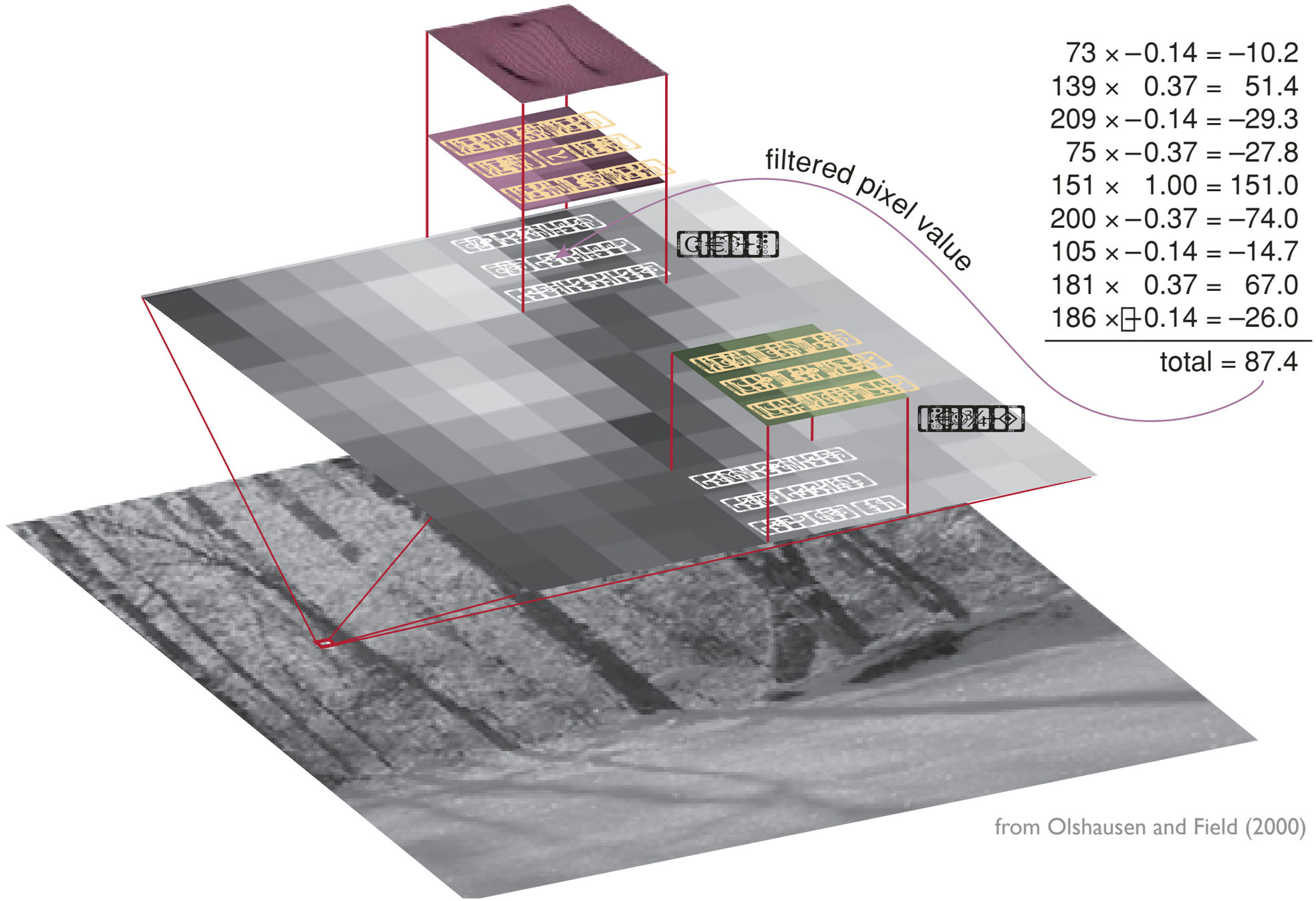
EECS 531

Computer Vision

Feature Detection and Classification

Convolution a natural image with a 2D Gabor function



from Olshausen and Field (2000)

The "response" of the filter at position (x,y) will be largest when the (normalized) feature pattern exactly matches the image.

separable linear filter kernels

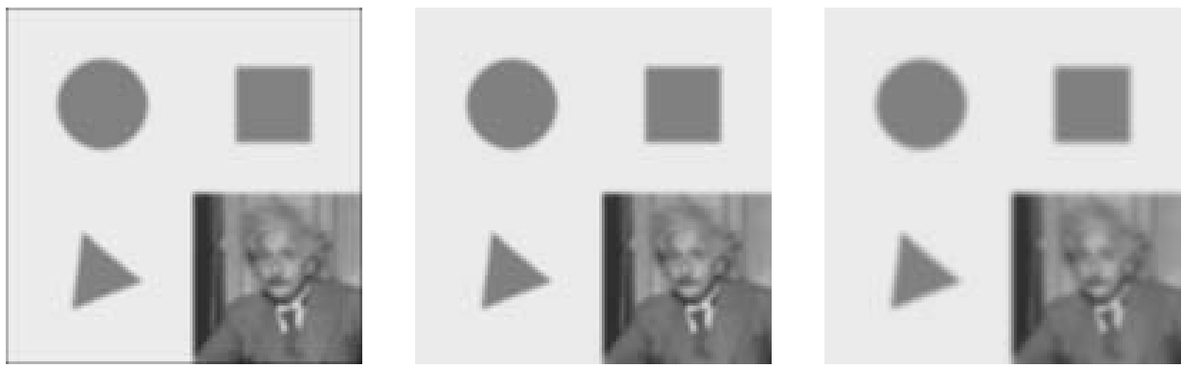
K

$$\frac{1}{K^2}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & 1 & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \quad \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 2 & 8 & 12 & 8 & 2 \\ 6 & 24 & 36 & 24 & 6 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

These blur the image by taking a weighted average of the local pixels.

v = h

$$\frac{1}{K}\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$



(a) box, $K = 5$ (b) bilinear (c) "Gaussian"

- These kernel matrices are "separable" — they are defined by the outer product of two one-dimensional kernels:

$$\mathbf{K} = \mathbf{v}\mathbf{h}^T$$

  where **K** is the kernel matrix and **v** and **h** are vectors.

- Can do 2D convolution with horizontal and vertical convolutions with **v** and **h**.
- This allows the convolution operation to be performed in O(2K) multiply-add operations per pixel instead of $O(K^2)$, where K is the size of the kernel.

Image Processing

Origin

Smoothed

Sharpened

Letter Detection

does seem to work well in the face of these challenges is the human visual system. It makes eminent sense, therefore, to attempt to understand the strategies this biological system employs, as a first step towards eventually translating them into machine-based algorithms. With this objective in mind, we review here 19 important results regarding face recognition by humans. While these observations do not constitute a coherent theory of face recognition in human vision (we simply do not have all the pieces yet to construct such a theory), they do provide useful hints and constraints for one. We believe that for this reason, they are likely to be useful to computer vision researchers in guiding their ongoing efforts. Of course, the success of machine vision systems is not dependent on a slavish imitation of their biological counterparts. Insights into the functioning of the latter serve primarily as potentially fruitful starting points for computational investigations.

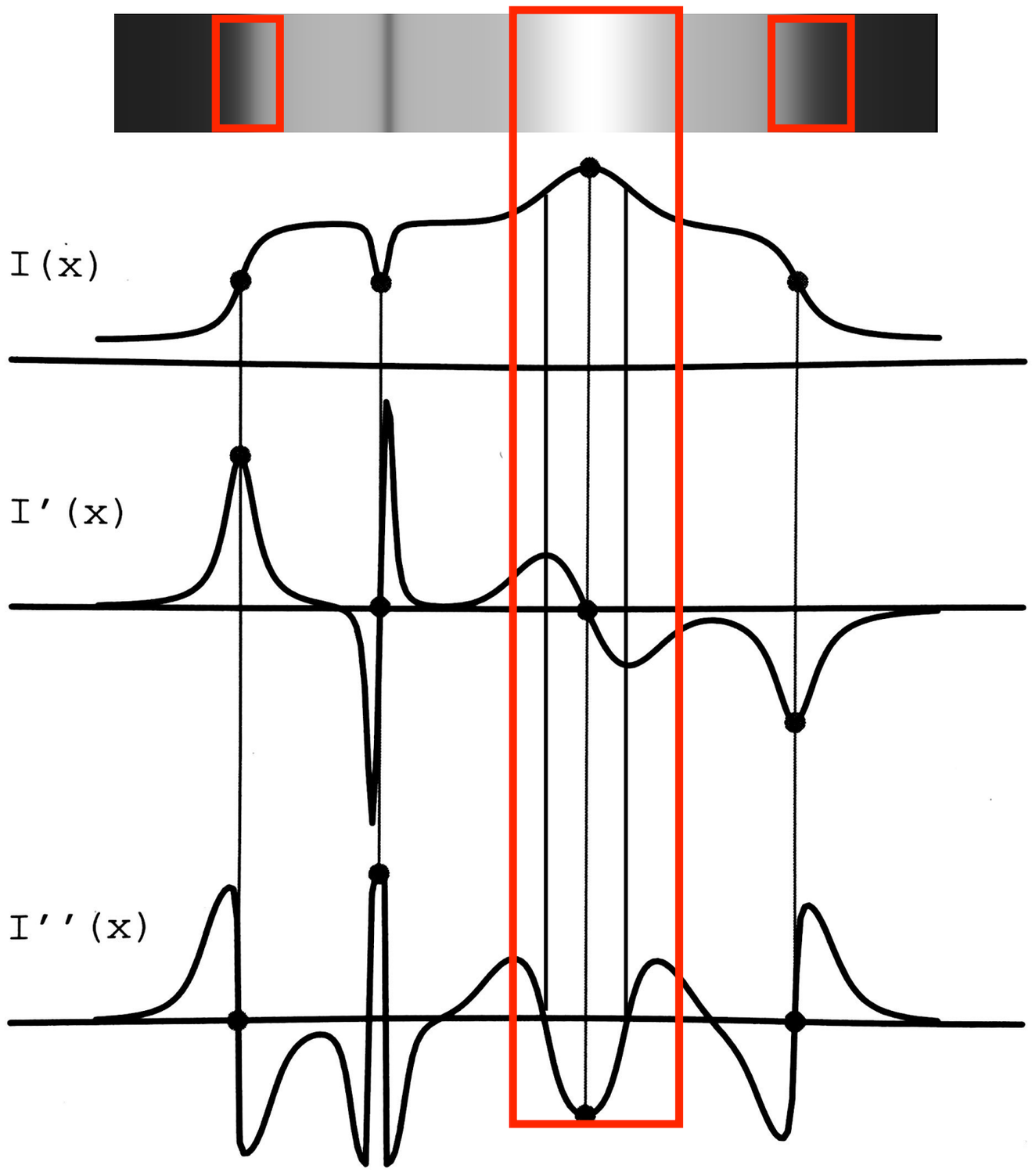
does seem to work well in the face of these challenges is the human visual system. It makes eminent sense, therefore, to attempt to understand the strategies this biological system employs, as a first step towards eventually translating them into machine-based algorithms. With this objective in mind, we review here 19 important results regarding face recognition by humans. While these observations do not constitute a coherent theory of face recognition in human vision (we simply do not have all the pieces yet to construct such a theory), they do provide useful hints and constraints for one. We believe that for this reason, they are likely to be useful to computer vision researchers in guiding their ongoing efforts. Of course, the success of machine vision systems is not dependent on a slavish imitation of their biological counterparts. Insights into the functioning of the latter serve primarily as potentially fruitful starting points for computational investigations.

Edge detection

- two basic methods
  - gradient based methods
  - matched filters
- gradient methods
  - compute the discrete image gradient in specific orientations
  - look for peaks in the 1st derivative
  - Or zero crossings in the 2nd derivative
- matched filters:
  - design an edge-like "feature"
  - convolve the image with the feature
  - threshold the output

Gradient approaches to edge detection

- Treat the image as a continuous function (valid?)
- Edges are rapid changes in intensity, `I(x)`
- For the first derivative, `I'(x)`, an edge will correspond to a maximum or minimum.
- For the second derivative, `I''(x)`, edges correspond to zero crossings.
- Note that these do not uniquely identify "edges".



Gradient approaches to edge detection

- Hundreds of strategies
- Often boil down to convolving a feature with the image
- Two classic approaches to edge detection
  - Sobel:
    - calculate the horizontal and vertical intensity gradient at each point (can be done with simple 2D filters)
    - threshold output
  - Canny:
    - smooth the image with a Gaussian filter
    - calculate the gradient in four directions
    - threshold

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

- Many, many variations on this basic design:
  1. *smooth*, i.e. eliminate noise or choose scale
  2. *compute gradient*, i.e. identify points (and directions) of high change

Calculating discrete gradients

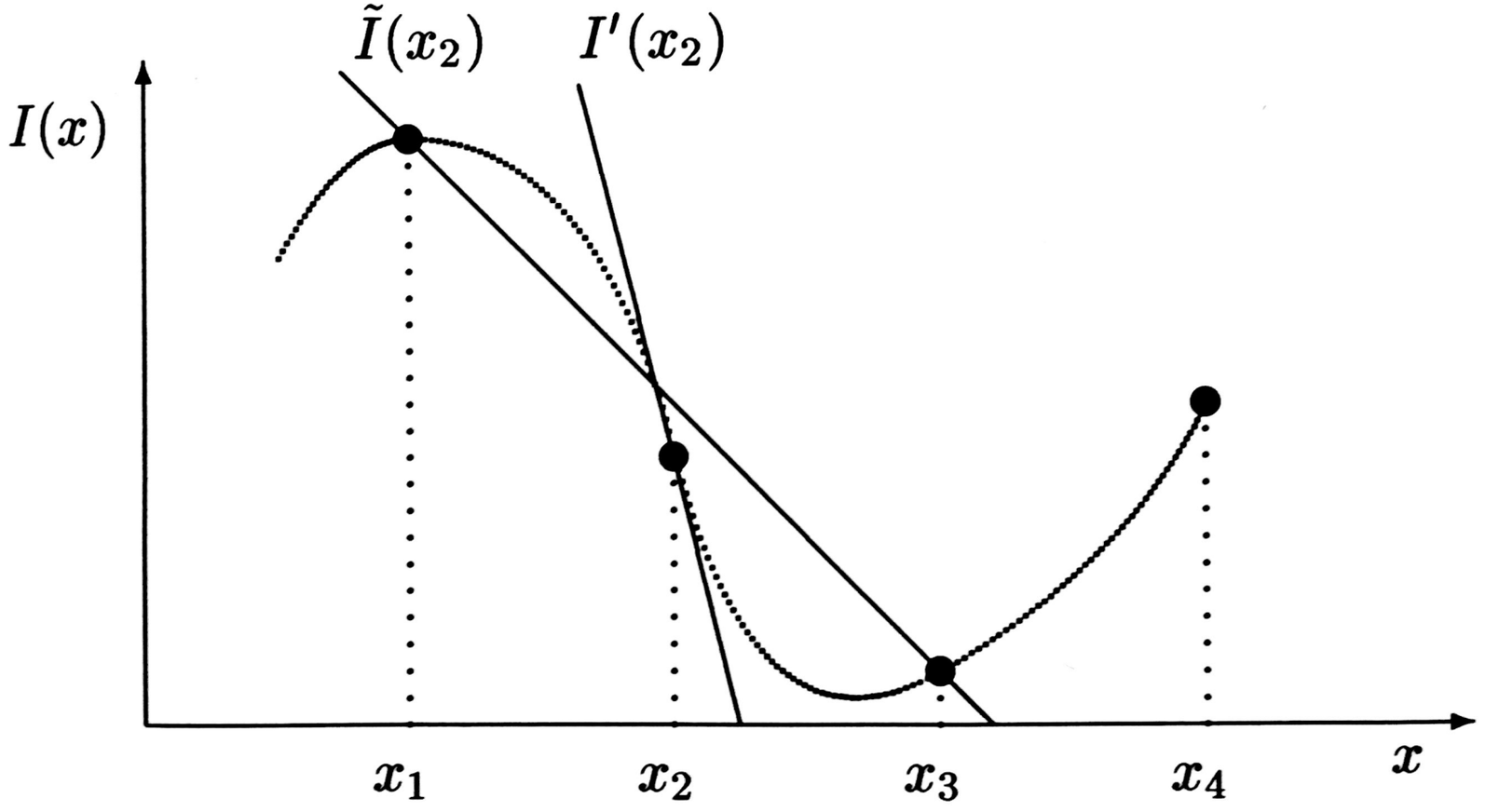
- A common way to define the derivative of sampled functions:

$$I'(x) = \lim_{h \to 0} \frac{I(x+h) - I(x-h)}{2h}$$

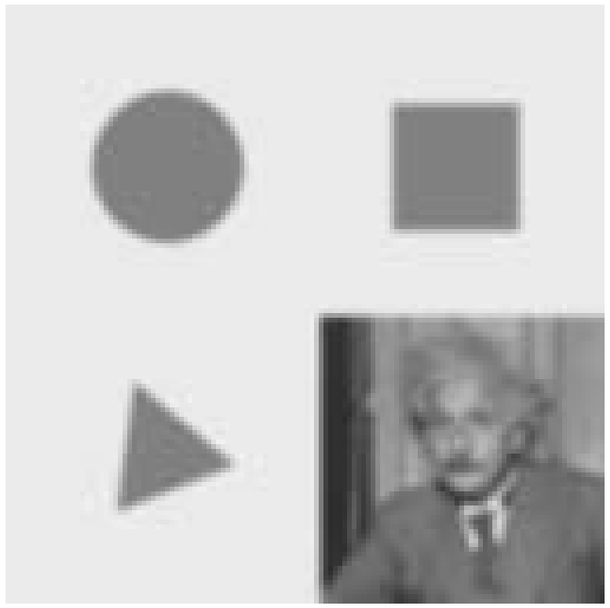
- With h=1, we have

$$I'(x) = \frac{1}{2}(I(x+1) - I(x-1))$$

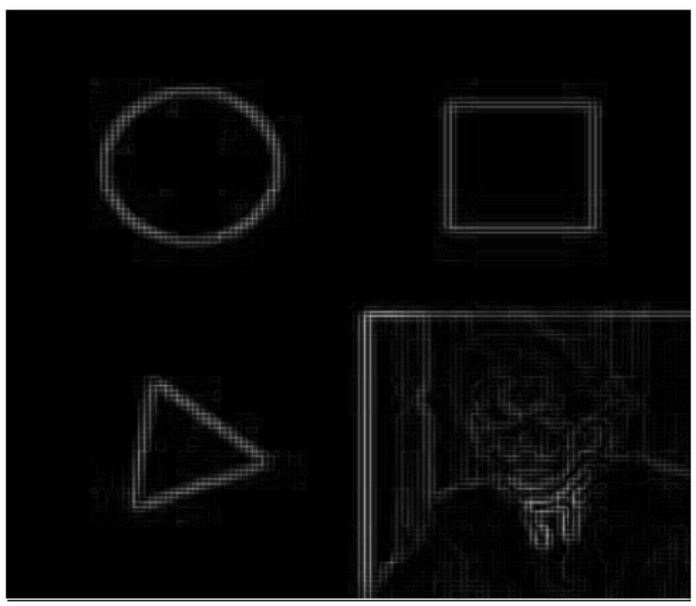
filter: [0.5, 0, -0.5]



Sobel filter kernels



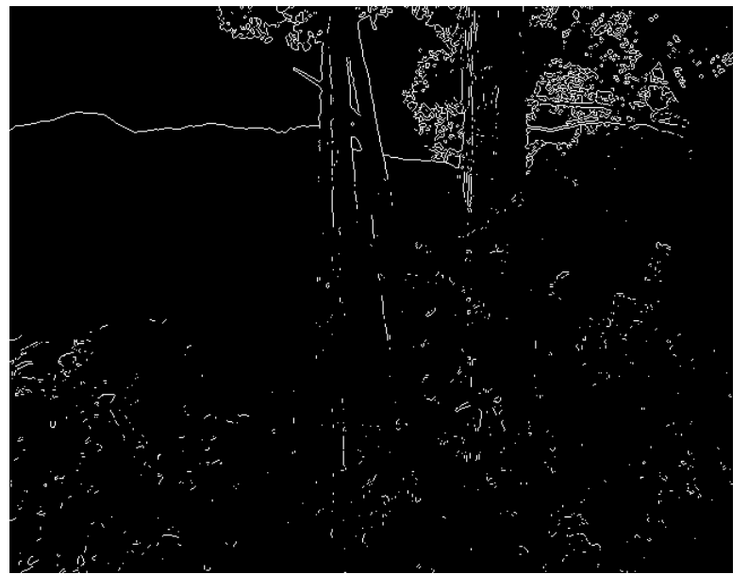
$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

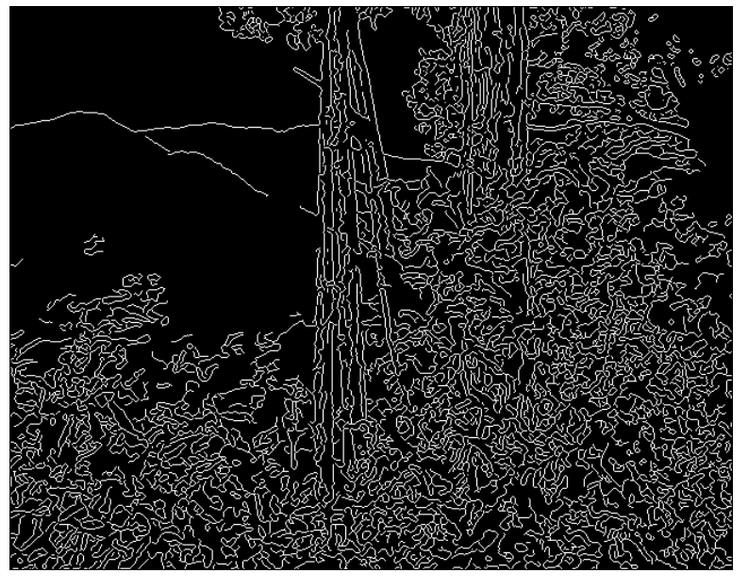


$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

Some matlab code for edge detection

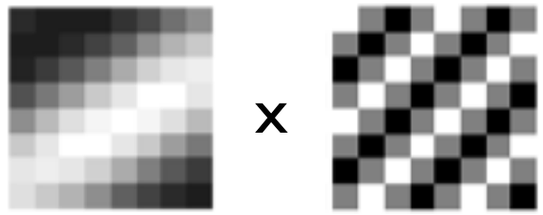
```
% read color image
I_rgb=imread('kyoto.png');
imagesc(I_rgb);

% convert to grayscale
I = mean(I_rgb,3);   % I is 500x640x3, take mean along 3rd dimension
imagesc(I);
colormap gray

% or more fancy:
I_hsv = rgb2hsv(I_rgb); % convert to Hue, Saturation, Value
I_v = I_hsv(:,:,3);     % pull out value (ie intensity)
figure(2);
imagesc(I_v);
colormap gray

% or yet another way, more directly:
I = rgb2gray(I_rgb);  % gray = 0.2989 * R + 0.5870 * G + 0.1140 * B

figure(1);
imagesc(I);
colormap gray

% now detect some edges:
E_sobel = edge(I, 'sobel');
figure(2);
imagesc(E_sobel);
colormap gray

E_canny = edge(Ig, 'canny');
figure(3);
imagesc(E_canny);
colormap gray
```





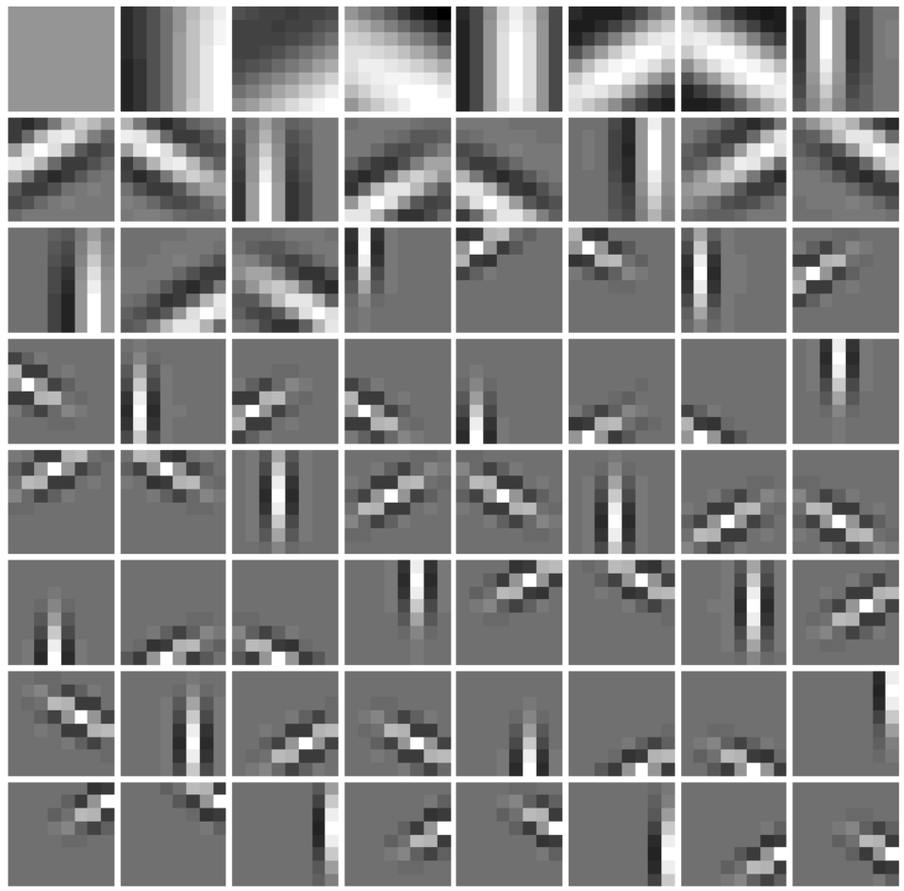
Gabor functions

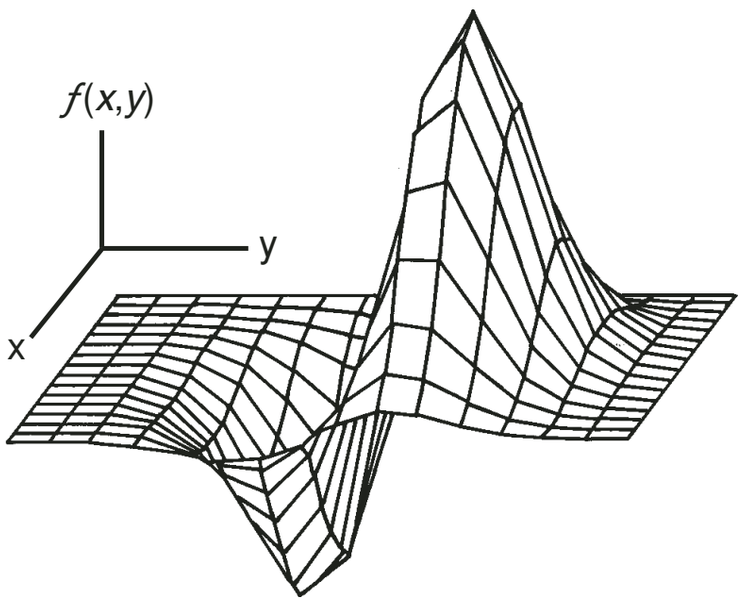
- Gabor functions are often called "edge-detectors"
- They're defined by:
  (2D) Gaussian x Sinusoidal Grid

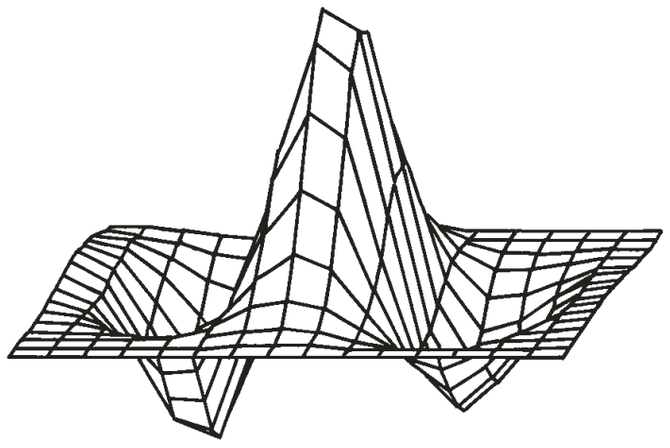


- parameters: x-y position, orientation, width, aspect ratio, phase (which shifts the sine wave grating)

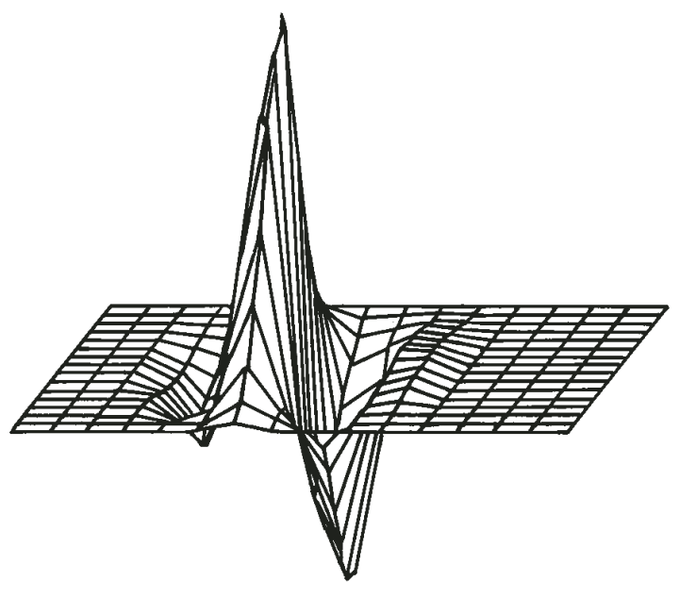
Gabor Features





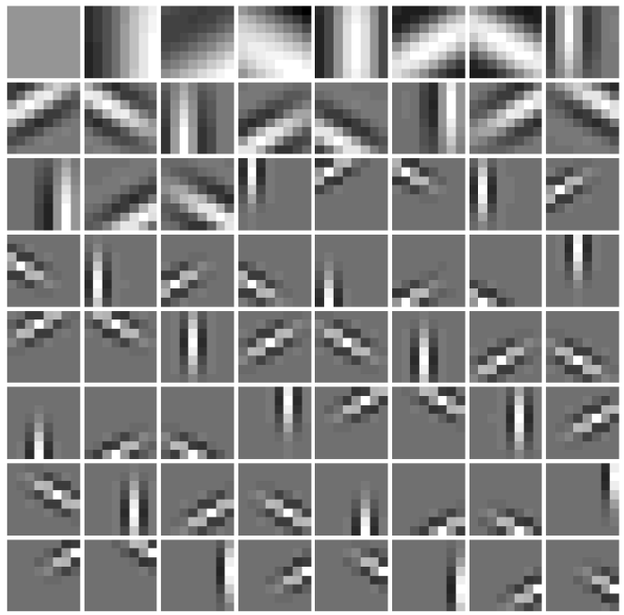


Gabor functions

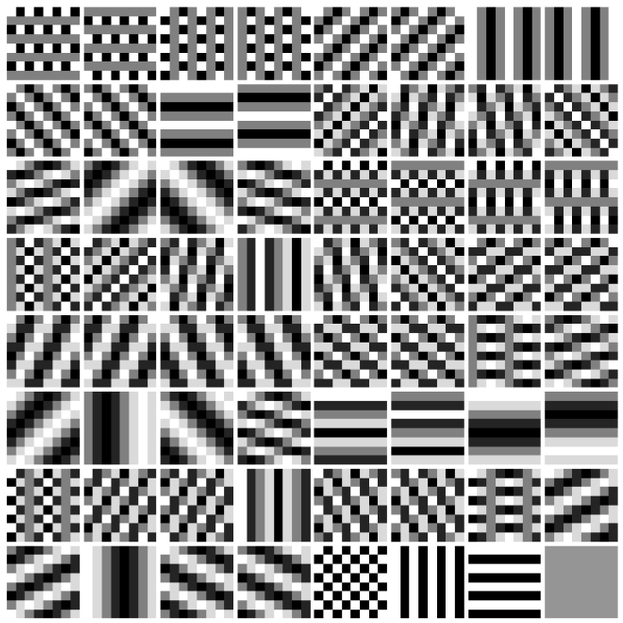


Some common image features

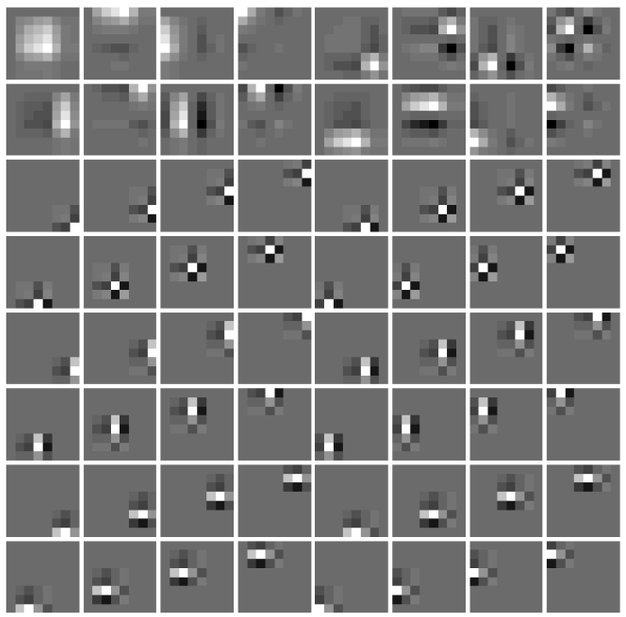
Gabor



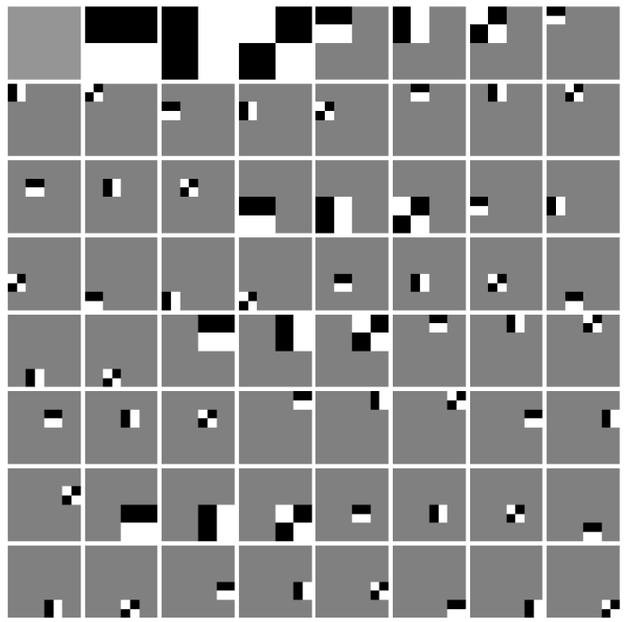
Fourier



Wavelet



Haar

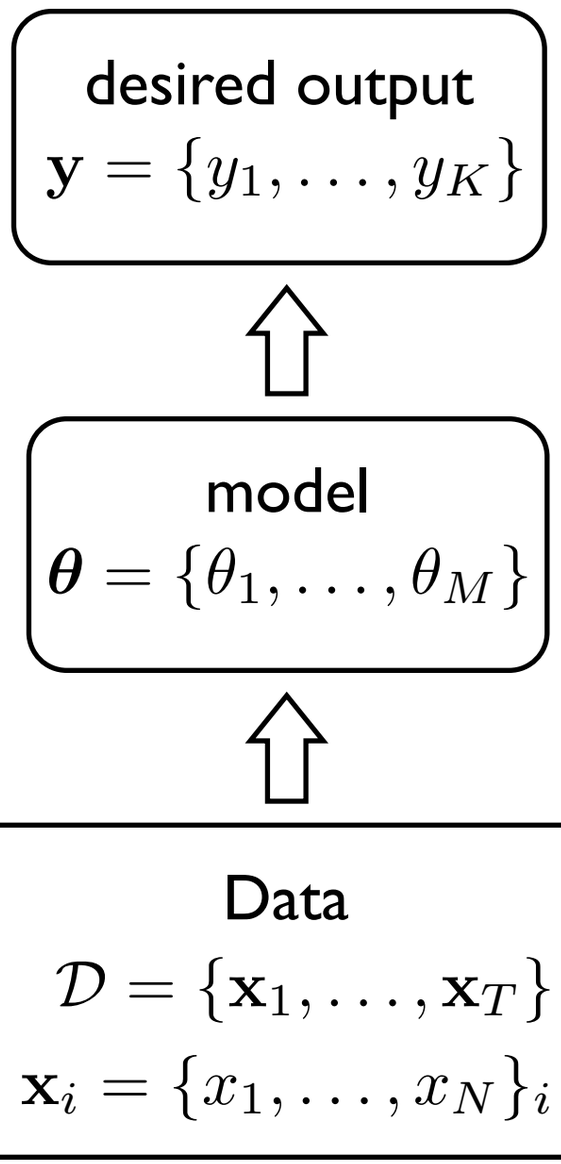


These are also image *bases*, which we'll discuss in a future lecture. For now, they are simply feature sets.

Problems with feature detectors

- How do you choose the features?
- How do you detect? Threshold?
- How do you account for variability?
- Next: *Separating signal from noise*

The general classification problem



output is a binary classification vector:

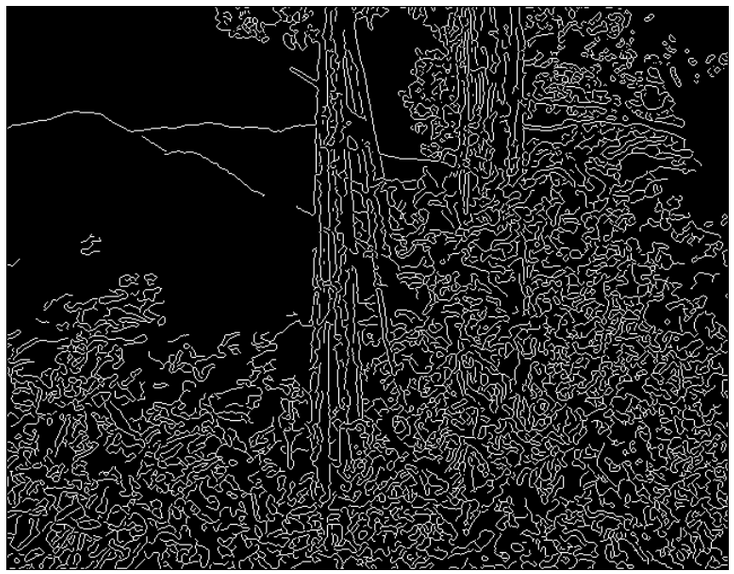
$$y_i = \begin{cases} 1 & \text{if } \mathbf{x}_i \in C_i \equiv \text{class } i, \\ 0 & \text{otherwise} \end{cases}$$

model, defined by M parameters.

input is a set of T observations,
each an N-dimensional vector
(binary, discrete, or continuous)

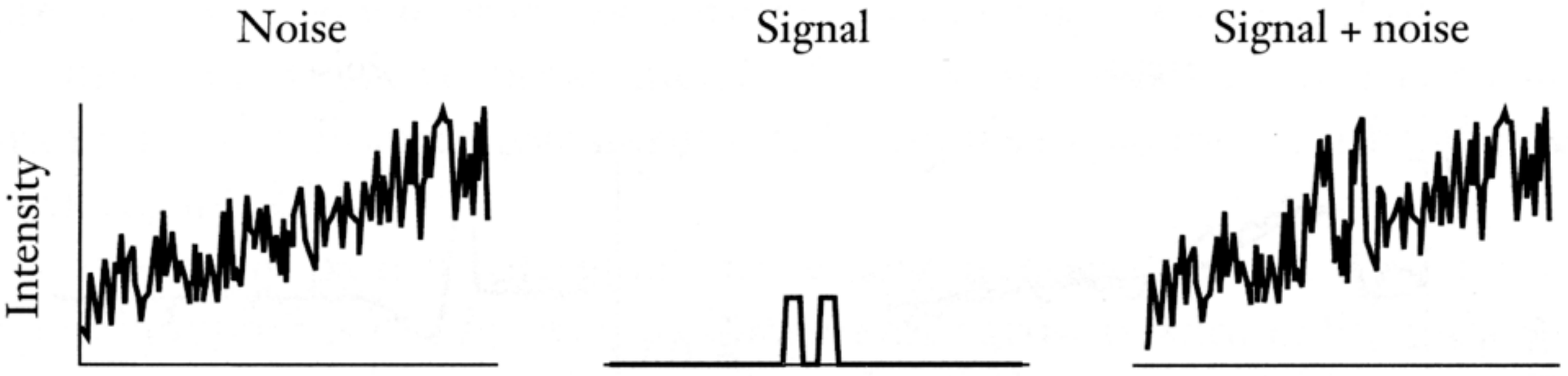
Given data, we want to learn a model that
can correctly classify novel observations.

Signals and noise



Signals and noise

- both signals and noise can have arbitrarily complex structure
- **signal**: generically defined, something that contains useful information
- **noise**: anything that interferes with detecting the signal:
  - *intrinsic*: variation in the signal itself
  - *environmental*: interfering signals or other sources of randomness
  - *receptor*: noise introduced in signal transduction
  - *processing* (or circuit): noise introduced in processing or extracting the signal

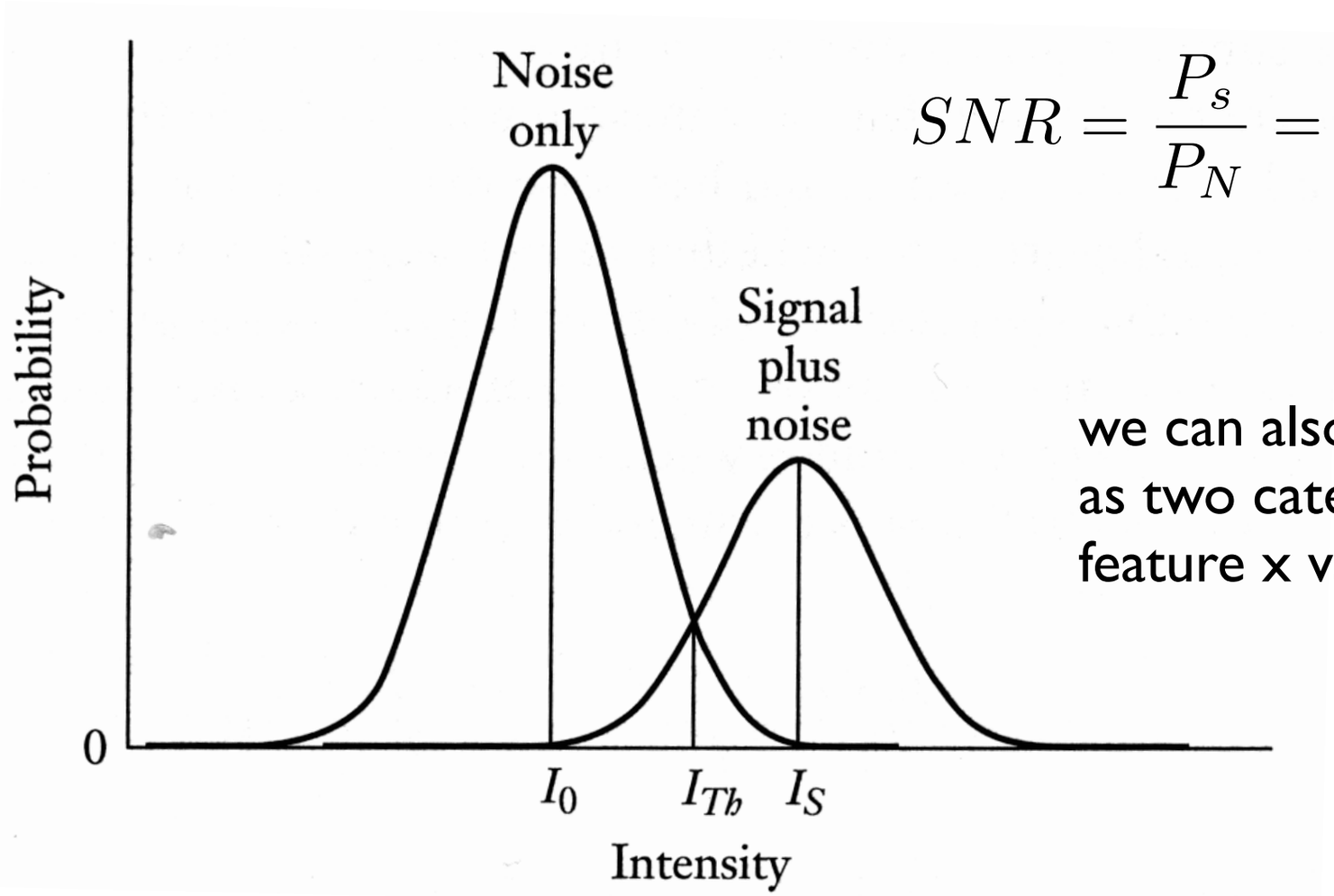


from Dusenbery, 1992

Separating signal from noise

- consider the statistical distribution of the signal and noise
- for some values the signal is not detectable
- for others it is detectable with high reliability
- the separation between these two distributions determines the degree of detectability: signal to noise ratio (S/N or SNR)

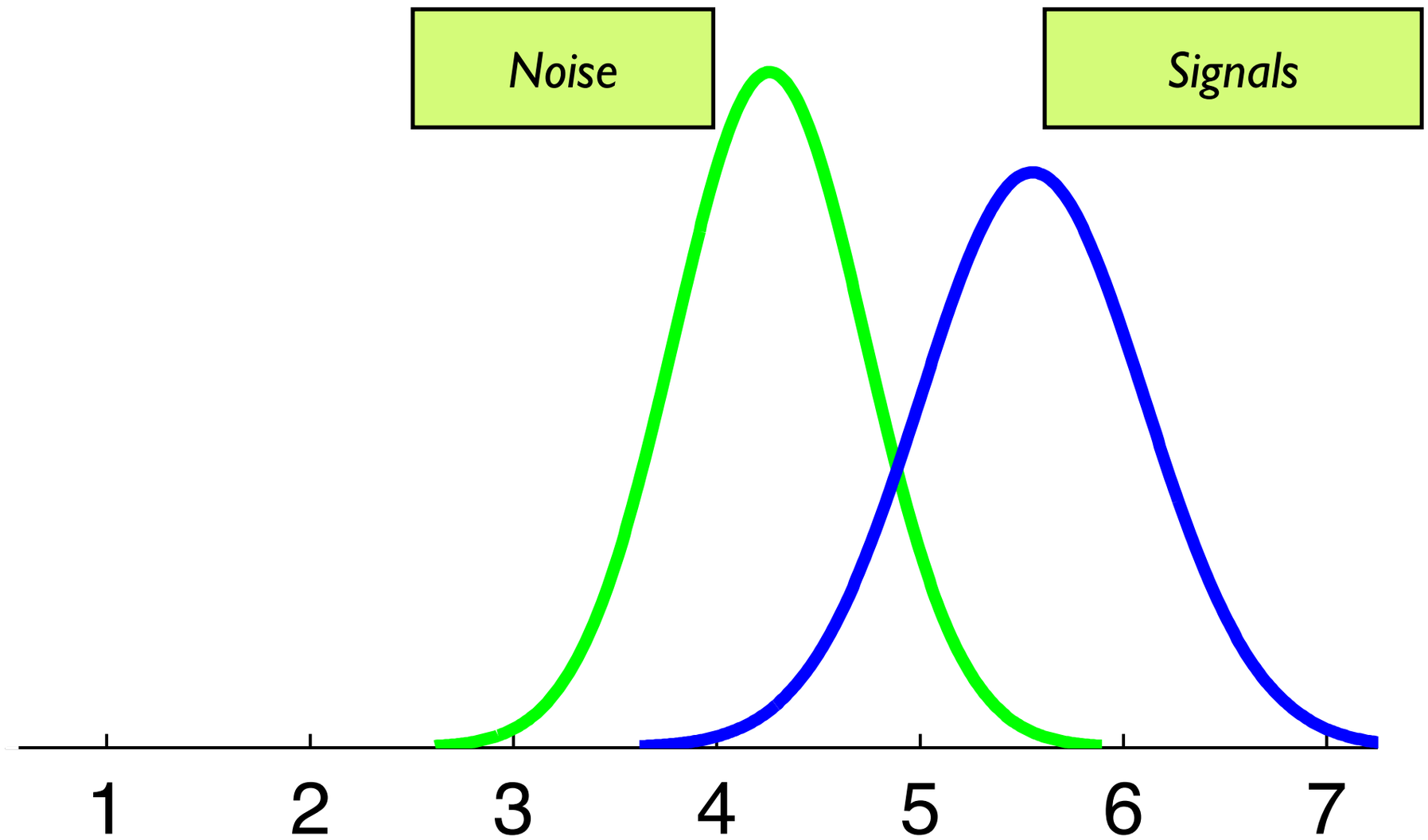
$$SNR = \frac{P_s}{P_N} = \frac{\int s^2(t)dt}{\sigma^2}$$



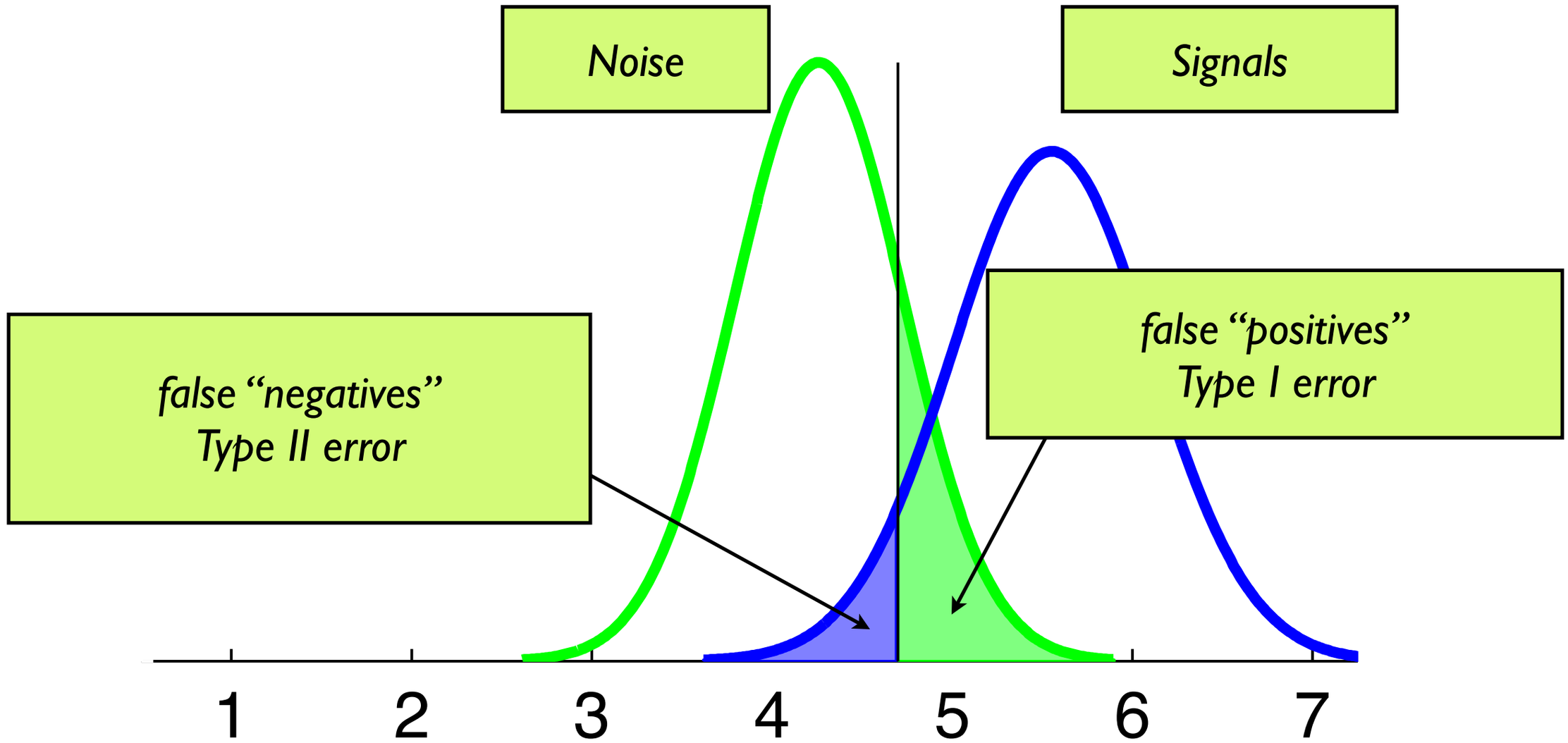
we can also think of these as two categories, e.g. feature x vs not feature x

from Dusenbery, 1992

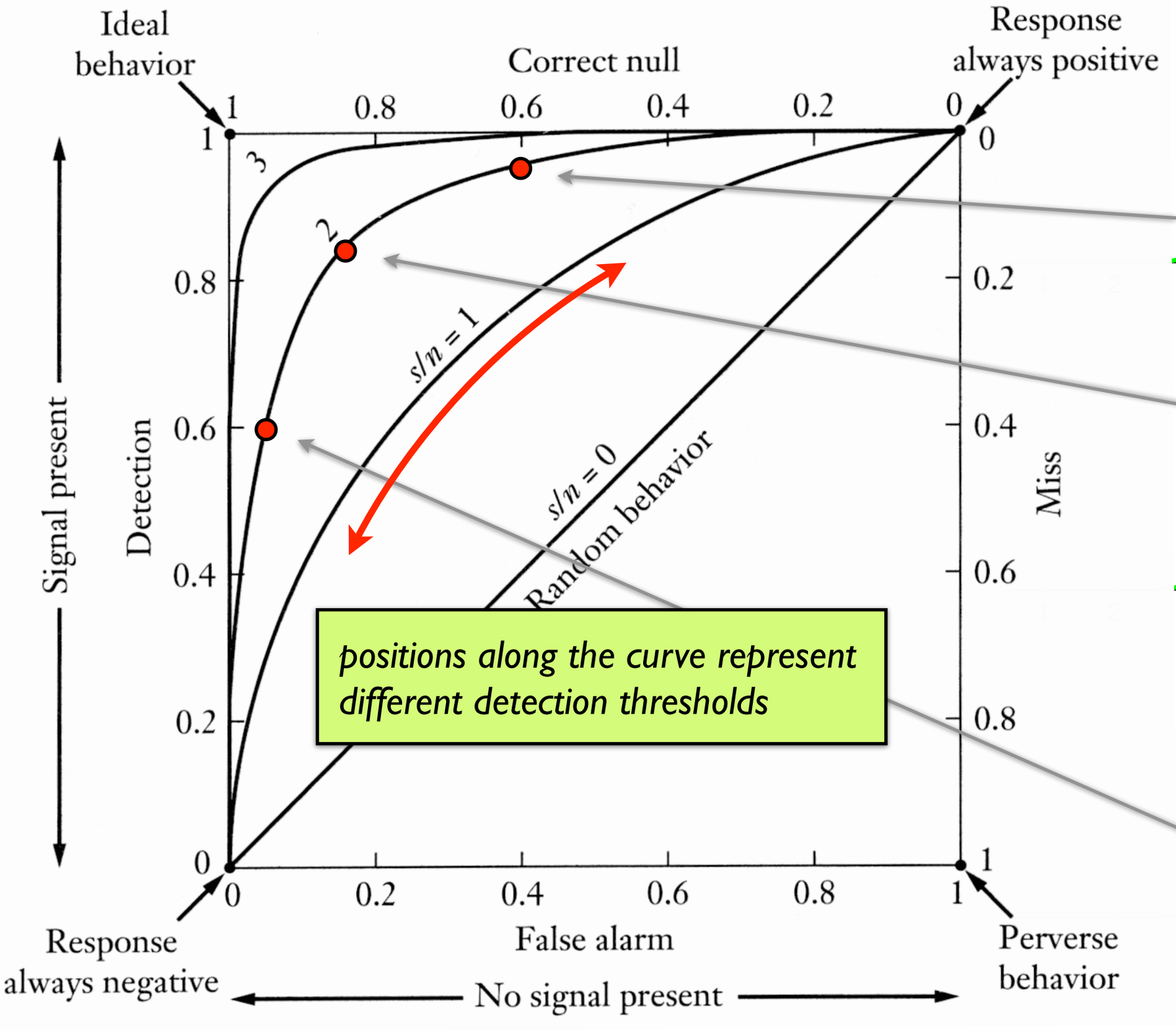
Signals and noise

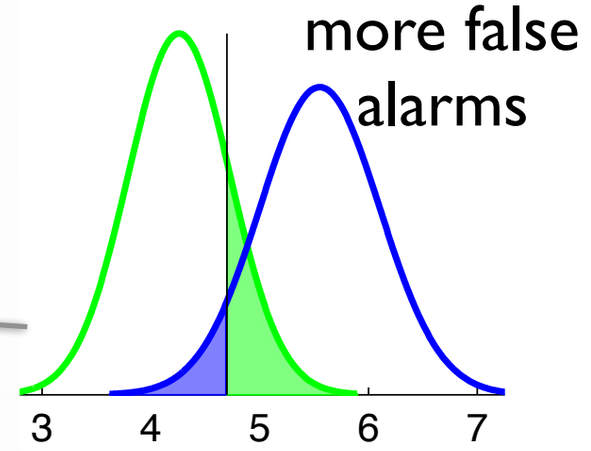


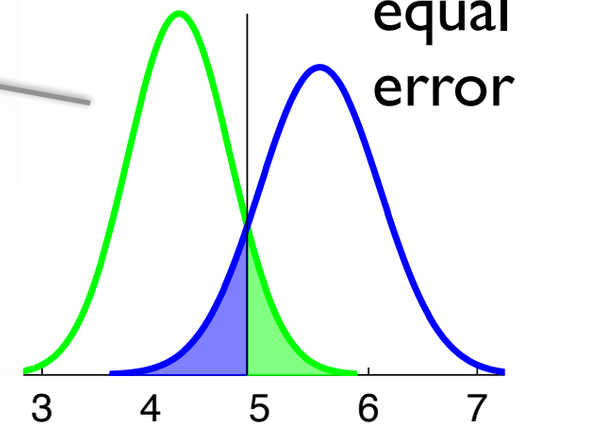
Signals and noise

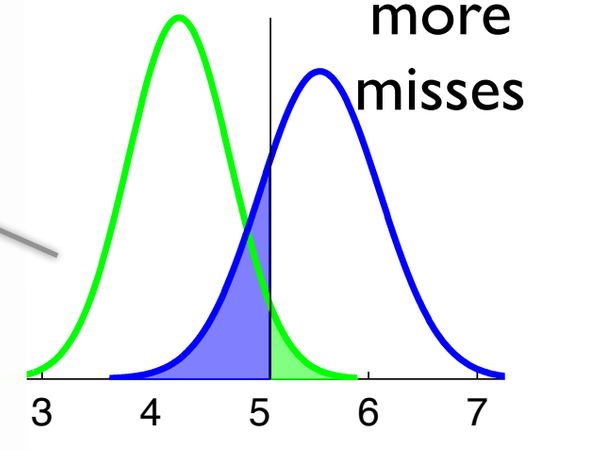


The "receiver operating characteristic" (ROC) curve



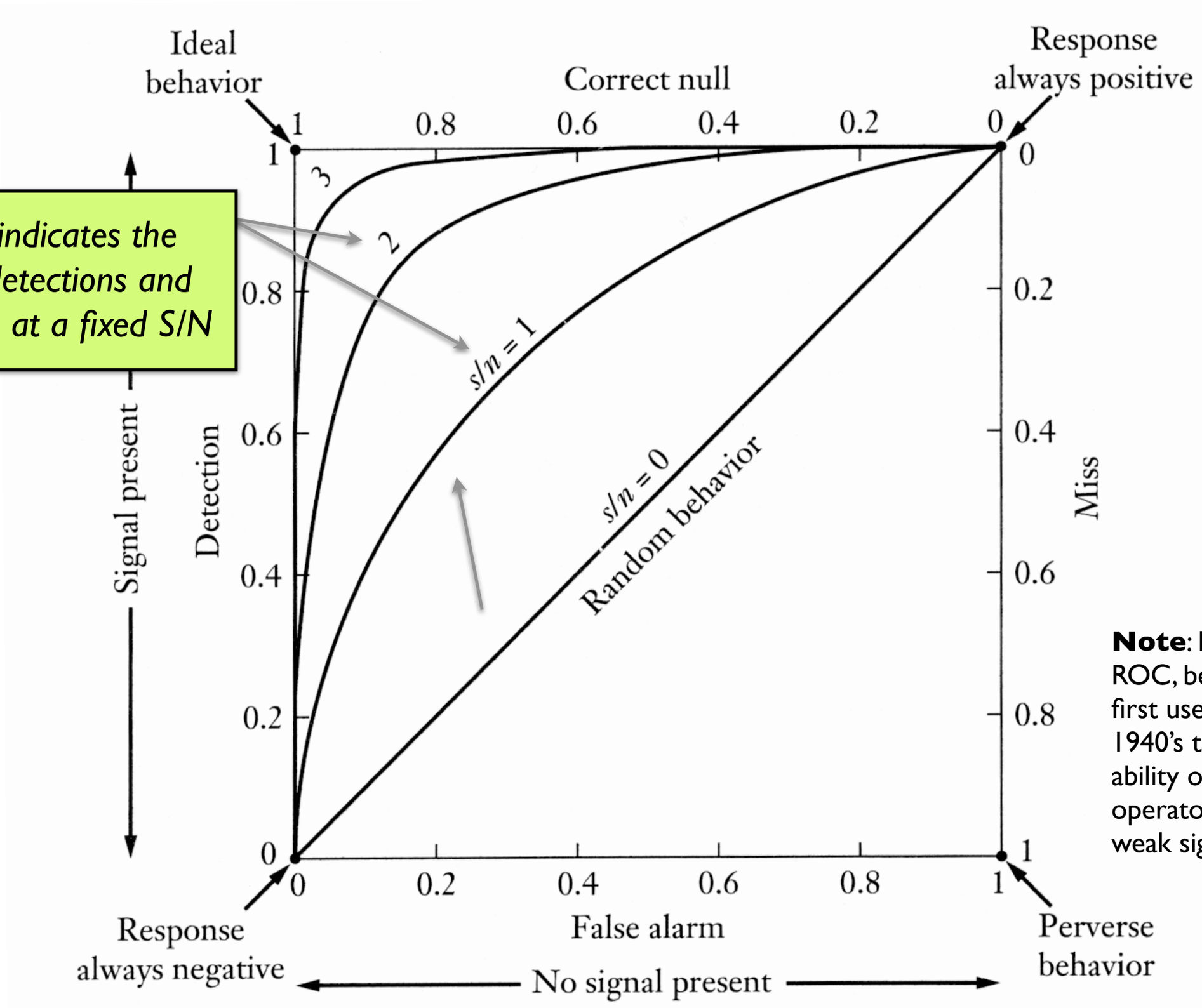






from Dusenbery, 1992

The "receiver operating characteristic" (ROC) curve

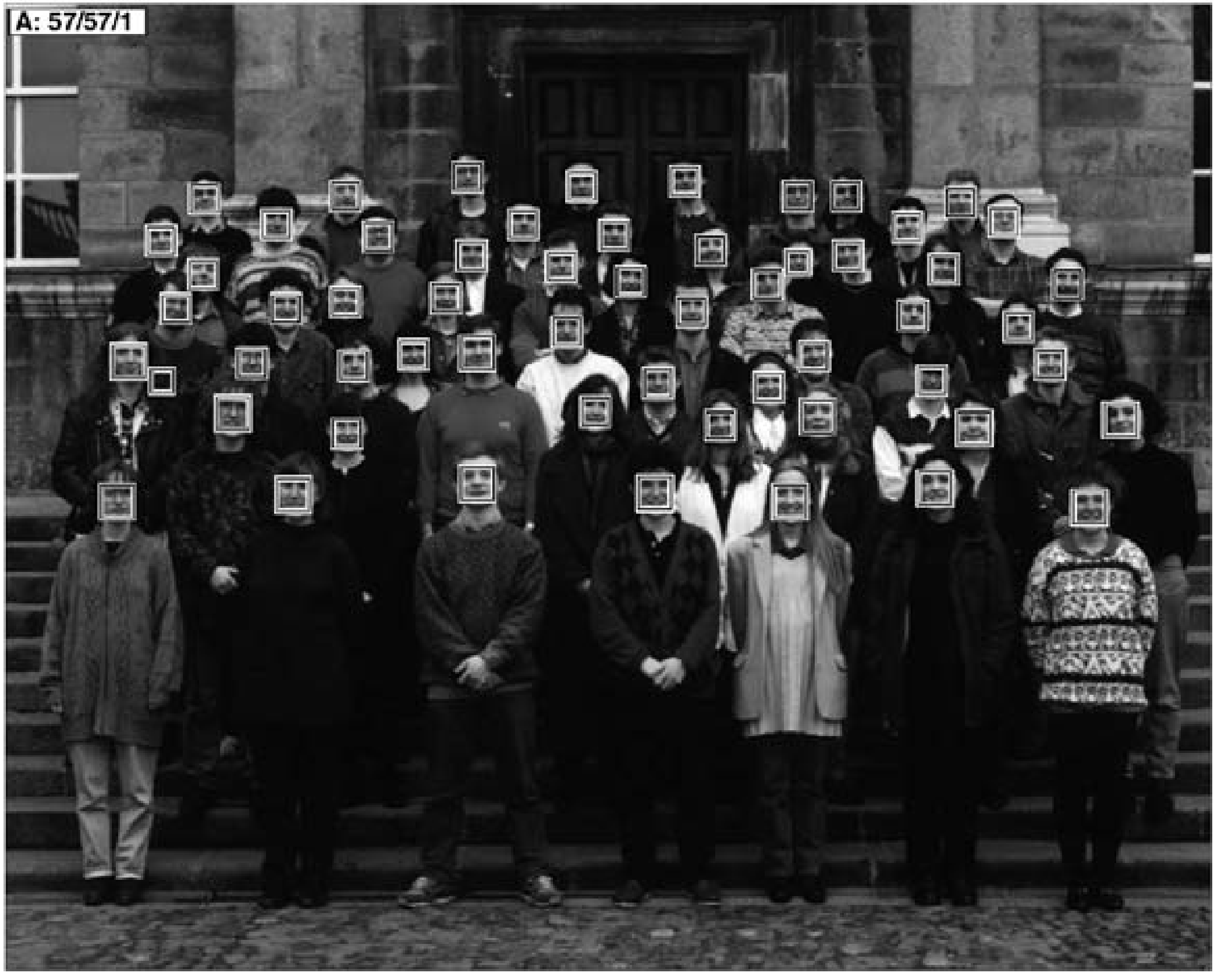


Each curve indicates the fraction of detections and false alarms at a fixed S/N

Note: It's called ROC, because it was first used in the 1940's to asses the ability of radio operators to detect weak signals in noise.

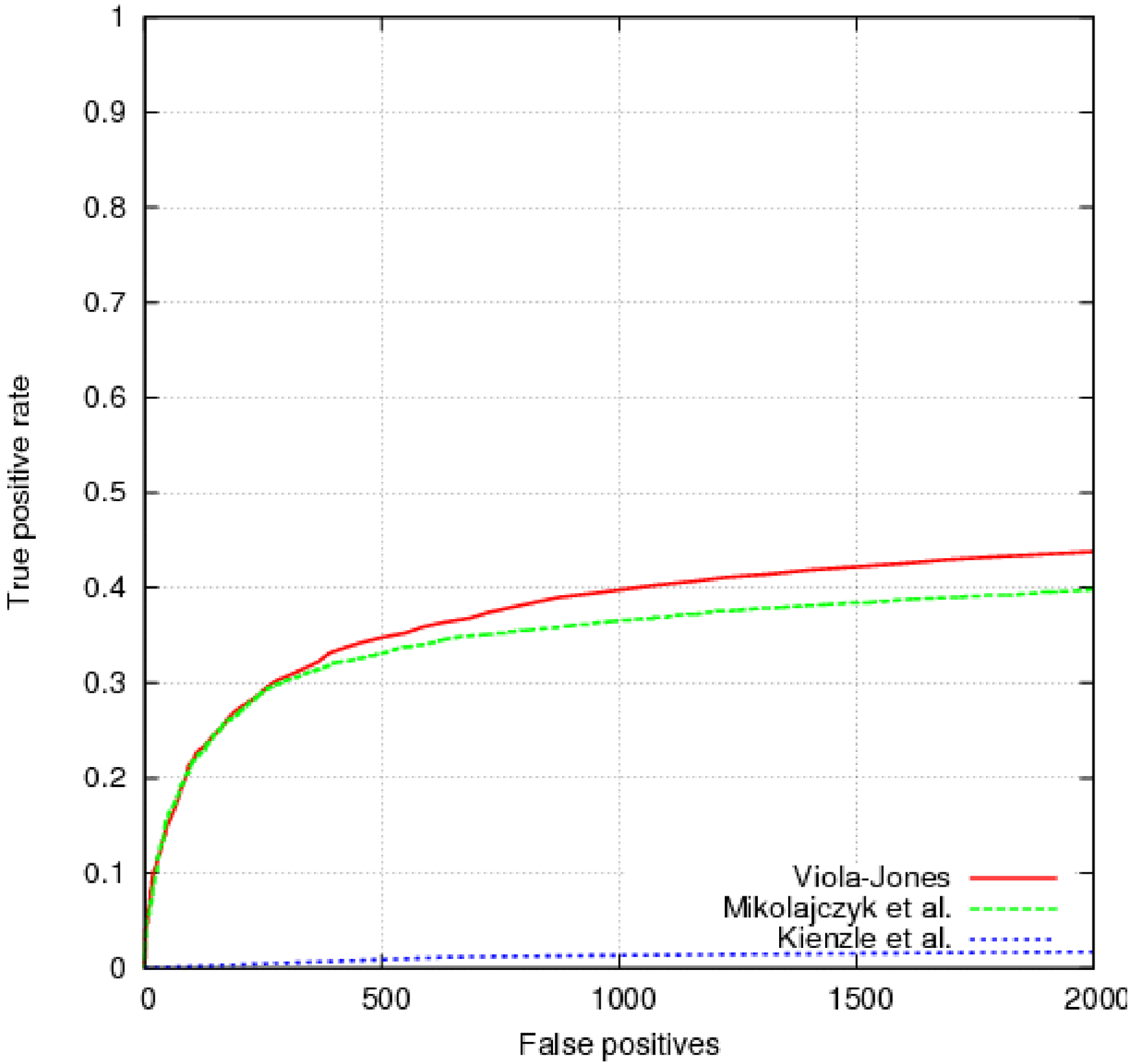
from Dusenbery, 1992

Face detection



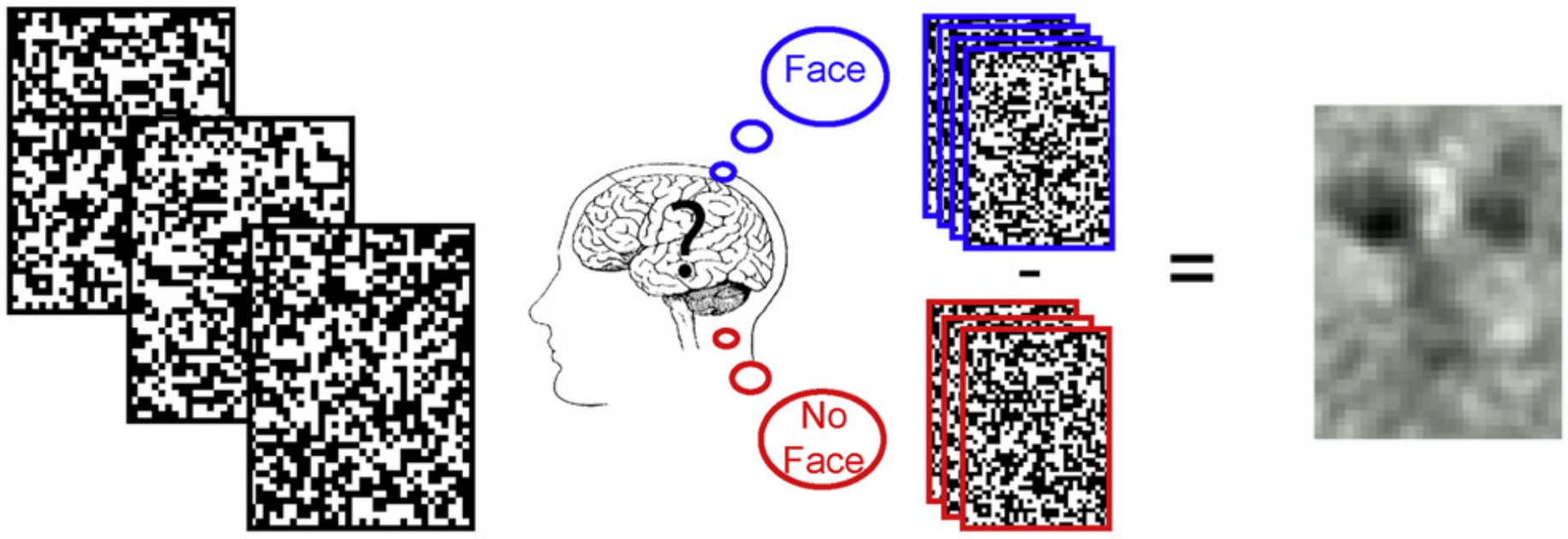
Can you find the false positive?

A ROC curve from a face-detection paper



Which one of these contains a face? (Smith et al, 2012)

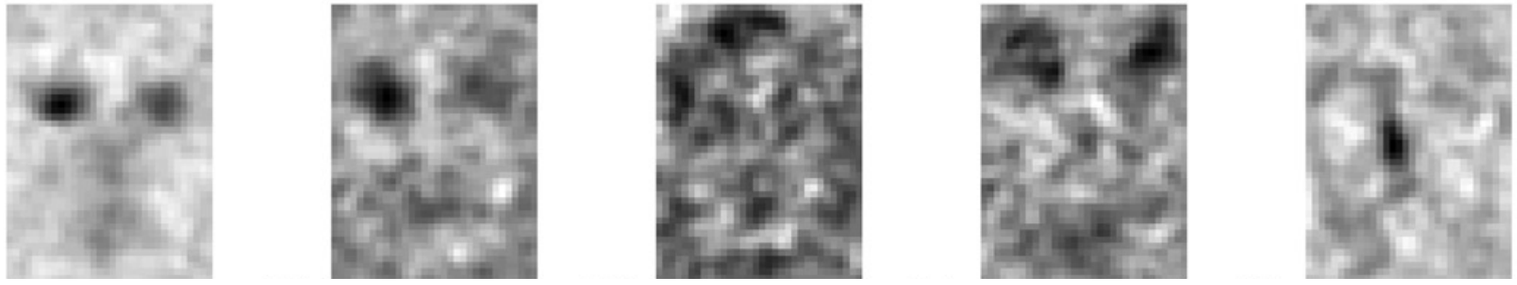
Trick question -
they're all random!



Subject judge 10,500 images.
Told faces are in half the trials.

avg. of "face" patterns minus
avg. of "no face" patterns

results from
5 subjects



Summary

- Linear filtering
- Edge detection
- SNR
- ROC curve

	Real positive	Real negative
Predicted positive	True positive (TP)	False positive (FP)
Predicted negative	False negative (FN)	True negative (TN)

False positive rate (FPR), Fall-out, probability of false alarm

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$$

False negative rate (FNR), Miss rate

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{TP}+\mathrm{FN}}$$

True positive rate (TPR), Recall, Sensitivity, probability of detection

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$

True negative rate (TNR), Specificity (SPC)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}}$$