3

Software tools and apparatus

3.1 An architecture for object tracking

The methods for hand tracking to be described in Chapters 4, 5 and 6 of this thesis were developed within the context of the assistive workbench illustrated earlier in Figure 1.5 of the introductory chapter.

From that sketch, it can be surmised that the principal task modules are (i) rigid object tracking; (ii) hand tracking; (iii) head tracking; and (iv) hand tracking from the wearable robot. In addition there are the tasks of (v) data fusion and (vi) 3D visualisation. In this thesis we are concerned with tasks (i), (ii), (iv) and (vi). They have been implemented as separate processes in a client-server CORBA-like architecture, with communication via TCP sockets, as shown in Figure 3.1.

An object is tracked on the client side and visualised (and reasoned about) on the server side of this architecture. To avoid communicating graphics primitives, two instances of the object are maintained, one on either side. One way to do this would be to construct the instances simultaneously from a common configuration file which would specify the object type, communication port and initial configuration and pose. However, specifying the port in this way would require all objects to exist throughout. Instead, to enable the dynamic addition and removal of objects, the object is created first on the client side, and a predefined port is defined on the server side via which new objects can register their existence. When one does, the object is duplicated on the server, and a mechanism invoked which dynamically allocates a dedicated port for communication between the two instances.
3.1 An architecture for object tracking

Figure 3.1: The communication of object position, orientation and other configuration requires the existence of a corresponding object in the viewer application. The tracking blocks are independent processes.

This system and all video-rate code has been implemented in C++, using the Active Vision Laboratory’s Vision Workbench (VW) library. This library incorporates basic computer vision methods and some modules of numerical processing based on VXL [The03]. The graphical user interface is based on GTK- - and 3D visualisation methods uses OpenGL, which takes full advantage of any available accelerated hardware for 3D graphics. This frees the cpu from the heavy processing needed to render 3D objects. VW also defines standard interfaces to acquire images from cameras on-line, or from disk off-line.

All data files for information on 3D objects, camera configurations, colour classification data, and so on, are written in XML, where each information block lies between two readable and meaningful tags. This allows complex articulated objects to be modified by editing text files, without re-compilation. Figure 3.2 shows a sample graphical user interface created using these libraries showing the visualisation of cameras, hand, object and underlying desk.

The server and clients run under Linux. This is not a real-time operating system and cannot guarantee completion times. Even with careful algorithm design the occasional frame is dropped, particularly when
images are captured to disk while being processed. Care has been taken to account for the occasional variation in inter-frame duration. As suggested by Figure 3.2, up to three cameras are used in this work. In most of the experiments in this thesis, Sony VL500 digital cameras have been used with up to three sitting on the same Firewire (IEEE 1394) interface. Table 3.1 shows the frame rate obtained for various sizes of image and capture modes. If \( c \) cameras are connected to a single interface, all the cameras deliver one frame sequentially, so the maximum interval between the acquisition of an image from each camera is \( \frac{1}{rc} \), where \( r \) is the frame-rate. Although the Sony VL500s can be externally synchronized, other cheaper cameras cannot. In practice the speed of movement is sufficiently small for the time skew to be tolerable.

Mis-calibration of the cameras, however, is much less tolerable.
### 3.2 Camera calibration

The methods of object pose recovery using explicit 3D models described in Chapters 5 and 6 require cameras whose internal and external parameters are known. To estimate these parameters, a method based on Section 2.5 of [Tor02] is employed: the radial distortion and calibration parameters are estimated iteratively. To ease initialisation, the user clicks on the hinge of the calibration grid (shown in Figure 3.3b). Corner features are located using Harris’ corner detector [CH88] and the algorithm tries to fit lines to the un-distorted location of the corner features. The distortion is modelled using Harris’ formulation [Har92b], in which the relation between the displacement of an ideal image point and its radial distance $r_u$ from the centre of distortion is modelled as

$$r_d = r_u \left( \frac{1}{\sqrt{1 - 2\kappa_1 r_u^2}} \right)$$  \hspace{1cm} (3.1)

This is the forward distortion equation, where $\kappa_1 < 0$ models barrelling distortion and $\kappa_1 > 0$ models pin-cushion distortion. The backward equation (below) corrects measured distorted image points back to their ideal position:

$$r_u = r_d \left( \frac{1}{\sqrt{1 - 2\kappa_1 r_d^2}} \right)$$  \hspace{1cm} (3.2)

The matching error between the undistorted image and the ideal image is minimised with Levenberg-Marquardt to estimate $\kappa$, which is initialised with zero. An estimate of the calibration is then performed and the process is iterated until convergence.

The camera calibration step is based on the method described by Faugeras [Fau93] (at least for the intrinsic parameters as will become clear below). Given a 3D point $X$ represented in homogenous coordinates in the world coordinate frame, its projection $x$ onto the image plane of a camera (with radial distortion)

<table>
<thead>
<tr>
<th>Image Size</th>
<th>YCbC Mode</th>
<th>Max frame rate (Hz)</th>
<th>Max. no. cameras</th>
</tr>
</thead>
<tbody>
<tr>
<td>320×240</td>
<td>4:2:2</td>
<td>30</td>
<td>3</td>
</tr>
<tr>
<td>640×480</td>
<td>4:2:2</td>
<td>15</td>
<td>3</td>
</tr>
<tr>
<td>640×480</td>
<td>4:1:1</td>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 3.1: The maximum frame rate achievable for the given number of Firewire cameras, image size and capture mode.
3.2 Camera calibration

distortion corrected for) is taken as

$$\lambda x = PX,$$

(3.3)

where $P$ is the $3 \times 4$ projection matrix and $\lambda$ is a scale factor. If $X$ is in a Euclidean frame, $P$ can be decomposed as two meaningful geometric entities: the internal and external calibration parameters

$$P = K(R,t)$$

(3.4)

where the external parameters are the rotation and translation that transform points defined in the world coordinate frame into those defined in the camera frame, and where the internal calibration describes the transformation between an ideal image and the pixel image

$$K = \begin{pmatrix} f & \bar{s} & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix}.$$

(3.5)

Here $f$ is the focal length, $\alpha$ is the aspect ratio, $(p_x, p_y)$ defines the principal point, and $\bar{s} = -f_x s$ describes the often negligible image skew. As there are 6 rotational and translational DOF and 5 internal calibration parameters, a minimum of 6 correspondences $\{X_i \leftrightarrow x_i\}$ between known scene points and measured image point correspondences are required to recover $P$. However, a useful rule of thumb [HZ01] is that for a good estimation the number of constraints should exceed the number of unknowns by a factor of five, suggesting that around 30 correspondences is a practical minimum. The set of world points is defined by the corners of the squares on the ubiquitous 3D calibration grid (Figure 3.3), and the image positions determined to around $\pm 0.1$ pixel by fitting extended straight lines to the edgels computed along the edges of the squares, and then intersecting the lines.

Initial values for the elements of $P$ are found by a direct linear transformation. In practice it is safe enough here to set the scale by fixing $p_{34} = 1$ and recovering the other 11 elements, but more generally one should guard against $p_{34} \approx 0$ by recovering all 12 using a null space method. These initial values are then refined by non-linear minimization

$$P = \arg \min_{P'} \sum_i d(x_i, P'X_i)^2$$

(3.6)

where $d(x_i, P'X_i)$ is the Euclidean distance observation and estimation. Here the Nelder-Mead simplex method has been used [NM65] but others have used Levenberg-Marquardt [Lev44, Mar63] with equal
3.2 Camera calibration

![Image 1](https://via.placeholder.com/150)

![Image 2](https://via.placeholder.com/150)

Figure 3.3: (a) The Sony VL500 camera, three of which are used in this work. (b) The calibration grid in which the corner of the squares are used to compute \( \{x_i \leftrightarrow X_i\} \) correspondences. For calibration, best results are obtained if the grid occupies the whole image.

success. With \( P \) determined, the rotation and internal matrix is found by applying QR-decomposition to the inverse of the leftmost \( 3 \times 3 \) block of \( P \)

\[
\lambda' R^{-1} K^{-1} = Q R \leftarrow P_L^{-1}
\]

so that \( R = Q^{-1} \) and \( K = \lambda' R^{-1} \), where the scale is fixed so that \( k_{33} = 1 \). (There are also other sign ambiguities that between rows and columns of \( K \) and \( R \) that are resolved by requiring the focal length, aspect ratio and principal point coordinates to be positive.) The translation is determined by:

\[
t = R(p_{14}, p_{24}, 1)^\top.
\]

To generalise the calibration data for any image resolution used, and to benefit from statistical centring, a normalised image is employed. For an image with width \( w \) and height \( h \), the conversion of parameters is done as follows:

\[
\hat{f} = \frac{f}{w} \quad \hat{p}_x = \frac{p_x}{w} - 1 \quad \hat{p}_y = \frac{1}{w}(p_y - h).
\] (3.7)

3.2.1 Interpolation over zoom

The Sony cameras have controllable zoom lenses and it is convenient to be able to adjust these without performing a full re-calibration.

The process described above was repeated for values of zoom motor setting 40, 100, 200, 300, \ldots, 1300, 1400 from the accessible range of 40, 41, \ldots, 1432. For each calibration position, ten images of the grid were acquired (under small variations in lighting) for each of these zoom positions and the internal
3.2 Camera calibration

The plots in Figure 3.4 show the estimated focal length, aspect ratio and principal points for one of the cameras throughout the zoom range. The parameters are shown for normalised images. Note that the principal points are close to the image centre up to odometry position 900. The aspect ratio, which should be constant across the range of odometry positions, presented some small variation explained by the “mopping up” of errors elsewhere in the system.

The error bars are relatively small for most estimates because the image data was acquired with small intervals and small pose changes for each zoom position. But the obtained polynomials provided good generalisations across the zoom range, considering that calibration estimates were available from a very limited set of zoom positions.
3.2 Camera calibration

Figure 3.5: Variation of the principal point coordinate (for normalised images) with zoom for 3 cameras of the same model. Continuous curves show \( p_x \) and dashed curves show \( p_y \).

For other two cameras of the same model the focal length and aspect ratio showed all but identical behaviour, but there was greater variation in the principal point as shown in Figure 3.5. Again this is expected depends on the alignment of all the lens elements and image plane and would be harder to control in manufacture than the distance of the lens to the image plane.

3.2.2 Re-working the external calibration

In normal use, the internal calibrations of the Sony cameras, including the variation over zoom, have been found to remain valid over long periods of time. However, the external calibration is much more susceptible to change, by accidental bumping of furniture and so on. To avoid having to perform a complete
recalibration (and, hence, completely wiping out the benefits of interpolation) a more convenient planar calibration of the external parameters has been used. It has the advantage too of forcing the world’s $Z$-axis to be perpendicular to the desk.

Figure 3.6 shows the planar calibration object in use. The planar object defines the world $Z = 0$ plane, and its oriented pattern defines the $X$- and $Y$-axes. The projection equation (Eq. 3.3) is simplified to that of a plane to plane homography

$$
\lambda \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = P_{3 \times 3} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix}
$$

(3.8)

where $P$ is the homography whose elements $p$ can again be estimated linearly, up to scale, as the null space of $A$ in $Ap = 0$ where

$$
A = \begin{pmatrix} X_i & Y_i & 1 & 0 & 0 & -X_i x_i & -Y_i x_i & x_i \\ 0 & 0 & X_i & Y_i & 1 & -X_i y_i & -Y_i y_i & y_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
$$

(3.9)

for $n \geq 4$ points. Again one can refine the initial estimate by non-linear optimization. As the internal calibration is known, one can recover

$$
H = (h_1 \ h_2 \ h_3) = K^{-1} P,
$$

(3.10)

where $h_3$ is the translation and where $h_1$ and $h_2$ are the first two columns on the rotation matrix, all modulo a scale factor. Rotation matrices are orthogonal and have unit norm, and the actual translation and rotation matrix rows are first estimated as

$$(r_1 \ r_2 \ t) = \frac{2}{\|h_1\| + \|h_2\|} (h_1 \ h_2 \ h_3).$$

(3.11)

The third column of the rotation is first determined as $r_3 = r_1 \times r_2$. However, due to image noise and discretization problems, these columns will not be mutually orthogonal and of unit norm. This is corrected by means of the singular value decomposition (SVD) [Str88]

$$
U W V^T \leftarrow \tilde{R} = (r_1 \ r_2 \ r_3)
$$

$$
R = U V^T.
$$

(3.12)
(Here \( \mathbf{w} \) is the diagonal matrix of singular values of \( \tilde{\mathbf{R}} \), and \( \mathbf{U} \) and \( \mathbf{V} \) are both orthonormal matrices. The columns of \( \mathbf{U} \) form a basis in \( \mathbb{R}^3 \) for the range of \( \tilde{\mathbf{R}} \) and the columns of \( \mathbf{V} \) form the basis of the nullspace of \( \tilde{\mathbf{R}} \). If all the singular values are set to 1, the reconstructed matrix \( \mathbf{R} \) is the orthonormal rotation matrix closest to \( \tilde{\mathbf{R}} \) in the Frobenius distance sense \cite{PTVF88}.)

The pose estimation methods described both here and earlier estimate the rotation and translation of the observed object in the camera coordinate frame. Later it will be more convenient to describe the camera position in the world frame. This is simply the inverse Euclidean transformation

\[
\mathbf{R}_C^W = \mathbf{R}^{-1} \quad \mathbf{t}_{CW} = \mathbf{R}_C^W (\mathbf{t}) .
\] (3.13)

### 3.2.3 Detection and localisation of the calibration object

Some care has been taken to automate the detection and localization of the calibration object (Figure 3.7(a)). The initial detection is based on ring templates (Figure 3.7(b)), which are rotationally invariant but not easily confused with random darkspots in the image. To improve the robustness, the template is generated with a number of registered sample images obtained with slightly different scales and perspective.

The template is correlated with the image (Figure 3.7(b)) the maxima detected, and straightforward geometric reasoning determines the appropriate correspondence in the scene. The least squares estimation method of Section 3.2.2 (referred to as the linear method hereinbelow) gives the first estimate of the transformation which can be refined non-linearly using, e.g., Nelder-Mead simplex method over the sum of squared distances between the projected and measured disc centres (Equation 3.6). Examples results are shown in Figure 3.8.

It is of course necessary to improve the calibration by using more image evidence. This could be done just as earlier by using a grid to generate points and line and capturing one image, but here a different approach is taken. The estimate of the transformation is handed over to a tracker which refines the rotation and translation to best fit the projection of the known triangular and rectangular objects to the observed edges in the image, as shown in Figure 3.9. The RAPiD tracker is first described in its proper context in Chapter 4. However, in addition to its using more image data, the tracker makes repeated estimates allowing an assessment of the error that arises because of unmodelled variations in lighting,
3.2 Camera calibration

Figure 3.7: (a) Calibration pattern used for camera pose estimation, showing the origin and axes of the world coordinate. (b) Template for ring detection obtained from 15 images of rings; (c) Result of template match for a view of the scene shown in Figure 3.8.

image noise, and vibration, and because of the stochastic nature of robust pose and robust collinearity methods (described in Section 4.2.4).

3.2.4 Calibration results

The accuracy of the camera external parameters was assessed by determining the error in the angles from the world origin to the camera and the error in the distance.

The arrangement of cameras was as in Figure 3.6: they were widely separated, and zoomed out sufficiently far that the entirety of the 0.5m sided manipulation cube was visible, volume consistent with that based on biomechanical analysis and used by Mayol et al. [MTM02]. Figure 3.10 shows a view of the scene rendered using the information return from the calibrations of the three cameras.

To assess the likely error in the individual camera external calibrations, and hence the likely error in the position of an object recovered in three cameras, 21 images per camera were acquired at different times (without moving the cameras or target object) and the pose estimation algorithm was applied 1000 times per image. On average, the RAPiD tracker used 700 control points. Robust pose and robust
3.2 Camera calibration

Figure 3.8: Example View from camera 2 showing the detected feature points as asterisks and the projection of the corresponding scene points after linear and non-linear estimation.

Camera 2

Figure 3.9: Refinement of the pose estimations obtained in Figure 3.8 using the RAPiD 3D rigid object tracker with edge features.

collinearity (see Section 4.2.4) were applied using 20 and 10 iterations, respectively. The whole process was iterated 10 times per frame to ensure that starting transients had decayed.

Following a procedure suggested by Thompson et al. [TRMM01], the mean translation $\bar{t}_{WC}$ from the world origin to the camera was derived as was the mean unit direction $\bar{n}$ of the camera axis in world coordinate frame. The object pose in the camera coordinate frame, is expect to be estimated with less accuracy in translation along the camera axis and rotations in depth. These translate to camera pose estimates in the world coordinate frame (centred on the calibration object) as translation errors that are highly correlated in the three dimensions. Thence, the inaccuracy in translation was computed using a measure-
ment that is suitable for multivariate cases. Assuming that the distribution of the pose estimations can be modelled by a Gaussian, one can determine the principal axes by obtaining the eigenvalues and eigenvectors of the covariance matrix of the pose estimations \( \Sigma(t) = \sum_j (t_{WC,j} - \bar{t}_{WC})(t_{WC,j} - \bar{t}_{WC})^\top \). Each eigenvalue \( \lambda \) corresponds to the variance in the direction of its eigenvector. An inaccuracy measurement can be defined by the volume of the ellipsoid defined by these eigenvectors and eigenvalues through this formula (see Figure 3.11a):

\[
V = \frac{4}{3} \pi \lambda_1 \lambda_2 \lambda_3
\]  

(3.14)

A more meaningful measurement is obtained if the expected standard deviation \( \sigma \) can be used (recall that \( \sigma = \sqrt{\lambda} \)). This can be thought of as an expected value of the distance between the pose estimation and the mean of the estimated pose. This measurement can be defined by

\[
\Delta t = 3 \sqrt{\sigma_1 \sigma_2 \sigma_3}
\]

(3.15)

Note that the equality \( \sigma_1 \sigma_2 \sigma_3 = \sqrt{\det(\Sigma(t))} \) simplifies the implementation of the last equation.

The error in rotation was expressed by two angles, \( \Delta \alpha \) which is the standard deviation in the angle between the individual direction vectors and the mean \( \bar{n} \), and \( \Delta \theta \) the standard deviation in the cyclotorsion about \( \bar{n} \), as shown in Figure 3.11. Table 3.2 shows the inaccuracy of the pose estimation, computed with the 21000 trials for each camera.
3.2 Camera calibration

![Diagram of ellipsoid and angles](image)

Figure 3.11: (a) Ellipsoid defined by the principal standard deviations of the position estimation in the translation space. (b) Angles used to evaluate the deviation in rotation axis \( \alpha \) and in rotation angle \( \theta \).

<table>
<thead>
<tr>
<th>Camera</th>
<th>( \Delta t ) mm</th>
<th>( \Delta \theta )°</th>
<th>( \Delta \alpha )°</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8.1</td>
<td>0.48</td>
<td>0.12</td>
</tr>
<tr>
<td>1</td>
<td>4.6</td>
<td>0.46</td>
<td>0.15</td>
</tr>
<tr>
<td>2</td>
<td>9.3</td>
<td>0.85</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Table 3.2: Pose estimation inaccuracy for the position of the cameras \( \Delta t \), the orientation of their optical axis \( \Delta \theta \) the rotation about this axis \( \Delta \alpha \). Position and orientation are expressed in the world coordinate frame.

Now it should be noted that these errors are from individual camera’s calibration. If it is assumed that the camera position is correct, the error can be transferred back to a point in the scene. But because the cameras are close to orthogonal, the error ellipsoids for a point viewed close to the centre of each camera will intersect orthogonally, and the resulting error covariance is not ellipsoidal, but can be approximated by a sphere of radius

\[
r \sim \Delta \alpha \left( \frac{\pi}{180} \right) D
\]

where \( D \) is the typical distance from camera to scene. Here \( D \sim 1000 \) mm, giving a translational error in an observed object of \( r \) between 2 mm and 4 mm. One expects this error to scale proportionally with depth, but inversely with focal length because a fixed error in the image corresponds to a smaller angular error. For this reason, where zoom lens is available, the pose estimate is done by zooming into the calibration object. Using the interpolation method described earlier, once the pose estimate is done, the cameras can zoom out to increase the field of view for the tracking experiments.
3.3 On the detection of hands images: a skin colour classifier

The third competence developed to support the research in the remaining chapters is that of hand detection. Any markerless visual method designed to detect and track an object without intervention must confront the question of how in the first instance to associate features observed in the image (be they pixels, edges, corners, etc.) with the object itself. Locating hands is likely to be difficult compared with, for example, face detection because hands are articulated objects that present both high variation in their shape and in their degree of self-occlusion. However, there is a useful uniformity in human skin colour allowing the development of a localization method based on pixel colour classification. As the review has indicated, and Chapters 4 and 7 will show, the resulting silhouette is sometimes all that is needed for 3D pose estimation. If internal edges are to be used, as they are in Chapters 5 and 6, the silhouette is still valuable in restricting search for an initial pose.

3.3.1 Eliminating brightness from the colour space

Classification based on colour requires pixels imaged from skin to form a tight cluster in some colour space. Although we loosely describe skins as being of different colour, the spectral variability is dependent mainly on the amount, density and distribution of melanin pigment in the skin, not on its colour [Mar02]. Thus, to a large extent it is the brightness of the skin that varies, not its colour [YLW98a]. Brightness normalisation involves reducing the dimensionality of a colour space (typically from three dimensions to two) by projecting points into a plane of constant brightness in the space. It is inevitable that un-modelled variations result in some overlap in the 2D space between skin and non-skin clusters, but the drop in dimensionality substantially cuts the volume of data and time required for training. Moreover, if the colour space decouples brightness and colour information from the outset, the task of brightness normalisation can be achieved by neglect rather than computation.

3.3.2 The choice of a colour space

Researchers into colour science, an important area long before the digital era, have proposed a large number of colour spaces each tailored to a different task. Among colour spaces that are decoupled, the most common — and commonly used for skin detection — are the CIE Chromatic space (used in, for ex-
3.3 On the detection of hands images: a skin colour classifier

ample, \([\text{YLW98b}]\), the HSV (hue, saturation and value) space (e.g. \([\text{RMG98}], \text{[AP96]}, \text{[ZYW00]}\)), and the YUV/YCbCr space (e.g. \([\text{Coh, Fri99}, \text{YLW98a}, \text{FdC00}])\). Several comparisons of spaces for skin detection have been carried out, but Martinkauppi’s thesis \([\text{Mar02}]\) suggests that there is no definitive conclusion as to which is the best, in part because different databases with different illumination conditions have been used, but principally because the differences in output quality are marginal. Explanatory and exploratory notes about these spaces are given in Appendix A.

Since the goal here is to implement a video-rate method, processing time becomes the key criterion by which to assess methods. Now the decision becomes straightforward. The conversion to HSV or HSL requires a non-linear transformation algorithm, making this the least efficient in terms of computational cost. Conversion to CIE is linear and fast enough. However, it turns out that many digital colour cameras, like the Sony VL500 used here, deliver images already encoded by hardware in the YCbCr space. As explained in Appendix A the Y channel holds the luminance information, which is to be neglected, and the Cb and Cr channels hold the chrominance information which is to be used for classification.

3.3.3 Classifying pixels for skin detection

Several methods have been applied to the problem of classification of skin pixels. \([\text{DHS00}]\). The simplest “manually” carve out a portion of colour space to be classified as skin \([\text{CnN98}]\), defining it by thresholds or a lookup table. More common is to allow different colours to have a probability of arising from skin, and to learn the underlying PDF. Within this approach there are variations in how the distribution of skin samples is modelled.

Yang and Waibel \([\text{YW96}]\) (and see \([\text{FdC00}])\) argue that a Gaussian PDF is good enough for their small dataset of skin colour samples. However, is not able to account for subtle variations in large databases. Nor do Yang and Waibel include training data to model the background colour distribution. This is assumed to be uniform, and a simple threshold in the PDF of the skin class defines the decision boundary. Multi-modal Gaussian mixtures were proposed by Jebara \([\text{JP97}]\): indeed any two-class classifier could be applied to this problem, but doing so would miss the point that the feature space has only three or fewer dimensions and the classes do not need to be modelled analytically.

Jones and Rehg \([\text{JR98}, \text{JR02}]\) have shown that non-parametric histogram models provide higher accuracy and lower computational cost than using multi-modal Gaussian mixtures. However, their clas-
3.3 On the detection of hands images: a skin colour classifier

Figure 3.12: Screen shot of the application to train the classifier for skin detection: the classification result is shown in the top left (skin is indicated by red, background by black and unknown by white). The panel on the bottom left shows the training areas already selected by the user for skin (dashed) and background (solid).

Skin colour detection was done in RGB colour space, which is less robust to illumination changes. This problem is largely eliminated in the truncated (Y)CbCr colour space, and it is this approach which is developed here.

Histogram-based classification in the CbCr colour space

Skin colour detection is modelled as a maximum a posteriori classification problem, using histograms to model discrete PDFs [FP03]. During training histograms are built in the colour space for each class involved — here there are just two, skin $S$ and background $B$. If a pixel with colour $(C_b, C_r)$ is known to be in the class $S$ the bin count $c_S[uv]$ is incremented, where

$$ u = \text{floor} \left( \frac{C_b}{b} \right) \quad v = \text{floor} \left( \frac{C_r}{b} \right) ,$$

and $b$ is the bin size. The resulting bin counts are normalized so that

$$ P(uv|S) = \frac{c_S[uv]}{T_S}$$

where $T_S = \sum_{uv} c_S[uv]$ is the total number of pixels labelled as skin during training.

During classification the posterior is determined using Bayes’ rule

$$ P(S|uv) = \frac{P(uv|S)P(S)}{P(uv|S)P(S) + P(uv|B)P(B)}$$

(3.18)
where the prior probability is \( P(S) = T_S/T \) and \( T \) is the total number of pixels used in training. The likelihoods and priors involving the background are defined similarly (and as this is a two-class problem can be derived without storing a second histogram). Then a particular pixel with its colour \((u, v)\) is labelled as skin if

\[
P(S|uv) > P(B|uv),
\]

which can for this two-class problem be simplified to \( c_S[uv] > c_B[uv] \). (When there is neither skin nor background training sample for a given \( uv \) bin, an uncertainty arises, as \( P(S|uv) = P(B|uv) = 0 \). For the skin detection application, what matters in this case is that it is known that this \( uv \) bin does not represent a skin value, so it is classified as background.)

**Iterative Training Method**

The training process consists of selecting skin and background regions of images. This task is performed manually and can be very tedious for a large training set. But often only a few images are enough to create a good model for classification. To inform the user, a train-and-classify system was implemented. Before the user starts segmenting a new image, the system shows the classification result for this image using the current training set. The user then judges whether it is necessary to use this image for training or not depending on the size of miss-classified areas in the image. For each image, the software shows a track of the areas already selected by the user and it does not add samples from areas selected before. This idea is similar to the iterative training method proposed by Saxe and Fouls [SF96]. Figure 3.12 shows a screen shot of this software.

**3.3.4 Qualitative evaluation**

**Tuning the generalisation power via the bin size**

The training method described in Section 3.3.3 was used to populate the CbCr space with more than 500 thousand skin samples and more than 1.2 million background samples obtained from the image database described in Section A.5. Both Cb and Cr ranges were 0-255, and the bin size was \( b = 1 \). Figure 3.13(a) shows the histogram of the skin samples in the CbCr colour space, and Figure 3.13(b) shows the area of this colour space populated by skin samples some 1273 CbCr locations. The same is done for the
background in figure parts (c) and (d), where 5144 CbCr locations are populated. The skin “area” is quite large because samples were acquired under different illumination conditions.

Lookup tables were built from these histograms. The first, in Figure 3.13(e), retained the bin size of 1, but it was found to leave gaps in the skin region. Figure 3.13(f) shows the result of increasing the bin size to \( b = 2 \). Many of the unknown CbCr classification values are extinguished. The effect of increasing the bin size on an arbitrary image is shown in Figure 3.14.
3.3 On the detection of hands images: a skin colour classifier

Figure 3.14: (a) An arbitrary image, and the effect on increasing the bin size from \( b = 1 \) in (b) to \( b = 2 \) in (c) for skin detection. (The original image (a) is ©2006 http://www.palhacomatraca.com.br, reproduced with permission.)
3.3 On the detection of hands images: a skin colour classifier

Results in uncontrolled conditions

Further results are shown in Figure 3.15. Some of these images were obtained with the camera set to adjust the brightness and contrast automatically. Such automatic modes also normalise image colours to “improve” the appearance to the human eye under variations in illumination. Others have cluttered background and include wooden objects whose colour is often close to that of skin.

![Figure 3.15: Input images and their classification results. Some of them present challenging background with wooden objects whose colour is similar to skin colour.](image)

3.3.5 Sources of noise and dealing with them

Even when the illumination is controlled and the camera parameters are static, there are several sources of noise that lead to mis-classification. Table 3.3 lists some of the sources of noise and some possible methods to reduce their effect.
3.3 On the detection of hands images: a skin colour classifier

### Table 3.3: Sources of noise and methods that can be used to reduce their effect.

<table>
<thead>
<tr>
<th>Source of noise</th>
<th>Alleviated by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saturated white pixels and black shadows</td>
<td>Adaptive iris</td>
</tr>
<tr>
<td>Compression artifacts</td>
<td>Interfaces with no compression</td>
</tr>
<tr>
<td>Light oscillations</td>
<td>Increase generalisation of the classifier</td>
</tr>
<tr>
<td>Colour subsampling</td>
<td>Smoothing or morphological operations</td>
</tr>
</tbody>
</table>

One of the less expected of these is a systematic mis-classification of pixels at the edges of objects, where the colour appears to belong neither to the object nor the background. In digital colour cameras, the image is acquired on a planar CCD array composed by grey level photosensors laid behind colour filters. These colour filters are usually arranged in the Bayer pattern [Bay76], as shown in Figure 3.16. Each pixel is composed of four subpixels, one red, one blue, and two green. These proportions acknowledge the human eye’s greater resolving power in green light.

![Bayer arrangement of colour filters on the pixel array of an image sensor.](image)

Spatial aliasing occurs at sharp edges since each colour is acquired from a different position. This is illustrated in Figure 3.17(a), where pixels on the edge of a white square on a black background are assigned to intermediate colour tones. An example from a real image in 4:1:1 YCbCr format is shown in Figure 3.17(b), where various tints are visible around the ring.
3.3 On the detection of hands images: a skin colour classifier

![Image](image1.jpg)

Figure 3.17: (a) An illustration of colour aliasing due to subsampling for CCD arrays with Bayer filters. (b) Detail of a YCbCr 4:1:1 image of a black disc on a white background where colour aliasing is evident near the edges. Subject to limitations of colour reproduction on paper, the enlarge region is tinted yellow.

3.3.6 Noise reduction

A variety of more or less principled methods can be applied to reduce noise, all implementing spatial low-pass filtering of some sort, and all assuming that the hands are of substantial size in the image. Routinely used are computing connected regions and thresholding on their area, followed by median filtering (e.g. [GW00]). Figure 3.18 shows a typical result of applying both these techniques. Among other possible alternatives is the successive application of opening and closing morphological operations.

Another approach (perhaps one that is acceptable only because the emphasis of research is elsewhere)

![Image](image2.jpg)

Figure 3.18: Noisy classification results, such as that shown in (b), can be improved using large blob segmentation followed by the median filter, leading to the result shown in (c).
3.3 On the detection of hands images: a skin colour classifier

Figure 3.19: Classification under more constrained situation: (a) skin and background clusters; (b) classification look up table obtained using bin of size 10. Black represents skin colour and grey represents background.

is to apply the low pass filter to the environment, reducing the variability in the lighting, the degree of clutter, and cameras settings. In this kind of situation, the clusters of skin and background can be very compact and have little intersection. As an example, Figure 3.19(a) shows the clusters obtained from a background that consists of a dark wooden table, and from the hands of four subjects under stable illumination. Note that the clusters are so well separated that a simple threshold in the Cb channel could classify them. To allow more generalisation and noise in the background, the histogram-based classifier was used with bins of size 10, making the lookup table very compact and classification very fast: the average processing time for $640 \times 480$ images was 10ms using a 1 GHz Pentium 4. The resulting hand segmentations are shown in Figure 3.20 where no post-processing has been applied.
3.3.7 A note on adaptation

Although brightness normalisation provides robustness to light intensity variations, it will not account for changes in the colour of the ambient lighting or reflected light. There are two approaches to retaining the compact skin clusters already computed: first, adapt the camera parameters to the environment, so that the colour appearances are the same as during training; or, second, adapt the lookup table to the illumination conditions by using samples of skin coloured objects in the image.

The first option is achieved by the process of white balance, which is usually implemented in the hardware of video cameras or digital still cameras. White balancing a camera is done by acquiring the image of a white region of the scene. The camera then shows true white as white and adjusts all the other colors accordingly [WS00]. The second consists of locating an area of the image which is known to be skin coloured. Once such area is located, the maximum and minimum values in each channel (CbCr) in this area can be used to translate and scale clusters of the training set and the lookup table can be updated. In [May04], for example, a skin patch is selected manually in the beginning of acquisition, and in [SWP98b] part of the user’s face is guaranteed always to be visible in the bottom of the image. Alternatively, a face detection method that works independently of colour such as Viola and Jones’ method [VJ01] could be used to locate a skin patch. For situations in which the only skin coloured object are hands, Han et al. [HASW06] collect samples of skin from the first frame using a rough skin colour model in RGB space. A region grow-based algorithm is applied to collect more training samples of skin and these are used to train a SVM classifier.

3.4 Summary

In this chapter, the software tools and apparatus used throughout this thesis have been described.

First, the architecture and software system underlying the video-rate tracking of multiple objects with multiple cameras was outlined. This is a modular system, built in the context of a larger project, that can distribute tasks over processors and communicates results via sockets.

The use of calibrated cameras facilitates model-based tracking of objects in 3D using multiple cameras. The second section of the chapter described the two methods of camera calibration. The first method, based on a 3D calibration grid, was used to recover the internal calibrations of the three Sony
3.4 Summary

digital cameras over their complete range of zoom settings. In order to avoid having to repeat this complete calibration if the cameras were moved, a second method based on a planar tile was used to recover the external parameters. A method for detecting and determining the alignment of the planar object was described, and the accuracy of recovery when three cameras are used was assessed.

The last section of the chapter outlined the classification method used to detect initially where the hand is located in the image. Because skin varies predominantly in brightness, not colour, brightness normalisation reduces the dimensionality of the pixel and makes skin colour clusters more compact, while retaining distinguishability from background pixels. Results in Appendix A confirmed the view in the literature that there is little difference in terms of colour separation between those colour spaces that use brightness as one of its axes, so the choice of colour space was determined by computational cost.

Classification of colour pixels was achieved by learning the likelihood of a pixel of some colour arising from a particular class, and using Bayes’ rule to determine the posterior. This is a simple and effective method that is able to model classes that have multimodal and discontinuous PDFs, essential for large datasets of skin colour pixels acquired with different cameras under different illumination, and essential for modelling the background class. The bin size of the histogram can be set to be inversely proportional to the number of training samples. The larger the bin, the more general is the model and the faster is the classifier.