Chapter 7

Object Recognition-1: Template Matching

template matching, decisions are non-linear, spaces of functions, projection as an operator

7.1 Introduction

Visually-mediated behavours are driven by signals from the environment—this was the observation underlying the perception/action cycle from the earliest lectures. In the previous lecture we studied how images were processed by a network characterized by its impulse response function. This corresponded to putting an impulse of light on a single ommatidium and observing the output. Now we start to address the use of this information, by making a decision based upon it. This embodies the ability to act differentially upon the information.

Decisions are made when, for example, we *recognize* something in the environment and reorient to move toward it. This word – re-cognize – has two parts. The second 'cognition' has its root in the Latin: 'get to know'; the first is (by my translation!) to 'again.' The 'get to know' part is to ask whether the network is actually storing a pattern of interest; if so, then by identifying where (in the environment) this pattern is we might find something good. It is, in effect, the simplest possible form of a match between the "neural image" of a structure in the environment and a "stored image" of something sought.

In the next section we show how these ideas are embodies, firstly, in the lateral plexus of *Limulus'* and, secondly, in a threshold (decision) to locate a position in the environment. This provides a basis for Limulus to orient toward a possible female, thus providing a basis for mate discovery. The result is related to our description of the acuity experiment in the last lecture, when a "dip" between peaks needed to be detectable to separate the image of a single point of light from two nearby points. Such "just detectable differences" are a cornerstone of visual psychophysics. But for

animals to survive in the world, false positives must be sufficently infrequent to not waste much energy (or time) and false negatives sufficently rare that targets are likely to be found.

7.2 Limulus Finds a Mate

We are now ready to turn template matching into a function that really matters for *Limulus* by asking what does an image (template) of a possible mate look like under the proper viewing and lighting conditions. For *Limulus* remember this is in shallow water along the coastline under a full moon. Hence as a possible mate is approached she will first appear like a dark spot (the carapice) against a noisy, but lighter background (the sand).

A cartoon of limulus approaching mate is in Fig. 7.1. From the use of cement castings it can be confirmed that visual signals do trigger the decision by males to orient their trajectory toward females. Working out the visual angle spanned by each ommatidium, it can be confirmed that a significant response is required.

A sketch of the response is indicated in Fig. 7.1(bottom). Reverse contrast is shown here, so that the female appears as a "bright spot" against the noisy, dark sand. When she first comes into view, a single pixel is bright, but this is only 1% of the visual field on an ommatidium; at a distance of about a meter, the scale is just right (Fig. 7.1(bottom, middle)), which triggers the trajectory change. Think of this trigger process as a threshold on the output:

Orient toward position x if Output(x) exceeds threshold.

Equivalently we could say that Limulus classifies the possible outputs as 'interesting', or exceeding threshold, or 'uninteresting'; that is below threshold. (Recall also the spiking property of integrate-and-fire neurons when the voltage threshold is exceeded.)

A more accurate simulation of this process is shown in Fig. 7.2.

For *Limulus* there was a characterization of the visual signature of mates—dark spots of a particular size against a light background—and this structure was embedded directly in the function of its visual system. But this was quite local. For an example of a global match we turn to the cuttlefish. This sets the stage in this lecture for more general matching of patterns and how decisions might be made from them.

7.3 Template Matching = Image Processing + Decision

Of all the bumps and dips in intensity on the beach, Limulus only orients toward large ones. This involves a decision, or a choice between alternatives. The test for acuity

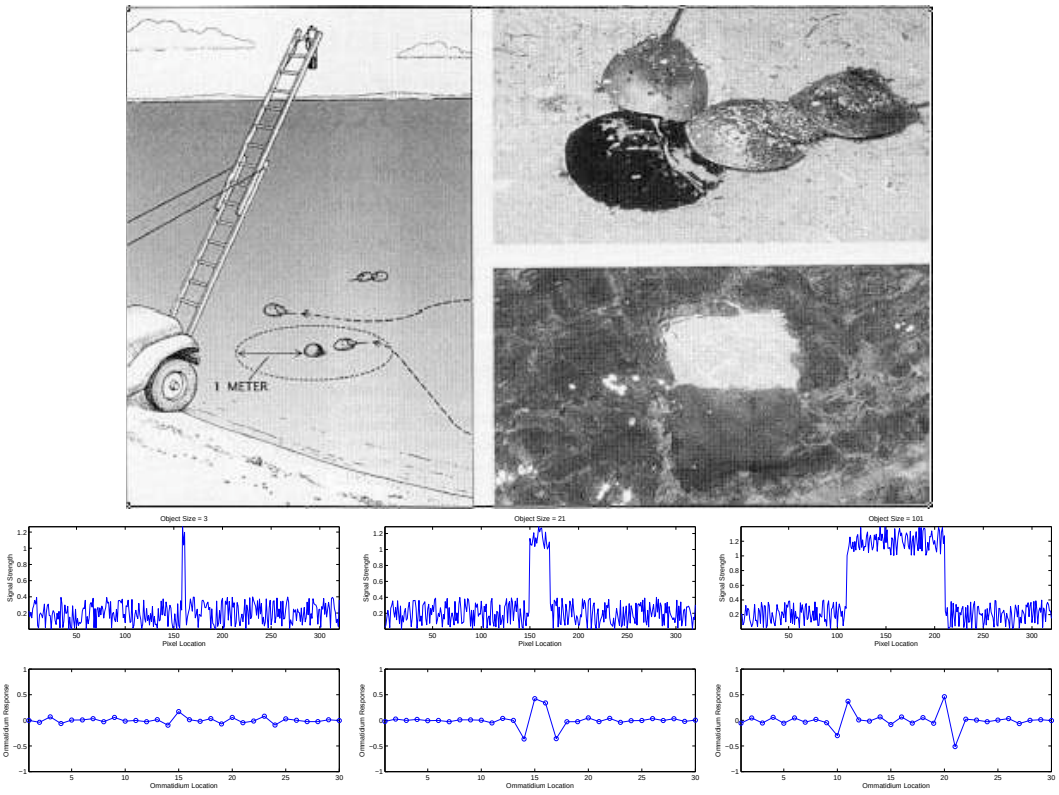

Figure 7.1: (Top) The setup used by Robert Barlow in studying mating behavour of Limulus. (left) Imaging the horseshoe crabs in shallow water. (right) Males approach a cement casting with the proper contrast (top), confirming that vision plays a role in mate selection. When the casting exhibits incorrect contrast, males do not remain interested. (Bottom)Cartoon simulation of Limulus finding a mate. The top plot is the intensity and the lower plot in each panel is the output of the lateral inhibitory network. Notice how when the female is far from the male, she casts no signature at the output of the lateral inhibitory network. For a particular distance she casts just enough to cause a reliable signature in the output; psychophysical evidence with the limulus indicates that it is at this point that he reorients toward the female. Taking into account the ommatidium sizes, this works out to be about a meter for adults. Scale: each ommatidium spans 10 pixels.

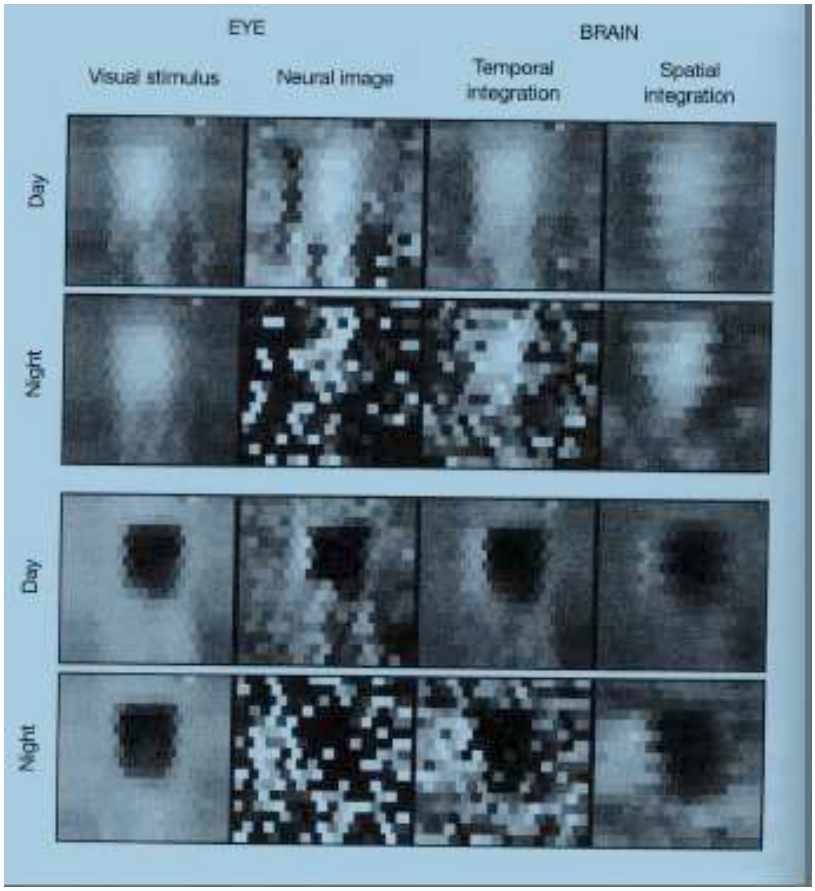

Figure 7.2: (Top) Barlow simulation of the neural image of a mate in Limulus. A video camera was mounted on a crab's carapice to obtain images in "natural circumstances." The left column shows a simulation of actual images under the eye's optics for low (upper) and high (lower) contrast objects on day and night viewing. The neural image shows ensembles of optic nerve activity, with firing rate mapped to gray level. Notice how noisy the images are. Temporal smoothing (low pass filtering in time) and spatial integration help to remove the noise. Scale: each image represents a simulation of 16 x 16 ommatidia.

involved a decision: the subject had to state whether one or two dots were present. Such decisions are inherently non-linear; they imply a condition which is either met (and an affirmative response given) or not (and a negative response given).

In effect, this test turned on the location of a local minimum in intensity surrounded by two local maxima—in other words, a template for a particular type of structure. Templates can be generalized beyond this elementary example, and provide the first way in which an algorithm can be designed to actually recognize structure.

The key idea is to think of the template as a small image of exactly the pattern that is sought. For example, it might be a picture of a chair under certain viewing and lighting conditions, or–more generally–a picture of an object of interest under given (and fixed) lighting and viewing conditions.

> KEY IDEA: THINK OF TEMPLATES AS PERFECT OR IDEAL IMAGES OF AN OBJECT. IF THE DIFFERENCE BETWEEN THE TEMPLATE AND AN IMAGE IS SMALL ENOUGH – THE SIMILARITY IS HIGH ENOUGH – THEN THE TEMPLATE MATCHES THE IMAGE.

Since the image typically indicates a picture of the scene domain (the 3-D world) and the template is an image of a single object possibly in this scene, the template image is often much smaller than the full image. *We seek to find that location in the image that matches the template; this is the location, in image coordinates, of the object.*

7.3.1 Templates as Stored Image Structure

The visual structure of interest to Limulus was a dark spot in a bright surround; the impulse response of the lateral inhibitory network responded nicely to this. We can think of this pattern as a little "image, " which lets us think about more general structures represented this way. Finding "waldo" is just such an example. We shall refer to this general image template as $T(i, j)$, the brightness distribution at pixel (i, j).

Assume given a template $T(i, j)$ of the object that we seek to match against an image $I(i, j)$. (Notice that we have switched to discrete (i, j) coordinates to indicate pixel locations.) Since the template is typically much smaller than the image, suppose we center it at some location (m, n) completely within the image and compare the two. The template will then overlap the image (well, a portion of the image). If it's a perfect representation, then for each overlapping pixel location we would have perfect agreement between the intensity of the template and the intensity of time image. This implies that their difference in intensity is an interesting measure of agreement, and this agreement at each pixel can be integrated (summed) over the entire template. The resulting difference between the template and the image:

$$\text{Diff}(m,n) = \underbrace{\sum_{\text{overlapping pixels}} \overbrace{\{I(i,j) - T(i-m, j-n)\}}^{\text{local}}}_{\text{global}} \tag{7.1}$$

should be small when there is a match. Important: This involves both local comparisons (how different are the two "overlapping" pixel values) and a global measure (the average, or mean, or maximum over the template of this difference).

Neither of these two key operations: the first one that takes some measure over the full template and the second that specifies what we mean by the "difference" between pixel values, is uniquely dictated by the philosophy of template matching. For example, we might want to stress the total difference in pixel values; the "least mean square" difference, or the max difference. The amounts to setting up a measure of how well they agree; or a *metric* between images (the template image and the observed image).

More generally, a metric is based on the concept of *norm*. The general form is:

$$\text{Match}(m,n) = ||I(i,j) - T(i,j)||_{L_\alpha} \tag{7.2}$$

where the $||\cdot||$ is a notation for distance between functions (in this case, images), and L_α denotes the norm. The distance between images is evaluated over all pixels (i,j) that are in the overlap; i.e., that are in the domain over which the template is defined, Dom(T). Three values of α are particularly interesting:

· Max norm (L_1)

$$\text{Match}_{L_1}(m,n) = \sum_{i,j s.t. (i-m,j-n) \in \text{Dom}(T)} |I(i,j) - T(i-m, j-n)|. \tag{7.3}$$

This norm treats all pixel intensities the same by simply adding up the absolute value of the difference. (Note: absolute value is necessary so that opposite differences don't cancel.) It is interesting to consider what this norm counts when the image and the template are binary (i.e., pixels can take on only the values of 0 or 1).

· Euclidean norm (L_2)

$$\text{Match}_{L_2}(m,n) = \left[\sum_{i,j s.t. (i-m,j-n) \in \text{Dom}(T)} (I(i,j) - T(i-m, j-n))^2 \right]^{1/2}. \tag{7.4}$$

This norm emphasises large pixel differences more than small ones, by squaring.

· ∞-norm (L_∞)

$$\text{Match}_{L^infty}(m,n) = \max_{i,js.t.(i-m,j-n)\in \text{Dom}(T)} |I(i,j) - T(i-m,j-n)|. \quad (7.5)$$

This norm only cares about the largest pixel difference.

These different formulas all define a different distance between the template and the image; but each is a distance in the mathematical sense.

When we know that there is a match, and only have to find it, it makes sense to select that point with the minimum distance (over the entire image); this then 'makes the decision' as to where the template matches. (Note: this differs from the threshold that we had before, because it is only searching for the smallest distance; not necessarily one that exceeds a similarity threshold.) That is, we seek those points where the "distance" between the template and the image is as small as possible.

This selection process is highly non-linear. It gives rise to a *characteristic function*, in this case a matrix of 1's and 0's, such that each entry is 1 at the minimal (matching) point(s) and 0 elsewhere. As we will shortly see, this measure of similarity can be thought of as a distance.

$$\text{Output}(i,j) = \begin{cases} 1 & \text{distance}(\text{Image}(i,j), \text{Template}) \leq \text{threshold} \\ 0 & \text{otherwise} \end{cases}$$

Of course, we need to Template to be positioned over the image coordinates (i,j).

7.3.2 Normalized Correlation

Considering the Euclidean norm, we notice an interesting effect. Squaring both sides we obtain:

$$\text{Match}^2_{L^2}(m,n)$$
$$= \sum_{i,js.t.(i-m,j-n)\in \text{Dom}(T)} \left(\underbrace{I^2(i,j)}_{\text{image "energy"}} - 2I(i,j)T(i-m,j-n) + \underbrace{T^2(i-m,j-n)}_{\text{template "energy"}} \right).$$

The last term, the "template energy", does not vary as the template is shifted over the image, so it cannot really participate in the matching process and we can ignore it for now. (Higher energy templates have more variation, so might be more discriminative.)

The image energy in the window–the first term–varies, in general, with where the template is located (the coordinates (m,n)). Suppose for a minute that this is small; then the second term will also be small (because of the product). However, it is in this second term—the *cross correlation* between the image and the template—that the matching is really taking place. Therefore we define:

$$\text{CrossCorrelation} = \sum_{i,j s.t.(i-m,j-n)\in \text{Dom}(T)} I(i,j)T(i-m,j-n) \tag{7.6}$$

and the *normalized cross correlation* as

$$\text{NormalizedCrossCorrelation} = \frac{\text{CrossCorrelation}}{\left[\sum_{i,j s.t.(i-m,j-n)\in \text{Dom}(T)} I(i,j)^2\right]^{1/2}} \tag{7.7}$$

Matching using normalized correlation implies selecting those points where the correlation is maximal. Alternatively, we could select all those points in the image where the match exceeded some predefined threshold and then perhaps examine them further. See Fig. 7.3.

Since correlation is very close to convolution (do you see the differences; when will they matter?) perhaps we should seek out a significant event for *Limulus* to discover what he "sees."

7.4 The Geometry of Correlation

At this point one might be struck by two very different types of observations. First, the formula for correlation that we just derived involves a product of an image function against a template function under a summation (the sum is over all discrete image points). This is highly reminiscent of the formula for the coefficients in the Fourier series developed in the previous lecture. But, second, correlation against a template provides a single number, which we interpreted as how well the template matched the signal. If the match is high, then the signal and the template are very similiar, in the following operational sense. Suppose we paste the template onto a wall, as if it were wallpaper, and then we project, using a slide projector, the image right onto it. If the correlation is high, it will be difficult to find the differences between them: bright places will project to bright places, and dark to dark. If the correlation between them is low, however, the projection will be a mess with almost nothing overlapping properly. Dark will often be illuminated by bright, and bright spots on the wall will have only occasional illumination. The result will tend toward average gray.

Can we take this word—projection—more seriously, and if so, how can it relate to the Fourier series? Let's briefly consider each in turn, but in the opposite order.

7.4.1 Vectors and Vector Algebra

Let's return to the simple example of vectors from Lecture 2, where we expanded a vector in the plane as a sum of components:

$$\vec{X} = x_i\vec{\imath} + x_j\vec{\jmath}$$

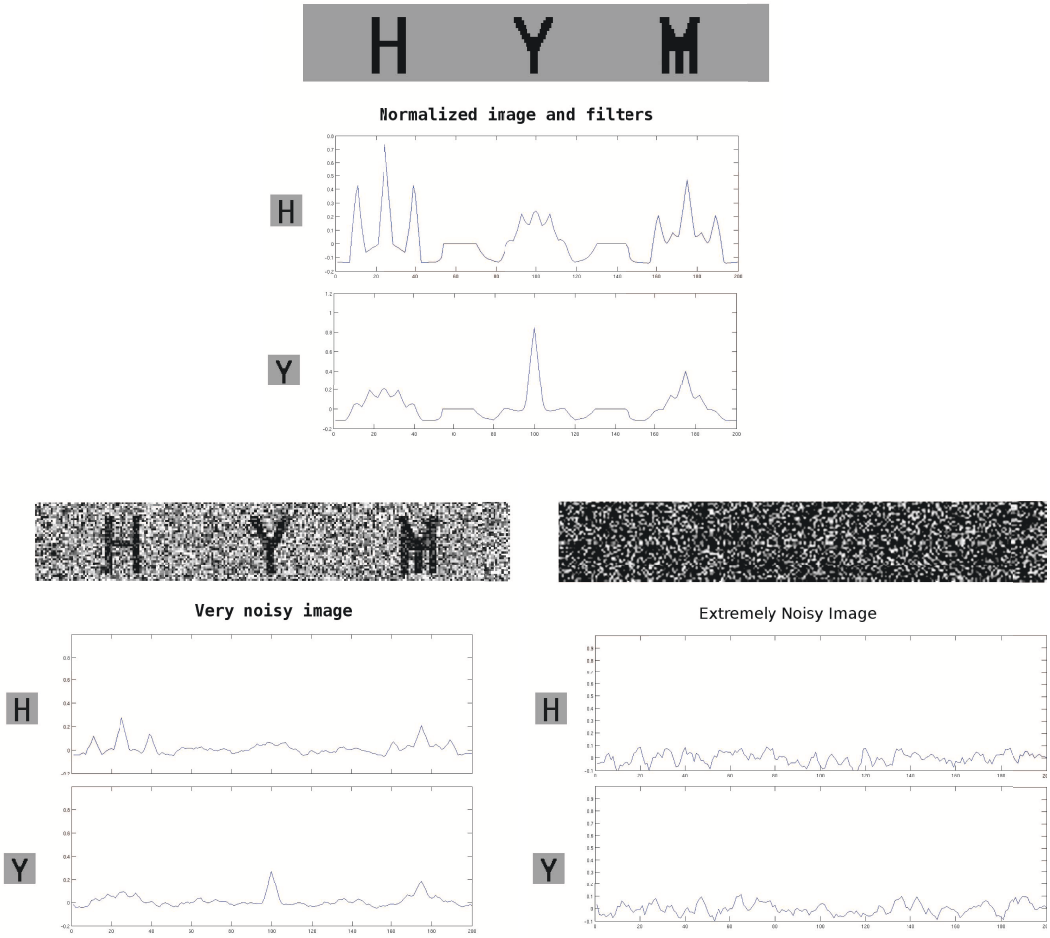

Figure 7.3: Illustration of the template correlation model for structure finding. Two normalized templates, for the block letters "H" and "Y" were created and then correlated with an image of each of them separately and then both superimposed. (top) no noise; notice how the template correlates with part of the patterns as well as with the correct one. (bottom) Increasing noise makes it increasingly difficult to find a "threshold" on the correlation function to declare a match.

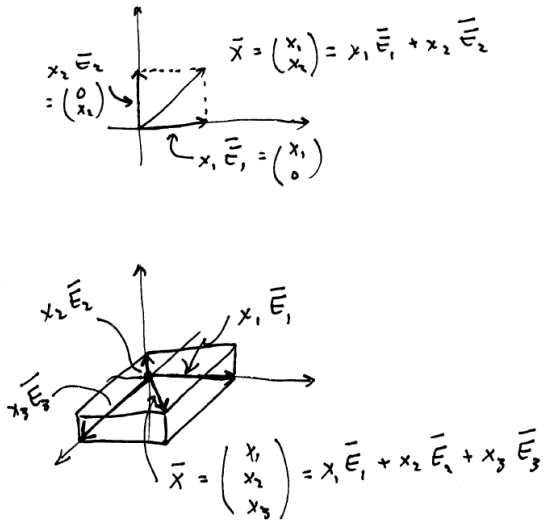

Figure 7.4: Vectors are described by their components in an ortho-normal coordinate basis. (a) (b)

where $\hat{i}$ is a unit vector in the x-direction and $\hat{j}$ is a unit vector in the y-direction. The a's are the magnitude of the components in each of these directions. A vector in 3-space would just require one more unit vector, and we could continue in this fashion as long as necessary; Fig. 7.4.

Note the general form: (i) a magnitude times a unit vector. The unit vectors are (ii) mutually orthogonal and (iii) span the plane (or space). Finally, (iv) the projection of any vector onto the axes defined by the unit vectors defines the coefficents and (v) they combine additively. Each of these points is an important one, and we'll first develop a little of the standard machinery for manipulating them. This is called LINEAR ALGEBRA.

To be consistent about developing a notation for vectors, we write them in the column form:

$$\vec{X} = \begin{pmatrix} x_i \\ x_j \end{pmatrix} \tag{7.8}$$

and realize that the above (eq. 7.4.1) represents an addition of two vectors. Such an addition is done component-wise:

$$\vec{X} + \vec{Y} = \begin{pmatrix} x_i + y_i \\ x_j + y_j \end{pmatrix}. \tag{7.9}$$

From this we can easily deduce that for every vector $\vec{X}$ there is another one $\vec{Y}$ such that $\vec{X} + \vec{Y} = 0$, or $\vec{Y} = -\vec{X}$. This negative vector is an additive inverse.

From this coordinate view we can write out the unit vectors above,

$$\vec{\imath} = \vec{E}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; \qquad \vec{\jmath} = \vec{E}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{7.10}$$

where we have introduced the more general notation $\vec{E}_i$. In general these are the NATURAL COORDINATES.

Scalar multiplication is simply multiplying the coordinates by this number; for example multiplying the first by any real-valued scalar $\alpha \in \Re$ gives

$$\alpha \vec{E}_1 = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

In this sense these COORDINATE AXES or BASIS VECTORS are special because, for any vector,

$$\vec{X} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = x_1 \vec{E}_1 + x_2 \vec{E}_2. \tag{7.11}$$

Two vectors that point in the same direction are said to be LINEARLY DEPENDENT, two vectors that do not are LINEARLY INDEPENDENT. To span the plane, in 2 dimensions, we clearly need two coordinate axes that are not identical; in fact, it is most convenient if they are linearly independent. Any vector can then be expressed in terms of (that is, as a weighted sum) of them. Of course, if they are not orthogonal (linearly independent) but are not identical, they can still be used as a basis but not a cleanly. The "overlap" or the difference from independence, just muddies the matrices.

A length (magnitude) can be assigned to a vector with the Pythagorean theorem: the distance between the point $\vec{X} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ and the origin $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ is $\sqrt{x_1^2 + x_2^2}$; we denote this $|\vec{X}|$. Note that length is never negative (we take the positive branch of the square root). And it doesn't matter whether we take the length from the origin to the tip of the vector, or the other way around; they're the same. This length formula is also called EUCLIDEAN.

Again for simplicity, let the basis vectors have unit length $|E_i| = 1$. All the unit length vectors from the origin trace out a unit circle parameterized by the POLAR ANGLE θ; see Fig. 7.5. An arbitrary vector $\vec{X}$ points in the direction of the unit vector $\frac{\vec{X}}{|\vec{X}|} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$.

To determine the magnitude of an arbitrary vector on one of the coordinate axes, we use the DOT PRODUCT or INNER PROJECT of the vectors: in symbols

$$\vec{X} \cdot \vec{Y} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = x_1 y_1 + x_2 y_2. \tag{7.12}$$

The dot product for vectors has many properties of ordinary multiplication between numbers, but an important difference. In ordinary multiplication, if the product is

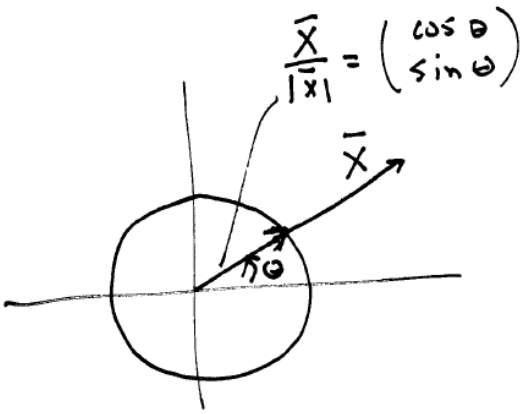

Figure 7.5: Unit vectors around the origin trace out a circle.

0 then at least one of the numbers had to be 0. For the dot product this is not the case; for two coordinate vectors we have:

$$\vec{E}_1 \cdot \vec{E}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 1 \cdot 0 + 0 \cdot 1 = 0. \tag{7.13}$$

When the dot product between vectors is 0, and at least one is not 0, the two vectors must be perpendicular to one another.

We can collect a lot of this discussion together by noting that an arbitrary vector $\vec{X}$ and be expressed in the coordinate basis $(\vec{E}_1, \vec{E}_2)$ by the formula:

$$\vec{X} = \underbrace{(\vec{X} \cdot \vec{E}_1)}_{\text{magnitude}} \overbrace{\vec{E}_1}^{\text{unitvector}} + \underbrace{(\vec{X} \cdot \vec{E}_2)}_{magn} \overbrace{\vec{E}_2}^{\text{vector}} \tag{7.14}$$

where we have stressed that the magnitude component in each direction is given by the PROJECTION of the vector onto the relevant basis vector. And the sum makes sense because we are adding vectors properly.

In general, for a vector in N-dimensions, $\vec{X} \in \Re^n$, we have

$$\vec{X} = \sum_{i=1}^{N} (\vec{X} \cdot \vec{E}_i)\vec{E}_i = \sum_{i=1}^{N} \alpha_i \vec{E}_i \tag{7.15}$$

Note how the definitions of length and orthogonality extend directly to N dimensions; e.g.,

- $|\vec{X}| = \sqrt{x_1^2 + x_2^2 + ... + x_N^2} = (x_1^2 + x_2^2 + ... + x_N^2)^{\frac{1}{2}}$
- $\vec{E}_i \cdot \vec{E}_j = 0, i \neq j$
- $\vec{E}_i \cdot \vec{E}_i = 1, \forall i.$

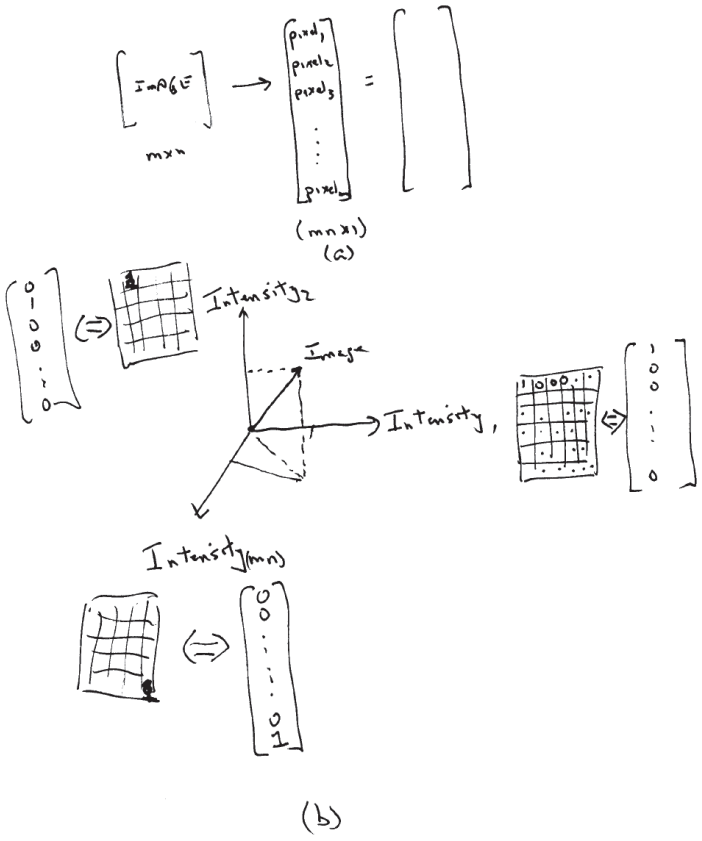

Figure 7.6: The standard basis in the space of images. (a) Think of an image as an $m \times n$ matrix of intensities. Stacking each row on top of the next yields an $mn \times 1$ vector. (b) Plot these vectors as points in an mn-dimensional space, where each axis indicates an individual pixel. Shown are the first, second, and mn^{th} directions.

7.4.2 Building Intuition about Image (Vector) Spaces

Now, let's review the model of an image as a point in a space of images (recall Figs. 7.10 and 7.11). If we had an image and "vectorized" it, this simply transforms a matrix of size $(m \times n)$ into a vector of size mn by rotating each row of the image into a column vector and then placing these columns on top of each other; see Fig. 10.4a.

This mn-dimensional vector can then be plotted in an mn-dimensional vector space. A (real) VECTOR SPACE is a set of vectors on which two operations are defined: vector addition and multiplication by a (real) number. These operations imply that if we add two vectors in the space we obtain another vector that is also in the space. Or if we multiply a vector by a number we get another vector (with different length) that is also in the space. (If we think of 0 as a number, what does this imply? The 0-vector is in the space.)

Because of the equivalence that we set up between images (matrices) and vectors, it follows that we could also have thought of the vector space of $(m \times n)$ matrices. (This helps a little in thinking about the role of abstraction in mathematics.)

Now, let's be a little more careful about thinking of the vector space in which we're representing images. To do calculations in this space, we have to determine in which directions the coordinates are pointing. Here we set it up so that one could think of each axis as a unit vector "indicating" a single pixel. There are just enough of them to represent each of the pixels. Each pixel's intensity is then the projection of the image onto this. Since the pixels are in different positions these vectors are "independent." It's natural to think of these as column vectors with a 1 indicating the position of the pixel i and a 0 elsewhere. Symbolically,

$$\vec{e_i} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \tag{7.16}$$

A BASIS is a set of vectors that are INDEPENDENT and that SPAN the space. Independence means that there is no linear relationship among them and span means that any vector in the space can be expressed in terms of them. In a 3-dimensional vector space, for example, three basis vectors are required and they cannot lie in the same plane. Any other vector is a linear combination of these three basis vectors.

Our image (Fig. 10.4b) can thus be expressed:

$$I = \sum_{i=1}^{mn} p_i \vec{e_i}. \tag{7.17}$$

Is this the only basis for this space? Is this the "best" basis? We'll be working in this lecture to find another one and a suitable notion of "best."

7.4.3 Correlation as Inner Product

Returning to our discussion of image correlation, we can now think of it in geometric terms by, first, thinking of images as vectors and then applying the above machinery.

Remembering that, for an $(n \times n)$ image there are n^2 pixels, we can string these out into a vector of length n^2 with the value of each component simply being the intensity at that pixel.

In symbols, the lexicographic map of a matrix into a vector strings out the rows:

$$\begin{pmatrix} \text{row } 1 \\ \text{row } 2 \\ \vdots \\ \text{row } M \end{pmatrix} \Rightarrow \begin{bmatrix} \text{r} \\ \text{o} \\ \text{w} \\ 1 \\ \vdots \\ \text{r} \\ \text{o} \\ \text{w} \\ \text{M} \end{bmatrix} \tag{7.18}$$

For now, let us think about the correlation between images of the same size. These image vectors can be plotted in a space with the natural coordinate vectors

$$\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

with a "1" in the i^{th} position denoting pixel i in the image. Since correlation was the pointwise product of the image intensities at the same location, with the "vector view" of images this is exactly the inner product: *correlation amounts to projecting one image onto the other.*. If the images are very similar, their locations in n^2-space will be close, and their projection close to 1; if they are far apart their projections will be closer to 0. Uncorrelated images can be thought of as orthogonal to one another. See Fig. 7.7

7.5 Recognition 101: Clustering in Image Space

We're now ready to introduce the first approach to OBJECT RECOGNITION. Suppose we were given a set of images for each class or group of objects of interest. For example, think about faces.[1] Suppose each of these is properly normalized (in size, pose, ...) as much as possible, but there will still be some variation. So, suppose further that we have lots of (normalized) images of each person. More generally, we might think of categories of objects (people, airplanes, toasters, bicycles, ...) Now,

[1]Face recognition by computer is an enormous field; these introductory remarks barely set the stage and certainly do not give a sense of the commercial possibilities. See e.g `www.face-rec.org/` for some entry points into this literature.)

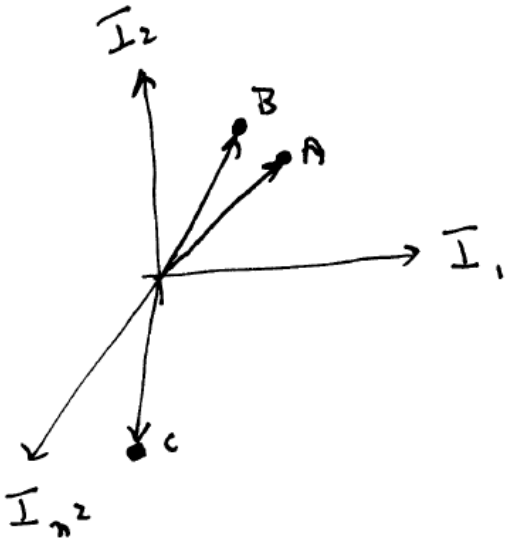

Figure 7.7: Images can be thought of as living in a high-dimensional vector space. Think of the coordinates in this space as the intensity of each pixel; so coordinate I_1 is the intensity of pixel 1, coordinate I_2 is the intensity of pixel 2, and no on. Each (normalized) image is a vector. Images nearby in this space (e.g., A and B) have high correlation, while images far away (e.g., A and C) are nearly orthogonal. This diagram also explains the need for normalization in correlation: we seek vectors of the same length – because it is the pattern of brightnesses that matter, not their actual values.

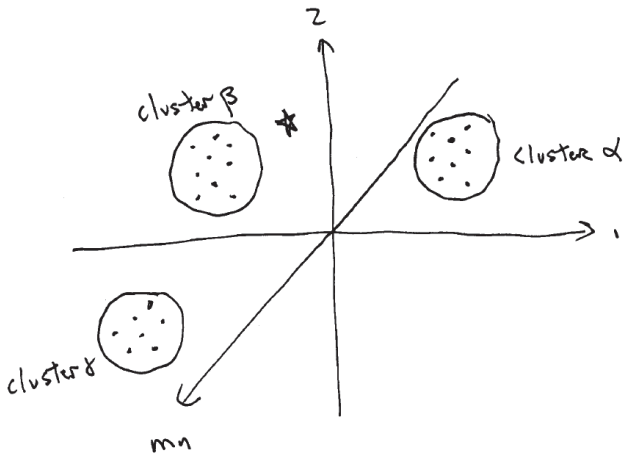

Figure 7.8: The geometry of classification in image space. Suppose we have been given a set of images taken from different individuals (call them α, β, γ. In the ideal, well-structured case, the subset of images from each individual forms a distinct cluster from the others. Now, given a previously unseen image ($\star$), it is natural to classify it as being part of the closest cluster.

given an image of an unknown person (or category of objects), find that group of known images to which it "most likely" belongs.

Using the representation just developed, we plot everything in image space (Fig. 10.5. The idea is to first build a set of clusters in image space, one for each "individual." Hopefully these clusters are smaller in diameter than the distance between them. Now, when a previously unseen image $\star$ of one of these individuals comes along it can be associated with the "nearest" cluster. (Since the unseen image captures directly the APPEARANCE of the individual, it will vary between photographs; thus one wouldn't expect it to overlap exactly with a previously acquired image.)

7.5.1 K-Means Clustering

The lateral inhibitory network of limulus can be viewed as a mechanism for evaluating each position across the compound eye as a candidate image of a mate. In effect, there were two possibilities: (1) move toward the mate or (2) keep looking for a mate. How might these two classes of images be "learned?" I put "learned" in quotes because, in this section, we shall be introducing our first (unsupervised) algorithm for organizing data into clusters, so it's machine learning not biological learning. Other, more biological approaches to learning will be introduced later.)

To begin, we need lots of images. Denote each image (data vector) $x_i, i = 1, 2, ..., N$ (for now let the size of each image vector be P pixels). If we plot these in

P-dimensional space then, hopefully, they organize into natural catagories, or clusters (Fig. 7.9(a)). For Limulus, the majority of these are background or otherwise uninteresting images, and some are images that include a mate. Of course, in general there might be many different clusters: how many categories of objects can you imagine? Let K be the number of clusters.

To start, we'll assume that we don't know very much about the world – we don't yet have a model for it – but we have to start somewhere. Building on what we've been doing in the previous section, then, suppose that we do have a MEASURE OF DISSIMILARITY between each pair of images: $d(x_i, x_j)$. Notice that this measure is related to a distance.

It is most natural to group those images that are most similiar ("closest") into the same cluster. This assignment process is represented in the map:

$$C(i) = k$$

that places image (datum) x_i into cluster k. How good is a particular assigment? Define a LOSS FUNCTION that measures the total dissimilarity across all points in all cluster by another type of "energy" (how spread out they are) by the formula:

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i=k)}\sum_{C(i'=k)} d(x_i, x_{i'}). \tag{7.19}$$

The brute force approach would be to try every combination of data points to clusters but, combinatorially, this is impossible because the number of combinations grows fast (For N = 20 and K = 4, the number of possibilities exceeds 10^{10} significantly!).

If our space of images has a Euclidean (distance) structure, it's natural to work with squared distance:

$$d(x_i, x_{i'}) = ||x_i - x_{i'}||^2 = \sum_{p=1}^{p=P}(x_{ip} - x_{i'p})^2 \tag{7.20}$$

so that the loss function becomes:

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{C(i=k)}\sum_{C(i'=k)} ||x_i - x_{i'}||^2 = \sum_{k=1}^{K}\sum_{C(i=k)} ||x_k - \bar{x_k}||^2 \tag{7.21}$$

where $\bar{x_k} = (\bar{x_{1k}}, \bar{x_{2k}}, ...\bar{x_{Pk}})$ is the mean coordinate vector for cluster k.

That is, we seek the clustering that minimizes the average dissimiliarity from the mean for all the data points within each cluster:

$$C^* = \min_C \sum_{k=1}^{K}\sum_{C(i=k)} ||x_k - \bar{x_k}||^2. \tag{7.22}$$

This can be achieved by the alternating optimization procedure called

K-means Clustering Algorithm

Given:
data points $x_i, i = 1, 2, ..., N$ and K, the number of clusters;
Choose
K cluster means at random.
Iterate:
Assign each data point to the nearest cluster k;
Recompute the means for each cluster;
Until:
No assignments change.

See Fig. 7.9. Note that this is a greedy approach; hardly any of the possible assignments are attempted. As such it may get stuck at a local min. Thus, as with *A. coli*, it makes sense to restart several times from different random positions.

7.5.2 Transformations of Images

Thinking of an image as a vector, a transformation of that vector is a matrix. Simplest example: the identity matrix – see Fig. 7.10.

A slightly more interesting example introduces a shear deformation (imagine "squeezing" your eye or camera in a lateral direction – DO NOT TRY THIS AT HOME!). In some directions the pixel values remain invariant while in others they change – fig. 7.11.

7.6 Applications and Limits of Template Matching

The application of template analysis in engineering is immense. Bar code readers are essentially reading a template of square waves in one dimension, with a specific code. Many other examples abound, from face recognition to record-album covers.

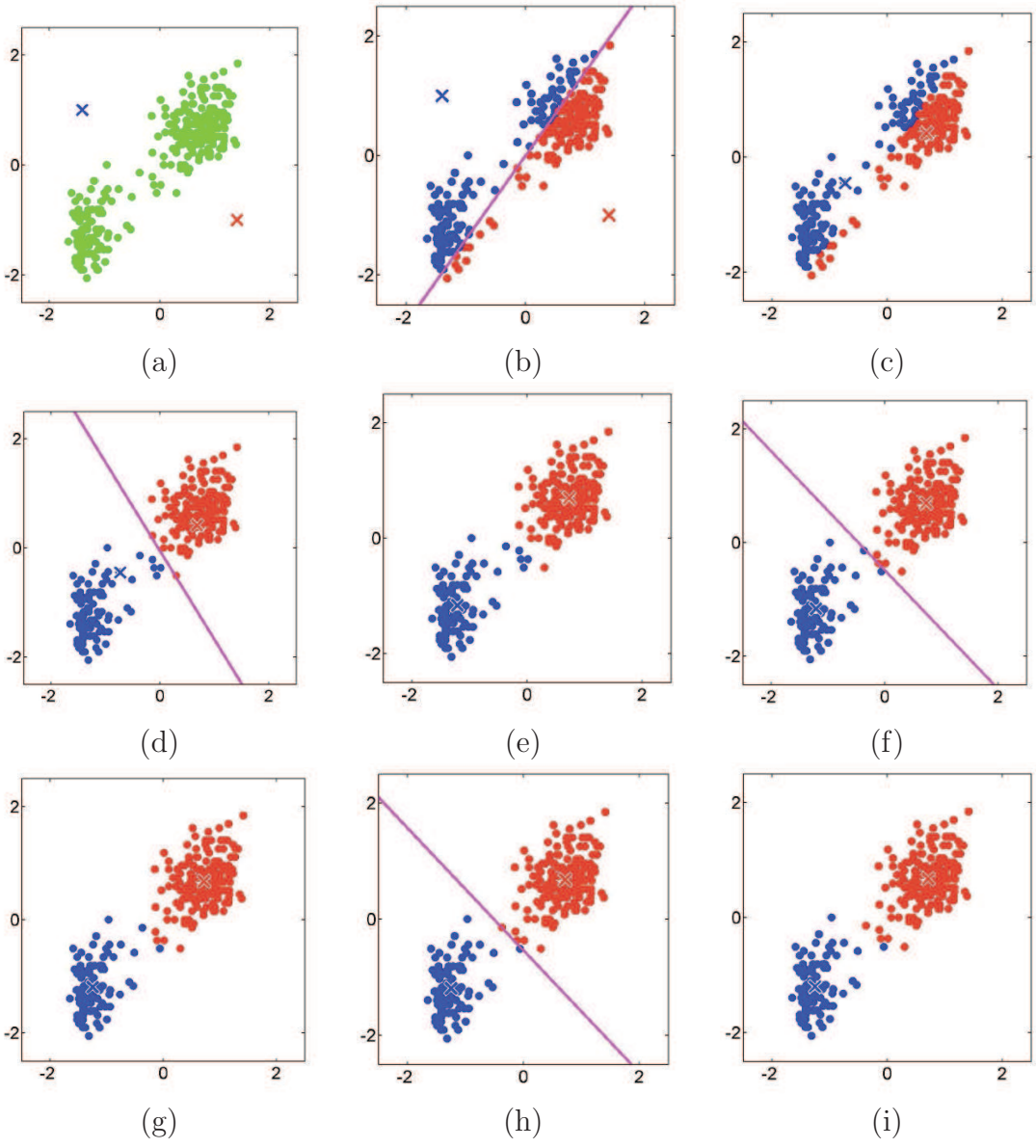

Figure 7.9: Illustration of k-means clustering. (a) Original data points, shown in green, plotted in $p = 2$ dimensions. Initial random guesses for the cluster centers. (b) Assignment of data points to nearest cluster center. Line indicates the cluster (decision) boundary. (c) Recalculation of centers. (d - h) Iteration of above until the assignments don't change (i). Figures from P. Bishop.

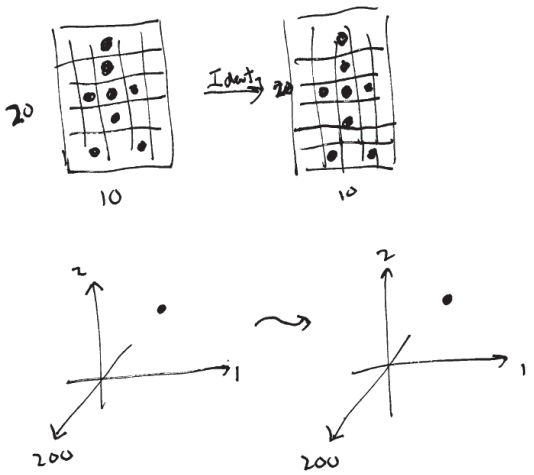

Figure 7.10: Example showing a 20 x 10 pixel image; the pixels are either black or white (a binary image). Under the identity mapping, nothing changes. This is shown abstactly at the bottom, where the image is a point in 200 = 20 x 10 dimensions. The identity maps every point in this space to itself.

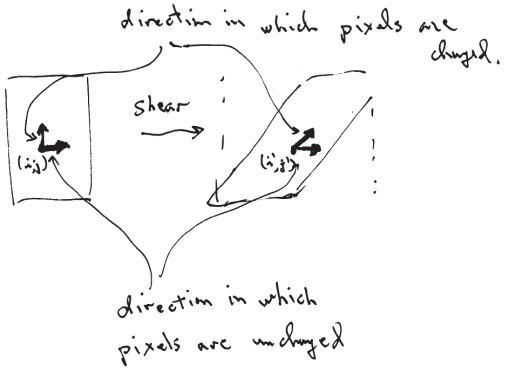

Figure 7.11: Shearing the image leaves pixels unchanged in one direction, horizontal in this example, but not in the other (vertical) direction. Hence there's something special about the horizontal dimension being unchanged when compared to the vertical.

Figure 7.12: Different typefonts for the letter "A"; can you make an image template that would work for all of them? How about zip codes?

7.6.1 Letter and Handwriting Recognition

How would you define a template for the letter "A" in a realistic verion of the task we considered in this lecture? See Fig. 7.12. How about reading a zip code automatically?

7.6.2 Recognition at Multiple Scales

The example of letter recognition considered earlier illustrated the fundamental interaction between noise and correlation-based matching: if there's enough noise, then the pattern is, simply put, not matched.

That example assumed that the pattern was defined at a particular scale: the size (in pixels) at which relevant pattern structure was represented. We assumed, in particular, that strokes – the lines of which letters were comprised – were a couple of pixels thick and that the noise was added "pixelwise". That is, the image consisted of the true image plus a noise image:

$$\hat{I}(i,j) = I(i,j) + N(i,j)$$

(Notice how we've begun to do "image algebra" here, where we're considering a new image to be a sum of other images. We also could subtract images. What would, for example, the "identity" image be for the algebraic operation of image addition?)

We saw earlier that there is a blurring (averaging) filter, and it seems clear that some amount of blurring can eliminate noise. Thus, preceeding template matching in noise we might consider the following steps:

- Take as input a noisy image.
- Blur the image.
- Perform template matching.

Would this have better performance in noise than what we considered earlier? (We shall focus on such statements more seriously in the next lecture.)

More generally, this suggests putting some algorithmic variation between the image and the match, and this can greatly expand the capabilities of template matching. For example, if the object may appear at different sizes (i.e., be viewed from different distances) then one might have a family of templates, one for each possible size (or orientation or whatever variation is of interest), and then seek the best one from among this set:

- Take as input an image of an object at unknown size.
- Calculate a series of templates at different scalings.
- Perform template matching for each one;
- Select the maximum correlation score over the different templates.

In general one could imagine a family of such templates based on e.g., the "group" of transformations allowing translation, rotation, and scaling. The max over this group would then provide some degree of INVARIANCE to the matching over the group of distortions, in the sense that we are invariant to distance and rotation effects in facial recognition (well, at least to some extent).

But there's another way to think about scales, and this is very closely tied to the Fourier analysis that we'll be doing in the next lecture. One can imagine a different concept of scale that goes from "coarse" to "fine", and the Fourier basis is relevant to thinking about this. Can you imagine, for example, what a very coarse version of a face would look like: will this provide a general idea of face vs. non-face, rather than Richard Nixon vs. Bill Clinton.

How might such "coarse" scale representations be related to Fourier? To blurring filters? We shall consider these issues in later lectures as well.

7.6.3 Curiosities of Recognizing Faces

Face recognition is harder than you might think. We'll develop a few more algorithms but none are totally satisfactory. Look at Fig. 7.13 and try to identify the person at the top from among the other figures; now try Fig. 7.14. The answer can be found at the end of the Chapter. In fact, people make about 30% errors on this task (This says something interesting about suspect identification procedures!)

7.7 Conclusions

Limulus finds a mate by a simple matching operation. It is highly non-linear, because once an action is taken it is taken. But there are strong limitations to this approach:

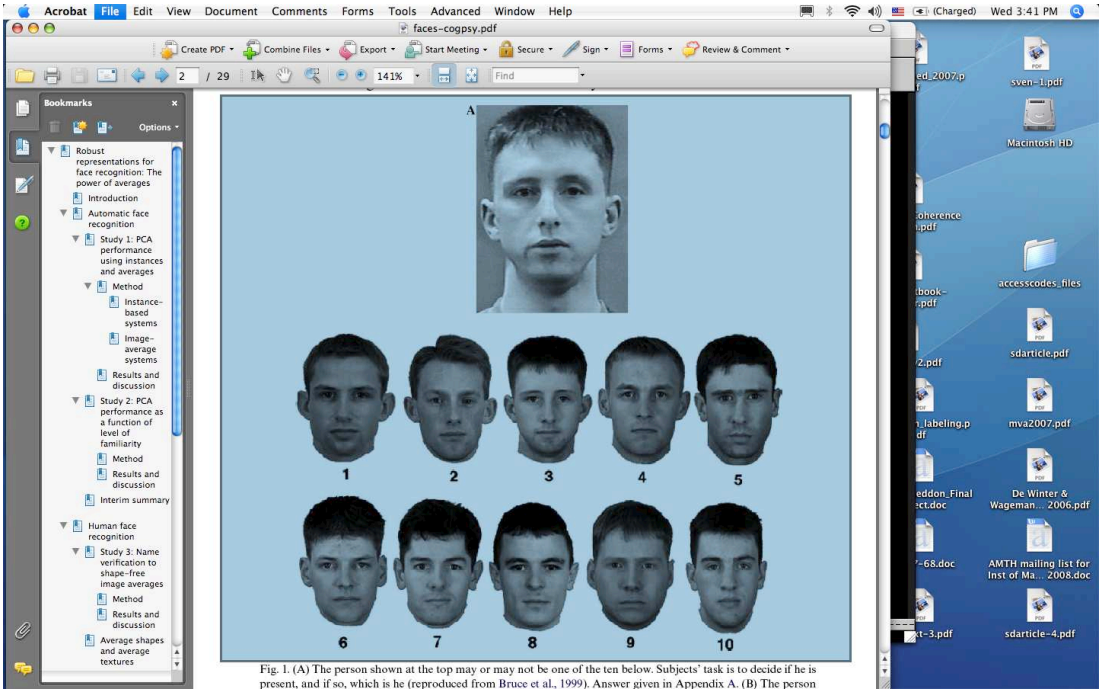

Figure 7.13: Can you find the person in the top image among those in the bottom? Answer at the end of the chapter.

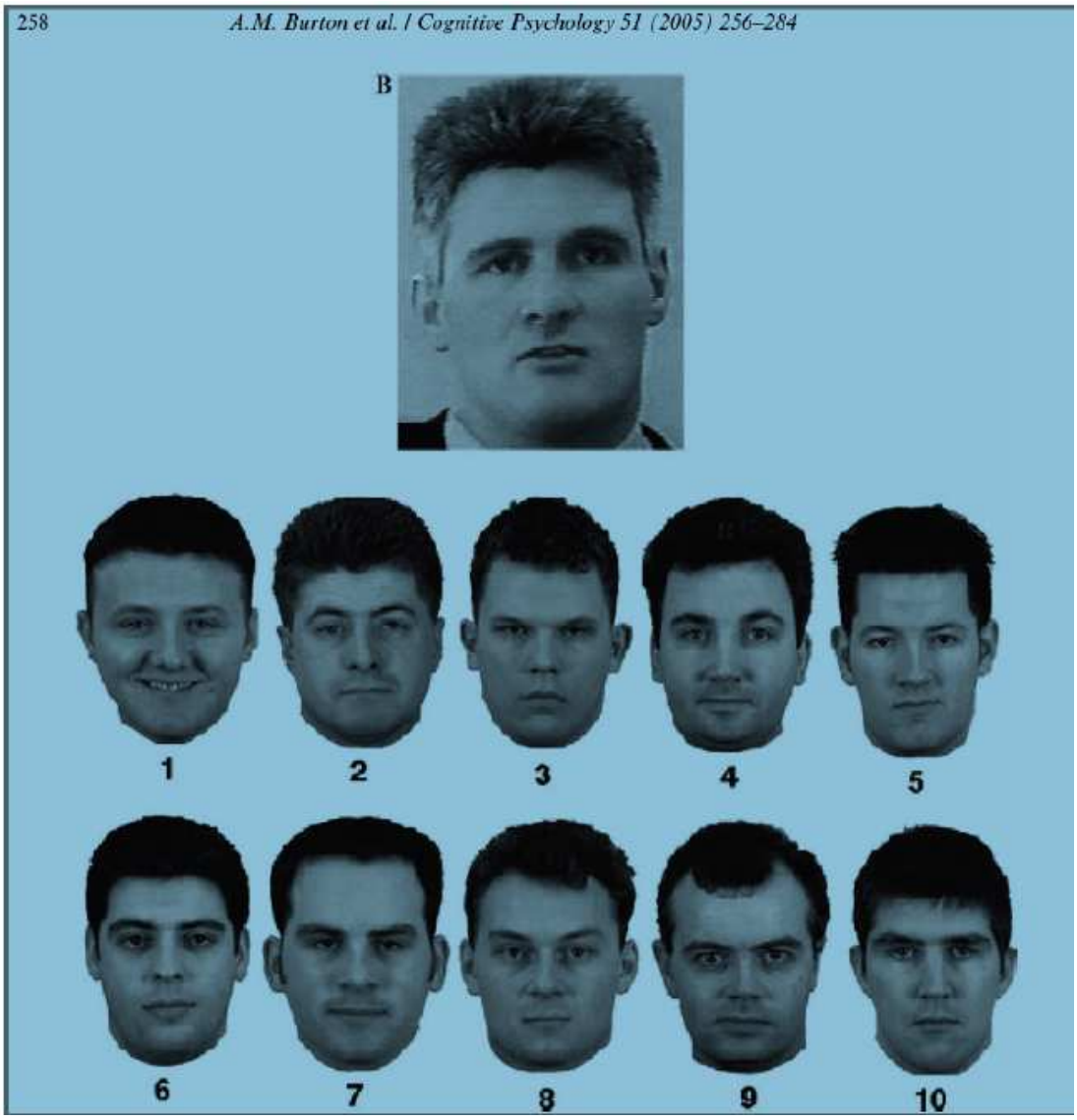

Figure 7.14: Can you find the person in the top image among those in the bottom? Answer at the end of the chapter.

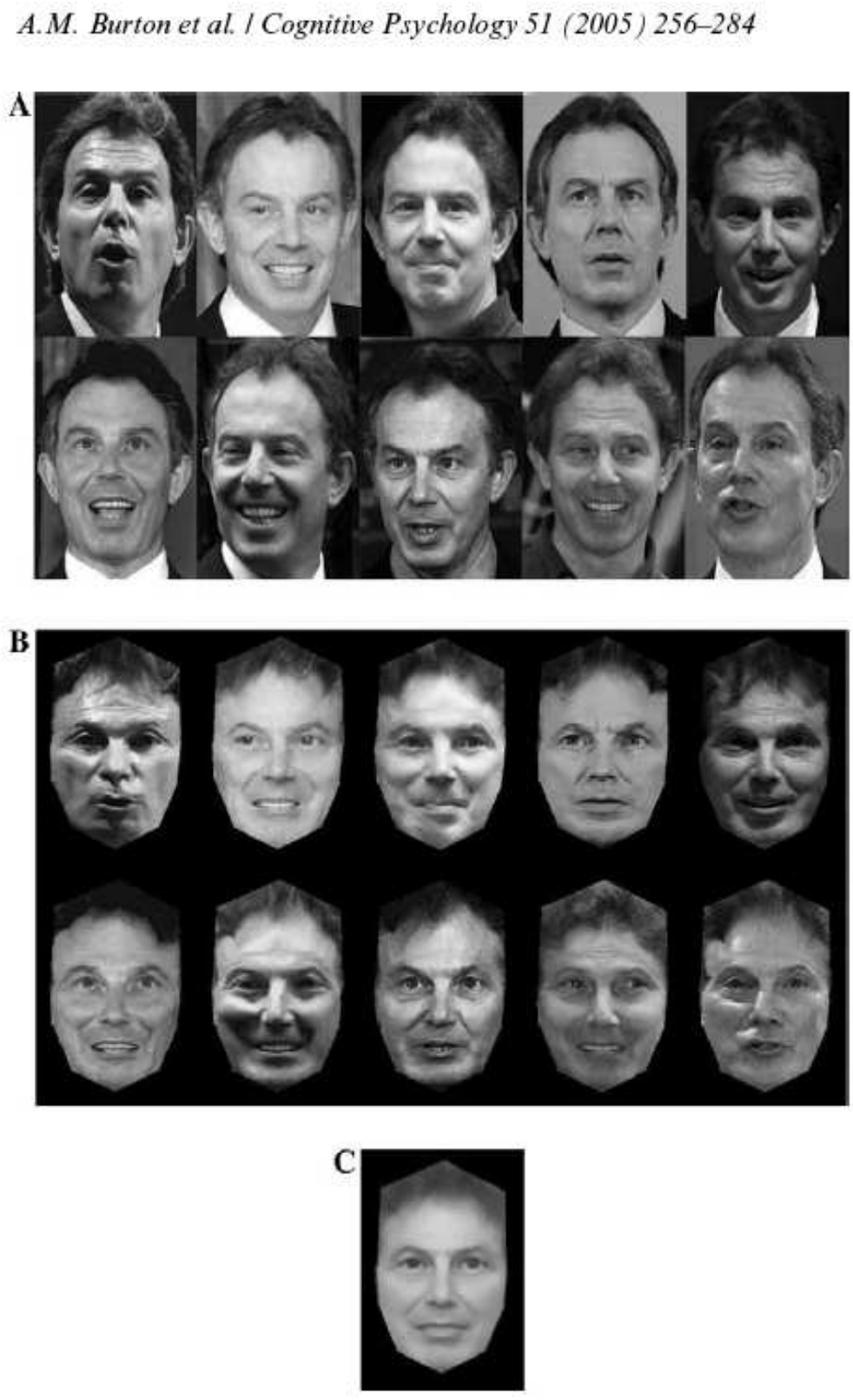

Figure 7.15: Snapshots of Tony Blair, and an average "composite" image of Tony Blair. The average was computed by an extremely complex, user-interactive procedure outlined in the text. Recognition of such averages is more accurate than for an individual photo.

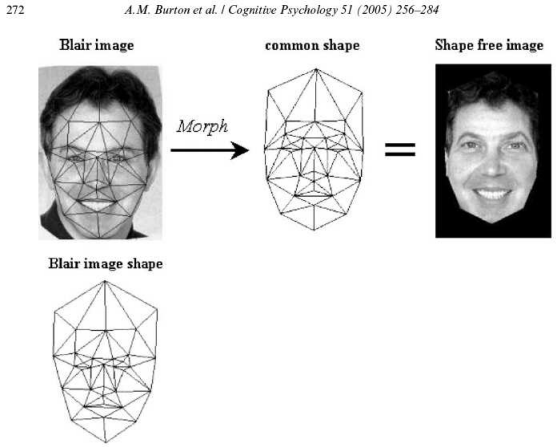

Figure 7.16: To compute the "average" image of Tony Blair, a complex template is fitted by hand to accurately capture the key points of interest on a face. This part of the procedure would be extremely difficult to automate, but corresponds to the flexible template necessary to fit images of even a single face together. Later in the class we will study such graph-based representations.

how to deal with articulated objects; with the fact that different objects may appear more similar when projected onto a template than different instances of the actual object. This is non-trivial, since there are so many different "objects" that we can recognize.

Answers: Fig. 7.13: the target matches panel 3. Fig. 7.14: the target is not present.

7.8 Notes

Limulus from Carl N. Shuster, Jr, Robert B. Barlow, and H. Jane Brockmann, The American Horseshoe Crab, Harvard U. P., 2003; figures from Chapter 4.

Norms and template matching after Duda and Hart, Pattern Classification and Scene Analysis, 1973.

For a review of linear algebra, see the books or on-line lectures by Gilbert Strang.

K-means discussion taken from Hastie, Tishirani, and Friedman, The Elements of Statistical Learning; the example is from Bishop, Pattern Recognition and Machine Learning.

Stuff still to come: