Face Verification for Real-time Applications*

Raquel Romano†
David Beymer
Tomaso Poggio
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Abstract

In this paper we describe a system that performs automatic real-time face verification for use in such human-machine interface applications as automated security systems. We demonstrate that simple correlation strategies on template-based models are sufficient for many applications in which the identity of a face in a novel image must be verified quickly and reliably from a single reference image. We present the results of testing the system on over 1000 face images, including images acquired interactively by the interactive system operating under realistic office-like conditions. The system has been integrated into a screen locking application which permits users access to workstations by performing face verification in lieu of password authentication.

1 Introduction

While researchers have made numerous attempts at solving the elusive problem of face recognition, many approaches focus on constructing elaborate models for representing human faces that in turn require time-consuming recognition algorithms and/or extensive training sessions. However, for certain practical applications, speed and ease of use are overriding considerations, and the conditions under which faces are imaged are stable enough that simple models with fast recognition algorithms will suffice.

This paper presents a real-time face verification system that operates in realistic conditions and is directly applicable to the task of automated security. The choice of representation and verification strategy is influenced by the following requirements demanded by such practical applications:

- Fast, real-time execution.
- Inexpensive hardware.
- Convenience and ease of use.
- Low rate of false entries.
- Flexible security level.

The system models a face with templates of facial features extracted from raw image data and compares face models and images using normalized cross-correlation as a distance measure. The models stored in the system's library are built from only a single reference image per person. The narrow range of targeted applications allows us to exploit the following assumptions about the conditions under which faces presented to the system are imaged:

- The user provides a consistent facial expression.

*This research is sponsored by a grant from ARPA-ONR under contract N00014-92-J-1879.
†This author is supported by the AT&T Bell Laboratories Cooperative Research Fellowship Program.

- The user presents a frontal face view.
- The background scene is unconstrained, but the face to be verified is not occluded.
- The scene illumination varies uniformly among images.

This paper first outlines relevant existing work in face recognition in Section 2. Section 4 describes the template-based representation of human faces. Section 3 outlines the normalized cross-correlation coefficient that performs image comparison. Section 5 presents the algorithm used by the verification system, and Section 6 describes experiments and results.

2 Related work

Terzopolous and Waters [Waters,*et.al.*,91] and Essa and Pentland [Essa,*et.al.*,95] model faces with physical 3D models that are useful for facial expression recognition and potentially for recognition under varying expression. Gordon [Gordon,92] uses features extracted from 3D range data to model faces for recognition, an approach that requires special hardware for data acquisition.

Feature-based approaches locate a collection of facial characteristics in a face image and build a model from the spatial configurations of these feature points [Manjunath,*et.al*,92]. This approach may be problematic due to error in the measurements of feature configurations taken from images of a single person. If the error is of the same order as the variation among measurements taken from images of different people [Brunelli,*et.al.*,93], the technique will discriminate poorly among faces.

The most common face recognition strategies that use representations based directly on raw grey-level intensity data are the template-based models of Beymer [Beymer,93] and Brunelli and Poggio [Brunelli,*et.al.*,93] and the eigenvector decompositions of Turk and Pentland [Turk,*et.al.*,91, Moghaddam,*et.al.*,94]. The two methods are analogous in that both compare face images by measuring the sum of squared differences between subimages containing facial features. While template-matching measures this distance in the full N-dimensional image space for an N-pixel template, the eigenvector decomposition technique measures this distance in a lower-dimensional subspace, called the eigenspace, whose $k << N$ basis vectors represent directions of highest variance among images. Comparing two images in the full image space requires $O(N)$ operations while comparing their projections into the eigenspace requires only $O(k)$ operations. This dimensionality reduction may improve efficiency in general face recognition tasks when a single image must be compared to a large set of images. However, in terms of applicability to real-time verification tasks that require the comparison of only one pair of images, the projection of an image into the eigenspace incurs an extra cost of $O(kN)$ that outweighs the computation time saved by computing distance in the eigenspace.

3 Image Comparison

Our face verification system is based on the work of Beymer [Beymer,93] and Brunelli and Poggio [Brunelli,*et.al.*,93], who compare two face images by computing the distances between pairs of subimages depicting salient facial features using normalized cross-correlation as a distance measure. The normalized cross-correlation coefficient between a template T and a subimage S of identical dimensions is defined as

$$r_n(T,S) = \frac{1}{\sigma_t \sigma_s}\left(\sum_{i=1}^{N}(t_i s_i) - N\mu_t\mu_s\right),$$

where

$$\mathbf{T} = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{pmatrix} \quad \text{and} \quad \mathbf{S} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{pmatrix}$$

are the N-pixel template and subimage treated as vectors of grey-level values, σ_t and σ_s are the respective standard deviations of the template and the subimage, and $\mu_t = \frac{1}{N}\sum_{j=1}^{N} t_j$ and $\mu_s = \frac{1}{N}\sum_{j=1}^{N} s_j$ are their respective means.

The normalized cross-correlation coefficient reduces the influence of ambient scene illumination on the image by effectively adjusting the pixel values to have zero mean and unit variance If we assume T and S have been normalized in this manner, maximizing the cross-correlation $\sum t_i s_i$ is equivalent to minimizing the Euclidean distance $\sum (t_i - s_i)^2$ between two images, since $\sum s_i^2 = 1$.

4 Face Representation

This section presents the model used by the system to represent faces. The system expects as input a novel image with a proposed identity and returns confirmation or denial of the face's identity using a reference library of one face model per person. Under the assumption that the face of a given person will have roughly the same expression in all input images, only a single reference image is required to build a face model.

Offline entry of a new user into the library of models consists of capturing a single frame of the user's face and extracting templates of facial features from the reference image. To ensure that the model is accurately built, six points demarcating prominent facial features, the left and right eye centers, the left and right nose lobes, and the left and right mouth corners, are manually labeled. The left and right eye centers are reference points for automatically normalizing the new library entry to ensure that every model is scaled, rotated, and translated to a standard width, orientation, and position. The six feature points also guide the automatic extraction of subimages of each eye, the nose, the mouth, and the whole face.

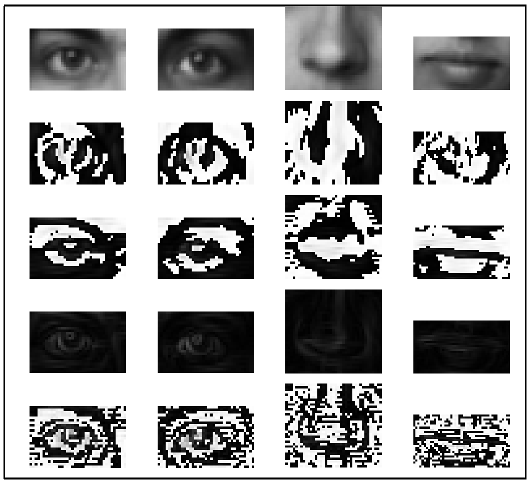


Figure 1: Grey-level and filtered templates comprising a face model.

Figure 1 displays an example set of feature templates extracted from normalized reference images. In addition to the set of grey-level templates, each model includes templates extracted from the result of filtering the reference image with the following differential operators: the horizontal and vertical components of discrete approximations of the image gradient, the magnitude of the gradient image, and a discrete approximation of the Laplacian of the image.

The eye templates and full face image are stored at several scales to compensate for the variable size of a face in an image when performing face and feature detection. Low resolution versions of eye and face templates are also stored for use in the hierarchical correlation strategy used in feature detection. Section 5 describes in detail the use of multiple resolutions and scales for locating the face and eyes. Figure 2 illustrates the scales and resolutions at which eye and face templates are stored. The total storage requirement of the full template set is less than 40 kilobytes.

Scale	Image Resolution		
	$\frac{1}{4}$	$\frac{1}{16}$	$\frac{1}{64}$
0.8	left eye right eye	left eye right eye	face
1.0	left eye right eye	left eye right eye	face
1.2	left eye right eye	left eye right eye	face

Figure 2: Organization of the multiresolution template set used for feature detection at multiple scales.

5 Face Verification System

The real-time interactive system performs the following steps to perform verification:

1. Capture a novel image and a proposed identity from the environment.
2. Normalize the face in the image to a standard position, orientation, and scale.
3. Verify the match between the face and its identity by matching templates of facial features.
4. Return either a confirmation or a denial of the face's identity.

Since the verification is complete in several seconds, the system may be modified to continuously grab new images as long as verification fails. Then if an authentic user is rejected initially due to an unusual pose or expression, one of the successive frames is more likely to be accepted.

The following two sections describe the normalization and verification components of the system.

5.1 Normalization

Normalization of a face image reduces the effects of variation in the distance, location, and rotation of the head relative to the camera. The number of scales at which templates are stored determines the range of distances from the camera that the system can handle. This range may be set to handle more or fewer distances falling within a smaller or larger range by building more templates at varying scales into the library models.

The normalization process is composed of four stages: face detection, eye detection, eye refinement, and geometric registration. We adopt a coarse-to-fine search strategy to use the first stage's estimate for the face location to guide the second stage's search for rough eye locations. The final stage then uses the optical flow between these eye location estimates and stored eye templates to perform fine adjustments to the eye positions. Once the eye locations have been found, the system brings the face into registration with the model face by performing a two-dimensional rigid transformation and uniform scaling using the eye centers as reference points.

Coarse face detection is performed on a version of the input image that has been multiply subsampled by a factor of two and smoothed. As indicated in Figure 2, the face model of the person to be verified contains low resolution face templates at three scales relative to the standard interocular distance in order to compensate for variation in camera distance. Each of the face templates is compared to a subimage centered at every pixel in the low resolution image using the normalized cross-correlation coefficient defined in Section 3. Exhaustive search returns a correlation coefficient corresponding to each pixel in the low resolution input image. For each scaled face template, a map of correlation coefficients indicates how well each image location matches the face template. The map that attains the highest correlation coefficient indicates the scale that most closely approximates the face size in the input image. The other two sets of face candidates are discarded, and future processing uses only the templates at the chosen scale. Pixels that reach local maxima in the correlation map for the chosen scale are thresholded and returned as potential face candidates. For each possible face center, regions above the face center that are likely to contain the eyes are delimited and passed to the eye finder to restrict the search area to be explored at higher resolutions. The face detector acts primarily as an initial filter to immediately discard image regions that bear little holistic resemblance to the face in the model's reference image.

The eye detection component processes the input image at a finer resolution level than that of the face detection component since the eye is a more detailed feature than the face and requires more precise localization. The eye search uses only the eye templates at the scale chosen by the face finder as most closely approximating the scale of the face in the input image. For each eye, the eye template at the current resolution level is correlated with the image only at those points lying within the left or right eye regions selected by the face finder. For each eye, correlation at pixels in the selected regions returns a map of correlation coefficients. The system thresholds the pixels at the local maxima of each map, and returns the surviving pixels as candidate eye locations to be considered at the next finest resolution level.

At each successive level, pixels not centered at eye locations will return lower results because at higher resolutions the correlation coefficient is more sensitive to detail. In order to save computation time, the system stops performing hierarchical correlation once the image resolution climbs to $\frac{1}{64}$ the original sampling density and returns the pixel with the highest correlation coefficient for each eye.

Since the estimated locations have been chosen at a low resolution, transforming the coordinates to the corresponding locations in the full resolution image may place the eye positions several pixels away from the actual eye centers. The refinement stage uses optical flow to perform fine adjustments to these locations.

To calculate the displacement of an eye center estimate from its actual location, the refinement step computes the optical flow field between the stored eye template and the subimage of identical dimensions centered at the current estimate. Each eye's position is then adjusted by the flow vector at its estimated center. The flow computation and adjustment of eye labels may be iterated in order to further improve precision.

Once the eye locations have been found, the system brings the face into registration with the model face by performing a two-dimensional rigid transformation and uniform scaling using the eye centers as reference points. These two points determine the rotation of the face with respect to the camera in the plane parallel to the image plane. Rotation of the image by this angle fixes the line through the eye centers, the interocular axis, at a horizontal orientation. The distance between the eye centers, or interocular distance, reflects the distance between the camera and the face. Scaling the image to fix the interocular distance to the standardized interocular distance in the model brings the face to the standard template size. Once the image is rotated and scaled, the eye positions guide the extraction of a subimage containing only the face. Extraction of the face subimage effectively translates the face to a known position, so that the input image is now geometrically registered with the normalized model image.

5.2 Verification

The verification stage receives a normalized image from the eye detection stage, computes the similarity between the image and the model, classifies the list of similarity measures as a good match or a poor match, and makes the final decision to accept or reject the individual.

After normalization, the eye locations in the input image have been spatially registered with the eye locations in the reference image. Under the assumption that the expressions in the two images are similar, the other facial features should also be well-registered when these two reference points are aligned. The positions of the four feature templates in the model guide the extraction of subimages around the corresponding features in the normalized input image. Each feature template of the model is correlated with the corresponding subimage of the input image using the normalized cross-correlation coefficient.

Each comparison between template and subimage returns a correlation coefficient. If there are m templates of facial features and n types of filters, a total of mn correlation coefficients form a vector representing the similarity between the input image and the model. Figure 3 shows the subimage comparison between the input and the model for each of four templates and five preprocessing types. From this 20-dimensional vector, the system must decide whether the similarity scores indicate a match or a mismatch between the user and the suggested identity and accept or reject the user accordingly.

In the last stage of the verification system, each list of d correlation coefficients is a point $\mathbf{x} = (x_1, \ldots, x_d)$ in $\mathbb{R}^d$, where $x_i \in [-1, 1]$, to be classified as a positive or negative example. The classifier must accept the correlation vector if the correlation scores indicate a match, and reject if the scores suggest a false entry. Geometrically, an ideal classifier defines a surface in $\mathbb{R}^d$ that separates the positive examples from the negative examples. Figure 4 shows a set of points representing 20-dimensional correlation vectors projected from $\mathbb{R}^{20}$ onto $\mathbb{R}^3$.

Our system's classifier is a modified nearest mean classifier. In our case, there are only two classes, a positive class and a negative class, so if $\mathbf{m}_p$ and $\mathbf{m}_n$ are the positive and negative class means respectively, then the classifier's decision rule is as follows:

$$\text{If } d(\mathbf{m}_p, \mathbf{x}) < d(\mathbf{m}_n, \mathbf{x}), \text{ accept.}$$
$$\text{If } d(\mathbf{m}_p, \mathbf{x}) > d(\mathbf{m}_n, \mathbf{x}), \text{ reject.}$$

When d is the Euclidean metric, the decision boundary, $d(\mathbf{m}_p, \mathbf{x}) = d(\mathbf{m}_n, \mathbf{x})$, is linear and the decision rule is equivalent to thresholding a weighted sum of the correlation vector components. The decision boundary is therefore a hyperplane in $\mathbb{R}^d$.

Since the negative examples with low coefficients are easy to identify as mismatches, while those with higher coefficients are difficult to distinguish from positive examples, we tune the classifier to separate the points lying near the boundary between positive and negative examples. We consider only those negative examples with correlation scores high enough to be potentially mistaken for positive examples. We call such negative examples *near misses*, and define them to be the set of negative examples falling within a fixed spherical neighborhood around the positive mean. Only the near misses are considered in the computation of the negative class mean.

The choice the radius r that defines the set of near misses will determine the value of $\mathbf{m}_n$, which in turn dictates the weights $\mathbf{w}$ and the threshold T of the classifier. The linear search for the ideal radius is therefore a method for estimating the weights and threshold of a linear classifier. The system

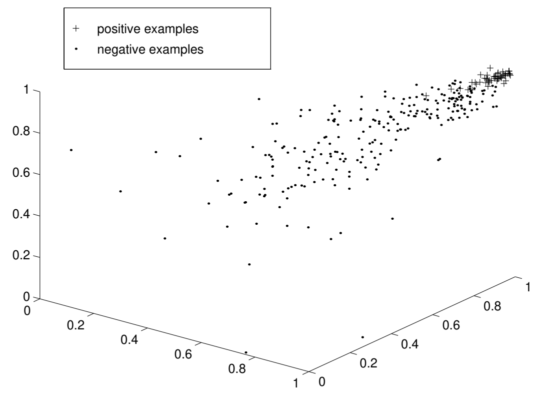


Figure 4: Projection of a set of examples in $\mathbb{R}^{20}$ onto $\mathbb{R}^3$.

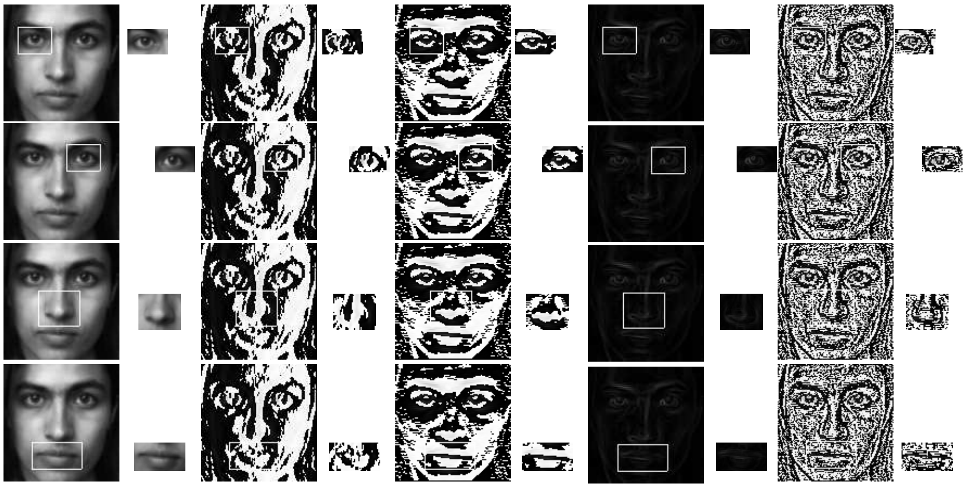


Figure 3: Input and model subimage comparisons for each of 4 templates (left eye, right eye, nose, mouth) and 5 preprocessing types (grey-level, horizontal gradient component, vertical gradient component, gradient magnitude, and Laplacian operator).

applies the linear decision rule to the new correlation vector returned by the correlation stage and accepts or rejects the input image according to the classifier's decision.

6 Results

The face verification system has been incorporated into a screen locking tool for a workstation with a camera. The program runs a screen saver until it receives a keystroke or mouse click. The system then grabs a new frame, performs verification with the current user's identity, and either unlocks the screen if verification succeeds or continues running the screen saver if verification fails.

The system's reliability was tested in batch mode on a publicly available image set and on a set of images acquired interactively under realistic office conditions by the real-time system.

The first set of images used for testing belongs to the University of Essex face database [Essex]. Each 180×200 pixel JPEG compressed 24 bit color image was decompressed and converted to an 8 bit grey-scale image. Images of faces with glasses have been eliminated because the specularities introduced by the lenses often saturate the image resulting in a loss of information that impedes recognition. The remaining image set contains 20 images per person for 81 people, totaling 1620 images of frontal views under constant illumination in uncluttered background scenes.

The second set of images used for testing was taken at the MIT Artificial Intelligence Laboratory by the real-time verification system. The system runs on an SGI Indy workstation and captures images with the IndyCam digital color video camera bundled with the workstation as a standard peripheral device. For each image, the view is frontal, the eyes are open, visible, and free of glasses, and the expression is fixed across all images of a given individual. The set consists of 158 360×240 pixel images of 48 people with an average of 4 images per person.

For each data set, the images are divided into non-overlapping sets of training and testing images. The correlation vectors used for training are computed from images drawn exclusively from the training set, and the correlation vectors used for testing are computed from images drawn only from the testing set. The purpose of training is to compute the positive and negative means needed to perform the nearest mean classification. At the same time that the training stage computes the positive and negative class means, it can tune the system to provide a particular level of reliability for a given database. Just as varying the threshold of a linear classifier effects a change in the decision boundary, so does varying the neighborhood from which negative examples are drawn to compute the negative class mean.

A larger set of near misses generally pulls the negative mean away from the positive class mean and increases the system's leniency, since a mean vector with low-valued components will encourage misclassification of high-scoring negative examples. A smaller set of near misses pulls the negative class mean closer to the positive examples and increases the system's level of security, since a mean vector with high-valued components leads to misclassification of low-scoring positive examples. We make the assumption that only a limited number of images may be available for training and therefore only use one image per person to compute the positive and negative class means.

To evaluate the success rate of the system and examine the trade-off between leniency and security, figure 5 plots the ROC curves for the both of the image sets. The varying parameter for the nearest mean classifier is the radius of the neighborhood around the positive mean from which near misses are drawn to compute the negative mean. The varying parameter for the thresholded unweighted sum of features is simply the value of the threshold. As expected, the two linear classifiers perform comparably well.

To evaluate the system's overall reliability, we maximize the verification rate, defined as the total percentage of correctly classified examples, both positive and negative. Table 1 displays the maximum verification rates attained by the two classifiers on both image sets. Since the total number of examples is dominated by negative examples, this accuracy rate is skewed in favor of the system's ability to reject false negatives. For extremely high security applications, a false acceptance rate of 99.999% can be attained at the expense of a false rejection rate of 90%.

Classifier	AI Lab	Essex
Nearest Mean	99.5%	99.85%
Thresholded Sum	99.5%	99.80%

Table 1: Comparison of verification accuracy rates between two classifiers on both image sets.

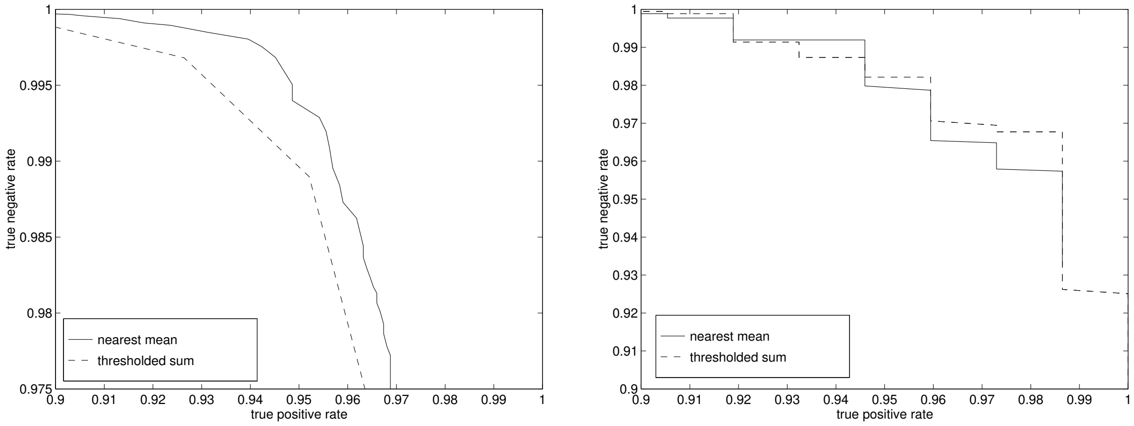
The average execution time of the real-time verification system alone is 2.7 seconds on an Indy workstation with a MIPS R4400 processor. The entire system with a built-in user interface that displays video and image windows runs in an average of 5 seconds.

7 Conclusions

The real-time system demonstrates the feasibility of using simple models and algorithms to perform fast face verification for specific applications. It may be installed on any SGI Indy workstation with video input from the IndyCam or an alternate video source and provides easy entrance of a new person into the model library within several seconds. The system yields an estimated false entry rate of less than .5% and may be tuned to be more tolerant or less tolerant depending on the security level demanded by the application.

References

[Beymer,93] David J. Beymer. Face recognition under varying pose. A.I. Memo 1461, Massachusetts Institute of Technology Artificial Intelligence Library, Cambridge, Massachusetts, December 1993.



(a) (b)

Figure 5: ROC curves for (a) the Essex University image set and (b) the Artificial Intelligence Laboratory database.

[Brunelli,*et.al.*,93] Roberto Brunelli and Tomaso Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, October 1993.

[Burt,88] Peter J. Burt. Smart sensing within a Pyramid Vision Machine. *Proceedings of the IEEE*, 76(8):1006–1015, August 1988.

[DeLeo,93] James M. DeLeo. Receiver operating characteristic laboratory (ROCLAB): software for developing decision strategies that account for uncertainty. In *International Symposium on Uncertainty Modeling and Analysis*, pages 318–325, 1993.

[Essa,*et.al.*,95] Irfan A. Essa and Alex P. Pentland. Facial expression recognition using a dynamic model and motion energy. In *International Conference on Computer Vision*, pages 360–367, 1995.

[Gilbert et.al,93] Jeffrey M. Gilbert and Woodward Yang. A real-time face recognition system using custom VLSI hardware. In *1993 Computer Architectures for Machine Perception*, pages 58–66, New Orleans, Louisiana, 1993.

[Gordon,92] G.G. Gordon. Face recognition based on depth and curvature features. In *Computer Vision and Pattern Recognition*, pages 808–810, 1992.

[Horn,86] Berthold K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[Kurita,*et.al*,92] T. Kurita, N. Otsu, and T. Sato. A face recognition method using higher order local autocorrelation and multivariate analysis. In *Eleventh International Conference on Pattern Recognition*, volume 2, pages 213–216, The Hague, The Netherlands, 1992.

[Manjunath,*et.al*,92] B.S. Manjunath, R. Chellappa, and C. von der Malsburg. A feature based approach to face recognition. In *1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 373–378, 1992.

[Moghaddam,*et.al.*,94] Baback Moghaddam and Alex Pentland. Face recognition using view-based and modular eigenspaces. Technical Report 301, MIT Media Laboratory, Cambridge, Massachusetts, 1994.

[Essex] University of Essex `http://hp1.essex.ac.uk/projects/vision/faces/male`.

[Sinha,94] Pawan Sinha. Object recognition via image invariants. *Investigative Ophthalmology and Visual Science*, 35:1735–1740, 1994.

[Sung,*et.al.*,94] Kah-Kay Sung and Tomaso Poggio. Example-based learning for view-based human face detection. A.I. Memo 1521, MIT Artificial Intelligence Laboratory, Cambridge, Massachusetts, 1994.

[Therrien,89] Charles W. Therrien. *Decision Estimation and Classification.* John Wiley & Sons, 1989.

[Turk,*et.al.*,91] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[Waters,*et.al.*,91] K. Waters and D. Terzopoulos. Modelling and animating faces using scanned data. *Journal of Visualization and Computer Animation*, 2(4):123–128, 1991.

[Xie,*et.al.*,94] X. Xie, R. Sudhakar, and H. Zhuang. On improving eye feature extraction using deformable templates. *Pattern Recognition*, 27(6):791–799, 1994.

[Yuille,*et.al.*,92] Alan L. Yuille, Peter W. Hallinan, and David S. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.