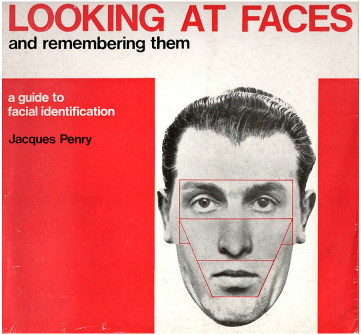
# Looking at Humans

Adrian F. Clark
CSEE, University of Essex
alien@essex.ac.uk

# Recognizing Faces

There is evidence that suggests we all have ‘circuitry’ in our brains dedicated to recognizing faces and associating names with them

Computer recognition of faces started in earnest with policing and has improved *vastly* since then



# Computer face recognition

There are normally three stages:

1. Locating the faces in an image: face *location* (*localization*)
2. Making face shapes align: face *normalization*
3. Face *recognition* (*identification*)

We shall look at each of these in turn

# Locating faces by colour

Skin is coloured by a compound called *melanin* which has a characteristic colour in HSV space

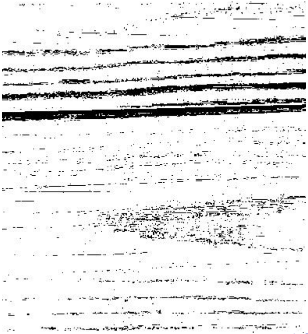
The idea is to convert an image to HSV space and identify regions with that hue

# . . . but. . .

Illumination is a problem: moving from sunlight to (say) an incandescent lamp changes the hue

Humans recognize faces in black-and-white images and so clearly don't do it this way

Lots of other things have the same hue



# The Viola and Jones algorithm

They worked for Mitshubishi, who patented it

The algorithm is based around machine learning

Training is quite slow but running the trained system is fast, a desirable characteristic

They collected a large database of images containing faces (at known locations) **and** the same number of non-face images

They reduced the images to $24 \times 24$ pixels

They computed all possible Haar features: 162,336 such features exist

# Haar features

Haar features are computed by summing the pixels under the dark and light regions and then calculating their difference

Some features help in locating faces or rejecting non-face regions, though none work particularly well — they are *weak classifiers*

# Learning which features to use

They then used a machine learning algorithm, *adaptive boosting ("Adaboost")* to combine sets of the most useful weak classifiers into stronger ones

The result was a series or *cascade* of 38 stages using 6,000 Haar features which classified all images in their face database correctly

Quick-to-compute features are used first to reject regions that are clearly non-face, and so on

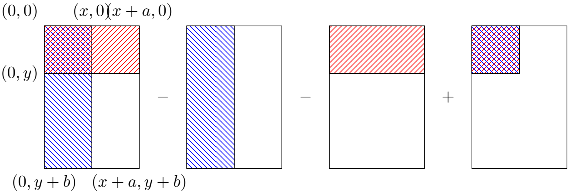
In OpenCV, cascades are stored in XML files, and there are also ones to be found for faces, eyes, mouths, cats, Russian number plates. . .

# Calculating Haar features quickly

The obvious approach to computing Haar features has four nested loops: see the code on pp93–4 of the notes

A vast speed-up is possible through the use of an *integral image* representation — known as a *summed-area table* in computer graphics for texture-mapping and widely implemented on graphics cards

$$ab = (x + a)(y + b) - x(y + b) - (x + a)y + xy$$



# What is an integral image?

The value at a pixel is the sum of itself and *all the pixels* above and to the left of it

It is quick to compute (by exploiting the equation on the previous slide)

See the lecture notes for code to calculate it in the 'obvious' and fast ways

The time taken to find the sum of pixels in a region **is constant** irrespective of its size — there are very few cases where this is happens

# Measuring attractiveness (for your amusement)

Based around two characteristics:

- the more symmetric a face appears, the more attractive it is supposed to be
- humans apparently find that things arranged according to the so-called *golden ratio* $\phi$ are pleasing. For $a > b > 0$, this is

$$\phi \equiv \frac{a+b}{a} = \frac{a}{b}$$

The *Parthenon* in Athens is the classic example of the use of golden ratio proportions

# Model-based image coding

We track motion of the head and facial features in a video

We infer how the head moves in 3D

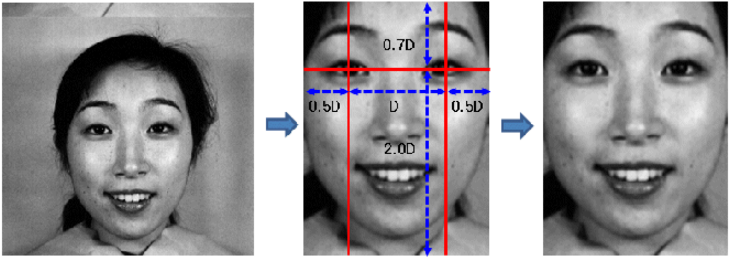
We send animation data over a link — the analogue telephone network when we did our research

We animate a 3D model of the head to mimic the motion

The result is good enough for lip-reading

# Normalizing faces

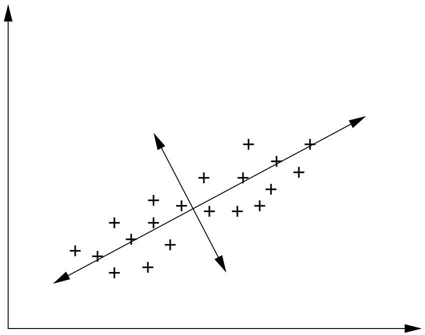
The idea is to scale faces so that the eyes are always the same size and in the same place




*From https://www.semanticscholar.org/paper/Geometric-Feature-Based-Face-Normalization-for-Kim-Sohn/8ff1f263d91f192269f6f3b324bdb1d30761ae41*

# Recognizing faces: *eigenfaces*

The classic way to recognize faces is through an algorithm called *eigenfaces* — though it is not limited to faces



*Principal component analysis*

# PCA fits a new set of axes (“basis vectors”)

The ideas is to fit a new coordinate system to a cloud of data so that its spread along each axis is maximized

How does one measure the spread of data? By its variance — or, as more than one axis is involved, its *co-variance*

The axis-fitting uses a mathematical technique known as *Eigen decomposition*

We can describe any point in the cloud as a weighted sum of the *principal components*, the new axes

# Eigenfaces describes people by their new axes locations

We store a person's ID and the set of coordinates along the new axes that correspond to the image of their face

When a *probe* (test) image is used, the ID of the closest set of weights to the probe is how the person is 'recognized'

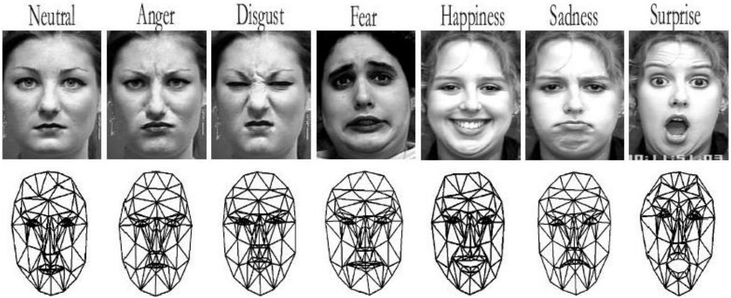
This works moderately well, though more modern techniques out-perform it

# Affective computing

This is based around the idea that expressions, and hence the appearance of a face, depend on the person's emotional state

This is what affective computing attempts to identify

The problem is that many of the databases available are poor



# Human body motion and HOG

HOG is the *histogram of oriented gradients*, computed in non-overlapping regions of the image

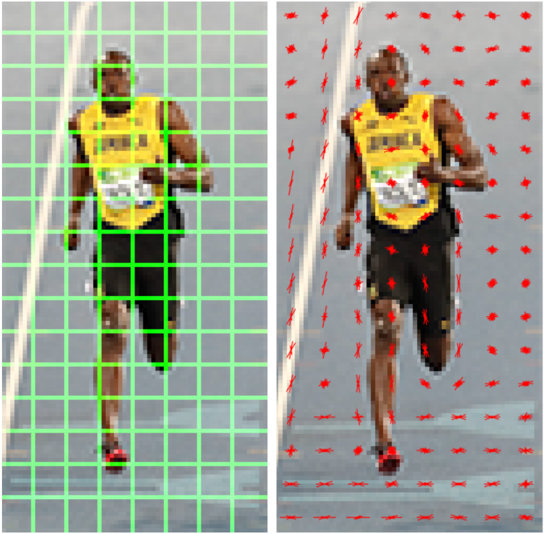
In each region, convolve with

$$\begin{pmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

to find vertical and horizontal edges respectively

Use these $x$ and $y$ components to compute an angle (as in Canny's edge detector) and histogram the angles — the largest histogram value give the dominant angle

If you do this in a time sequence, you can identify (say) a person running by regular, cyclic changes in the angles



# OpenPose

You might be asking yourself whether it would be easier to identify where the body is

Long thought to be impossibly difficult to do... but then OpenPose appeared, a real game-changer

Based around a convolutional neural network

Gathering enough training data was a major undertaking

There are now several alternatives too


*OpenPose in action*

Do watch the video