Høgskolen i Gjøviks rapportserie, 2001 nr 1

# Experimental Comparison of Face/Non−Face Classifiers

Ivar Farup og Erik Hjelmås
Avdeling for Teknologi

Gjøvik 2001
ISSN 0806−3176

## Abstract

Most face detection algorithms can be divided into two sub-problems, initial visual guidance and face/non-face classification. In this paper we propose an evaluation protocol for face/non-face classification and provide experimental comparison of six algorithms. The overall best performing algorithms are the baseline template matching algorithms. Our results emphasize the importance of preprocessing.

# Contents

# 1 Introduction

Face detection is an important and necessary first step in most face recognition applications. Face detection serves to localize potential face regions in images and classify them as faces or non-faces. This is a difficult task due to the dynamic appearance and variability of faces as opposed to more static objects such as vehicles or weapons. In addition to face recognition, areas such as content-based image retrieval, intelligent human-computer interfaces, crowd surveillance, video coding and email content security also make use of face detection algorithms.

The last decade has shown a great deal of research effort put into face detection technology. A comprehensive survey can be found in Hjelmås and Low [5], where the algorithms are classified as feature-based or image-based. Several algorithms perform very well, and the choice of algorithm is mostly dependent on the requirement of the application to perform in real-time or not. However, not much work has been done on comparing existing algorithms. Some of the image-based algorithms report results on a common dataset (the CMU/MIT dataset), but there does not exist a specific evaluation protocol for this dataset. This has lead to different interpretations of testing parameters (such as how many faces are present,[1] or how to report correct detections vs. false alarms) for this set, which makes it hard to compare the algorithms.

In this paper we provide an experimental comparison of six face detection algorithms, categorized as two baseline, two image-based and two feature-based algorithms. One of the feature-based algorithms is a new version of an existing technique, while the rest are implemented based on previously published papers by other authors. The algorithms are selected based on findings in [5], and also to represent significantly different approaches. We also propose an evaluation protocol for the face/non-face classifier in face detection algorithms.

In section 2, we present an overview of the dataset we have selected for training and testing, while section 3 described the testing protocol in detail. Section 4 presents the algorithms, while section 5 and 6 contains the experimental results and discussion. Finally we conclude in section 7.

[1] Recently, a ground truth has been established and is available at `http://www.vasc.ri.cmu.edu//idb/html/face/frontal_images/index.html` but since this includes cartoons and line-drawn faces, it has not been applied consistently in face detection tests.

Figure 1: Example images from the XM2VTS training set.

## 2 The dataset

The dataset consists of images from the XM2VTS[2] [7] and AR[3] [6] face databases, and non-face images collected from the world wide web. The XM2VTS dataset is used for training. It contains 8 images of 295 subjects for a total 2360 images. All images are frontal view face images with a high degree of variation with respect to skin color, hair style, facial hair and glasses. However, there is not much variation in facial expression, most of the subjects have a neutral look with the mouth closed. The images are taken at four sessions with a month interval between sessions. For this training set, the coordinates of the eyes are available. Examples of the training images are shown in figure 1.

For testing, we use the AR dataset with 3313 images from 136 subjects where most of the subjects images have been captured during two sessions with a 2 week interval between the sessions, from which we define the following subsets:

**Easy** An easy dataset with 1783 face images. All subjects vary their facial expression, and there are large variations in lighting, but there are no facial occlusions. In 14% of the images the subjects where told to scream when the image was captured, thus these images have and extreme facial expression. Example images are shown in figure 2.

**Sunglasses** A difficult dataset with 765 face images. All subjects are wearing dark sunglasses. Example images are shown in figure 3.

**Scarf** A difficult dataset with 765 face images. All subjects are wearing a scarf covering the mouth area. Example images are shown in figure 4.

From the world wide web, we have collected manually a set of 67 large images with considerable structure, which might contain face-like patterns, which we use as the negative test set. In addition we have further collected

[2] `http://www.ee.surrey.ac.uk/Research/VSSP/xm2vtsdb/sample_front.htm`
[3] `http://rvl1.ecn.purdue.edu/~aleix/aleix_face_DB.html`

Figure 2: Example images from the **Easy** testing set.

Figure 3: Example images from the **Sunglasses** testing set.

Figure 4: Example images from the **Scarf** testing set.

a few large images for bootstrap training of the SNoW algorithm (described later).

The resolution of the training images (XM2VTS) are originally 720×576, but we only use an extracted window covering the center of the face (rescaled to 20×20 or 60 × 60 pixels, and geometrically normalized with respect to the eyes). Similarly, the resolution of testing images (AR) are originally 768×576, but we focus the search on subset covering the facial area (see the following section for details). The test sets and training sets are non-overlapping. All images are converted to 8 bit grayscale images (256 graylevels).

# 3 The evaluation protocol

Most face detection tasks can be divided into two steps, where the first step is an algorithm for visual guidance (focusing the search based on visual clues/low-level features such as color or motion, or simply an exhaustive search at all scales and locations) and the second step is the actual face/non-face classification. In this section we propose a protocol for evaluating the second step. Not all proposed face detection algorithms work in this two-step fashion, but since the general problem of face detection can be decomposed into these two steps, all face detection approaches would benefit (in terms of accuracy) from decomposing or combining their algorithm this way. Decomposing the problem leads to easier selection of the appropriate technique for the two sub-problems.

The key elements of the evaluation protocol are the following (tailored to the datasets used in our experiments) :

- The face classifiers generate a confidence score $s_{algo}$ where $algo \in \{\mathbf{bE}, \mathbf{bC}, \mathbf{PCA}, \mathbf{SNoW}, \mathbf{Gradient}, \mathbf{Gabor}\}$ indicates the face classifier algorithm.

- The multiresolution scanning algorithm: a $n \times n$ window $\omega$ scans the entire image with 1 pixel step size and the image is subsampled by a factor of 1.2 until all scales and locations have been included. The face classifiers are applied at each location and scale. $n$ is set to 20 for the image-based and baseline algorithms, and 60 for the feature-based algorithms. However, the $20 \times 20$ windows are just downscaled versions of the $60 \times 60$ windows in order to have the same number of testing windows $\omega$ for all algorithms.

- A correct detection of a face in a face image $I$ is registered if the window $\omega$ which produces the highest confidence score, $\max_{\omega}(s_{algo})$, is *correctly centered* in $I$ and has $s_{algo} > t_{algo}$. We have manually located the center $(x_c, y_c)$ of the face for all the test images, so we define $\omega$ correctly centered to be $\omega$ located such that its center region $\{(\frac{n}{2} \pm \frac{n}{4}, \frac{n}{2} \pm \frac{n}{4})\}$ encompasses $(x_c, y_c)$.

- The correct face detection rate $CD$ is simply

$$CD_{testset} = \frac{\text{number of images with face correctly detected}}{\text{total number of images}}$$

where $testset \in \{\textbf{Easy}, \textbf{Sunglasses}, \textbf{Scarf}\}$ indicates the test set used, and the total number of images is 1783 for the **Easy** dataset and 765 for the **Scarf** and **Sunglasses** dataset. We know that for the face images there is only one face present in each image.

- For the false alarm rate $FA$, we are simply interested in the number of false alarms relative to the total number of windows $\omega$ produced by the multiresolution scanning algorithm on the negative test set. This number is 5938360, so the false alarm rate is computed from

$$FA = \frac{\text{number of false detections}}{5938360}$$

A false alarm is a window $\omega$ where the face classifier produces a $s_{algo} > t_{algo}$. We do not count false alarms in the face images (the positive test sets).

- Results are reported in terms of ROC (Receiver Operator Characteristics) curves, which shows the trade-off between correct face detection rate $CD$ and the false alarm rate $FA$. The threshold $t_{algo}$ for the face classifier is varied in a range to produce a false alarm rate $10^{-4} \leq FA \leq 10^{-1}$.

# 4 The algorithms

Six algorithms are compared in the present work. They are categorized as two baseline algorithms, two image-based algorithms, and two feature-base algorithms. In this chapter the six algorithms are presented. The algorithms are implemented in C++ using the Qt library[4] by inheriting a new class `GrayProcessingImage` from the `QImage` class provided by the library.

## 4.1 Baseline: bE

Standard template matching is used as baseline algorithms for comparison. The training images are geometrically normalized such that a $20 \times 20$ window containing the eyes (in fixed positions), nose and mouth, can be extracted (see figure 5 for an example).

We compute the template by simply averaging these training images. The resulting template and the testing windows are preprocessed by extracting a best fit linear plane and by histogram equalization. The resulting templates are shown in figure 6.

[4] `http://www.troll.no`

Figure 5: Example of a geometrically normalized $20 \times 20$ face from the training set.

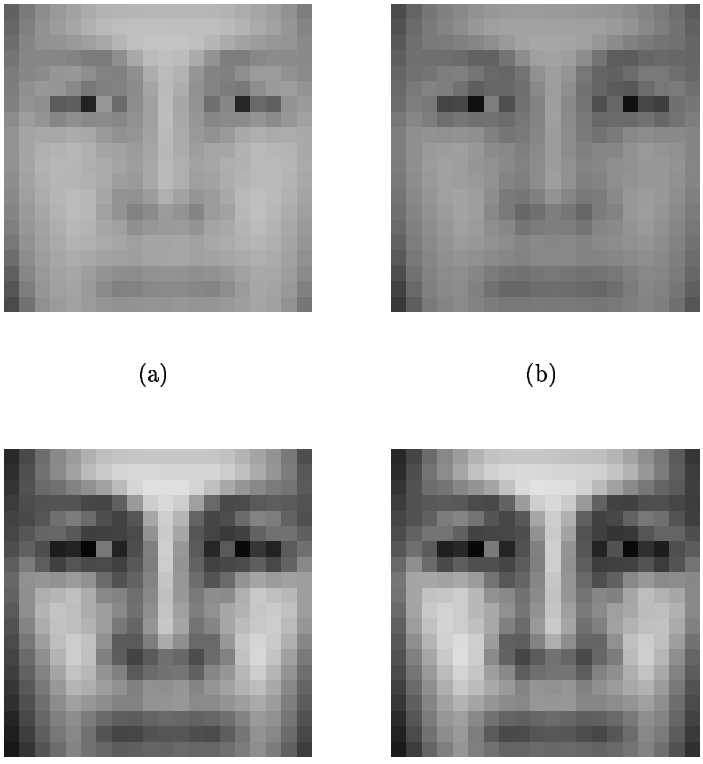



Figure 6: (a) The $20 \times 20$ face template obtained from averaging the training images, (b) preprocessed with the subtraction of a best linear fit plane, (c) histogram equalized, and (d) both best linear fit and histogram equalization.

Matching is performed by measuring Euclidian distance (L2) between template and testing window,

$$d_{\mathbf{bE}} = \sum_{i,j}(I_{ij} - T_{ij})^2, \tag{1}$$

where $I_{ij}$ is the pixel value of the testing window at position $(i, j)$, and $T_{ij}$ is the pixel value of the template at the same position.

### 4.2 Baseline: bC

The second baseline algorithm is exactly the same as the **bE** algorithm except that matching is performed by computing the correlation coefficient between the template and the testing window instead of the L2 norm. The distance is defined as

$$d_{\mathbf{bC}} = 1 - c(I, J), \tag{2}$$

where $c(I, J)$ is the correlation coefficient between the testing window $I$ and the template $T$ defined as

$$c(I, J) = \frac{\sum_{i,j}\left[(I_{ij} - \bar{I})(T_{ij} - \bar{T})\right]}{\sqrt{\sum_{i,j}\left[I_{ij} - \bar{I}\right]^2 \sum_{i,j}\left[T_{ij} - \bar{T}\right]^2}}, \tag{3}$$

where $\bar{I}$ and $\bar{T}$ denote the average pixel value in the testing window and the template, respectively.

### 4.3 Image-based: PCA

Principal Component Analysis (PCA) can be used to create a face space consisting of eigenfaces (some examples shown in figure 7) as an orthogonal basis, on which new faces can be projected to achieve a more compact representation. To our knowledge, this technique was first proposed by Sirovich and Kirby [12] and later further developed by Turk and Pentland [10]. In our implementation, we use the reconstruction error $\epsilon^2 = \|\tilde{\omega}\| - \sum_{i=1}^{n} y_i^2$ (where $y_i$ are projection coefficients and $\|\tilde{\omega}\|$ is the mean subtracted window) as a measure for the score $s_{pca}$. We only keep the first principal component for representation (thus $n = 1$).

#### 4.3.1 Computing face space

The basic procedure for computing face space is as follows:

We have a training matrix $\Omega = [\tilde{\omega}_1^{\nu}\tilde{\omega}_2^{\nu}...\tilde{\omega}_{2360}^{\nu}]$ of size $2360 \times 400$ where $\tilde{\omega}_i^{\nu}, i \in 1, ..., 2360$ denotes the pre-processed, mean-subtracted and vectorized extracted $20 \times 20$ training images (thus $400 \times 1$ in vectorized form). We form the covariance matrix

$$\Sigma = \Omega^T\Omega \tag{4}$$

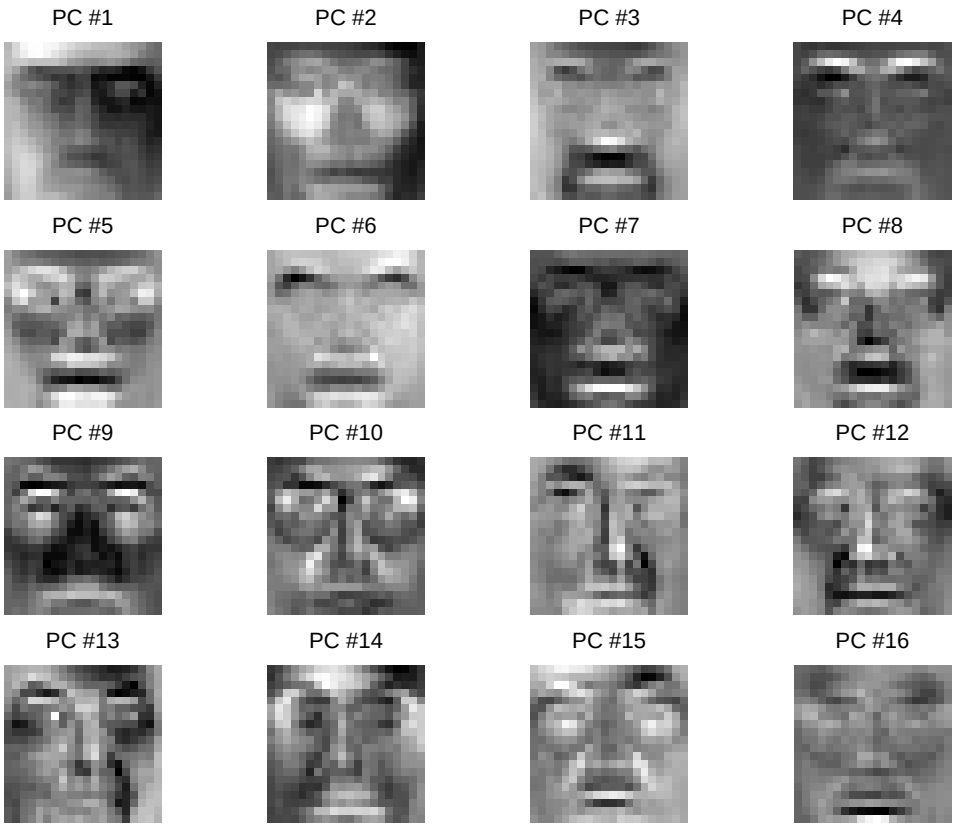



Figure 7: The first 16 eigenfaces (principal components) computed from the training set.

and solve the eigenvalue problem

$$\Lambda = \Phi^T \Sigma \Phi \tag{5}$$

where $\Phi$ is the eigenvector matrix of $\Sigma$ and $\Lambda$ the corresponding eigenvalues. The projection coefficients for a new pre-processed, mean-subtracted and vectorized extracted test window $\tilde{\omega}^\nu$ is computed by

$$\mathbf{y} = \Phi_m^T \tilde{\omega}^\nu \tag{6}$$

where $\Phi_m$ represents the selected subset of $\Phi$, which in our case consists simply of the eigenvector with the largest eigenvalue.

### 4.4 Image-based: Snow

The SNoW (Sparse Network of Winnows) learning architecture, proposed by Roth in [3], has been successfully applied to face detection by Roth et. al. in

[11]. We have implemented this algorithm using the software available from the website[5] of Roth for training, and our own implementation for testing. The technical details of of the algorithm are described in [11]. For simplicity, we have selected the primitive {`position` × `intensity`} features, which extract $n \times n$ Boolean features in a $n \times n \times g$ dimensional feature space from

$$f(x, y) = (g \times ((y \times n) + x)) + \omega(x, y) \tag{7}$$

where $n = 20$ is the size of the extracted window $\omega$ and $g = 256$ is the number of graylevels, which yields $f$ as a $20 \times 20$matrix (but used later in vectorized form) of indexes to the active features for $\omega$. We use the Winnow update rule for training, and an incremental training procedure similar to the bootstrap training proposed by Sung and Poggio [14]. The classifier consists of two linear units (similar to perceptrons), one representing the class of faces and non-faces respectively. The linear units have weights $w_i^u \neq 0, u \in \{face, nonface\}$ in the 102400 dimensional feature space only where a feature has been active during training. Classification is performed by computing

$$\mathcal{O}_{face} = \sum_{i=f(1)}^{f(400)} w_i^{face} \tag{8}$$

$$\mathcal{O}_{nonface} = \sum_{i=f(1)}^{f(400)} w_i^{nonface} \tag{9}$$

and using the score function

$$s_{SNoW} = \mathcal{O}_{face} - \mathcal{O}_{nonface}. \tag{10}$$

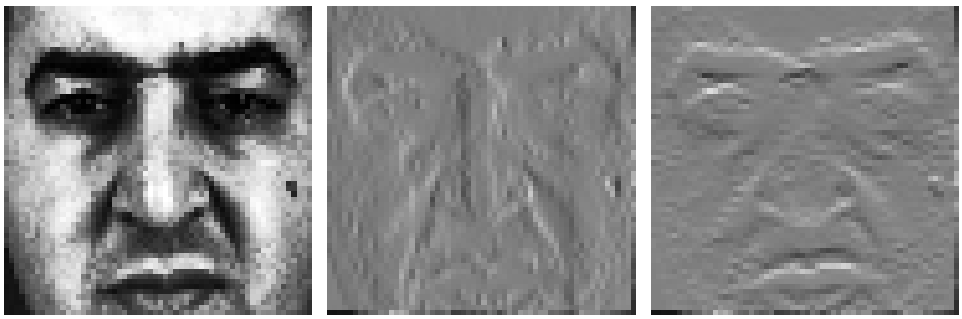
### 4.5 Feature-based: Gradient

This feature-based face/non-face classificator is a modification of an original idea by Maio and Maltoni [1]. The basic idea is to extract a directional image consisting of pairs of directions and strengths from the testing window. The directional image is then compared with an artificially constructed template. Since our approach on some points differs from the original work, we go through it in detail.

#### 4.5.1 Construction of the directional image

The computation of the directional image was described by Donahue and Rokhlin [2]. The algorithm works in a two-step fashion. First, the image point-normal is determined for each window of $2 \times 2$ pixels. Secondly, the normal vector field is projected onto the image plane, and $3 \times 3$ of these are averaged in order to reduce noise, and the directional vector is taken as the orthogonal vector to the resulting averaged projected normal vector.

[5] `http://l2r.cs.uiuc.edu/~cogcomp/`



(a) (b) (c)

Figure 8: (a) The $60 \times 60$ geometrically normalized face image (best linear fit plane subtracted and histogram equalization applied), and corresponding (b) $n_1$ and (c) $n_2$ normal vector component.

**Point-normal determination** The point-normal $\mathbf{n}(x, y) = (n_1, n_2, 1)$ is determined from the $2 \times 2$ neighborhood of $(x, y)$ constituted by the pixels at $(x + 1, y + 1)$, $(x, y + 1)$, $(x, y)$, and $(x + 1, y)$. The pixel values at these locations are denoted $a_1$, $a_2$, $a_3$, and $a_4$, respectively. The components $n_1$ and $n_2$ are then obtained by fitting the pixel values to the plane

$$p(x, y) = -n_1 x - n_2 y + c \tag{11}$$

using the least squares method, giving

$$n_1 = (-a_1 + a_2 + a_3 - a_4)/4 \tag{12}$$

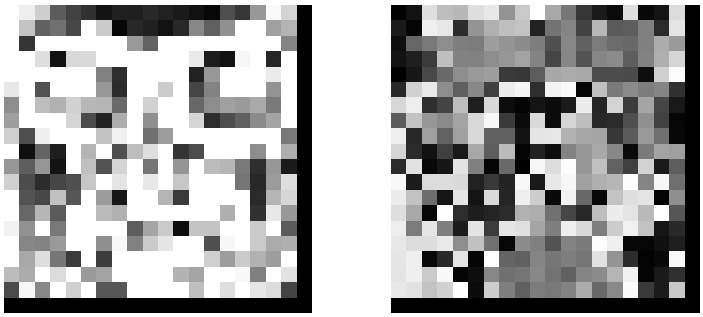
$$n_2 = (-a_1 - a_2 + a_3 + a_4)/4 \tag{13}$$

$$c = (a_1 + a_2 + a_3 + a_4)/4 \tag{14}$$

The resulting normal vector component images are shown in figure 8.

**Determining the average tangent direction** For determining the average tangent direction and strength in the image we choose an approach slightly different from the one outlined by Donahue and Rokhlin [2]. The projected normal vector is obtained by averaging the $n_1$ and $n_2$ components over a $3 \times 3$ window, resulting in two $20 \times 20$ normal images. The directional strength of this image element is taken as $\sqrt{n_1^2 + n_2^2}$. The corresponding undirected angle in the range $[-90°, 90°]$ is obtained from $\arctan(n_2/n_1)$ and encoded as an integer in the range $[0, 255]$, where 0 represents $-90°$, and 255 represents $90°$. The resulting images are shown in figure 9.

The combination of direction and strength represented by these two images can be visualized by drawing an image with line segments with corresponding lengths and directions, see figure 10. We observe that the directions



(a) (b)

Figure 9: (a) The $20 \times 20$ image representing the directional strength of the $60 \times 60$ face image shown in figure 8, and (b) corresponding directional image.

are close to horizontal in the regions of the eyes, eyebrows, and mouth, and close to vertical in the region of the root of the nose.

#### 4.5.2 Construction of the directional template

The directional image computed for each testing window by the method outlined in the previous section is to be compared to an artificially designed directional template. We start with an initially empty directional image, and add elliptical elements for eyes, eyebrows, nose and mouth. All elements are taken to have a horizontal direction except the nose, which is vertical (cf. figure 9). In contrast with the work of Maio and Maltoni [1], we only use one such template. The geometric parameters for the template ($a$ and $b$ in Reference [1]) are chosen such that the directional template resembles the averaged face template used in the baseline algorithms (cf. figure 6) as closely as possible. The relative weights are chosen in accordance with the original work being $58(\times 2)$, $81(\times 2)$, 138, and 253 for the eyebrows, eyes, nose, and mouth, respectively. The resulting template is visualized in figure 11.

#### 4.5.3 The metric algorithm

When the template and the directional image corresponding to the testing window are known, the distance between the two can be found. Let $\sigma_{Iij}$ and $\sigma_{Tij}$ denote respectively the directional strength of the testing window and the template at position $(i, j)$, and $\phi_{Iij}$ and $\phi_{Tij}$ the corresponding directions encoded as described earlier. The metric is then defined as

$$d_{\textbf{Gradient}} = \frac{\sum_{i,j} \sigma_{Ii,j}\sigma_{Tij} \times \Delta(\phi_{Iij}, \phi_{Iij})}{\sum_{i,j} \sigma_{Iij}\sigma_{Tij}}, \tag{15}$$

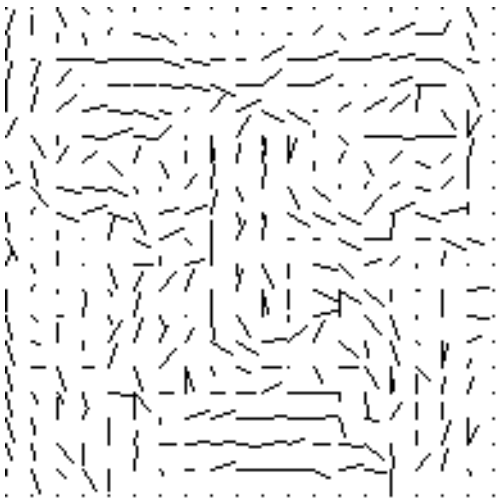


Figure 10: Visualization of the directional image represented in figure 9.

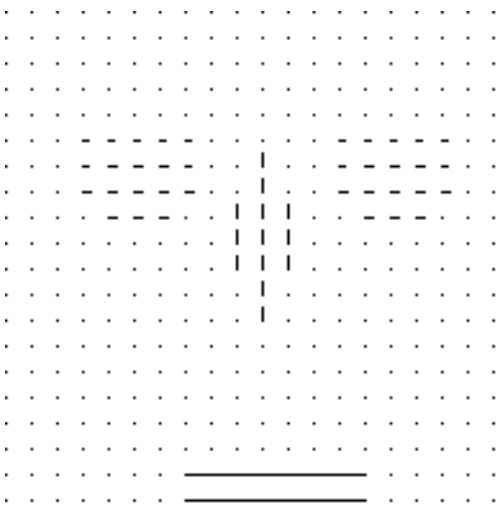


Figure 11: Visualization of the artificially constructed directional template. The dots correspond to locations with zero weight, and are not included in the distance function.

where $\Delta$ is a function returning the angle between the two direction arguments.[6]

### 4.6 Feature-based: Gabor

Gabor features are widely applied for local feature extraction in face recognition systems and have also been used for face detection [9] and facial feature detection [13]. Gabor features are extracted using a set of 2D Gabor filters [4]. In our implementation we use a set of 40 filters (5 sizes, 8 orientations, figure 12) $\psi_{\mathbf{k}}$ generated from a wavelet expansion of a Gabor kernel, parameterized (determining the wavelength and orientation) by the vector $\mathbf{k}$ [15]:

$$\psi_{\mathbf{k}}(\mathbf{x}) = \frac{\mathbf{k}^2}{\sigma^2} e^{-\frac{\mathbf{k}^2\mathbf{x}^2}{2\sigma^2}} \left( e^{i\mathbf{k}\cdot\mathbf{x}} - e^{-\frac{\sigma^2}{2}} \right), \tag{16}$$

where

$$\mathbf{k} = \begin{pmatrix} k_\nu \cos\phi_\mu \\ k_\nu \sin\phi_\mu \end{pmatrix}, \quad k_\nu = 2^{-\frac{\nu+2}{2}}\pi, \quad \phi_\mu = \mu\frac{\pi}{8}. \tag{17}$$

We create a Gabor template which is a $60 \times 60$ window where the set of 40 Gabor coefficients have been extracted at the two locations corresponding to the eyes. In other words, we have a template which simply represents the average eyes. We only keep the magnitude of the complex coefficients and compare the template with the extracted subwindow at each location using the normalized correlation coefficient.

## 5 Results

### 5.1 The effect of preprocessing

In order to determine what kind of preprocessing to apply for the training images, templates and testing images, we try out different approaches using the **bE**-algorithm. Two different kinds of preprocessing – subtraction of best fit linear plane and histogram equalization – are applied in different combinations. The results given in figure 13 show that the preprocessing is of major importance for the algorithm to work correctly. The highest $CD$ is obtained when both kinds of preprocessing is applied to both the training images and the template. This combination is thus applied for the remaining algorithms.

### 5.2 Face/non-face classification results

The results for all algorithms are shown in figure 14 for the **Easy** dataset, figure 15 for the **Scarf** dataset, and figure 16 for the **Sunglasses** dataset.

[6]Note that this cannot be obtained by simply subtracting the two angles, since, e.g., the directional difference between $-89°$ and $89°$ is $2°$ and not $178°$.

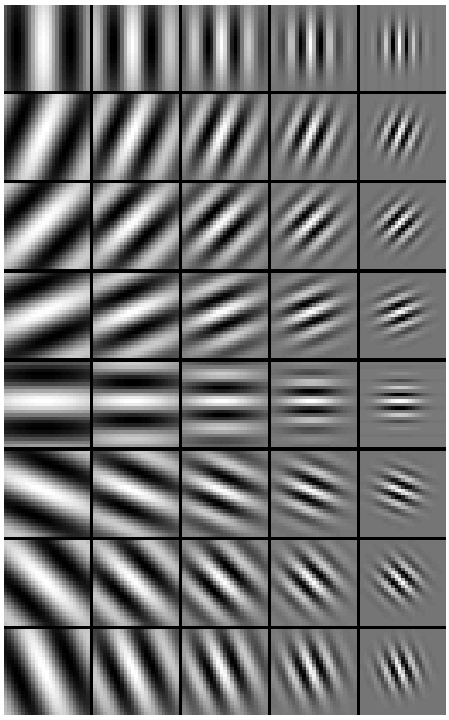


Figure 12: Example of Gabor filters (five sizes and eight orientations).

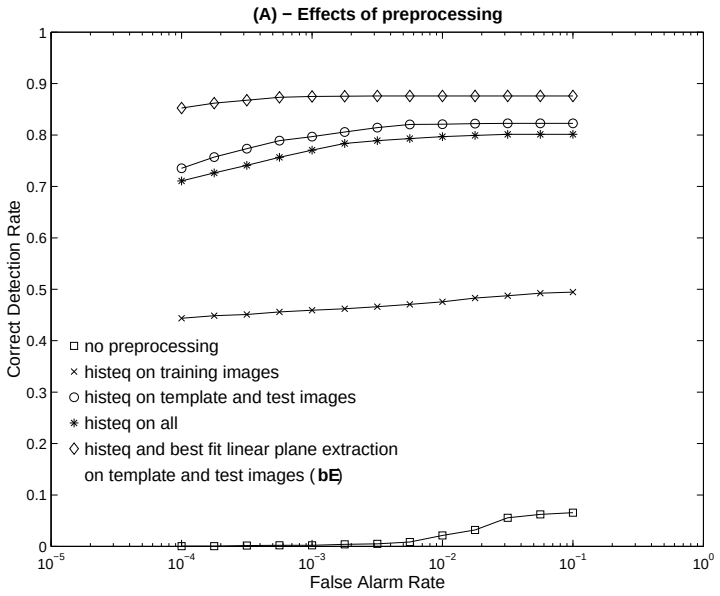



Figure 13: The effect of preprocessing on the ROC curve using the **bE** algorithm.

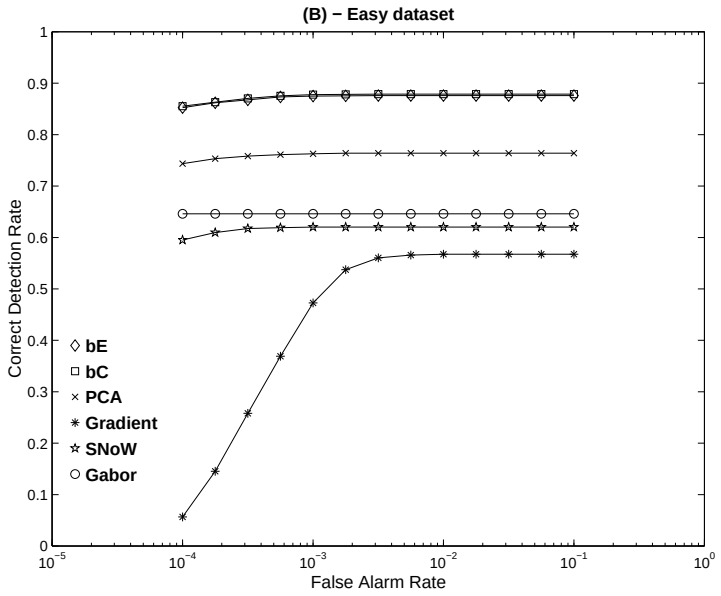



Figure 14: Comparison of algorithms on the **Easy** dataset.

The baseline template matching algorithms are the overall best performing algorithms.

# 6 Discussion

The **PCA** algorithm gives the best results when using only the first principal component, thus reducing the algorithm to a modified correlation measure. The reason for this is possibly that the size of the training set is not large enough to provide a general basis for representing the class of faces. We believe that this could be the reason since a general face class consisting of geometrically normalized faces should be Gaussian [8], and examination of the training data when plotting the projection coefficients of the first two principal components (figure 17) showed us that this is not the case.

The size of the training set is possibly also the reason to the poor performance of the **SNoW** classifier, since the classifier had no problems learning the face/non-face classification during training and initial testing. During training of the SNoW classifier, the bootstrap method had little effect, since the classifier hardly made errors. The results also show that it is not due to a high false detection rate that the classifier performs below expectations, but the correct detection rate is not high enough, which could indicate that we need a much larger positive training set. In the original application of the SNoW face detector in Roth et al. [11], additional positive training samples was generated by mirroring and slight rotation of the face images. This

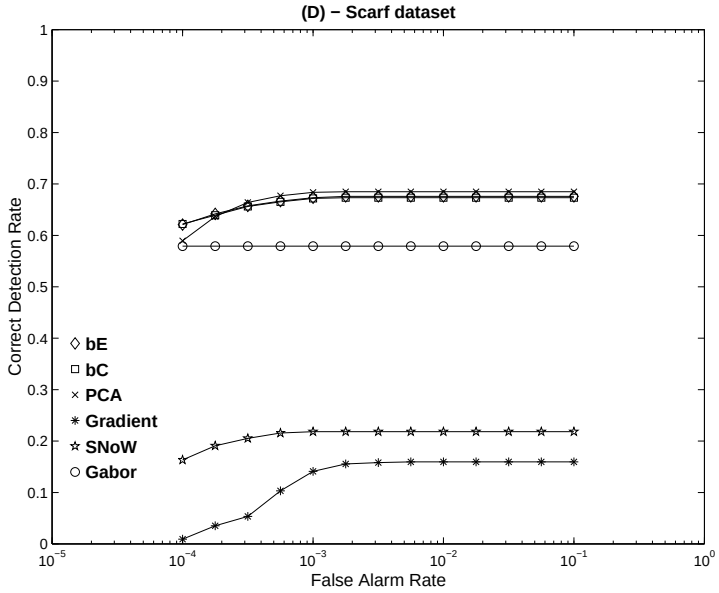



Figure 15: Comparison of algorithms on the **Scarf** dataset.

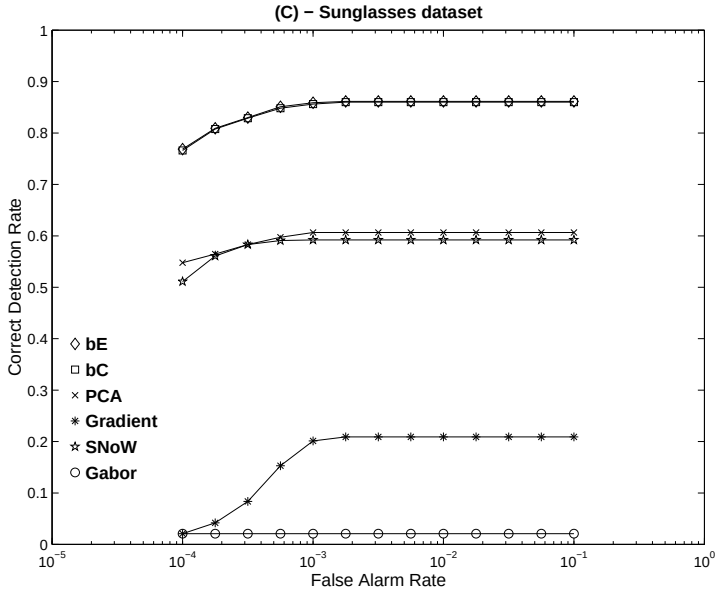



Figure 16: Comparison of algorithms on the **Sunglasses** dataset.

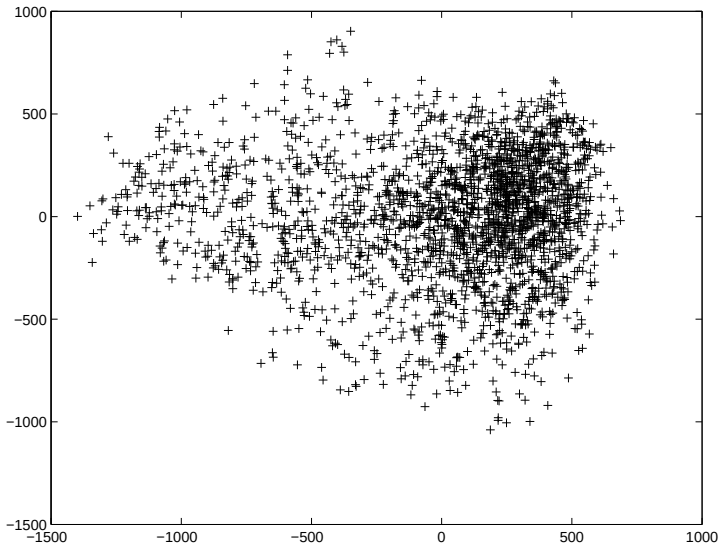



Figure 17: Plot of the training set coefficient represented by the first two principal components.

procedure might very well have improved performance significantly in our experiments also.

The abandoning of the ellipsis around the face introduced an important alteration for the **Gradient** algorithm compared to the original work of Maio and Maltoni [1]. This might explain why the algorithm performs less than ideally. In the original work, the total weight of the ellipsis in the distance function was approximately 2–3 times the weight of the remaining template, indicating the importance of the ellipses.

Selection of the Gabor filters for the **Gabor** algorithm was accomplished by manual inspection, and we have no reason to believe that these filters are optimal for representing the face class (in terms of the eyes here). This is a plausible explanation of the relatively poor performance of the Gabor classifier. However, we must note that this classifier is based only on locating the eyes at two fixed locations, and as we can see this leads to decent performance for the easy and scarf datasets, while the algorithm completely collapses for the sunglasses dataset.

# 7 Conclusions

To our knowledge, detailed comparison of the preprocessing effects in face detection has not been presented earlier, thus figure 13 is quite significant. Simple template matching algorithms are not always used as a baseline for comparison, and our results should be taken as a strong indication that this

is necessary. Due to the complexity of the other algorithms such as different selection of training set size, training parameters, template and filter design, improved performance can most likely be achieved. However, in our scenario, the simple baseline algorithms show impressive performance with the right kind of preprocessing.

## References


[1] D. Maio and Davide Maltoni. Real-time face location on gray-scale static images. *Pattern recognition* **33** (2000) 152–1539.

[2] M. J. Donahue and S. I. Rokhlin. On the Use of Level Curves in Image Analysis. *CVGIP: Image Understanding* **57** (1993) 185–203.

[3] A. J. Carlson, C. M. Cumby, J. L. Rosen, and D. Roth. SNoW user's guide. Technical Report UIUC-DCS-R-99-210, UIUC CS dept, 1999.

[4] J. G. Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research*, 20:847–856, 1980.

[5] E. Hjelmås and B. K. Low. Face detection: A survey. submitted.

[6] A. M. Martinez and R. Benavente. The AR face database. Technical Report CVC 24, School of Elec. and Comp. Eng., Purdue University, 1998.

[7] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, 1999.

[8] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1), 1997.

[9] K. Okada, J. Steffens, T. Maurer, H. Hong, E. Elagin, H. Neven, and C. v. d. Malsburg. The Bochum/USC face recognition system and how it fared in the FERET phase III test. In *Face Recognition: From Theory to Application*. Springer, 1998.

[10] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[11] D. Roth, M.-H. Yang, and N. Ahuja. A SNoW-based face detector. In *Advances in Neural Information Processing Systems 12 (NIPS 12)*. MIT press, 2000.

[12] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America*, 4:519–524, 1987.

[13] F. Smeraldi, O. Carmona, and J. Bigün. Saccadic search with Gabor features applied to eye detection and real-time head tracking. *Image and Vision Computing*, 18:323–329, 2000.

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

[15] M. Lades, J. C. Vorbrüggen, J. Buhmann, J. Lange, C. von der Malsburg, R. P. Würtz, and W. Konen. Distortion Invariant Object Recognition in the Dynamic Link Architecture. *IEEE Transactions on Computers*, 42(3), 1993.