

Eigenfaces for Recognition

Matthew Turk and Alex Pentland

Vision and Modeling Group
The Media Laboratory
Massachusetts Institute of Technology

Abstract

■ We have developed a near-real-time computer system that can locate and track a subject's head, and then recognize the person by comparing characteristics of the face to those of known individuals. The computational approach taken in this system is motivated by both physiology and information theory, as well as by the practical requirements of near-real-time performance and accuracy. Our approach treats the face recognition problem as an intrinsically two-dimensional (2-D), recognition problem rather than requiring recovery of three-dimensional geometry, taking advantage of the fact that faces are normally upright and thus may be described by a small set of 2-D characteristic views. The system functions by projecting

face images onto a feature space that spans the significant variations among known face images. The significant features are known as "eigenfaces," because they are the eigenvectors (principal components) of the set of faces; they do not necessarily correspond to features such as eyes, ears, and noses. The projection operation characterizes an individual face by a weighted sum of the eigenface features, and so to recognize a particular face this necessary only to compare those weights to those of known individuals. Some particular advantages of our approach are that it provides for the ability to learn and later recognize new faces in an unsupervised manner, and that it is easy to implement using a neural network architecture. ■

INTRODUCTION

The face is our primary focus of attention in social intercourse, playing a major role in conveying identity and emotion. Although the ability to infer intelligence or character from facial appearance is suspect, the human

can be important. Detecting faces in photographs, for instance, is an important problem in automating color film development, since the effect of many enhancement and noise reduction techniques depends on the picture content (e.g., faces should not be tinted green, while perhaps grass should).

Research supported by the National Science Foundation, Office of Naval Research, and the MIT Media Laboratory. This work was done while the authors were at the MIT Media Laboratory.

© 1991 International Association of Pattern Recognition

This paper is a U.S. Government work and, as such, is in the public domain in the United States of America.

Journal of Cognitive Neuroscience, Volume 3, Number 3

decoding face images may give insight into the information content of face images, emphasizing the significant local and global “features.” Such features may or may not be directly related to our intuitive notion of face features such as the eyes, nose, lips, and hair. This may have important implications for the use of identification tools, such as Identikit and PhotoFit (Bruce, 1989).

In the language of information theory, we want to extract the relevant information in a face image, encode it as efficiently as possible, and compare one face encoding with a database of models encoded similarly. A simple approach to extracting the information contained in an image of a face is to somehow capture the variation in a collection of face images, independent of any judgment of features, and use this information to encode and compare individual face images.

In mathematical terms, we wish to find the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images, treating an image as a point (or vector) in a very high dimensional space. The eigenvectors are ordered, each one corresponding to a particular eigenvalue, and the eigenvalue

of characteristic features or eigenpictures, perhaps an efficient way to learn and recognize faces would be to build up the characteristic features by experience over time and recognize particular faces by comparing the feature weights needed to (approximately) reconstruct them with the weights associated with known individuals. Each individual, therefore, would be characterized by the small set of feature or eigenpicture weights needed to describe and reconstruct them—an extremely compact representation when compared with the images themselves.

This approach to face recognition involves the following initialization operations:

1. Acquire an initial set of face images (the training set).
2. Calculate the eigenfaces from the training set, keeping only the M images that correspond to the highest eigenvalues. These M images define the *face space*. As new faces are experienced, the eigenfaces can be updated or recalculated.

3. Calculate the corresponding distribution of M of

in Figure 1. Such the correspondence between the rows in Figure 2, which represent individual face images, and the columns, which represent the eigenvectors, is shown in Figure 3. The rows are a set of principal components between faces.

Each individual face can be represented exactly in terms of a linear combination of the eigenvectors, and faces are also represented using only the “best” eigenvectors—those that have the highest eigenvalues, and which therefore account for the most variation within the set of face images. The best eigenvectors span an M -dimensional subspace—our *face space*—of all possible faces.

The idea of using eigenvectors was motivated by a well-known principle of low-dimensional data representation and dimensionality reduction: if you have a large set of vectors, you can find a small number of principal components that capture most of the variation in the data. In the case of face images, the principal components are the eigenvectors of the covariance matrix of the face images. The eigenvectors are ordered by their eigenvalues, which represent the amount of variation captured by each eigenvector. The first few eigenvectors capture the most variation, and the remaining eigenvectors capture less and less variation. By using only the first few eigenvectors, we can represent the face images in a much more compact way, while still capturing most of the variation in the data.

... eigenvectors are ordered by their eigenvalues, which represent the amount of variation captured by each eigenvector. The first few eigenvectors capture the most variation, and the remaining eigenvectors capture less and less variation.

1. Calculate a set of weights based on the input image and the eigenvectors by using the eigenvectors to reconstruct the image.

2. Compare the weights to the weights of the faces in the training set. If the weights are significantly different, the face is not in the training set.

3. If the weights are significantly different, the face is not in the training set.

4. If the weights are significantly different, the face is not in the training set.

5. If the weights are significantly different, the face is not in the training set.

Calculating the eigenfaces

Let's take a step back and look at the problem of face recognition. We want to find a way to represent a face image in a compact way, so that we can compare it to a set of known face images. The first step is to find a way to represent a face image in a compact way. One way to do this is to use eigenvectors. Eigenvectors are a set of vectors that are orthogonal to each other, and they capture the most variation in the data. By using only the first few eigenvectors, we can represent the face image in a much more compact way, while still capturing most of the variation in the data.

... eigenvectors are ordered by their eigenvalues, which represent the amount of variation captured by each eigenvector. The first few eigenvectors capture the most variation, and the remaining eigenvectors capture less and less variation.



Figure 1. (a) Face images used as the training set.



nent analysis (or Karhunen–Loeve expansion) is to find the vectors that best account for the distribution of face images within the entire image space. These vectors define the subspace of face images, which we call “face space.” Each vector is of length N^2 , describes an N by N image, and is a linear combination of the original face images. Because these vectors are the eigenvectors of the covariance matrix corresponding to the original face images, and because they are face-like in appearance, we refer to them as “eigenfaces.” Some examples of eigenfaces are shown in Figure 2.

Let the training set of face images be $\Gamma_1, \Gamma_2, \Gamma_3, \dots, \Gamma_M$. The average face of the set is defined by $\Psi = \frac{1}{M} \sum_{n=1}^M \Gamma_n$. Each face differs from the average by the vector $\Phi_i = \Gamma_i - \Psi$. An example training set is shown in Figure 1a, with the average face Ψ shown in Figure 1b. This set of very large vectors is then subject to principal component analysis, which seeks a set of M orthonormal vectors, $\mathbf{u}_n$, which best describes the distribution of the data. The k th vector, $\mathbf{u}_k$, is chosen such that

$$\lambda_k = \frac{1}{M} \sum_{n=1}^M (\mathbf{u}_k^T \Phi_n)^2 \quad (1)$$

is a maximum, subject to

$$\mathbf{u}_l^T \mathbf{u}_k = \delta_{lk} = \begin{cases} 1, & \text{if } l = k \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

The vectors $\mathbf{u}_k$ and scalars λ_k are the eigenvectors and eigenvalues, respectively, of the covariance matrix

$$\begin{aligned} C &= \frac{1}{M} \sum_{n=1}^M \Phi_n \Phi_n^T \\ &= A A^T \end{aligned} \quad (3)$$

where the matrix $A = [\Phi_1 \ \Phi_2 \ \dots \ \Phi_M]$. The matrix C , however, is N^2 by N^2 , and determining the N^2 eigenvectors and eigenvalues is an intractable task for typical image sizes. We need a computationally feasible method to find these eigenvectors.

If the number of data points in the image space is less than the dimension of the space ($M < N^2$), there will be only $M - 1$, rather than N^2 , meaningful eigenvectors. (The remaining eigenvectors will have associated eigenvalues of zero.) Fortunately we can solve for the N^2 -dimensional eigenvectors in this case by first solving for the eigenvectors of an M by M matrix—e.g., solving a 16×16 matrix rather than a $16,384 \times 16,384$ matrix—



Figure 1. (b) The average face Ψ .

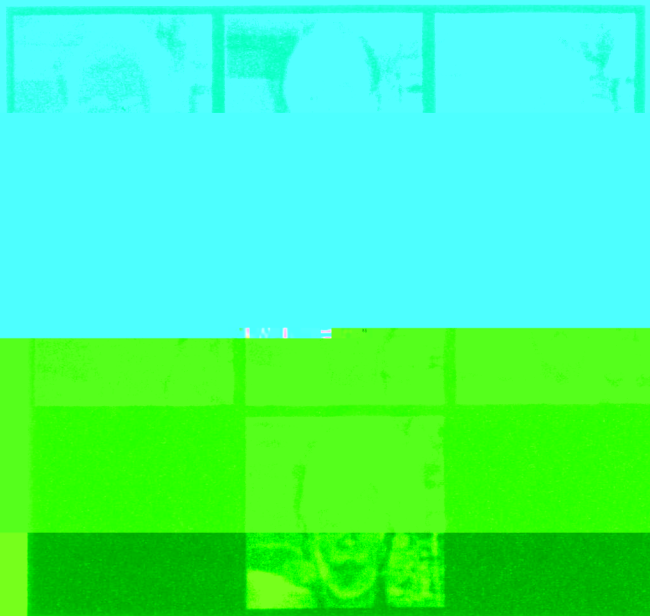


Figure 2. Seven of the eigenfaces calculated from the input images of Figure 1.

and then taking appropriate linear combinations of the face images Φ_i . Consider the eigenvectors $\mathbf{v}_i$ of $A^T A$ such that

$$A^T A \mathbf{v}_i = \mu_i \mathbf{v}_i \quad (4)$$

Premultiplying both sides by A , we have

$$A A^T A \mathbf{v}_i = \mu_i A \mathbf{v}_i \quad (5)$$

from which we see that $A \mathbf{v}_i$ are the eigenvectors of $C = A A^T$.

Following this analysis, we construct the M by M matrix

With this analysis the calculations are greatly reduced, from the order of the number of pixels in the images (N^2) to the order of the number of images in the training set (M). In practice, the training set of face images will be relatively small ($M \ll N^2$), and the calculations become quite manageable. The associated eigenvalues allow us to rank the eigenvectors according to their usefulness in characterizing the variation among the images. Figure 2 shows the top seven eigenfaces derived from the input images of Figure 1.

Using Eigenfaces to Classify a Face Image

The eigenface images calculated from the eigenvectors of Z span a basis set with which to describe face images. Sirovich and Kirby (1987) evaluated a limited version of this framework on an ensemble of $M = 115$ images of Caucasian males, digitized in a controlled environment.

For each image, the mean image Ψ is subtracted from the image, and the difference is then projected into the face space by multiplying by the matrix A . The resulting vector is then compared to the vectors for the training set images, and the image is classified as the one with the closest match.

Significant eigenvectors of the Z matrix are chosen as those with the largest associated eigenvalues. In many of our test cases, based on $M = 16$ face images, $M' = 7$ eigenfaces were used.

A new face image (Γ) is transformed into its eigenface components (projected into "face space") by a simple operation,

$$\omega_k = \mathbf{u}_k^T (\Gamma - \Psi) \quad (7)$$

for $k = 1, \dots, M'$. This describes a set of point-by-point image multiplications and summations, operations performed at approximately frame rate on current image processing hardware. Figure 3 shows an image and its projection into the seven-dimensional face space.

The weights form a vector $\Omega^T = [\omega_1, \omega_2, \dots, \omega_{M'}]$ that describes the contribution of each eigenface in representing the input face image, treating the eigenfaces as a basis set for face images. The vector may then be used

in a standard pattern recognition algorithm to find which of a number of predefined face classes, if any, best describes the face. The simplest method for determining which face class provides the best description of an input face image is to find the face class k that minimizes the Euclidian distance

$$\epsilon_k^2 = \|(\Omega - \Omega_k)\|^2 \quad (8)$$

where Ω_k is a vector describing the k th face class. The face classes Ω_k are calculated by averaging the normalized images of the known individuals belonging to each face class. The mean face vector Ω_k for the k th face class is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

$$\Omega_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \Omega_{ki} \quad (9)$$

where N_k is the number of individuals belonging to the k th face class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

where N_k is the number of individuals belonging to the k th face class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class. The mean face vector Ω_k is calculated by averaging the normalized face images of all the individuals belonging to that class.

with the highest associated eigenvalues. (Let $M' = 10$ in this example.)

3. Combine the normalized training set of images according to Eq. (6) to produce the $(M' = 10)$ eigenfaces u_k .

4. For each known individual, calculate the class vector Ω_k by averaging the eigenface pattern vectors Ω [from Eq. (8)] calculated from the original (four) images of the individual. Choose a threshold θ_k that defines the maximum allowable distance from any face class and a threshold θ_k that defines the maximum allowable distance from any face class.

5. For each unknown face image, calculate the class vector Ω by averaging the eigenface pattern vectors Ω [from Eq. (8)] calculated from the original (four) images of the unknown face image. Calculate the distance ϵ_k from the class vector Ω_k to the unknown face vector Ω using Eq. (8). If the distance ϵ_k is less than the threshold θ_k , then the unknown face image belongs to the class k .

6. If the distance ϵ_k is greater than the threshold θ_k , then the unknown face image does not belong to any of the known face classes. The unknown face image is classified as "unknown".

7. The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown".

8. The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown".

The unknown face image is classified as "unknown".

The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown". The unknown face image is classified as "unknown".

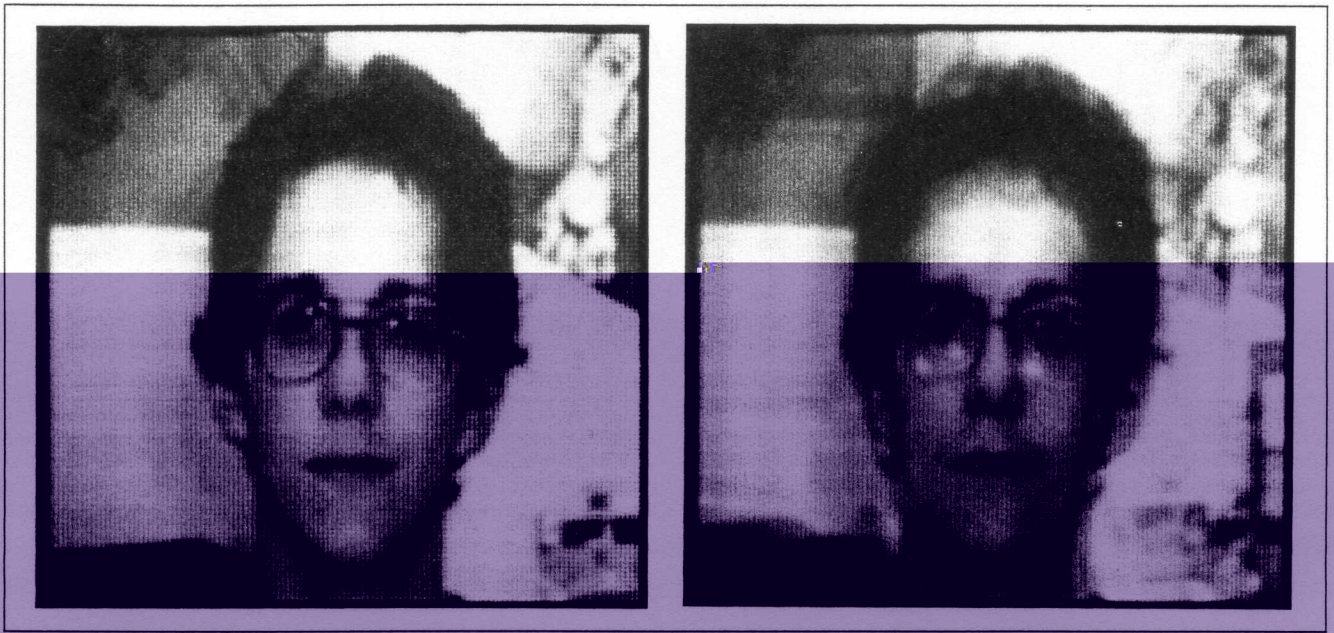


Figure 3. An original face image and its projection onto the face space defined by the eigenfaces of Figure 2.

Motion Detecting and Head Tracking

People are constantly moving. Even while sitting, we fidget and adjust our body position, nod our heads, look around, and such. In the case of a single person moving in a static environment, a simple motion detection and tracking algorithm, depicted in Figure 5, will locate and track the position of the head. Simple spatiotemporal filtering (e.g., frame differencing) accentuates image locations that change with time, so a moving person “lights up” in the filtered image. If the image “lights up” at all, motion is detected and the presence of a person is postulated.

After thresholding the filtered image to produce a binary motion image, we analyze the “motion blobs” over time to decide if the motion is caused by a person moving and to determine head position. A few simple rules are applied, such as “the head is the small upper blob above a larger blob (the body),” and “head motion must be reasonably slow and contiguous” (heads are not expected to jump around the image erratically). Figure 6 shows an image with the head located, along with the path of the head in the preceding sequence of frames.

The motion image also allows for an estimate of scale. The size of the blob that is assumed to be the moving head determines the size of the subimage to send to the recognition stage. This subimage is rescaled to fit the dimensions of the eigenfaces.

Using “Face Space” to Locate the Face

We can also use knowledge of the face space to locate faces in single images, either as an alternative to locating

faces from motion (e.g., if there is too little motion or many moving objects) or as a method of achieving more precision than is possible by use of motion tracking alone. This method allows us to recognize the presence of faces apart from the task of identifying them.

As seen in Figure 4, images of faces do not change radically when projected into the face space, while the projection of nonface images appears quite different. This basic idea is used to detect the presence of faces in a scene: at every location in the image, calculate the distance ϵ between the local subimage and face space. This distance from face space is used as a measure of “faceness,” so the result of calculating the distance from face space at every point in the image is a “face map” $\epsilon(x,y)$. Figure 7 shows an image and its face map—low values (the dark area) indicate the presence of a face.

Unfortunately, direct application of Eq. (9) is rather expensive. We have therefore developed a simpler, more efficient method of calculating the face map $\epsilon(x,y)$, which is described as follows.

To calculate the face map at every pixel of an image $I(x,y)$, we need to project the subimage centered at that pixel onto face space, then subtract the projection from the original. To project a subimage Γ onto face space, we must first subtract the mean image, resulting in $\Phi = \Gamma - \Psi$. With Φ_f being the projection of Φ onto face space, the distance measure at a given image location is then

$$\begin{aligned}\epsilon^2 &= \|\Phi - \Phi_f\|^2 \\ &= (\Phi - \Phi_f)^T (\Phi - \Phi_f) \\ &= \Phi^T \Phi - \Phi^T \Phi_f + \Phi_f^T (\Phi - \Phi_f) \\ &= \Phi^T \Phi - \Phi_f^T \Phi_f\end{aligned}\tag{10}$$

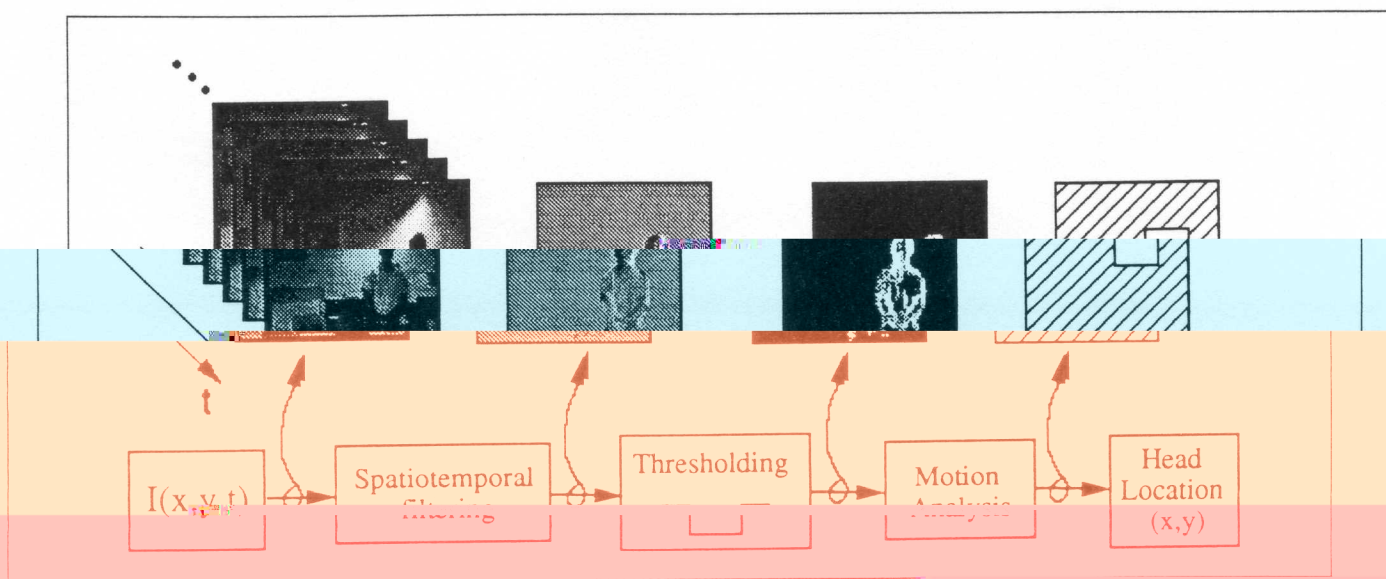


Figure 10. The head has been located. The output of the first three steps is the head location and the output of the fourth step is the head location and the head location.

Since the average value of the input image is $I(x,y,t)$, the average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$.

Using the assumption of the first step, the average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$.

The average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$.

Conclusion

The average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$. The average value of the input image is $I(x,y,t)$.

Acknowledgment

The authors would like to thank the authors for their contribution to the development of the system.

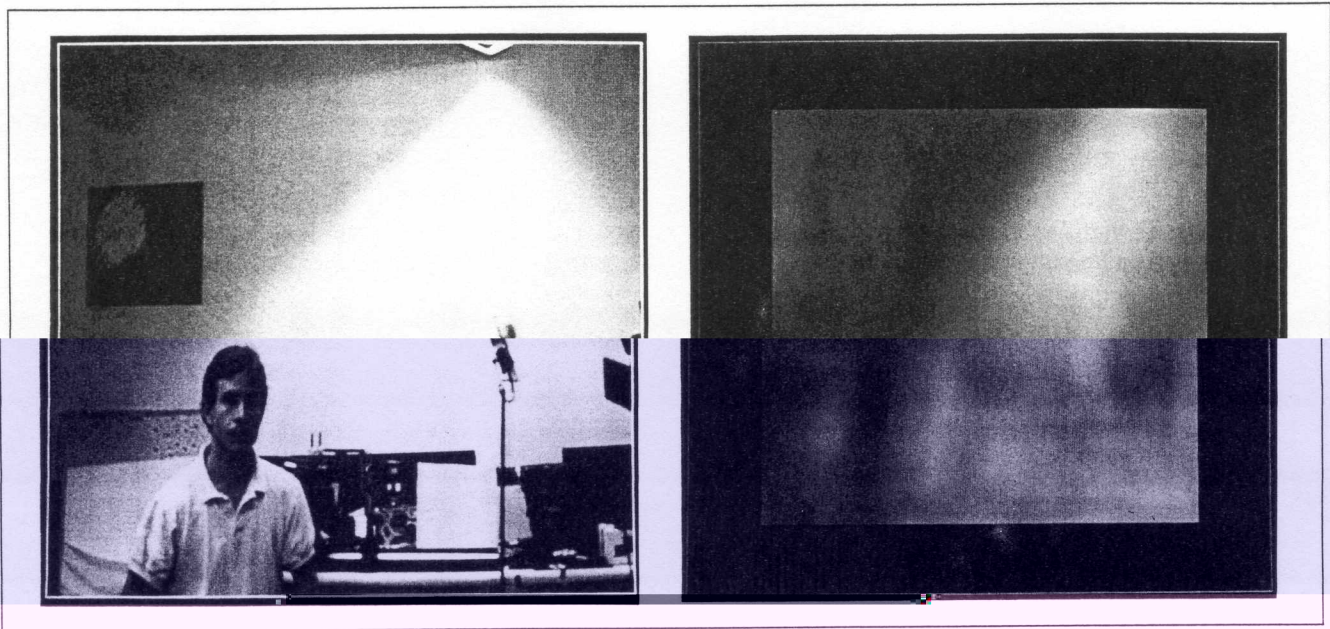


Figure 7. (a) Original image. (b) The corresponding face map, where low values (dark areas) indicate the presence of a face.

genface analysis as described above does not distinguish the face from the rest of the image. In the experiments described in the section on Experiments with Eigenfaces, the background was a significant part of the image used to classify the faces.

To deal with this problem without having to solve other difficult vision problems (such as robust segmentation of the head), we have multiplied the input face

scale the input image to multiple sizes and use the scale that results in the smallest distance measure to face space.

Although the eigenfaces approach is not extremely sensitive to head orientation (i.e., sideways tilt of the head), a non-upright view will cause some performance degradation. An accurate estimate of the head tilt will certainly benefit the recognition. Again, two simple methods have been considered and tested. The first is to

strategies of face recognition (Hay & Young, 1982) cite the importance of the internal facial features for recognition of familiar faces. Deemphasizing the outside of the face is also a practical consideration since changing hairstyles may otherwise negatively affect the recognition.

Scale (Head Size) and Orientation Invariance

The experiments in the section on Database 6: Face Images show that recognition performance decreases quickly as the head size in the image decreases. The head size in the image is a function of the distance from the camera to the face. The distance from the camera to the face is a function of the distance from the camera to the face. The distance from the camera to the face is a function of the distance from the camera to the face.

One of the most important factors in face recognition is the scale of the face in the image. The scale of the face in the image is a function of the distance from the camera to the face. The distance from the camera to the face is a function of the distance from the camera to the face.

metric patterns, at least for frontal views, we have used simple symmetry operators to estimate head orientation. Once the orientation is estimated, the image can be rotated to align the head with the eigenfaces.

The nearest-neighbor classification previously described assumes a Gaussian distribution in face space of an individual's feature vectors Ω . Since there is no a priori reason to assume any particular distribution, we want to characterize it rather than assume it is gaussian. Nonlinear networks such as described in Fleming and Cottrell (1992) are used to estimate the distribution of the feature vectors. The distribution of the feature vectors is estimated by comparing

Orientation Invariance

The first method for estimating head orientation is to use simple symmetry operators. The second method is to use a more complex method. The third method is to use a more complex method. The fourth method is to use a more complex method. The fifth method is to use a more complex method.

face view, side views, and right and left profile views. Under most viewing conditions these seem to be sufficient to recognize a face anywhere from frontal to profile view, because the real view can be approximated by interpolation among the fixed views.

EXPERIMENTS WITH EIGENFACES

To assess the viability of this approach to face recognition, we have performed experiments with stored face images and built a system to locate and recognize faces in a dynamic environment. We first created a large da-

under known variations of lighting, scale, and orientation. The results of these experiments and early experience with the near-real-time system are reported in this section.

Database of Face Images

The images from Figure 1a were taken from a database of over 2500 face images digitized under controlled conditions. Sixteen subjects were digitized at all combinations of three head orientations, three head sizes or scales, and three lighting conditions. A six level Gaussian pyramid was constructed for each image, resulting in image resolution from 512×512 pixels down to 16×16 pixels. Figure 8 shows the images from one pyramid level for one individual.

In the first experiment the effects of varying lighting, size, and head orientation were investigated using the complete database of 2500 images of the 16 individuals shown in Figure 1a. Various groups of 16 images were selected and used as the training set. Within each training set there was one image of each person, all taken under the same conditions of lighting, image size, and head orientation. All images in the database were then classified as being one of these sixteen individuals (i.e., the threshold θ_e was effectively infinite, so that no faces were rejected as unknown). Seven eigenfaces were used in the classification process.

Statistics were collected measuring the mean accuracy as a function of the difference between the training conditions and the test conditions. The independent variables were difference in illumination, imaged head size, head orientation, and combinations of illumination, size, and orientation.

Figure 9 shows results of these experiments for the case of infinite θ_e . The graphs of the figure show the number of correct classifications for varying conditions of lighting, size, and head orientation, averaged over the number of experiments. For this case where there was

lighting variation, 85% correct averaged over orientation variation, and 64% correct averaged over size variation.

As can be seen from these graphs, changing lighting conditions causes relatively few errors, while performance drops dramatically with size change. This is not surprising, since under lighting changes alone the neighborhood pixel correlation remains high, but under size changes the correlation from one image to another is largely lost. It is clear that there is a need for a multiscale approach, so that faces at a particular size are compared with one another. One method of accomplishing this is to make sure that each "face class" includes images of

followed, but the acceptance threshold θ_e was also varied. At low values of θ_e , only images that project very closely to the known face classes will be recognized, so that there will be few errors but many of the images will be rejected as unknown. At high values of θ_e most images will be classified, but there will be more errors. Adjusting θ_e to achieve 100% accurate recognition boosted the unknown rates to 19% while varying lighting, 39% for orientation, and 60% for size. Setting the unknown rate arbitrarily to 20% resulted in correct recognition rates of 100%, 94%, and 74% respectively.

These experiments show an increase of performance accuracy as the threshold decreases. This can be tuned to achieve effectively perfect recognition as the threshold tends to zero, but at the cost of many images being rejected as unknown. The tradeoff between rejection rate and recognition accuracy will be different for each of the various face recognition applications. However, what would be most desirable is to have a way of setting the threshold high, so that few known face images are rejected as unknown, while at the same time detecting the incorrect classifications. That is, we would like to increase the efficiency (the d-prime) of the recognition process.

One way of accomplishing this is to also examine the (normalized) Euclidian distance between an image and face space as a whole. Because the projection onto the eigenface vectors is a many-to-one mapping, there is a potentially unlimited number of images that can project onto the eigenfaces in the same manner, i.e., produce the same weights. Many of these will look nothing like a face, as shown in Figure 4c. This approach was described in the section on Locating Faces and is presented here as a method of identifying likely face subimages.

Real-Time Recognition

We have used the techniques described above to build a system that locates and recognizes faces in near-real-

proximately 96% correct classification averaged over

time. The system, which is implemented on a fixed camera, monitoring part of a room, is connected to a Datacube image



Figure 8. Variation of face images for one individual, three head sizes, three lighting conditions, and three head orientations.

processing system, which resides on the bus of a Sun 3/160. The Datacube digitizes the video image and performs spatiotemporal filtering, thresholding, and subsampling at frame rate (30 frames/sec). (The images are subsampled to speed up the motion analysis.)

The motion detection and analysis programs run on the Sun 3/160, first detecting a moving object and then tracking the motion and applying simple rules to determine if it is tracking a head. When a head is found, the subimage, centered on the head, is sent to another com-

puter, which is either rejected as not a face, recognized as one of a group of familiar faces, or determined to be an unknown face.

Recognition occurs in this system at rates of up to two or three times per second. Until motion is detected, or as long as the image is not perceived to be a face, there is no output. When a face is recognized, the image of the identified individual is displayed on the Sun monitor.

RELATIONSHIP TO BIOLOGY AND NEURAL NETWORKS

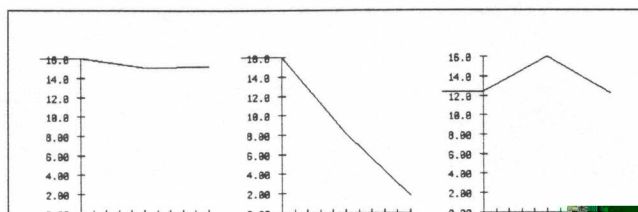


Figure 1. Relationship between the number of cells and the number of faces for different cell types.

cells were sensitive to identity, some to “faceness,” and some only to particular views (such as frontal or profile).

Although we do not claim that biological systems have “eigenface cells” or process faces in the same way as the eigenface approach, there are a number of qualitative similarities between our approach and current under-

Figure 2. Results of the eigenface experiment.



Figure 2. Results of the eigenface experiment. The top subplot shows the relationship between the number of cells (N) and the number of faces (F) for different cell types. The middle subplot shows the relationship between the number of cells (N) and the number of faces (F) for different cell types. The bottom subplot shows the relationship between the number of cells (N) and the number of faces (F) for different cell types.

standing of face processing in the visual system. For instance, the eigenface approach is based on the idea that faces are represented in the visual system as a set of features, and that the visual system is sensitive to the identity of these features. This is a key idea in the eigenface approach, and it is one of the reasons why it has been so successful in modeling face processing in the visual system.

Neural Networks

Another approach to modeling face processing in the visual system is the use of neural networks. Neural networks are a class of models that are designed to mimic the way in which the human brain processes information. They are composed of a large number of simple processing units, called neurons, which are interconnected in a complex way. The output of a neural network is determined by the weights of the connections between the neurons, and by the activation function of each neuron.

Neural networks have been used to model a wide range of cognitive tasks, including face processing. In the context of face processing, neural networks are typically used to model the way in which the visual system extracts features from faces and uses these features to make decisions about the identity of the faces. This is a key idea in the eigenface approach, and it is one of the reasons why it has been so successful in modeling face processing in the visual system.

There are many different types of neural networks, and each type has its own strengths and weaknesses. In the context of face processing, the most commonly used type of neural network is the convolutional neural network (CNN). CNNs are designed to be sensitive to the spatial relationships between features in an image, and they are able to learn to extract features that are invariant to translation and scaling. This makes them well-suited for modeling face processing, where the visual system is able to recognize faces regardless of their position or size in the image.

Despite their success in modeling face processing, neural networks are still a relatively simple model of the human brain. They do not take into account the complex, hierarchical nature of the visual system, nor do they take into account the role of attention and other cognitive factors in face processing. As a result, neural networks are still a long way from providing a complete understanding of face processing in the visual system.

There are many different types of neural networks, and each type has its own strengths and weaknesses. In the context of face processing, the most commonly used type of neural network is the convolutional neural network (CNN). CNNs are designed to be sensitive to the spatial relationships between features in an image, and they are able to learn to extract features that are invariant to translation and scaling. This makes them well-suited for modeling face processing, where the visual system is able to recognize faces regardless of their position or size in the image.

Despite their success in modeling face processing, neural networks are still a relatively simple model of the human brain. They do not take into account the complex, hierarchical nature of the visual system, nor do they take into account the role of attention and other cognitive factors in face processing. As a result, neural networks are still a long way from providing a complete understanding of face processing in the visual system.

There are many different types of neural networks, and each type has its own strengths and weaknesses. In the context of face processing, the most commonly used type of neural network is the convolutional neural network (CNN). CNNs are designed to be sensitive to the spatial relationships between features in an image, and they are able to learn to extract features that are invariant to translation and scaling. This makes them well-suited for modeling face processing, where the visual system is able to recognize faces regardless of their position or size in the image.

Figure 10. System diagram of the face recognition system.

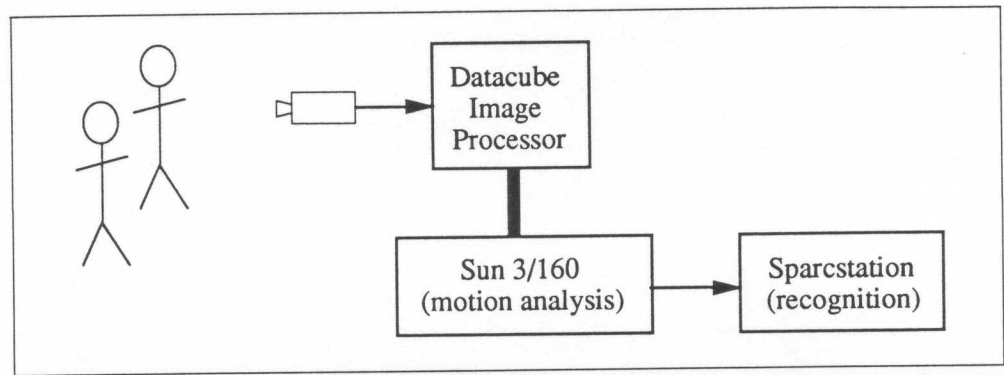
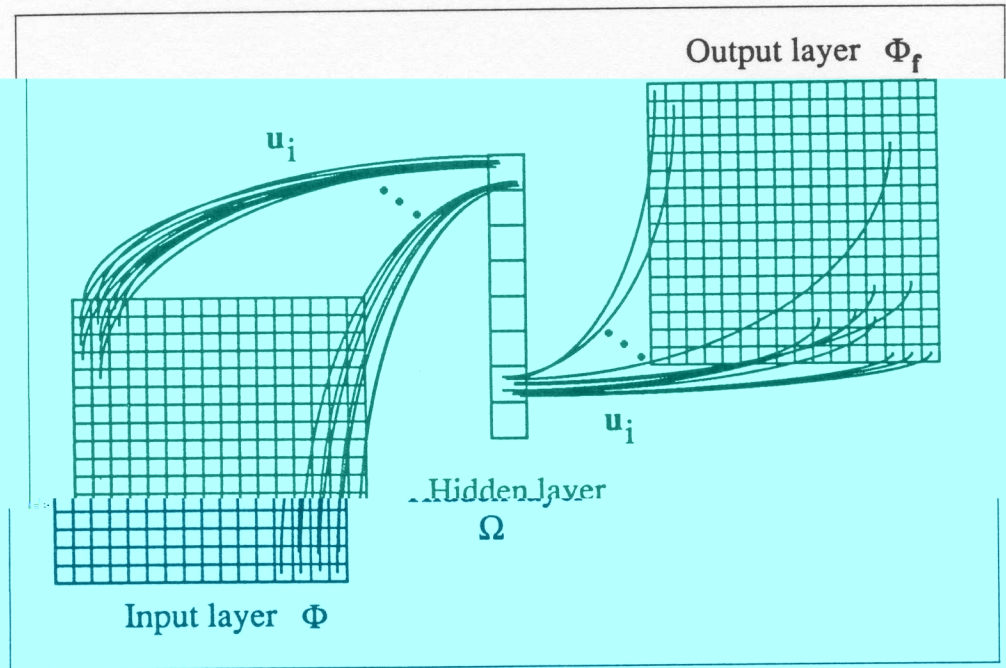


Figure 11. Three-layer linear network for eigenface calculation. The symmetric weights u_i are the eigenfaces, and the hidden units reveal the projection of the input image Φ onto the eigenfaces. The output Φ_f is the face space projection of the input image.



CONCLUSION

Early attempts at making computers recognize faces were limited by the use of impoverished face models and feature descriptions (e.g., locating features from an edge image and matching simple distances and ratios), assuming that a face is no more than the sum of its parts, the individual features. Recent attempts using parameterized feature models and multiscale matching look more promising, but still face severe problems before they are generally applicable. Current connectionist approaches tend to hide much of the pertinent information in the weights that makes it difficult to modify and evaluate parts of the approach.

The eigenface approach to face recognition was motivated by information theory, leading to the idea of basing face recognition on a small set of image features

eigenface approach does provide a practical solution that is well fitted to the problem of face recognition. It is fast, relatively simple, and has been shown to work well in a constrained environment. It can also be implemented using modules of connectionist or neural networks.

It is important to note that many applications of face recognition do not require perfect identification, although most require a low false-positive rate. In searching a large database of faces, for example, it may be preferable to find a small set of likely matches to present to the user. For applications such as security systems or human-computer interaction, the system will normally be able to "view" the subject for a few seconds or minutes, and thus will have a number of chances to recognize the person. Our experiments show that the eigenface technique can be made to perform at very high accuracy, although with a substantial "unknown" rejection rate, and

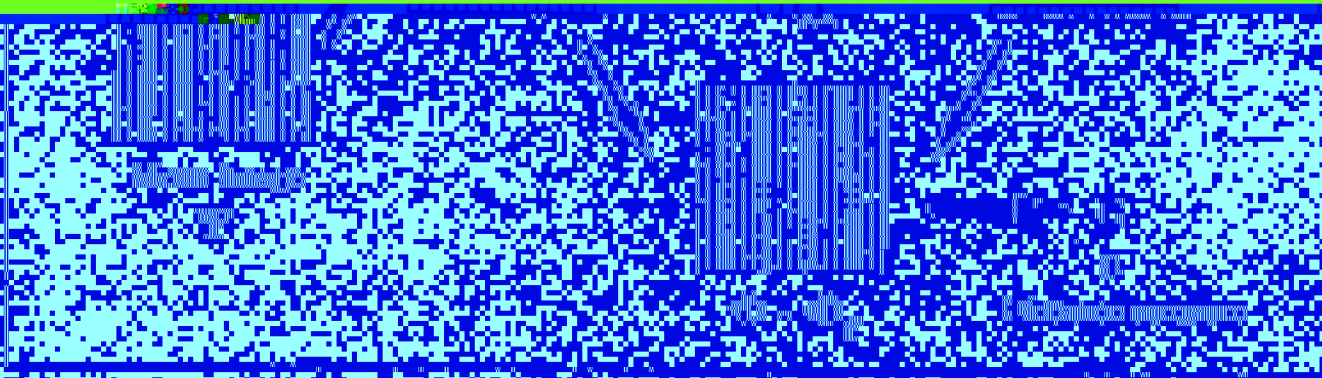
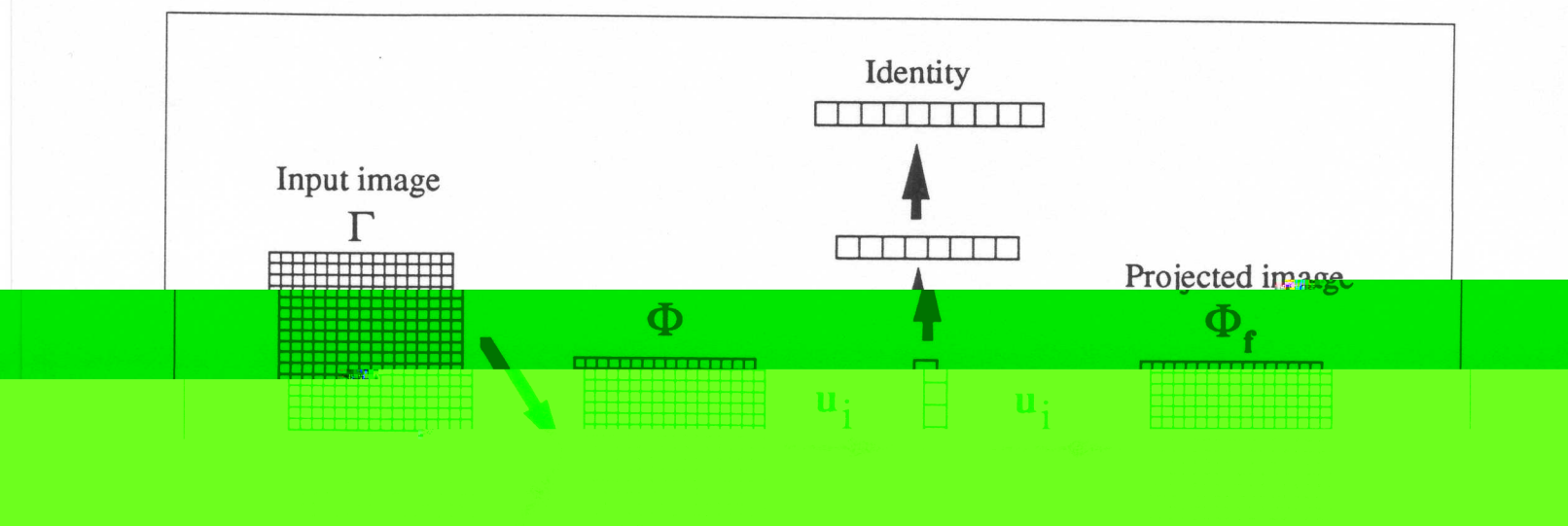


Figure 22: Visualization of feature maps and their corresponding identity outputs.

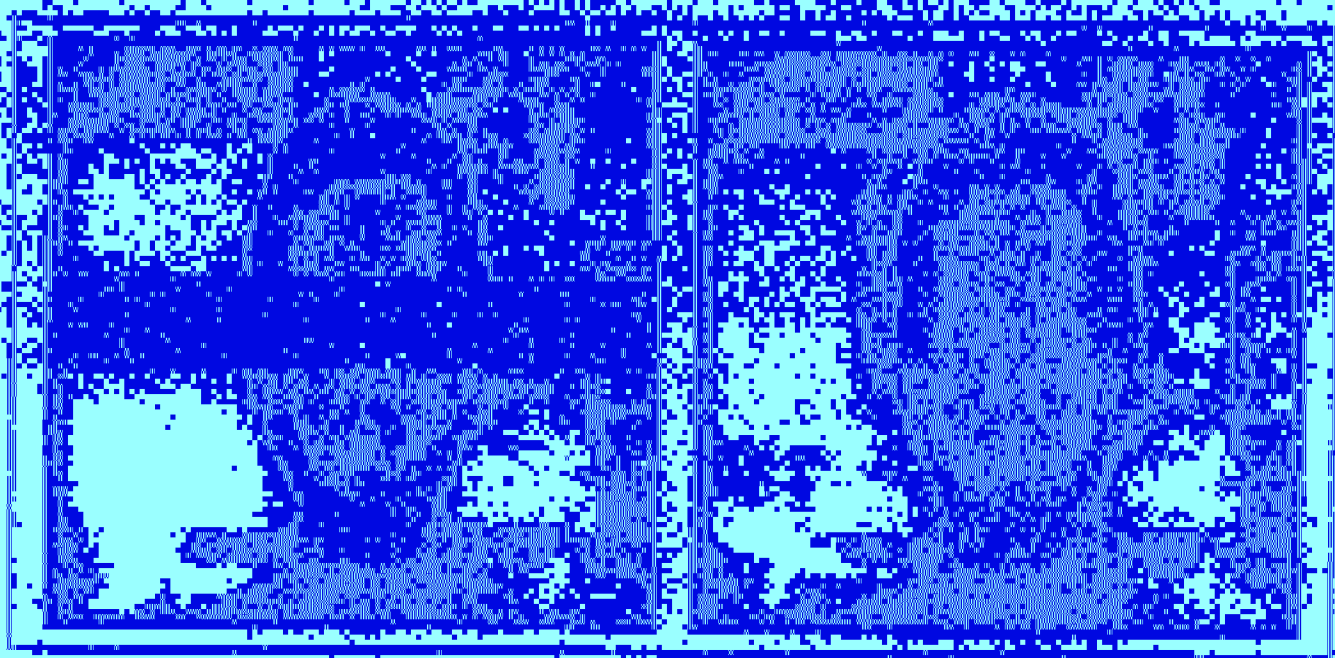


Figure 23: Left: Partially occluded face images and right: an experimental subject's identity.

rating a limited number of characteristic views for each individual, and the resulting frequency distribution of ratings. This approach focuses the investigation on the specific set of characteristics comprising the community. However, it is far more difficult to generalize from one set of community characteristics to another or to compare and contrast the results of the analysis with other studies. The use of a large number of characteristics, on the other hand, provides a more complete picture of the community and allows for more generalization.

1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 26

[illegible]

tion in biological and technical systems. Berlin: Springer-

1. **Author:** [Name of the author]
 2. **Title:** [Title of the document]
 3. **Date:** [Date of the document]
 4. **Page:** [Page number]
 5. **Section:** [Section number]
 6. **Subject:** [Subject of the document]
 7. **Keywords:** [Keywords of the document]
 8. **Abstract:** [Abstract of the document]
 9. **Introduction:** [Introduction of the document]
 10. **Conclusion:** [Conclusion of the document]
 11. **References:** [References of the document]

Figure 12. a) Schematic of the proposed 2D hexagonal lattice structure. b) The proposed 2D hexagonal lattice structure with the unit cell. c) The proposed 2D hexagonal lattice structure with the unit cell. d) The proposed 2D hexagonal lattice structure with the unit cell.

1. The authors are grateful to the Ministry of Higher Education and Science of the Republic of Armenia for the financial support of this work.

© 2004 Blackwell Publishing Ltd, *Journal of Internal Medicine* 255: 105–112

© 2004 Blackwell Publishing Ltd, *Journal of Internal Medicine* 255: 115–122

¹Wright, D. L., Jaramillo, A. J., and Collins, J. (2006). "Genetic and morphological relationships in the subgenus *Myrica* (Myricaceae)." *Journal of Biogeography* 33: 1001–1014.

© 2000 Blackwell Science Ltd, *Journal of Internal Medicine* 247: 101–108

Wang, H., Li, Y., Wang, H., et al., 1999. *Journal of Cell Biochemistry*, 72, 193-200.

faces. *Proceedings IEEE*, 59, 748.

Harmon, L. D. (1971). Some aspects of recognition of human faces. In O. J. Grusser & R. Klinke (Eds.), *Pattern recogni-*

Turner, R. L., Cohen, D. S., & Hallinan, P. W. (1989). Feature extraction from faces using deformable templates. *Proceedings of CVPR*, San Diego, CA, June.