Visual Perception System for a Social Robot (VISOR)

Deliverable 1

SKIN COLOUR DETECTION, FACE DETECTION AND FACE RECOGNITION

Due date of deliverable: December 15th, 2005
Actual submission date: December 15th, 2005
Start date of project: September 15th, 2005 Duration: 12 months

Organizational name of responsible for this deliverable:
Institute of Systems and Robotics

Revision: 0.2
Dissemination level: PU

Grupo de Ingeniería y Sistemas Integrados, University of Málaga (Spain)
Institute of Systems and Robotics, Coimbra (Portugal)

Deliverable 1.1

SKIN COLOUR DETECTION

Rebeca Marfil
Grupo de Ingeniería y Sistemas Integrados
Dpto. Tecnología Electrónica
Universidad de Málaga
Campus de Teatinos 29071 Málaga (Spain)
www.grupoisis.uma.es

Contents

List of Figures

1 Introduction

A social robot can be defined as a robot which is capable to not only perform some predefined tasks but also to interact with its environment in general and with humans in particular [1]. To do that a social robot must have a visual system with a biological-plausible attentional mechanism. That is, an attentional mechanism which attempts at imitating the behavior of the human attentional mechanism. This attentional mechanism should determine which regions or objects from the robot visual input are important for the robot and should to be more detailed studied. Using this information, the visual system of the social robot should be capable of solve some specific problems as: identify faces, measure head and hands poses, capture human motion, recognize gestures and read facial expressions to emulate human social perception. Therefore, the robot could identify who the human is, what the human is doing and how the human is doing it.

A first step to identify a human in the scene is to determine skin colour regions. These are the regions where a face or hand is likely located. Skin-color segmentation approaches can be grouped into two basic categories [2]: physically-based approaches [3, 4] and statistical approaches. Statistical approaches can be subdivided further into: parametric approaches [5, 6, 7, 8] and nonparametric approaches [9, 10]. Parametric statistical approaches represent the skin-color distribution in parametric form, such as a Gaussian model [5, 6, 7] or a mixture of Gaussian model [8]. The key advantages of parametric models are: i) low space complexity and ii) relatively small training sets are required. The major difficulty in the case of Mixture of Gaussians is order selection. Nonparametric statistical approaches [9], [10] use histograms to represent density in color space. A major advantage of the histogram representation is that the probability density function can be evaluated trivially regardless of the complexity of the underlying distribution. A major drawback is that the histogram approach requires a considerable amount of training data. Physically-based approaches [3, 4] made direct use of a physical model of skin reflectance. The reflectance model is used to discount a known, time-varying illuminant to obtain color constancy. Segmentation tends to be more accurate due to the algorithm's use of strong prior knowledge: camera and illumination parameters, as well as initial image segmentation. However, such information may not be readily available in analysis of everyday image sequences.

In this report, a parametric skin colour segmentation approach is pre-

sented. This approach is based on the work of Terrillon *et al*. [11, 12] which uses a skin chrominance model built over the TSL (Tint-Saturation-Luminance) colour space. It assumes that the chrominance of Caucasian skin can be modelled by an unimodal elliptical Gaussian joint probability density function. Once the model is built, the Mahalanobis metric is used to discriminate between skin and non-skin pixels.

2 Chrominance model of Caucasian human skin

2.1 Colour space

The first step to build a chrominance model is to select an adequate colour space. This space must provide robustness to illumination variation. That is achieved if the colour space efficiently separates the chrominance from the luminance. Then, only the chrominance component is used. Besides, the chosen colour space should provide a confined and easily to model skin color distribution. Therefore, the efficiency of the skin colour segmentation depends on the selected colour space, because the colour distribution of human skin depends on the colour space.

Terrillon and Akamatsu in [12] present an analysis of the performance of nine different colour spaces used in skin colour segmentation. They study the distribution of the skin colour in normalized r-g, CIE-xyz, TSL, CIE-DSH, HSI, YIQ, YES, CIE-L*u*v* and CIE-L*a*b* spaces. The CIE-DSH and HSI are discarded because the skin colour distribution is not confined. Terrillon and Akamatsu also propose a skin colour chrominance model for Asian and Caucasian skin which uses an unimodal elliptical Gaussian joint probability density function. In YIQ, ES, CIE-L*u*v* and CIE-L*a*b*, although the skin colour is confined it can not be modelled by a single model, so they discard these spaces too. The TSL and normalized r-g yield the best fit to this model. In their experiments, the best colour space to segment skin colour regions and detect faces is the TSL colour space.

In our work, we have implemented the skin colour chrominance model proposed by Terrillon [11, 12] using the TSL colour space. This space is obtained from the RGB space using the following transformation:

$$S = \sqrt{\frac{9}{5} * (r'^2 + g'^2)} \tag{1}$$

$$T = \begin{cases} \frac{\arctan(\frac{r^{'}}{g^{'}})}{2\pi+\frac{1}{4}} & g^{'} > 0 \\ \frac{\arctan(\frac{r^{'}}{g^{'}})}{2\pi+\frac{3}{4}} & g^{'} < 0 \\ 0 & g^{'} = 0 \end{cases} \tag{2}$$

$$L = 0.299R + 0.587G + 0.114B \tag{3}$$

where

$$r^{'} = (r - \frac{1}{3}) \tag{4}$$

$$g^{'} = (g - \frac{1}{3}) \tag{5}$$

$$r = (R - \frac{R}{R+G+B}) \tag{6}$$

$$g = (G - \frac{G}{R+G+B}) \tag{7}$$

T is the tint, S the saturation and L the luminance, normalized in the range [0,...,1.0].

2.2 Skin chrominance model building

The first step to build the skin colour model is to determine the portion of the TS colour space where the Caucasian skin colour is confined. The TS colour space is the TSL space where the luminance component has been removed. To do that, we have used a set of 108 indoor and outdoor colour images with faces and hands. These images have been manually segmented, extracting the skin colour regions of faces and hands from the background and computing a set of 108 skin colour masks. Fig.1 shows ten of the training images and their corresponding masks used in the skin chrominance model building process. The set of masks is used to compute a cumulative histogram with the TS values of the skin colour pixels. Fig.2 shows this histogram. It must be noted as the distribution of the skin colour is confined. We assume that the distribution shown in Fig.2 can be modelled by an unimodal elliptical Gaussian joint probability density function given by

$$p[\bar{X}(i,j)/W_s] = (2\pi)^{-1}|\bar{\bar{C}}_s^{\frac{-1}{2}}| \exp[\frac{-\lambda_s^2(i,j)}{2}] \tag{8}$$

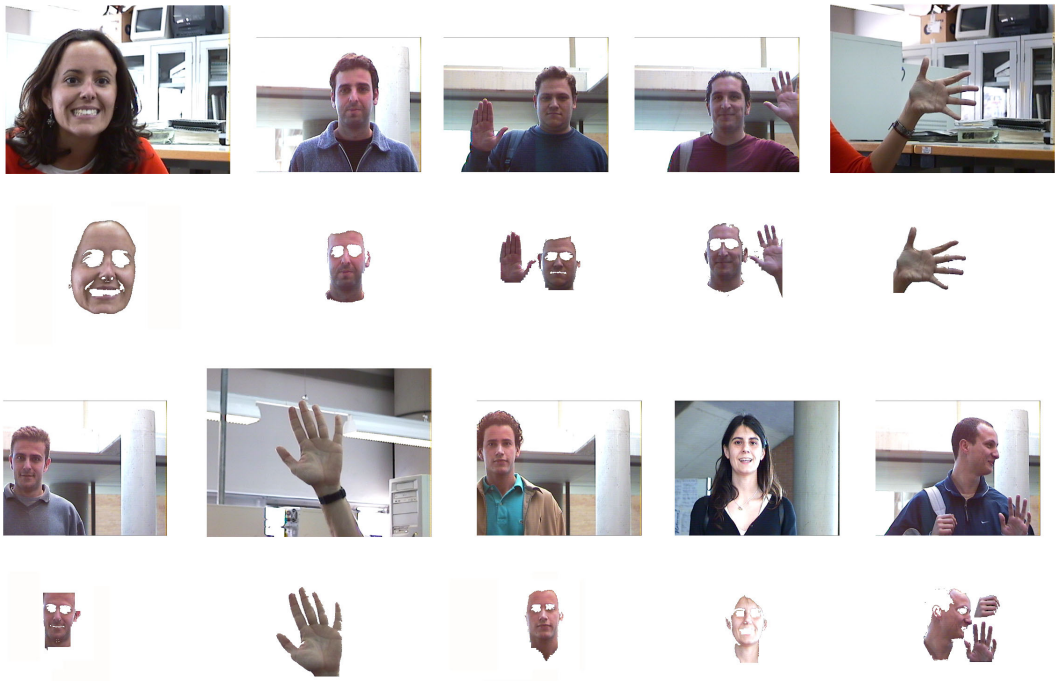

Figure 1: Training images and masks.

where $X(\bar{i},j) = [\bar{T}(i,j)\bar{S}(i,j)]^T$ represents the random measured values of tint and saturation of a pixel with coordinates (i,j) in an image. W_s is the class describing the skin colour. $\bar{\bar{C}}_s$ is the covariance matrix of the skin colour distribution:

$$C_s = \begin{bmatrix} \sigma_{T_s}^2 & \sigma_{TS_s} \\ \sigma_{TS_s} & \sigma_{S_s}^2 \end{bmatrix} \tag{9}$$

and $\lambda_s(i,j)$ is the Mahalanobis distance from vector $\bar{x}(i,j)$ to the mean vector $\bar{m}_s = [m_{T_s} m_{S_s}]^T$ obtained from the skin colour distribution. Equation (8) means that the probability of a pixel to be a skin colour pixel depends on the covariance matrix of the skin colour distribution as well as on the Mahalanobis distance between the pixel colour and the mean colour of the skin distribution. Therefore, the larger $\lambda_s(i,j)$, the lower the probability that the pixel be a skin pixel.The Mahalanobis distance is given by

$$[\lambda_s(i,j)]^2 = [\bar{X}(i,j) - \bar{m}_s]^T \bar{\bar{C}}_s^{-1} [\bar{X}(i,j) - \bar{m}_s] \tag{10}$$

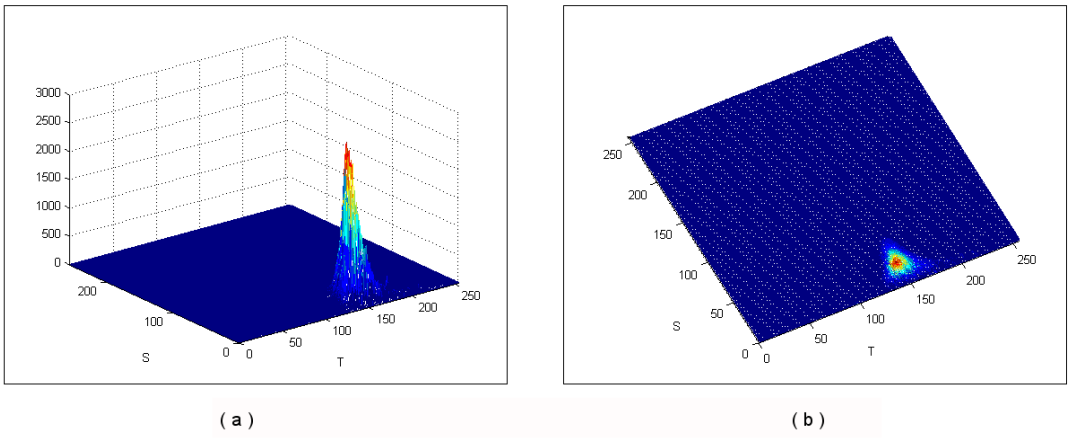

Figure 2: a) Histogram of the skin colour distribution; b) top view of the histogram.

Equation (10) defines elliptical surfaces in chrominance space of scale $\lambda(i,j)$, centered about $\bar{m}_s$ and whose principal axes are determined by $\bar{\bar{C}}_s$.

Equations (8) and (10) show that the skin colour chrominance model is whole described by $\bar{m}_s$ and $\bar{\bar{C}}_s$. The values obtained for the skin colour distribution shown in Fig.2 were the following:

$$\bar{m}_s = [\ 149.0228 \quad 23.0944\ ] \tag{11}$$

$$\bar{\bar{C}}_s = \begin{bmatrix} 0.0058 & 0.0009 \\ 0.0009 & 0.0094 \end{bmatrix} \tag{12}$$

3 Skin colour segmentation

Once the parameters of the model have been computed from the training masks, the model can be used to segment skin colour regions from real images. The process to segment an input image is the following:

1. The RGB input image is transformed in a TSL image applying equations (1), (2) and (3) to all pixels.

2. $\lambda(i,j)^2$ is computed for each pixel of the input image using equation (10). Each value is compared with a threshold λ_T^2.

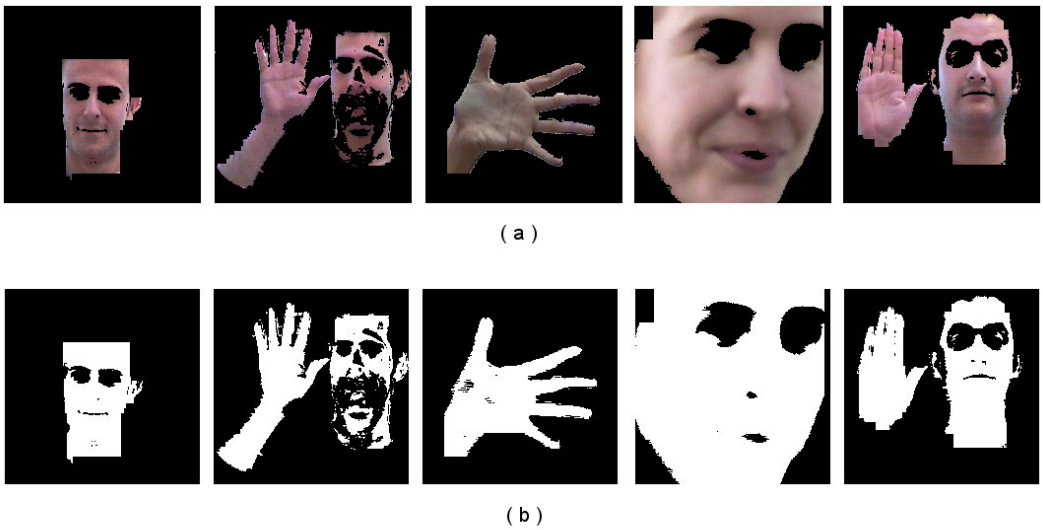

Figure 3: a) Masks; b) skin detection results $\lambda_T^2 = 10.0$.

3. A value of 1 is assigned to pixel (i, j) if $\lambda(i, j)^2 \leq \lambda_T^2$. Otherwise, pixel (i, j) is set to 0.

The output of the skin colour segmentation algorithm is a binary image where the skin colour pixels are set to 1 and non-skin colour pixels are set to 0.

The threshold λ_T^2 depends on the used camera and can be computed studying the portion of false positives and false negatives [12] in the segmentation. In our case, we have studied the Mahalanobis distance in the pixels of a set of 18 background images without skin regions. Using a histogram of distances, we compute the distance value in which the percentage of pixels classified as skin (false positives) is equal to a desired value. This distance is chosen as threshold. With this method a range of false positives between 10% and 28% produces $\lambda_T^2 \in [6..10]$.

Figs.3.b and 4.b show some results obtained with $\lambda_T^2 = 10.0$, using the training images as inputs. While the masks are correctly segmented (Fig.3.b), a lot of incorrectly classified pixels appear in the non-skin images (Fig.4.b). These pixels are in grey coloured regions. The grey colour is characterized only for its Saturation value in the TSL space. Specifically, grey pixels have a small saturation value while the tint and luminance can be a random value. Therefore, some gray values are included in the computed skin colour

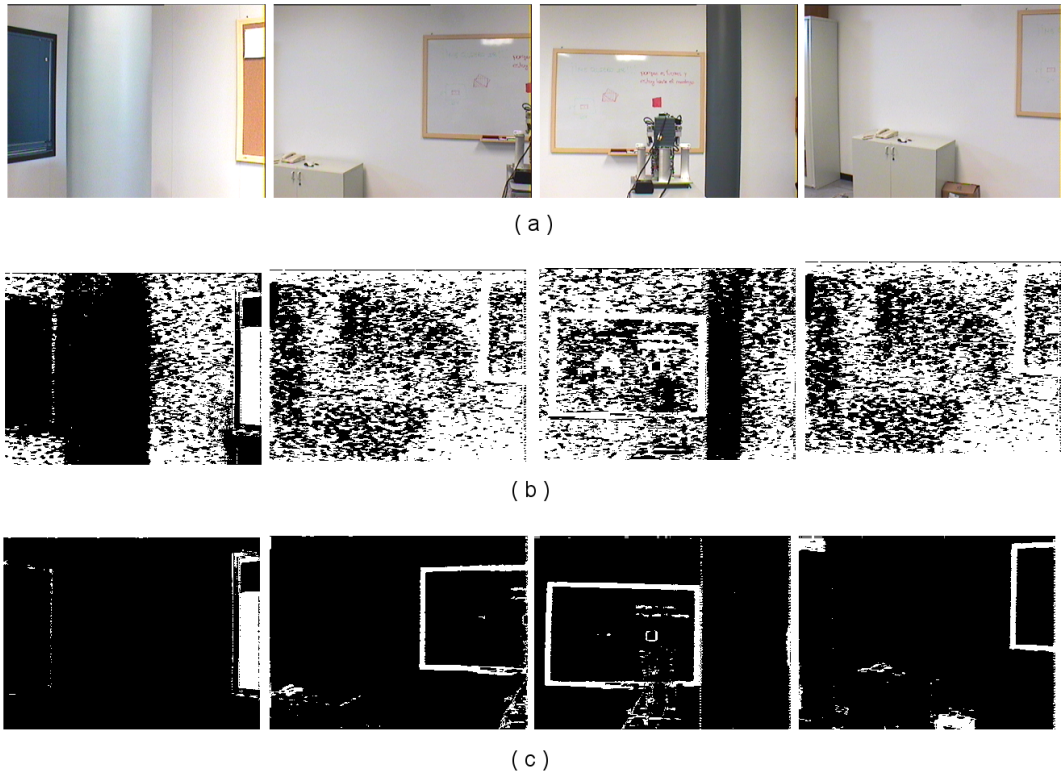

Figure 4: a) Non-skin images; b) skin detection results $\lambda_T^2 = 10.0$; c) Skin detection results $\lambda_T^2 = 10.0$ $S_T = 10$.

distribution (Fig. 2). In order to solve this problem a new threshold S_T (saturation threshold) has been included in the model. Only pixels with a saturation value higher than S_T are included in the model. Fig.4.c shows the new results obtained with $\lambda_T^2 = 10.0$ and $S_T = 10$.

Using the two previous thresholds λ_T^2 and S_T, the algorithm still has some problems with black regions, as it is shown in Fig. 5.b. Black colour is characterized in TSL space with its low L value. Black regions can have random values in T and S. As the model only takes into account the T and S values of colours, then it is possible to classify a black pixel like a skin coloured pixel. To avoid that, another threshold (L_T) is used in the model. Only pixels with $L > L_T$ are included in the model. Fig.5.c shows the new results obtained with $\lambda_T^2 = 10.0$, $S_T = 10$ and $L_T = 80$.

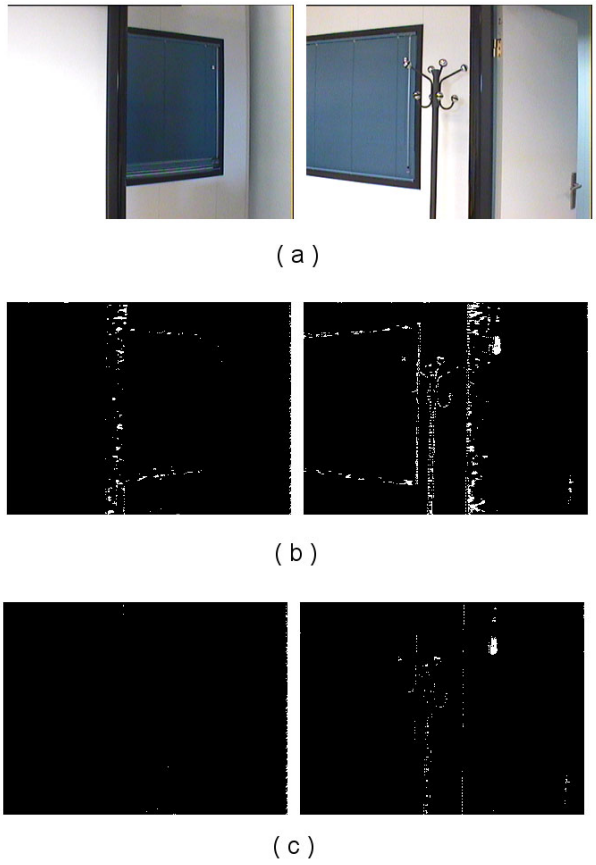

Figure 5: a) Non-skin images; b) skin detection results $\lambda_T^2 = 10.0$ $S_T = 10$; c) skin detection results $\lambda_T^2 = 10.0$ $S_T = 10$ $L_T = 80$.

4 Experimental results

In this section, a set of experimental results is shown. These results have been obtained using two different cameras: the camera used to take the training images and a different camera in order to illustrate the variation of the thresholds.

The thresholds used with the camera employed in the capture of the training images have been: $\lambda_T^2 = 10.0$, $S_T = 10$ and $L_T = 80$. Fig. 6 shows some results obtained with this camera.

Fig.7 shows the results obtained with another camera. The thresholds used in this case have been: $\lambda_T^2 = 40.0$, $S_T = 20$ and $L_T = 60$.

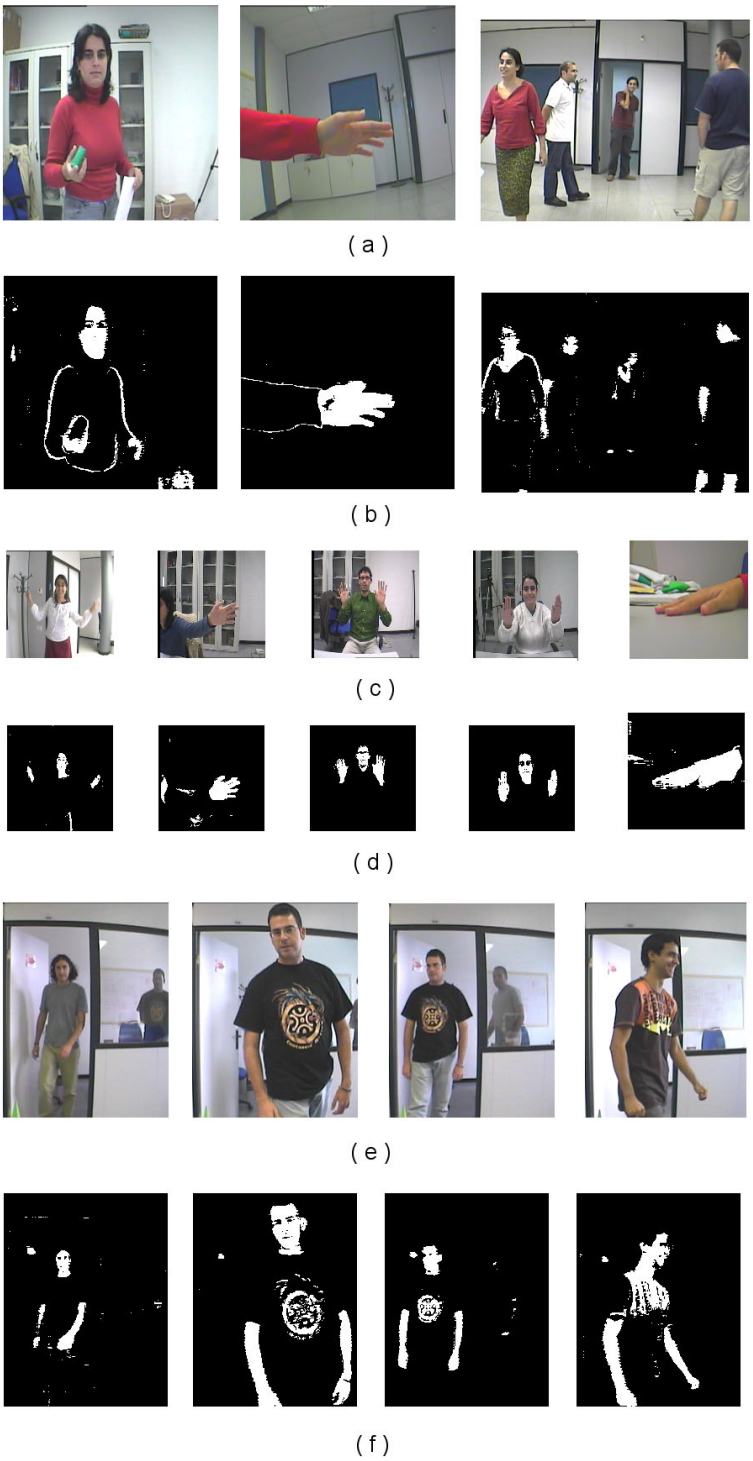

Figure 6: a-c-e) Original images; b-d-f) skin detection results $\lambda_T^2 = 10.0$ $S_T = 10$ $L_T = 80$.

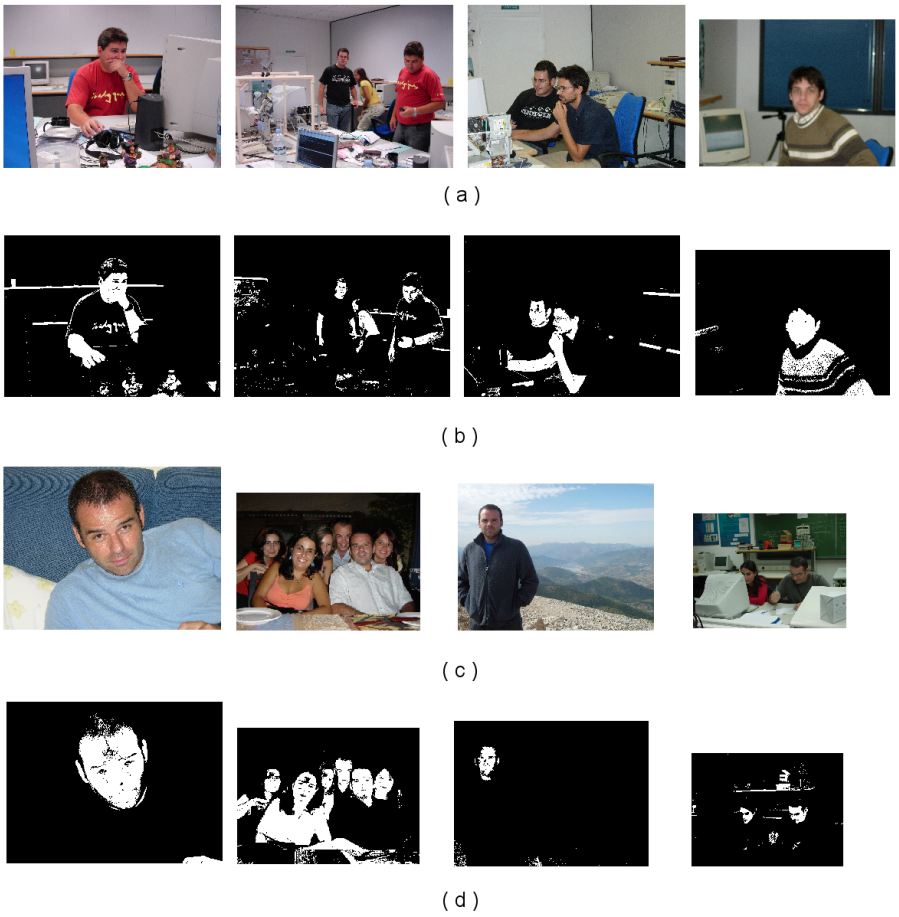

Figure 7: a-c) Original images; b-d) skin detection results $\lambda_T^2 = 40.0$ $S_T = 20$ $L_T = 60$.

References

[1] C. Breazeal, *Designing sociable robots*, M. Press, Ed. Cambridge-MA, 2002.

[2] L. Sigal, S. Sclaroff, and V. Athitsos, "Skin color-based video segmentation under time-varying illumination," *IEEE Trans. on Pattern Analysis ans Machine Intelligence*, vol. 26, no. 7, pp. 862–877, 2004.

[3] M. Storring, H. Andersen, and E. Granum, "Skin colour detection under changing lighting conditions," *Proc. Seventh Symp. Intelligent Robotics Systems*, pp. 187–195, 1999.

[4] ——, "Estimation of the illuminant colour from human skin colour," *Proc. Int. Conf. Automatic Face and Gesture Recognition*, pp. 64–69, 2000.

[5] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 601–607, 1998.

[6] W. Hafner and O. Munkelt, "Using color for detecting persons in image sequences," *Pattern Recognition and Image Analysis*, vol. 7, no. 1, pp. 47–52, 1997.

[7] J. Yang, L. Weier, and A. Waibel, "Skin-color modeling and adaptation," *Proc. Asian Conf. Computer Vision*, vol. II, pp. 687–694, 1998.

[8] X. Zhu, J. Yang, and A. Waibel, "Segmenting hands of arbitrary color," *Proc. Int. Conf. Automatic Face and Gesture Recognition*, pp. 446–453, 2000.

[9] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 232–237, 1998.

[10] K. Schwerdt and J. Crowley, "Robust face tracking using color," *Proc. Int. Conf. Automatic Face and Gesture Recognition*, pp. 90–95, 2000.

[11] J. C.Terrillon, M. David, and S. Akamatsu, "Automatic detection of human faces in natural scene images by use of a skin color model and

of invariant moments," *Proc. Int. Conf. Face and Gesture Recognition*, pp. 112–117, 1998.

[12] J. C.Terrillon and S. Akamatsu, "Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images," *Proc. 12th Conf. on Vision Interface*, vol. 2, pp. 180–187, 1999.

Deliverable 1.2

FACE DETECTION

Rebeca Marfil
Grupo de Ingeniería y Sistemas Integrados
Dpto. Tecnología Electrónica
Universidad de Málaga
Campus de Teatinos 29071 Málaga (Spain)
www.grupoisis.uma.es

Contents

List of Figures

1 Introduction

A first step for any face processing system (i.e. face recognition or facial expressions identification systems) is to detect if one or more faces are presented in the image and to compute their locations. Given an arbitrary image, the goal of a face detection system is to determine whether or not there are faces in the image and, if present, return the image location and area of each face [1].

Face detection is a challenging task because of the existence of factors which can modify the appearance of a face in the image. Some of these factors are:

- Presence of structural components in the face as beard, mustache, hat or glasses.
- Pose and orientation.
- Facial expression.
- Variations in the illuminance.
- Oclussions.
- Noise.

Face detection methods can be classified into four categories [1]:

- Knowledge-based methods [2]: this category includes any rule-based face detection approach. The rules, derived from the researcher's knowledge of human faces, usually describe the features of a face and their relationships. For example: a face often contains two eyes that are symmetric to each other, a nose and a mouth. The relationships between them are their relative distances and positions. The most important problem of this type of methods is the difficulty to extract the adequate human knowledge and represent it using rules.
- Feature-based methods: these methods try to find invariant features of faces. The most used features are eyebrows, eyes, nose, mouth and hairline [3]. A problem with these features is that they can be corrupted due to illumination, noise and occlusions. Other used features are texture [4] and skin colour [5].

- Template matching methods: a standard pattern of a face, or some patterns of face features as nose, eyes and mouth, is stored as template. This template can be a fixed template [6], which has problems with variations in scale, pose and shape, or a deformable template [7].

- Appearance-based methods [8]: in these methods, models are learned from a set of training images which should capture the representative variability of facial appearance. In general, appearance-based methods rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face and non-face images. The learned characteristics are in the form of distribution models or discriminant functions.

In this report, a feature-based method for face detection is proposed. This method is based on the previous work of Viola and Jones [9], which uses Haar-like features to detect faces. Haar-like features encode the existence of oriented contrasts in the input image. The Viola and Jones's method has proven to be very fast (15 frames per second in a conventional desktop). Two of the main characteristics of the Viola and Jone's method, that are exploited in the work proposed in this report, are the following:

- The use of a set of features which are reminiscent of Haar Basis functions. In order to compute these features very quickly at many scales, they introduce a new image representation called *integral image*. It can be computed from an image using a few operations per pixel. Once computed, any of the Haar-like features can be calculated at any scale or location in constant time.

- A simple an efficient classifier is used to select a small number of important features from the huge amount of potential ones. This classifier is built using the AdaBoost learning algorithm [10].

Although the proposed algorithm is based on the key ideas of [9], a main contribution is presented. While Viola and Jones compute the Haar-like features over the whole image, we propose to previously detect skin colour regions in the input image and then to compute the Haar-like features only in the set of skin colour regions where a face is probably located. In order to select these potential "face regions" a set of test is computed over each skin region.

Figure 1: a) "Skin image"; b) eroded and dilated image.

Therefore, the face detection method proposed in this report has two main steps:

1. Potential face regions of the input image are detected. This step can be subdivided into two stages: first, the skin colour pixels of the input image are detected using the algorithm explained in deliverable 1.1, and grouped into connected regions. Second, a set of structural tests is applied to the previously detected skin regions in order to discard the regions that clearly are not a face.

2. The remaining regions are classified as face or not face using the method proposed by Viola and Jones [9].

2 Potential face regions extraction

The first step of the proposed face detection system is to compute the skin colour pixels of the input image. The used skin colour detection algorithm is the previously presented in deliverable 1.1. Once the skin colour pixels are detected, the resulting "skin image" is eroded and dilated in order to remove small noisy regions (Fig. 1). Then, the connected skin colour regions are computed using a region labelling algorithm. In a first step of this algorithm, the input image is covered row by row. The 8-vicinity of each skin colour pixel is studied. If some of the eight neighbours of a skin colour pixel has been previously labelled, the skin colour pixel is labelled with the same label. It is labelled with a new label in other case. If there are two or more different

Figure 2: a) Eroded and dilated image; b) labelled image.

labels in the vicinity of a skin colour pixel, these two labels are stored in a list as equivalent labels. In a second step, the list of equivalent labels is studied in order to assign only one label to the same connected region. In Fig. 2 each extracted connected region is marked with a different colour.

Once the skin pixels of the skin image are grouped into connected regions, those whose dimensions are clearly not the dimensions of a face are discarded. In order to do that, four different tests are applied to the connected skin regions:

1. Test of minimum and maximum area: the skin regions whose area is less than the 1% of the total area of the input image are discarded. The skin regions whose area is higher than the 80% of the total area of the input image are discarded.

2. Test of elongated regions: each skin region whose bounding box height is less than the 40% of its bounding box width is discarded. Each skin region whose bounding box width is less than the 40% of its bounding box height is discarded.

3. Test of sparse regions: each region whose area is less than the 50% of the area of its bounding box is discarded.

4. Test of proportion: if the $height/width$ proportion of a region is higher than 1.6, the height of the region is reduced until $(height/width) < 1.6$.

All the previously used thresholds have been empirically obtained and they can be changed in order to control the flexibility of the tests.

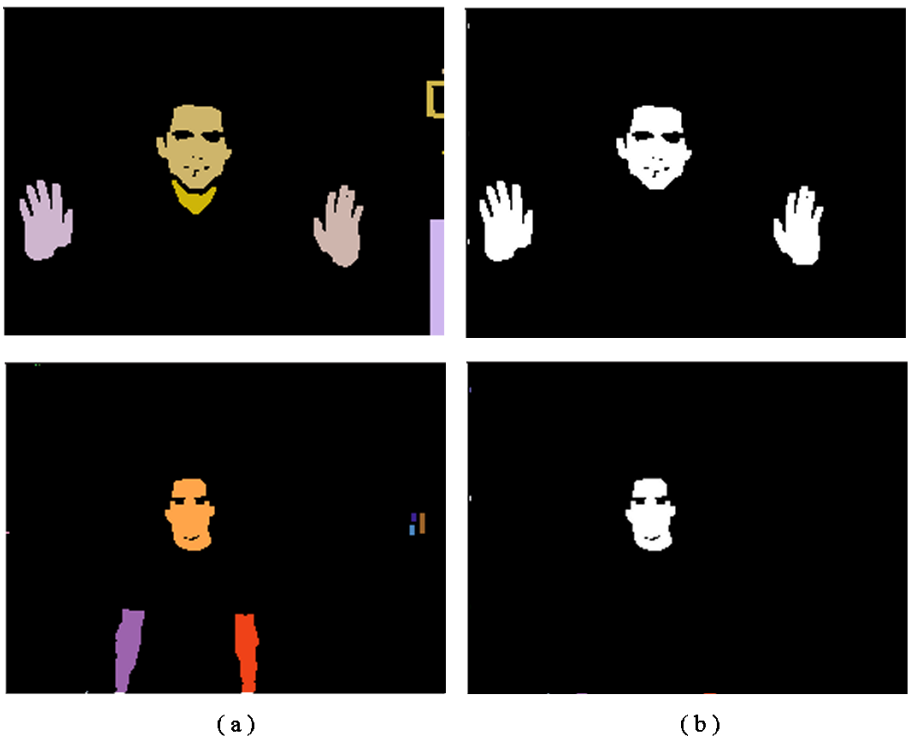

Figure 3: a) Labelled images; b) remaining regions after the tests.

Fig. 3 shows the regions obtained after applying the tests to the labelled images.

Once the previously explained set of regions is discarded, the remaining regions are used to build gray-scale images with fixed size. These images will be inputs of the classification algorithm employed to discriminate between face and non-face regions. Each of these images is built as follows:

1. A square bounding box around the region is used to select a subimage from the original colour image which previously has been converted in a grey scale image.

2. The previously extracted subimage is interpolated using a bicubic interpolation algorithm to achieve an image of 24x24 pixels (Fig. 4).

Figure 4: Examples of interpolated images which will be inputs of the classifier.

3 Classification algorithm

The 24x24 images extracted in the previous section are the inputs of a classifier which is built using the Adaboost learning algorithm [10]. This algorithm selects a small set of critical face features from a large set of features. The used features are the Haar-like features. This classifier has been proposed by Viola and Jones [9].

3.1 Features

The used features are the Haar-like features proposed by Papageorgiou *et al.* [11]. We use four kinds of features (Fig. 5):

- *Two-rectangle feature*: It is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent. It is subdivided in:
  - vertical two-rectangle: two rectangles vertically adjacent.
  - horizontal two-rectangle: two rectangles horizontally adjacent.

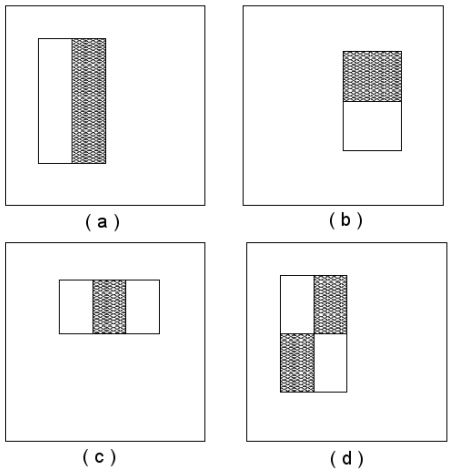

Figure 5: a) Vertical two-rectangle; b) horizontal two-rectangle; c) three-rectangle; d) four-rectangle.

- Three-rectangle feature: it computes the sum within two outside rectangles subtracted from the sum in a center rectangle.
- Four-rectangle feature: it computes the difference between diagonal pairs of rectangles.

Fig.5 shows the used features. The sum of the pixels which are within the white rectangles are subtracted from the sum of the pixels in the grey rectangles.

3.2 Integral image

In order to compute the previously explained features very quickly, Viola and Jones [9] propose to use the *integral image*. The integral image is an intermediate representation for the image which at location (x, y) contains the sum of the pixels above and to the left of (x, y) inclusive. It can be represented as:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{1}$$

being $ii(x, y)$ the integral image and $i(x, y)$ the original image. In our case the original images are the set of subimages computed in the skin regions

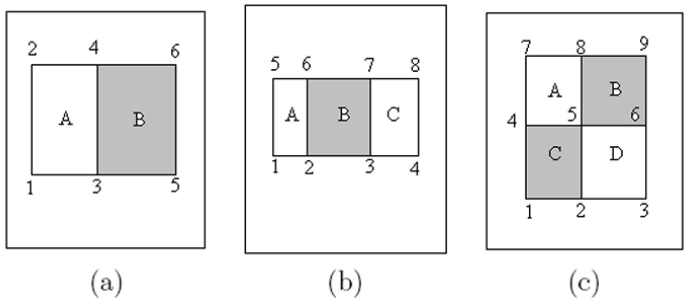

Figure 6: Memory accesses needed to compute: a) a two-rectangle feature; b) a three-rectangle feature; c) a four-rectangle feature. For example the two-rectangle feature is computed as: $B - A = (5-6-3+4)-(3-4-1+2)$

extraction step. The integral image can be computed in only one pass using equations (2) and (3).

$$s(x,y) = s(x,y-1) + i(x,y) \tag{2}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{3}$$

where $s(x,y)$ is the cumulative row sum.

Using the integral image it is possible to compute the sum of the pixels within any image rectangle with only four memory accesses [9]. Therefore, a two-rectangle feature is computed with six memory accesses. A three-rectangle feature needs 8 memory accesses and nine memory accesses are needed to compute a four-rectangle feature (Fig. 6).

3.3 Classifier learning process: Adaboost algorithm

The Adaboost algorithm is used to select a small set of critical features (T) from the huge set of previously computed features. Besides, it is used to train the classifier. The Adaboost learning algorithm consists of T weak classifiers (one for each feature) which are combined to form a strong classifier. Each weak classifier is designed to select the single rectangle feature which best separates the positive and negative examples. For each feature, the weak learning process determines the optimal threshold classification function, such that the minimum number of training images are misclassified. Therefore, a weak classifier ($h(x, f, p, \theta)$) consists of a feature (f), a

threshold (θ) and a polarity (p) indicating the direction of the inequality:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The process to select the optimum set of weak classifiers from the whole set of possible weak classifiers is the following [9]:

- Given example images $(x_1, y_1), ..., (x_n, y_n)$ where $y_i = 0$ for negative (non-face) examples and $y_i = 1$ positive (face) ones.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, ..., T$:
  1. Normalize the weights, $w_{t,i,norm} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
  2. Select the best weak classifier with respect the weighted error

$$\epsilon_t = \min_{f,p,\theta} \sum_i w_i |h(x_i, f, p, \theta) - y_i| \tag{5}$$

     This process is more detailed explained below.
  3. Define $h_i(x) = h(x, f_t, p_t, \theta_t)$ where f_t,p_t and θ_t are the minimizers of ϵ_t.
  4. Update the weights $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$, where $e_i = 0, 1$ if the image x_i is classified correctly or incorrectly respectively. $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
  5. Compute the weights α of the final strong classifier $\alpha_t = \log \frac{1}{\beta_t}$.

The final strong classifier is a lineal combination of the previously selected weak classifiers. It is represented using equation (6).

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$C(x) = 1$ when the input image is classified as face and 0 in another case.

The process to select the best weak classifier in the iteration t is the following:

- For $i = 1, ..., K$ (being K the total number of potential features f):

1. Initialize $\epsilon_{t,i,min} = 1$.
2. Compute and sort the feature values for each example image $f_i(x_1), ..., f_i(x_N)$.
3. For $j = 1, ..., N$:
   (a) Set $\theta_j = f_i(x_j)$.
   (b) Compute $\epsilon_{t,i,\theta_j} = \min\{S^+ + (T^- - S^-), S^- + (T^+ - S^+)\}$, where T^+ is the total sum of positive example weights, T^- is the total sum of negative example weights, S^+ is the sum of positive weights below the current example in the sorted list, and S^- is the sum of negative weights below the current example in the sorted list.
   (c) If $\epsilon_{t,i,\theta_j} < \epsilon_{t,i,min} \Rightarrow \epsilon_{t,i,min} = \epsilon_{t,i,\theta_j}$, $\theta_{t,i} = \theta_j$ and $p_t = p_j$.

- Select the feature f_i with less $\epsilon_{t,i,min}$

4 Results

In order to perform the training of the classifier, we have used a set of $N = 200$ positive (face) and negative (non-face) images. Specifically, 100 positive and 100 negative images have been used.

The number of computed potential features has been $K = 108,241$. Although the total number of features is 134,736 in a 24x24 subimage, we have discarded some of them because their contribution to the training process is not important. Specifically, the discarded rectangle features have been the following: 2-pixels width vertical two-rectangle, 2-pixels height horizontal two-rectangle, 3-pixels width three-rectangle and 2-pixels width 2-pixels height four-rectangle. The number of training iterations and final features has been $T = 150$. This number has been empirically obtained. A comparison using different numbers of final features is presented at the end of this section. The computational time of the training process has been 5 hours in a 2,4 GHz Pentium IV PC. It must be noted that the training process is performed only once.

The features obtained in the training process are used to detect the faces presented in real images. The detection process is the following:

1. Detection of the skin colour regions.
2. Erosion and dilatation of the "skin image".

Table 1: Percentages of rightly classified faces, false positives and false negatives

	Indoor Images	Outdoor images
Rightly classified	86%	82%
False negatives	14%.	18%
False positives	10%.	8%

3. Extraction of the connected skin regions.

4. Computation of the tests to discard the regions which are not clearly a face.

5. Generation of the 24x24 subimages using the skin regions which have passed the tests.

6. Classification of the previous images using the strong classifier computed in the training process.

Fig. 7 and Fig. 8 show the results of the face detection process in some indoor and outdoor images, respectively. Table 1 shows some percentages obtained using a set of 50 images. The face detection process has proven to be very fast: 0,033 seconds (30 fps) with 192x256 images and 0,043 seconds (24 fps) with 256x320 images using a 2,4 GHz Pentium IV PC.

In order to study the behaviour of the classifier using different sets of features, we have repeated the training of the Adaboost algorithm with different values of T: $T = 1, 10, 50, 100, 150, 200$. The rest of parameters are $N = 200$ and $K = 108,241$. Figs. 9 and 10 show some of these results. In the case of only one feature, it should be appreciated that some faces are detected. The problem is the high number of false positives. The higher the number of features, the lower the number of false positives, until $T = 150$. The results obtained with $T = 200$ are worse than the results obtained with $T = 150$ because the accuracy of the classifier does not increase indefinitely with the number of features. There is an optimum number of iterations for the inputs used in the training process. In that case, the optimum T value is 150.

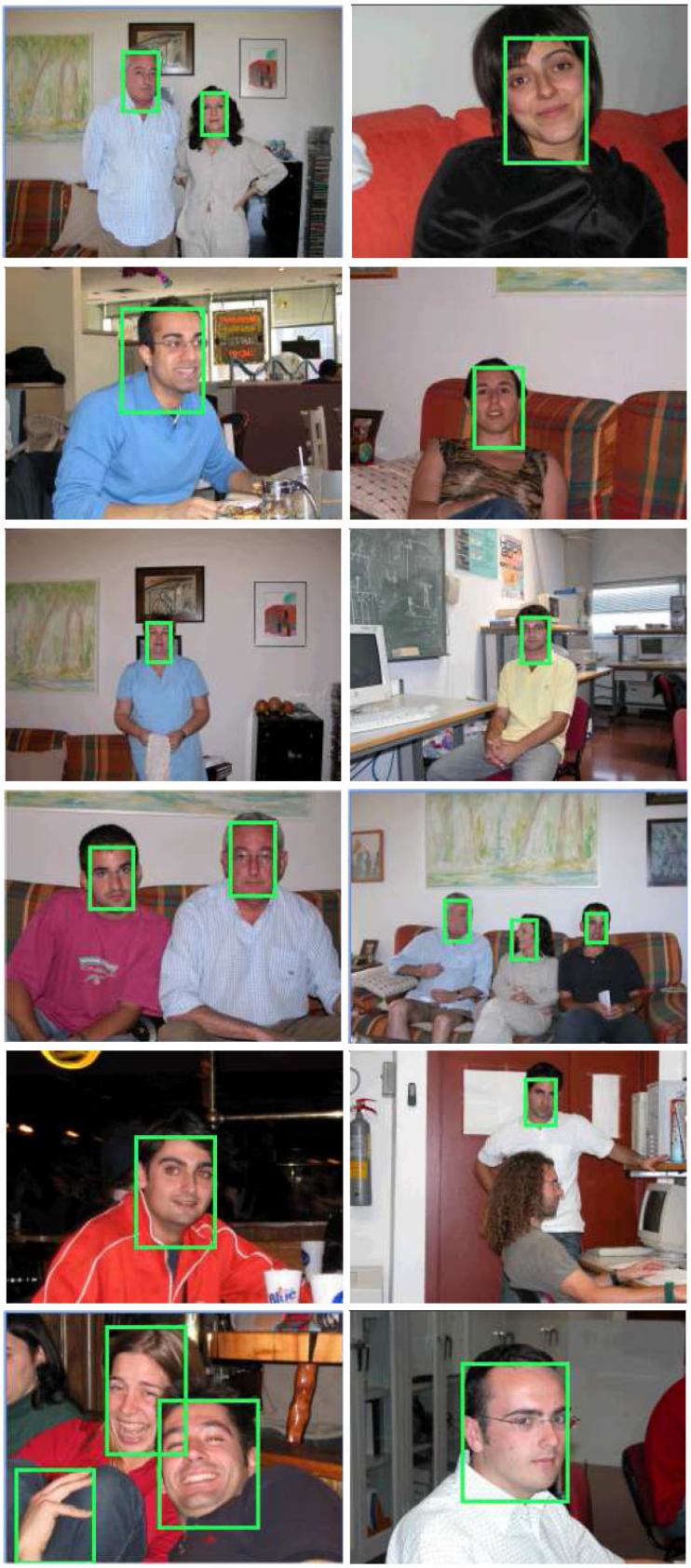

Figure 7: Indoor result images

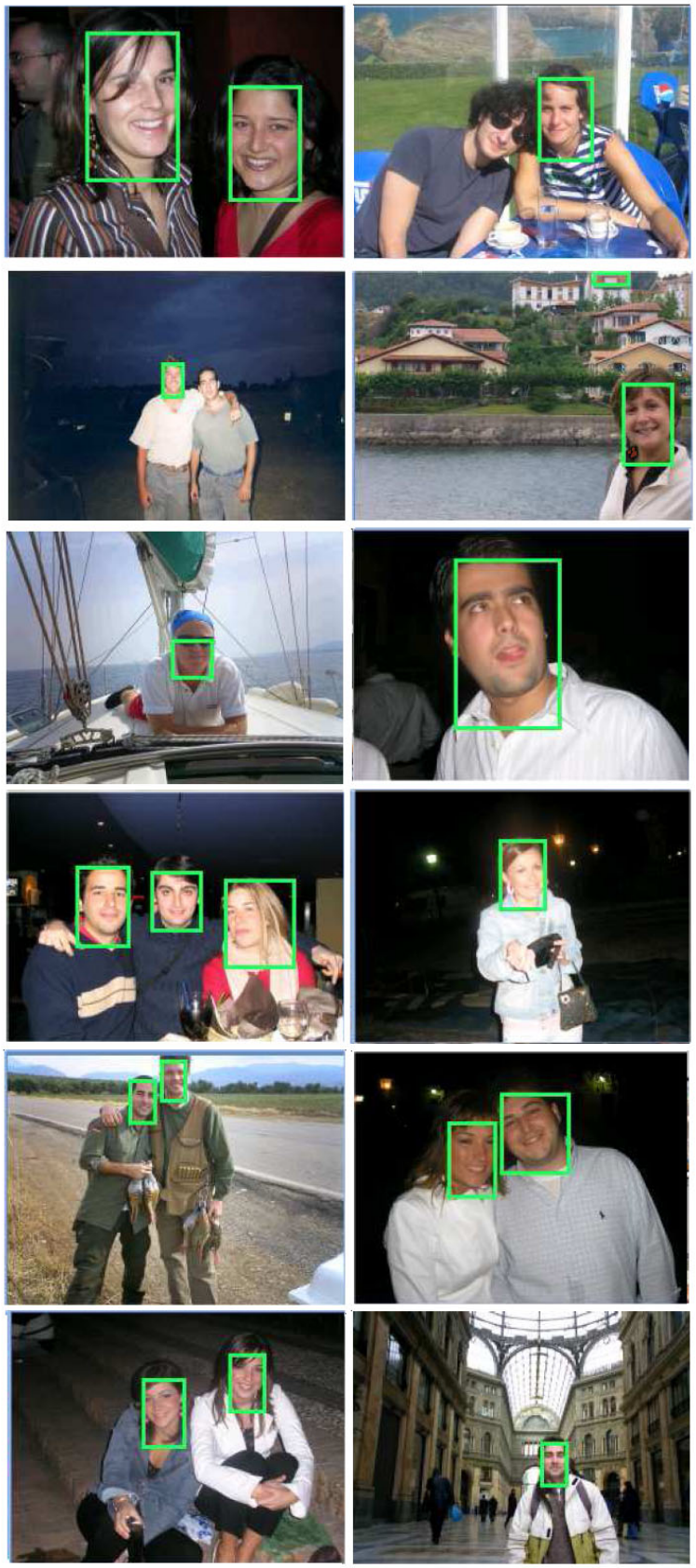

Figure 8: Outdoor result images

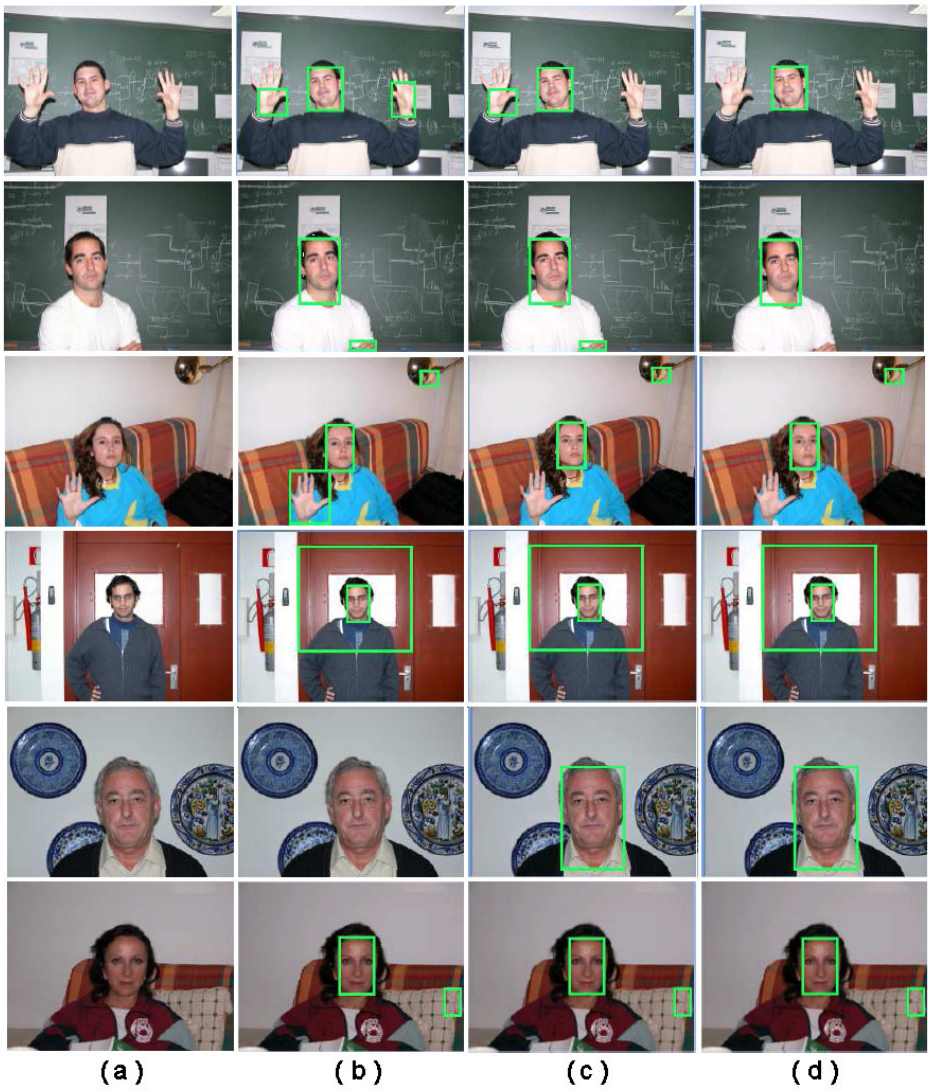

Figure 9: Face detection results; a) original images; b) T=1; c) T=10; d) T=50.

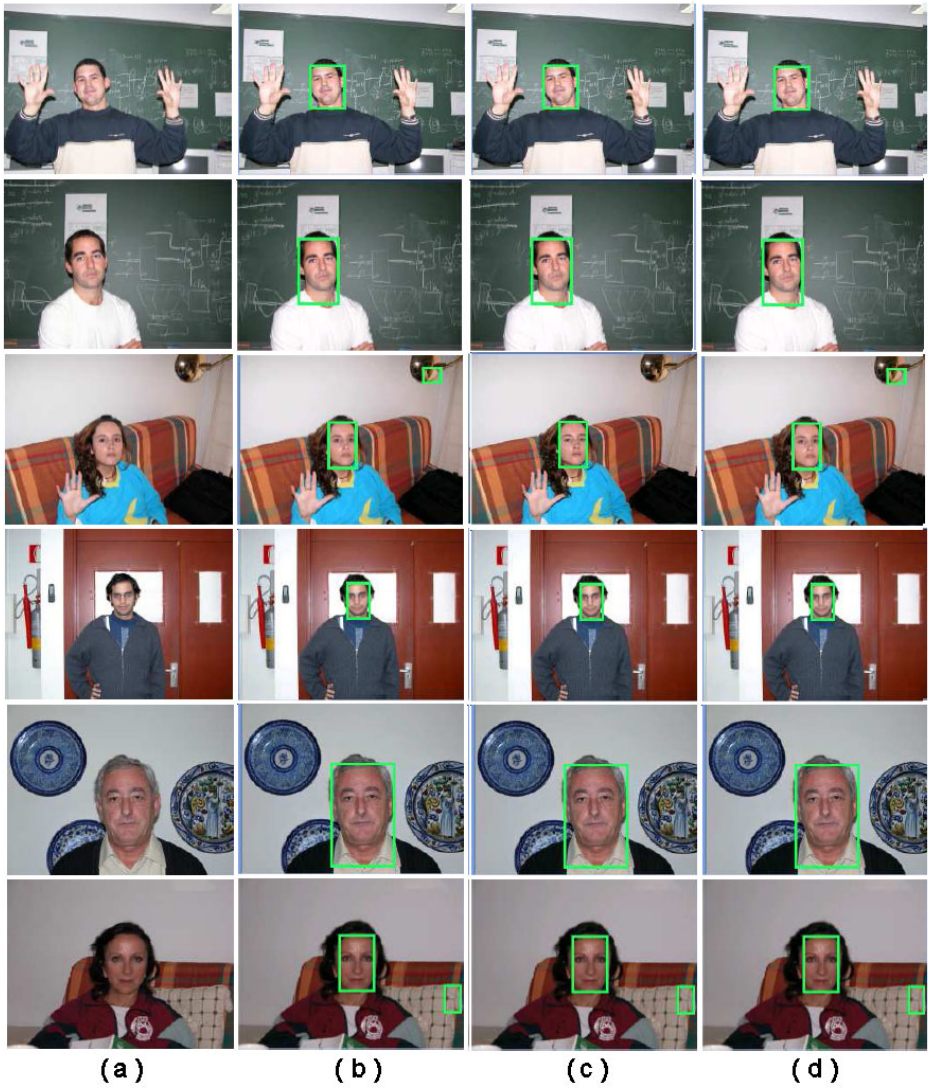

Figure 10: Face detection results; a) original images; b) T=100; c) T=150; d) T=200.

References

[1] M.-H. Yang, D. Kriegman, and N. Ahuja, "Detcting faces in images: A survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 34–58, 2002.

[2] G. Yang and T. Huang, "Human face detection in complex background," *Pattern Recognition*, vol. 27, no. 1, pp. 53–63, 1994.

[3] K. Yow and R. Cipolla, "Feature-based human face detection," *Image and Vision Computing*, vol. 15, no. 9, pp. 713–735, 1997.

[4] Y. Dai and Y. Nakano, "Face-texture model based on sgld and its application in face detection in a color scene," *Pattern Recognition*, vol. 29, no. 6, pp. 1007–1017, 1996.

[5] J. C.Terrillon, M. David, and S. Akamatsu, "Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments," *Proc. Int. Conf. Face and Gesture Recognition*, pp. 112–117, 1998.

[6] I. Craw, D. Tock, and A. Bennett, "Finding face features," *Proc. European Conf. Computer Vision*, pp. 92–96, 1992.

[7] A. Lanitis, C. Taylor, and T. Cootes, "An automatic face identification system using flexible appearance models," *Image and Vision Computinf*, vol. 13, no. 5, pp. 393–401, 1995.

[8] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[9] P. Viola and M. Jones, "Robust real-time face detection," *Int. Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[10] Y. Freund and R. Schapire, "A decision-theoric generalization of on-line learning an application to boosting," *Proc. European Conf. Computational Learning Theory*, pp. 119–139, 1995.

[11] C. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," *Int. Conf. Computer Vision*, pp. 555–562, 1998.

Deliverable 1.3

FACE RECOGNITION

Jörg Rett
Institute of Systems and Robotics
University of Coimbra
Polo II, 3030-290 Coimbra, Portugal
http://www.isr.uc.pt

Contents

List of Figures

1 Introduction

If the perceptual system of a robot is based on vision, interaction will involve *visual human motion analysis*. The ability to recognize humans and their activities by vision is key for a machine to interact intelligently and effortlessly with a human-inhabited environment [1]. Several surveys on visual analysis of human movement have already presented a general framework to tackle this problem [2], [1], [3] and [4]. Aggarwal and Cai point out in their survey [2] that one (of three) mayor areas related to the interpretation of human motion is motion analysis of the human body structure involving human body parts. The general framework consists of: 1. Feature Extraction, 2. Feature Correspondence and 3. High Level Processing.

The architecture we present relates to this framework in that we call the low level processing concerned with Feature Extraction - Observation Level see fig. 1. This level accomplishes tasks like detecting the presence of human using face detection, detecting important (with reference to interaction) body parts like hands and face using skin color detection and recognizing the person using eigenobjects. This report is concerned with the latter task.

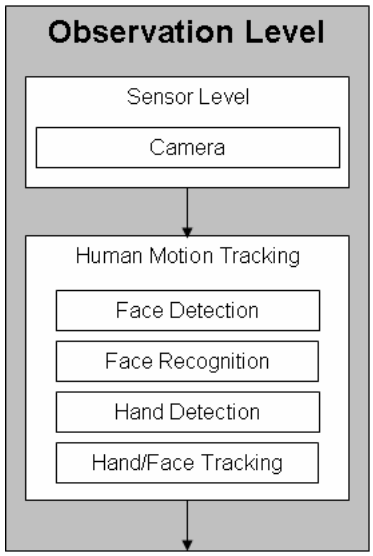

Figure 1: architecture of the system

Presuming a face has been detected using the method described in De-

liberable 1.2 we proceed by identifying the person. For the face recognition we choose a method using PCA and eigen objects. Face recognition systems based on eigenfaces methods has been introduced by Turk et al. [9]. Methods based on eigenvectors are used to extract low-dimensional subspaces which tend to simplify tasks such as classification. The Karhunen- Loeve Transform (KLT) and Principal Components Analysis (PCA) are the eigenvector based techniques used for dimensionality reduction and feature extraction in automatic face recognition. Face recognition using eigen objects and the implementation using Intel Opencv library are shown in the following section.

2 Recognition Database

The face recognition system is based on eigenspace decompositions for face representation and modeling. The learning method estimates the complete probability distribution of the faces appearance using an eigenvector decomposition of the image space. The face density is decomposed into two components: the density in the principal subspace (containing the traditionally-defined principal components) and its orthogonal complement (which is usually discarded in standard PCA).

Given a training set of WH images, it is possible to form a training set of vectors x^T. The basis functions for the Karhunen Loeve Transform (KLT) are obtained by solving the eigenvalue problem:

$$\Delta = \Phi^T \Sigma \Phi \tag{1}$$

where Σ is the covariance matrix, Φ is the eigenvector matrix of Σ and Δ is the corresponding diagonal matrix of eigenvalues λ_i. In PCA, a partial KLT is performed to identify the largest eigenvalues eigenvectors and obtain a principal component feature vector $y = \Phi_M^T \tilde{x}$, where $\tilde{x} = x - \overline{x}$ is the mean normalised image vector and Φ_M is a submatrix of Φ containing the principal eigenvectors([8]).

3 Implementation

The system architecture consists of three main modules: learning, face detection and face recognition.

3.1 Learning phase

During the (offline) learning process our system creates the eigenspace for a certain set of people. In our case we used a set of seven people (enguer, luis, elo, carlos, vinay, mannequin and joerg). From 50 captured images per person we use a certain number to build the database. The image capture is performed automatically using our face detection module (based on Haar-like features [5, 6, 7, 8]). The images are then resized to 32*32 pixels gray scale images. In the next step we compute the orthonormal eigen basis and the average image for each person using the function *cvCalcEigenObjects* from Opencv [10](see figure 2).

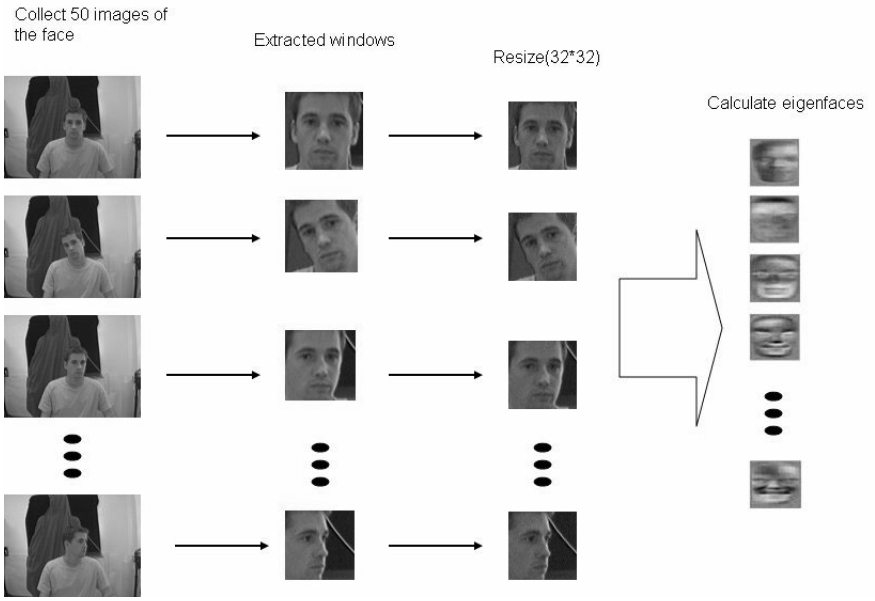

Figure 2: learning process

3.2 Recognition phase

Once this eigenspace is calculated the system is able to recognize the face of a person during the tracking process in real time. Again, we start by detecting the face within the image using our face detection module, resizing and converting it. We then calculate all decomposition coefficients (function

cvEigenDecomposite) for the extracted image using the eigen objects for each person in the database. In the next step we calculate the projection of the image from the eigen objects of the person and the coefficients calculated before (function *cvEigenProjection*). As a result we get one projection image for each person (see figure 3).

We finally simply compare which projected image match the best with the original extracted image. The confidence value is directly derived from a template matching function. In our case the two images are compared using the Opencv function cvMatchTemplate and choosing the squared difference normalized method :

$$R = \frac{\sum_{x,y} (Img2(x,y) - Img1(x,y))^2}{\sqrt{\sum_{x,y} Img2(x,y)^2 * \sum_{x,y} Img1(x,y)^2}} \tag{2}$$

The function gives a value between 0 and 1 (where 0 is a perfect match). We calculate the confidence value by $conf = (1 - match) * 100$. The person with the highest confidence value is defined as the recognized person. We only keep matching values under a certain threshold (here, we use 0.04 equivalent to 96%).

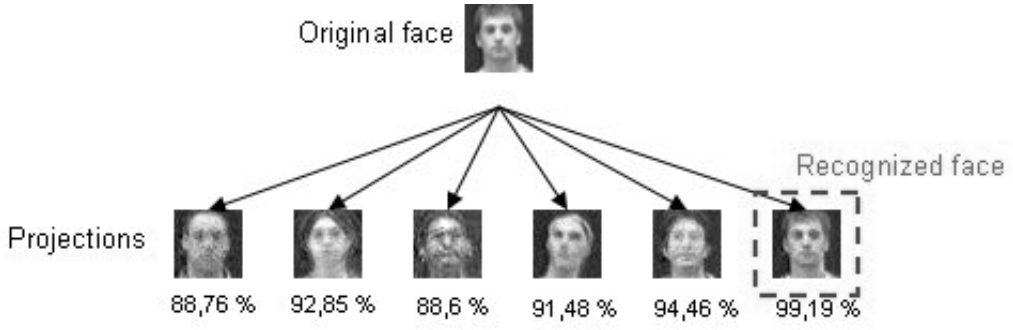

Figure 3: projections created from an extracted face, with the percentage of recognition

4 Discussion and Results

The amount of time needed to process the eigenfaces for recognition was between 2 and 3 ms per person in the database using our system (AMD64

	confidence values
wrong matches	$70\% to 94\%$
right matches	$90\% to 100\%$

Table 1: approximation of possible values return by the recognition system

3000+ processor with Suse 9.1 operating system). We were able to recognize several people at the same time like shown in figure 4.

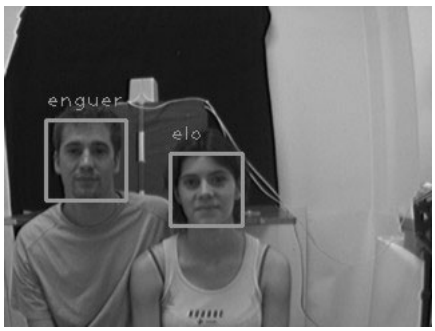

Figure 4: several person recognition

4.1 Confidence Value

Table 1 shows the confidence values for wrong matches (other people from the database) and the correct match. The confidence values shown in table 1 are reached by using similar background and illumination conditions of the environment for the learning phase as well as for the recognition phase. The same holds true for camera settings like shutter etc. In other cases the algorithm might still work but the confidence values will be lower due to the used a correlation method for matching. In this case, the threshold needs to be adapted.

4.2 Training Data

A question that arises during the training phase is that of how many training samples (images) do we need and what should they contain (in terms of facial orientation and expression). In the following we present results concerning

the size and the content of the database. First we compared some results using a big database (50 images) and a small database (10 images) for a image sequence. The sequence contains several facial expressions and orientations (the person to be recognize is enguer). In figure 5, the first chart represents

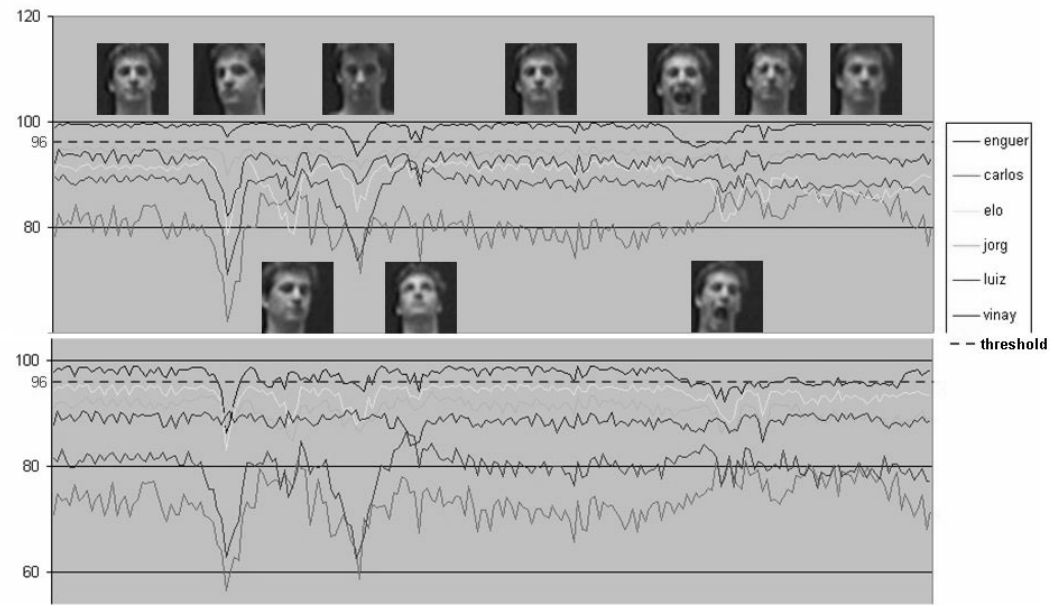

Figure 5: Comparison between two kind of databases for different facial expressions and orientations

the recognition using 6 people and 50 images but with similar orientation and expression (most are frontal faces). The second chart uses the same sequence but with a 10 images sized database and hand-picked images od different orientation and expression. The first chart shows better results for frontal faces. We can observe in both cases some downward peaks when the head changes its orientation or when the person produces different facial expressions. Finally looking at the threshold (red dotted line), we see that overall confidence value of the right match (enguer) is lower in the second chart.

The figure 6 represent the same sequence as before but using a big database (50 images) with diverse facial expressions and orientations. Again, the results for the probability value of the right match (enguer) are lower than

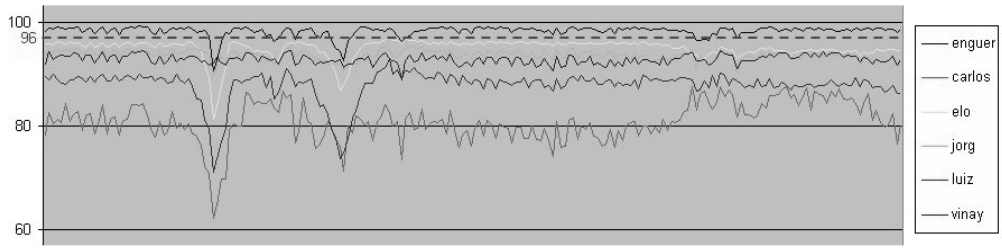

Figure 6: huge database composed of all kind of facials expressions and orientations

in chart one of figure 5, but some facial expressions are better recognized (e.g opened mouth at the end).
The results show that a database of 50 images with diverse orientations and facial expressions produces a more stable recognition though the overall probability values might be lower.

5 Conclusion

This report has shown the implementation of a recognition program for human motions using eigenfaces (PCA) from the Intel OpenCv library. We presented results concerning the face recognition, showing the feasibility of our approach.

name	number	date
carlos	100	2005-07-13
elo	76	2005-08-10
enguer	53	2005-08-10
jorg	96	2005-08-09
luiz	76	2005-07-13
mankin	100	2005-08-09
vinay	100	2005-07-13

Table 2: Full Image Database

name	number	date
elo	10	2005-08-10
enguer	10	2005-08-10
jorg	10	2005-08-09
vinay	10	2005-08-10

Table 3: Small Image Database

A Recognition Database

The full database consists of 53 - 100 images of seven people.

The small database consists of 10 handpicked images from the full database of four people. The images have a high diversity in terms of facial orientation and expression.

References

[1] D. M. Gavrila. The visual analysis of human movement: A survey. *CVIU*, 73(1):pp. 82–98, 1999.

[2] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *CVIU*, 73(3):428–440, 1999.

[3] Alex Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on PAMI*, 22(1):107–119, January 2000.

[4] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *CVIU*, 81(3):231–268, 2001.

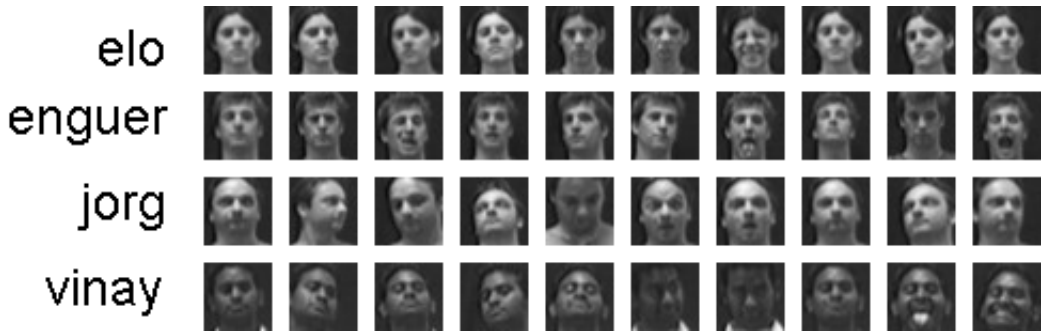

Figure 7: Small Database

[5] Paul Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, page 511, 2001.

[6] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *IEEE International Conference on Image Processing*, volume 1, pages 900–903, 2002.

[7] Paulo Menezes, Jose Barreto, and Jorge Dias. Face tracking based on haar-like features and eigenfaces. In *IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, 2004.

[8] Jose Barreto, Paulo Menezes, and Jorge Dias. Human-robot interaction based on haar-like features and eigenfaces. In *IEEE International Conference on Robotics and Automation*, 2004.

[9] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[10] Intelwebsite. `http://www.intel.com/technology/computing/opencv/overview.htm`.