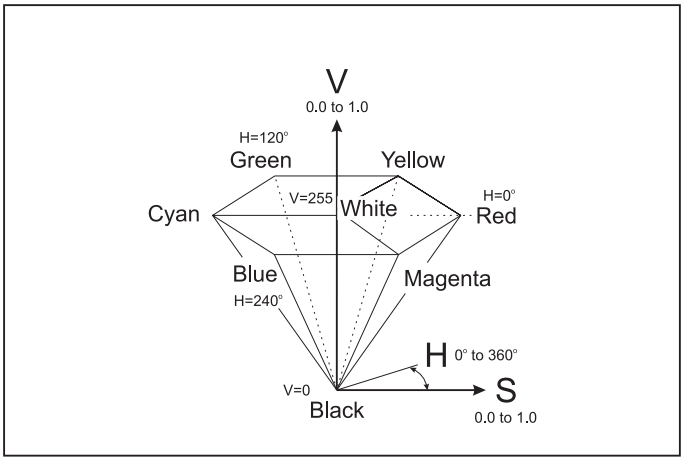



Figure B.1: The HSV hex cone

Figure B.1 provides an illustration of the HSV hexcone [32]. The *hue* component is an angular measurement, and increases in value in a counter-clockwise direction. A value of 0 indicates the colour red, 120 indicates green, and 240 indicates blue. The primary and secondary colours red, yellow, green, cyan, blue, and magenta occur at the vertices of the hexagons. The *saturation* component describes colour intensity where a value of 0 (on the vertical axis) indicates that the colour is gray, while a value of 1 (at the outer edge of a hexagon) specifies that the colour

is saturated. The *value* component describes the brightness where value of 0 represents black, while a maximum value of 1 indicates that the colour is at its brightest.

Treating the HSV colour space as a hexcone, the pseudo code for converting a RGB pixel to a HSV pixel is presented in Algorithm 1. This thesis conducts skin detection in the HS plane, and V is not included in the skin model.

**Algorithm 1** Conversion of RGB to HSV

$max = \max(R, G, B)$

$min = \min(R, G, B)$

$\delta = \max - \min$

Hue

**if** $(max == R)$ **then**

  $H = \frac{G-B}{\delta}$  (between magenta and yellow)

**else if** $(max == G)$ **then**

  $H = \frac{B-R}{\delta}$  (between yellow and cyan)

**else if** $(max == B)$ **then**

  $H = \frac{R-G}{\delta}$  (between cyan and magenta)

**end if**

Convert to Degrees

$H = H \times 60$  ($H$ is represented by a hexagon i.e. $6 \times 60°$)

**if** $(H < 0)$ **then**

  $H = H + 360$

**end if**

Saturation

$S = \delta / max$

Value

$V = max$

# Bibliography


[1] F. Aherne, N. Thacker, and P. Rockett. The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1997.

[2] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of International Joint Conference of Artificial Intelligence (IJCAI)*, pages 659–663, 1977.

[3] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 1(35):99–110, 1943.

[4] A. Blake. Invited speaker. In *BMVA Symposium on Spatiotemporal Image Processing*, 2004.

[5] A. Blake, R. Curwen, and A. Zisserman. A framework for spatio-temporal control in the tracking of visual contours. *International Journal of Computer Vision (IJCV)*, 11(2):127–145, 1993.

[6] A. Blake and M. Isard. *Active Contours*. Springer Verlag, 1998.

[7] N. Bojic and K.K. Pang. Adaptive skin segmentation for head and shoulder video sequences. In *Proceedings of Visual Communications and Image Processing (VCIP)*, pages 704–711, 2000.

[8] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 10(6):849–865, November 1988.

[9] R. Bowden. *Learning Non-Linear Models of Shape and Motion*. PhD thesis, Brunel University, Systems Engineering, October 1999.

[10] R. Bowden, P. KaewTraKulPong, and M. Lewin. Jeremiah: The face of computer vision. In *Smart Graphics'02, 2nd International Symposium on Smart Graphics, ACM Int. Conf. Proceedings Series*, pages 124–128, 2002.

[11] R. Bowden and T.A. Mitchell. Non-linear statistical models for the 3D reconstruction of human pose. In *Image and Vision Computing*, volume 18, pages 729–737, 2000.

[12] R. Brooks. The intelligent room project. In *Proceedings of Cognitive Technology Conference*, pages 271–275, 1997.

[13] J. Calvert. Eye toy: Mini games. *www.gamespot.com/ps2/action/eyetoy/news_6030310.htm*, March 2003.

[14] J. Carpenter, P. Clifford, and P. Fearnhead. An improved particle filter for non linear problems. Technical report, Department of Statistics, University of Oxford, 1997.

[15] R. Carpenter. Jabberwacky. *http://www.jabberwacky.com*, 2005.

[16] C.Garcia and G. Tziritas. Face detection using quantized skin colour regions merging and wavelet packet analysis. *IEEE Transactions on Multimedia*, 9(4):264–277, 1999.

[17] D. Chai and K.N. Ngan. Face segmentation using skin color map in videophone applications. *IEEE Transactions on Circuits and Systems for Video Technology (CSVT)*, 9(4):551–564, 1999.

[18] P. Chen and C. Grecos. A fast skin region detector. In *Proceedings of* $1^{st}$ *IEEE Conference of Visual Information Engineering (VIE)*, pages 35–38, 2005.

[19] I. Cohen and M. Wai Lee. 3D Body Reconstruction For Immersive Interaction. In *Proceedings of Articulated Motion and Deformable Objects Workshop*, 2002.

[20] T.F Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[21] D. Cristinacce, T. Cootes, and I. Scott. A multi-stage approach to facial feature detection. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 231–240, 2004.

[22] F.C. Crow. Summed-area tables for texture mapping. In *Proceedings of Computer Graphics and Interactive Techniques*, volume 18, pages 207–212, 1984.

[23] J. Deutscher, A. Blake, and I.Reid. Articulated body motion capture by annealed particle filtering. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2126–2133, 2000.

[24] J. Deutscher, A. Davison, and I. Reid. Automatic partioning of high dimensional search spaces associated with articulate body motion capture. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 669–676, 2001.

[25] Discreet. 3D Studio Max 6. *http://www4.discreet.com/3dsmax*, 2005.

[26] T. Hewett et al. *Curricula for Human-Computer Interaction*. The Association for Computing Machinery, 1992.

[27] P. Felzenszwalb and D. Huttenlocher. Distance transforms of sampled functions. Technical Report TR2004-1963, Cornell Computing, 2004.

[28] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient matching of pictorial structures. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 66 – 73, 2000.

[29] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Comm. of the ACM*, volume 24, pages 381–395, 1981.

[30] M.A. Fischler and R.A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 22(1):67–92, 1973.

[31] R.B. Fisher, K. Dawson-Howe, A. Fitzgibbon, C. Robertson, and E. Trucco. *Dictionary of Computer Vision and Image Processing.* John Wiley and Sons, England, 2005.

[32] J. Foley, A. Van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice, Second Edition in C.* Addison-Wesley, 1995.

[33] D.A. Forsyth and M.M. Fleck. Body plans. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 678–683, 1997.

[34] F. Gasparini, S. Corchs, and R. Schettini. Pixel based skin colour classification exploiting explicit skin cluster definition methods. In *Proceedings of International Colour Association AIC Colour*, pages 543–546, 2005.

[35] D. M. Gavrila, J. Giebel, and S. Munder. Vision-based pedestrian detection: the protector system. In *Proceedings of Intelligent Vehicles Symposium*, pages 13–18, 2004.

[36] D. M. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 1, pages 87–93, 1999.

[37] D.B. Gennery. Visual tracking of known three-dimensional objects. *International Journal of Computer Vision (IJCV)*, 7(3):243–270, 1992.

[38] G. Gordon, T. Darrel, M. Harville, and J. Woodfill. Background estimation and removal based on range and color. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 459–464, 1999.

[39] U. Grenander, Y. Chow, and D.M. Keenan. *HANDS. A Pattern Theoretical Study of Biological Shapes*. Springer-Verlag, New York, 1991.

[40] W. Grimson, C. Stauffer, R. Romano, and L. Lee. Background estimation and removal based on range and color. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 22–31, 1998.

[41] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active vision*, pages 59–73. MIT Press, 1992.

[42] D.C. Hogg. Model-based vision: A program to see a walking person. In *Image and Vision Computing*, volume 1, pages 5–20, 1983.

[43] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proceedings of Frame-Rate Applications Workshop*, pages 1–19, 1999.

[44] N. Howe, M. Leventon, and W. Freeman. Bayesian Reconstruction of 3D Human Motion from Single Camera Video. In *Neural Information Processing Systems (NIPS)*, volume 12, pages 820–826, 2000.

[45] I-S. Hsieh, K-C. Fan, and C. Lin. A statistical approach to the detection of human faces in colour nature scene. *IEEE Transactions on Pattern Recognition*, 35(1):1583–1596, 2002.

[46] R. Hsu, M. Abdel-Mottaleb, and A.K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(5):696–706, May 2002.

[47] Intel. Opencv. *http://www.intel.com/technology/computing/opencv*, 2005.

[48] S. Ioffe and D. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision (IJCV)*, 43(1):45–68, 2001.

[49] M. Isard and A. Blake. Visual tracking by stochastic propagation of conditional density. In *Proceedings of Automatic Face and Gesture Recognition*, volume 1, pages 38–44, 1996.

[50] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision (IJCV)*, 29(1):5–28, 1998.

[51] Y. Ivanov, A. Bobick, and J. Liu. Fast lighting independent background subtraction. In *Proceedings of Workshop on Visual Surveillance*, pages 49–55, 1998.

[52] I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[53] A.X. Ju, M.J. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Proceedings of Automatic Face and Gesture Recognition (FGR)*, pages 38–45, 1996.

[54] P. KaewTraKulPong and R. Bowden. An adaptive visual system for tracking low resolution targets. In *Proceedings of British Machine Vision Conference (BMVC)*, volume 1, pages 243–252, 2001.

[55] P. KaewTraKulPong and R. Bowden. A real-time adaptive visual survelance system. *Journal of Image and Vision Computing*, 21(10):913–929, 2003.

[56] H. Kang and S. Cho. Adaptive object tracking using Bayesian network and memory. In *Proceedings of Video Surveillance and Sensor Networks*, pages 104–113, 2004.

[57] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis. Background modeling and subtraction by codebook construction. In *Proceedings of International Conference on Image Processing (ICIP)*, volume 5, pages 3061–3064, 2004.

[58] M. Kölsch and M. Turk. Robust hand detection. In *Proceedings of Automatic Face and Gesture Recognition (FGR)*, pages 614–619, 2004.

[59] Creature Labs. Creatures. *http://www.gamewaredevelopment.co.uk/creatures_index.php*, 2004.

[60] I. Laptev and T. Lindeberg. Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features. *Lecture Notes in Computer Science*, 2106:63–76, 2001.

[61] M. Wai Lee, I. Cohen, and S Ki Jung. Particle filter with analytical inference for human body tracking. In *Proceedings of Motion and Video Computing Workshop*, pages 159–165, 2002.

[62] P. Li, T. Zhang, and A.E.C. Pece. Visual contour tracking based on particle filters. *Image and Vision Computing*, 21(1):111–123, 2003.

[63] R. Lienhart and J. Maydt. An extended set of Haar-like features for rapid object detection. In *Proceedings of International Conference on Image Processing (ICIP)*, volume 1, pages 900–903, 2002.

[64] S. H. Lin, S. Y. Kung, and L. J. Lin. Face recognition/tracking by probabilistic decision -based neural networks. *IEEE Transactions on Neural Networks*, 8(1):747–762, 1997.

[65] H. Lohninger. *Teach/Me Data Analysis*. Springer Verlag, 1999.

[66] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proceedings of International Conference on Computer Vision ICCV*, volume 1, pages 572–578, 1999.

[67] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In *Proceedings of European Conference on Computer Vision (ECCV)*, volume 2, pages 3–19, 2000.

[68] S. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 11(2):674–693, 1989.

[69] R. Marks. Eye toy. *www.eyetoy.com*, 2005.

[70] J. Micheletti and M. Wurpts. Applying chroma-keying techniques in a virtual environment. In *Proceedings of AeroSense Helmet and Head-Mounted Displays Conference*, pages 1–10, 2000.

[71] I. Mikic, M. Triverdi, E. Hunter, and P. Cosman. Articulated body posture estimation from multicamera voxel data. In *Proceedings of Computer Vision and Pattern Recognition*, pages 455–461, 2001.

[72] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust body part detectors. In *Proceedings of European Conference on Computer Vision (ECCV)*, volume 1, pages 69–82, 2004.

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

[74] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99–110, 2003.

[75] M. Oren, C.P. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 193–99, 1997.

[76] C. Palmer. *The Fundamentals of Drawing No. 2.* Vinciana, 1992.

[77] C.P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 552–562, 1998.

[78] A. Papoulis. *Probability, Random Variables, and Stochastic Processes, second edition.* McGraw-Hill, New York, 1984.

[79] A. Pentland. Classification by clustering. In *Proceedings of the IEEE Symposium on Machine Processing of Remotely Sensed Data*, 1976.

[80] P. Pérez, C. Hue, J. Vermakk, and M. Gangnet. Color-based probabilistic tracking. In *Proceedings of European Conference on Computer Vision (ECCV)*, volume 1, pages 661–675, 2002.

[81] N. Peterfreund. Robust tracking of position and velocity with kalman snakes. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 21(6):564–569, 1999.

[82] Son Lam Phung, A. Bouzerdoum, and D. Chai. A novel skin color model in ycbcr color space and its application to human face detection. In *Proceed-*

*ings of International Conference on Image Processing (ICIP)*, volume 1, pages 289–292, 2002.

[83] K.K. Pringle. Visual perception by a computer. In *Automatic Interpretation and Classification of Images*, pages 277–284, 1969.

[84] B. Triggs R. Ronfard, C. Schmid. Learning to parse pictures of people. In *Proceedings of European Conference on Computer Vision (ECCV)*, volume 4, pages 700–707, 2002.

[85] C. Rao. *Linear Statistical Inference and Its Applications*. John Wiley and Sons, New York, 1973.

[86] C. Rasmussen and G.D. Hager. Joint probabilistic techniques for tracking objects using multiple visual cues. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 16–21, 1998.

[87] T. Roberts, S. McKenna, and I. Ricketts. Human pose estimation using learnt probabilistic region similarities and partial configurations. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 291–303, 2004.

[88] P. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 347–356, 1995.

[89] H.A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(1):23–38, January 1998.

[90] R. Schapire. The boosting approach to machine learning: An overview. In *Proceedings of MSRI Nonlinear Estimation and Classification Workshop*, 2002.

[91] R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

[92] H. Sidenbladh. *Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences*. PhD thesis, Royal Institute of Technology, CVAPL, Stockholm, 2001.

[93] H. Sidenbladh, F. De la Torre, and M.J. Black. A framework for modeling the appearance of 3D articulated figures. In *Proceedings of Automatic Face and Gesture Recognition*, volume 1, pages 368–375, 2000.

[94] K. Sobottka and I. Pitas. Extraction of facial regions and features using colour and shape information. In *Proceedings of International Conference on Image Processing (ICIP)*, volume 3, pages 483–486, 1996.

[95] K. Sobottka and I. Pitas. A novel method for automatic face segmentation, facial feature extraction and tracking shape information. *Transactions on Signal Processing: Image Communication*, 12(3):263–281, 1998.

[96] C. Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 22(8):747–757, 2000.

[97] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla. Hand pose estimation using hierarchical detection. In *Workshop on HCI*, pages 105–116, 2004.

[98] K.K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(1):39–51, 1998.

[99] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake and A. Yuille, editors, *Active vision*, pages 3–20. MIT Press, 1992.

[100] M. E. Tipping. Sparse Bayesian learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1(211):211–244, 2001.

[101] S. Tsekeridou and I. Pitas. Facial feature extraction in frontal views using biometric analogies. In *Proceedings of European Signal Processing Conference (EUSIPCO)*, volume 1, pages 315–318, 1998.

[102] F. van den Bergh and V. Lalioti. Software chroma keying in an immersive virtual environment. *South African Computer Journal*, 24:155–162, November 1999.

[103] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

[104] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 511–518, 2001.

[105] P. Viola and M. Jones. Robust real-time object detection. In *Proceedings of Statistical and Computational Theories of Vision Workshop*, pages 1–25, 2001.

[106] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision (IJCV)*, 57(2):137–154, 2004.

[107] Wikipedia. Cell shading. *http://en.wikipedia.org/wiki/Cel-shaded_animation*, 2005.

[108] Wikipedia. Lambertian reflectance model. *http://en.wikipedia.org/wiki/Lambertian_reflectance*, 2005.

[109] O. Williams, A. Blake, and R. Cipolla. A sparse probabilistic learning algorithm for real-time tracking. In *Proceedings of International Conference on Computer Vision (ICCV)*, volume 2, pages 353–360, 2003.

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

[111] D. Wright. The empirical rule. *http://www.intel.com/technology/computing/opencv*, 2003.

[112] S. Wright. *Digital Compositing for Film and Video.* Focal Press, 2001.