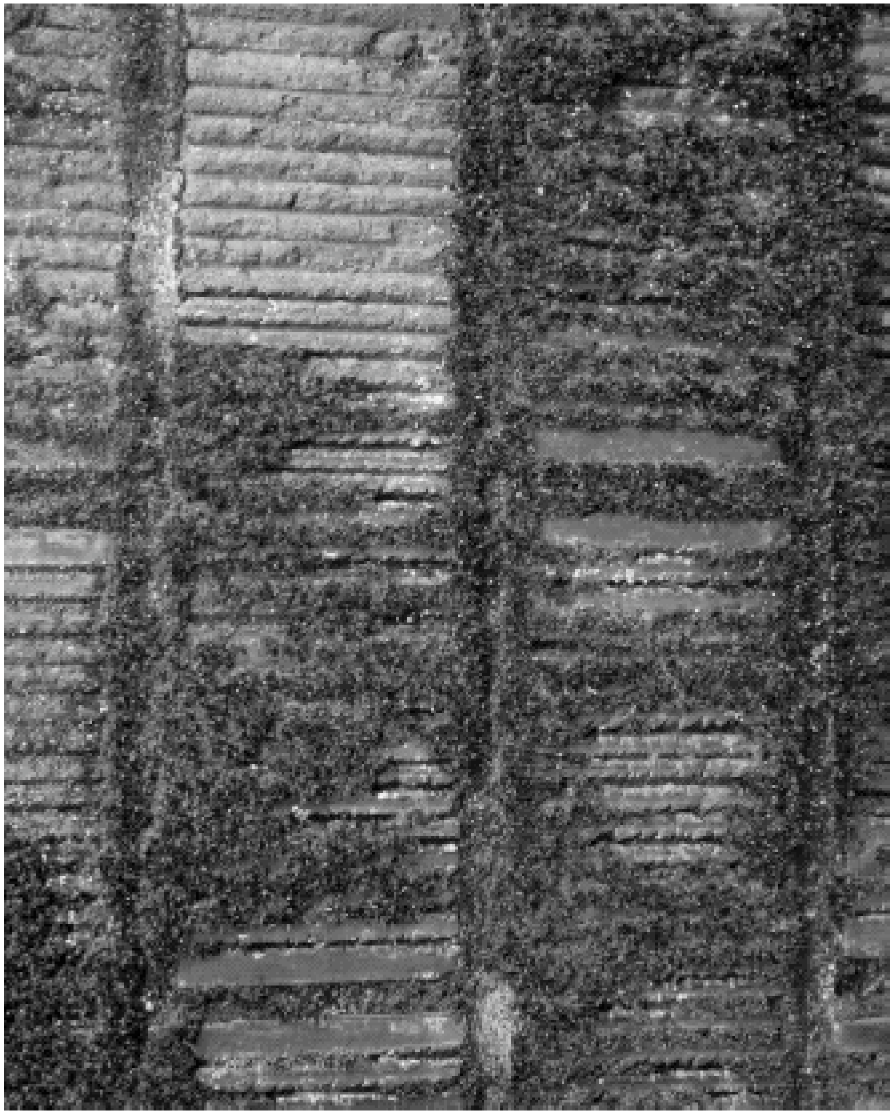
Texture

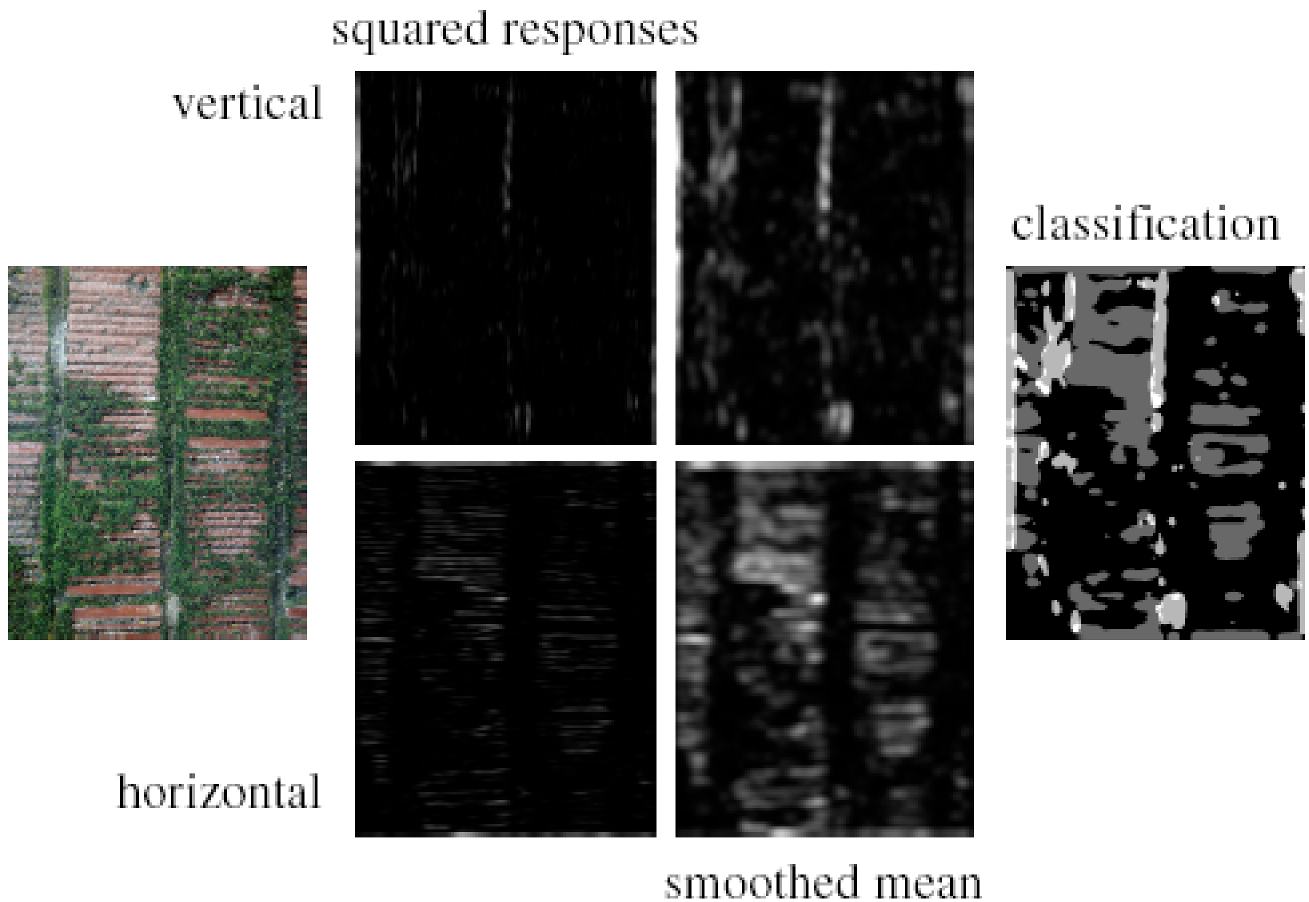
- Key issue: representing texture
  - Texture based matching
    - little is known
  - Texture segmentation
    - key issue: representing texture
  - Texture synthesis
    - useful; also gives some insight into quality of representation
  - Shape from texture
    - cover superficially

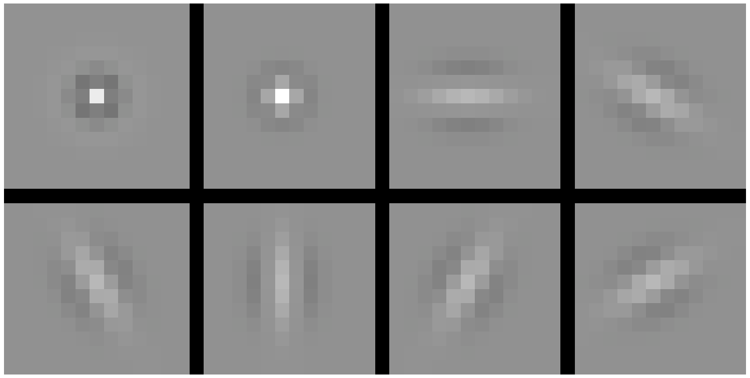


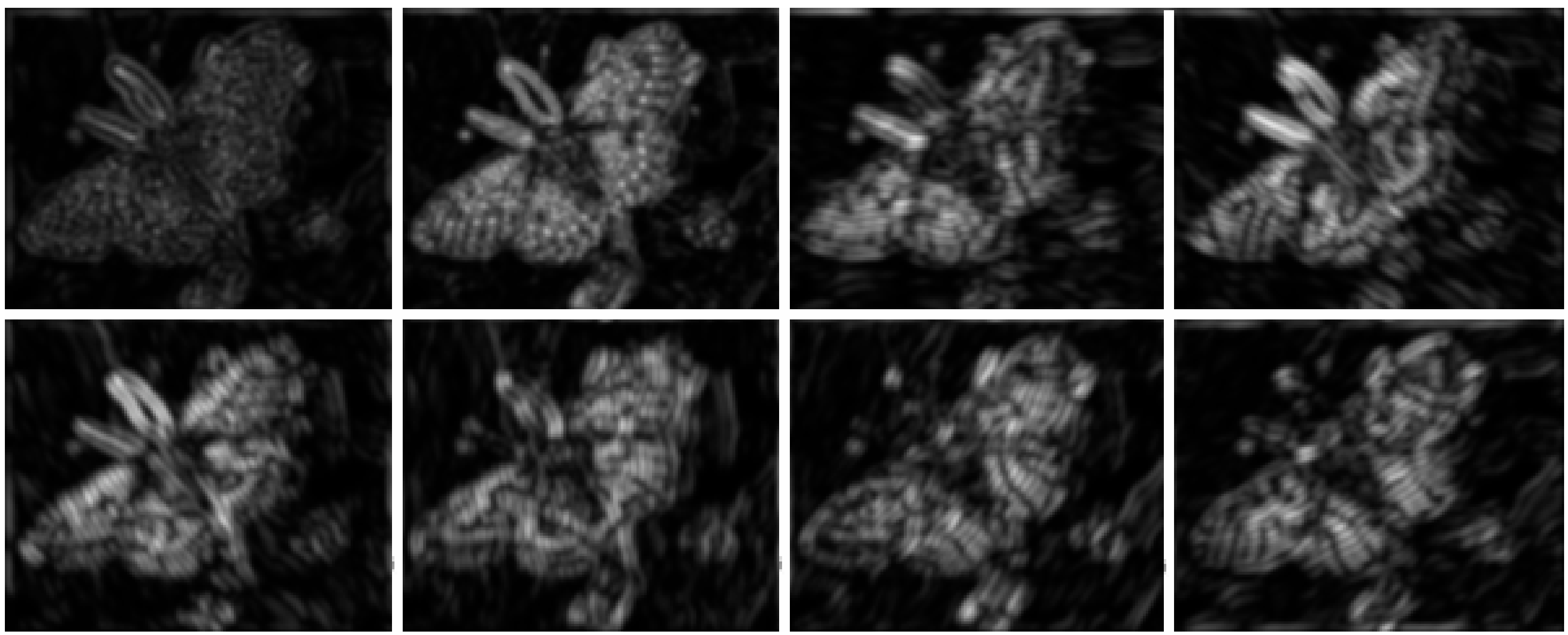
Representing textures

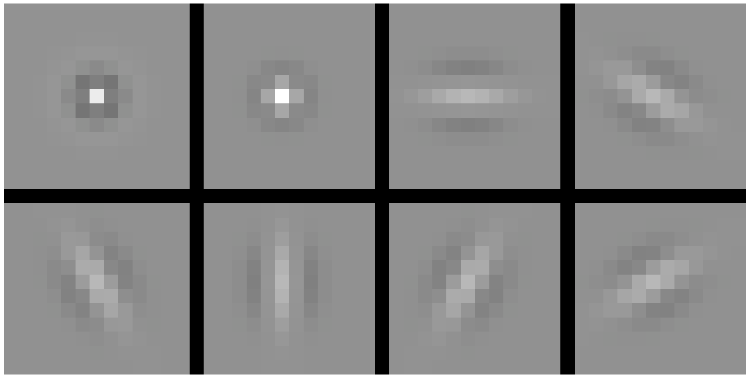
- Textures are made up of quite stylised subelements, repeated in meaningful ways
- Representation:
  - find the subelements, and represent their statistics
- But what are the subelements, and how do we find them?
  - recall normalized correlation
  - find subelements by applying filters, looking at the magnitude of the response

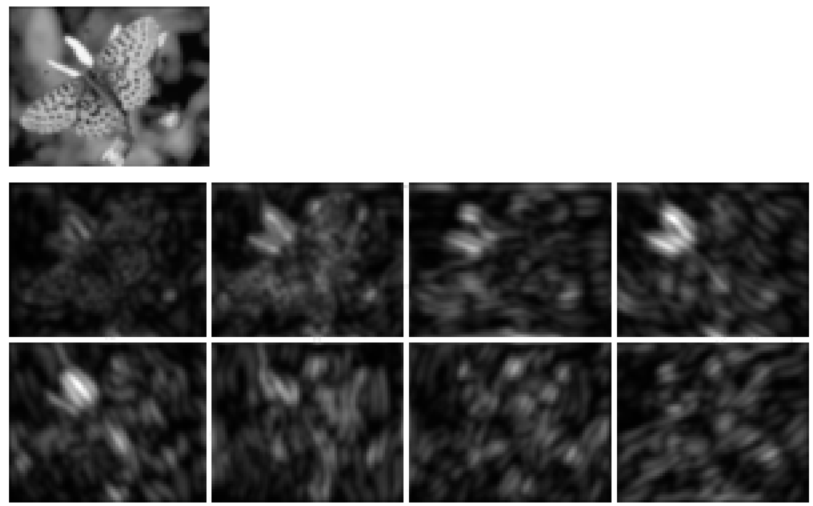
- What filters?
  - experience suggests spots and oriented bars at a variety of different scales
  - details probably don't matter
- What statistics?
  - within reason, the more the merrier.
  - At least, mean and standard deviation
  - better, various conditional histograms.

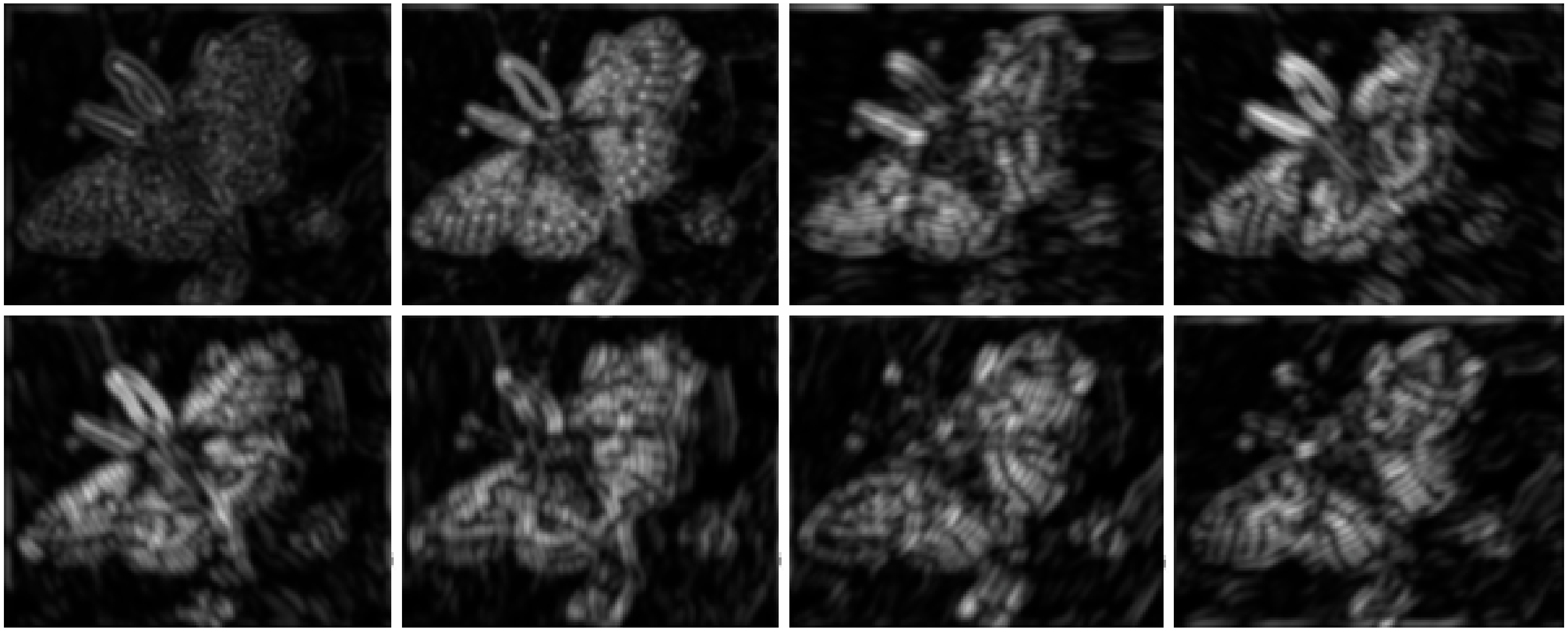

squared responses
vertical
classification
horizontal
smoothed mean

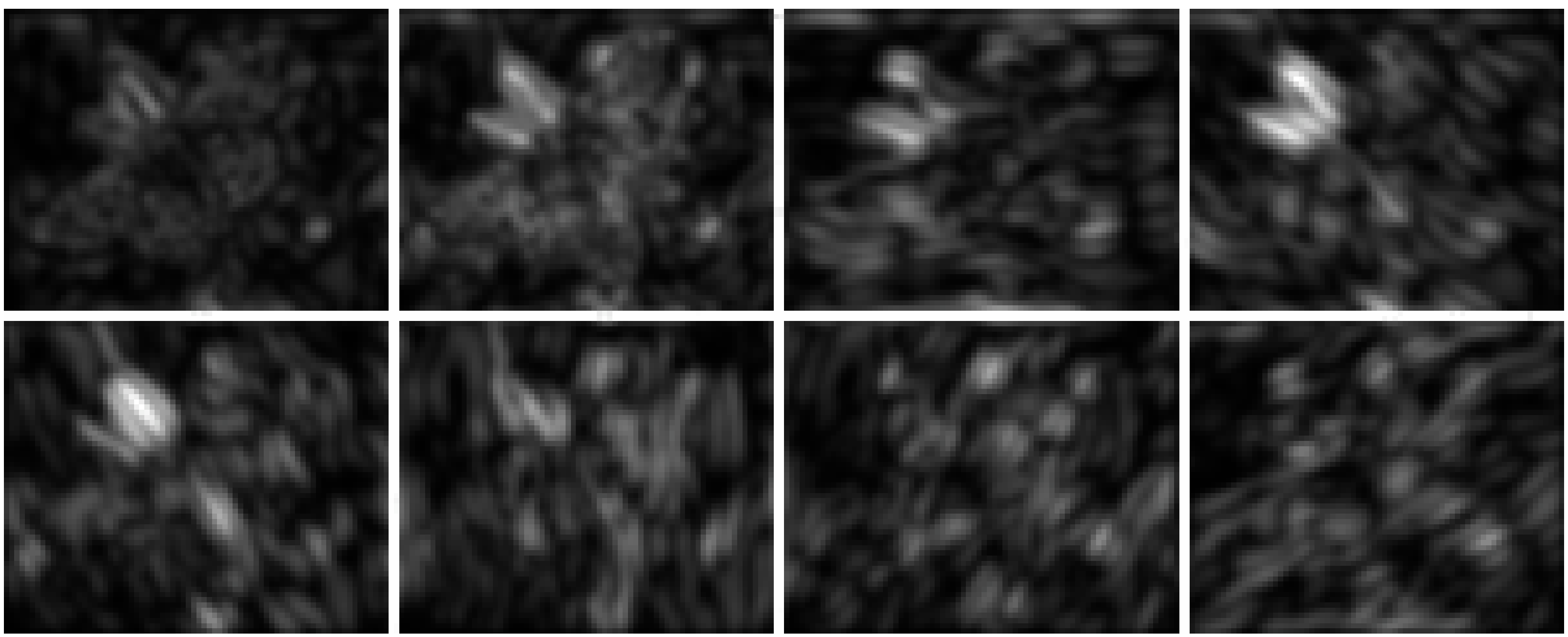


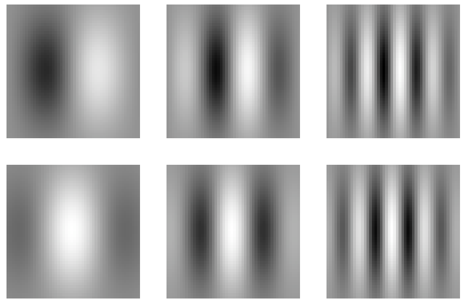






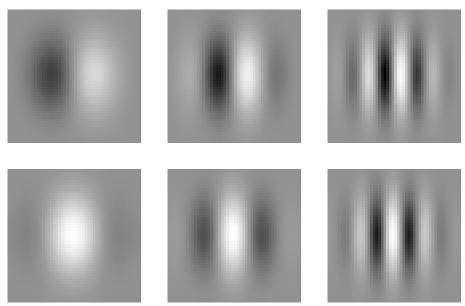


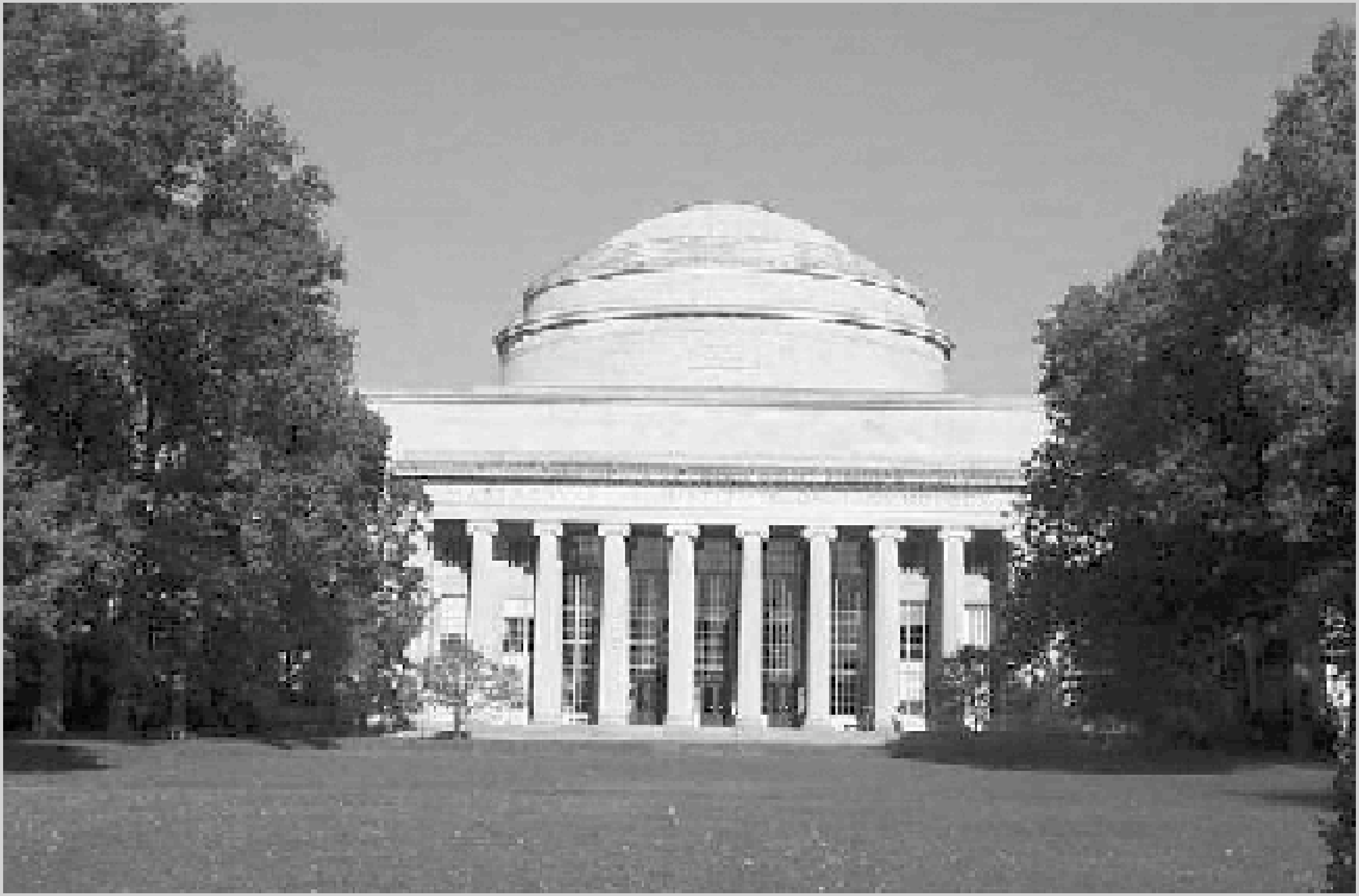




Gabor filters at different scales and spatial frequencies

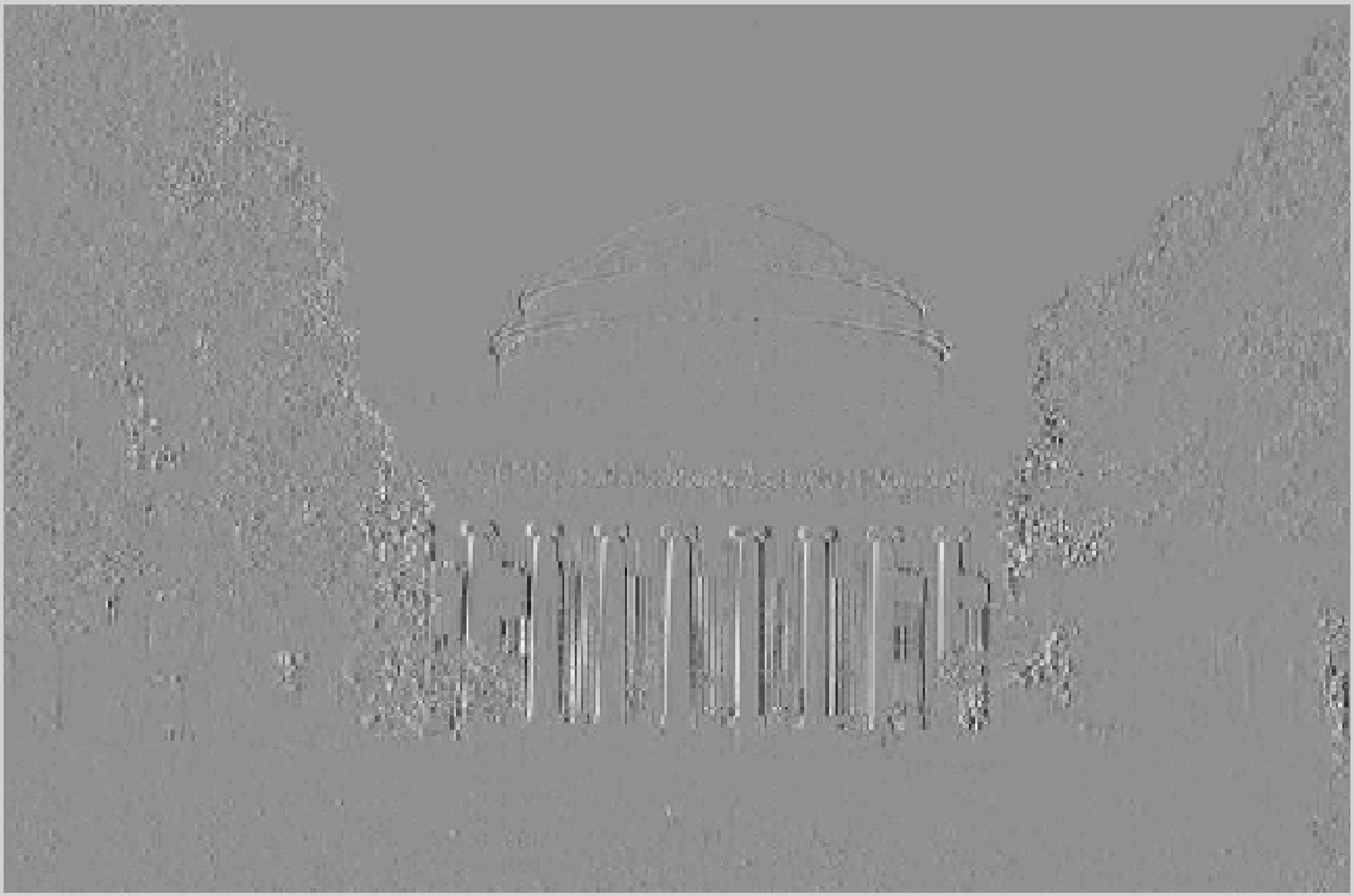
top row shows anti-symmetric (or odd) filters, bottom row the symmetric (or even) filters.





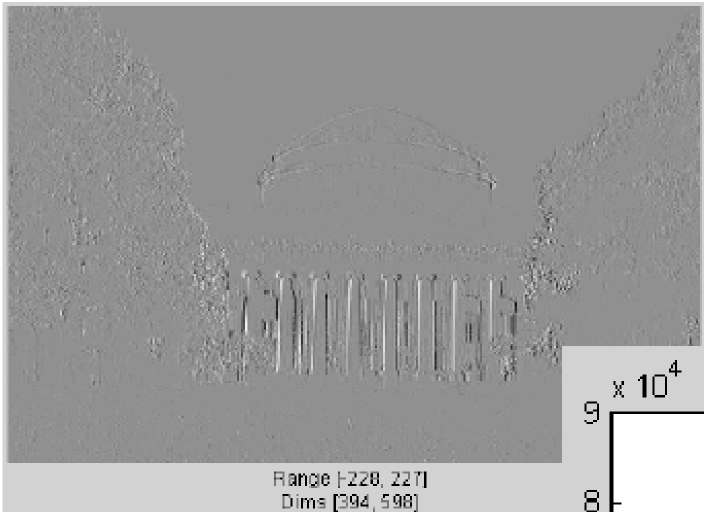
Range [0, 255]
Dims [394, 599]

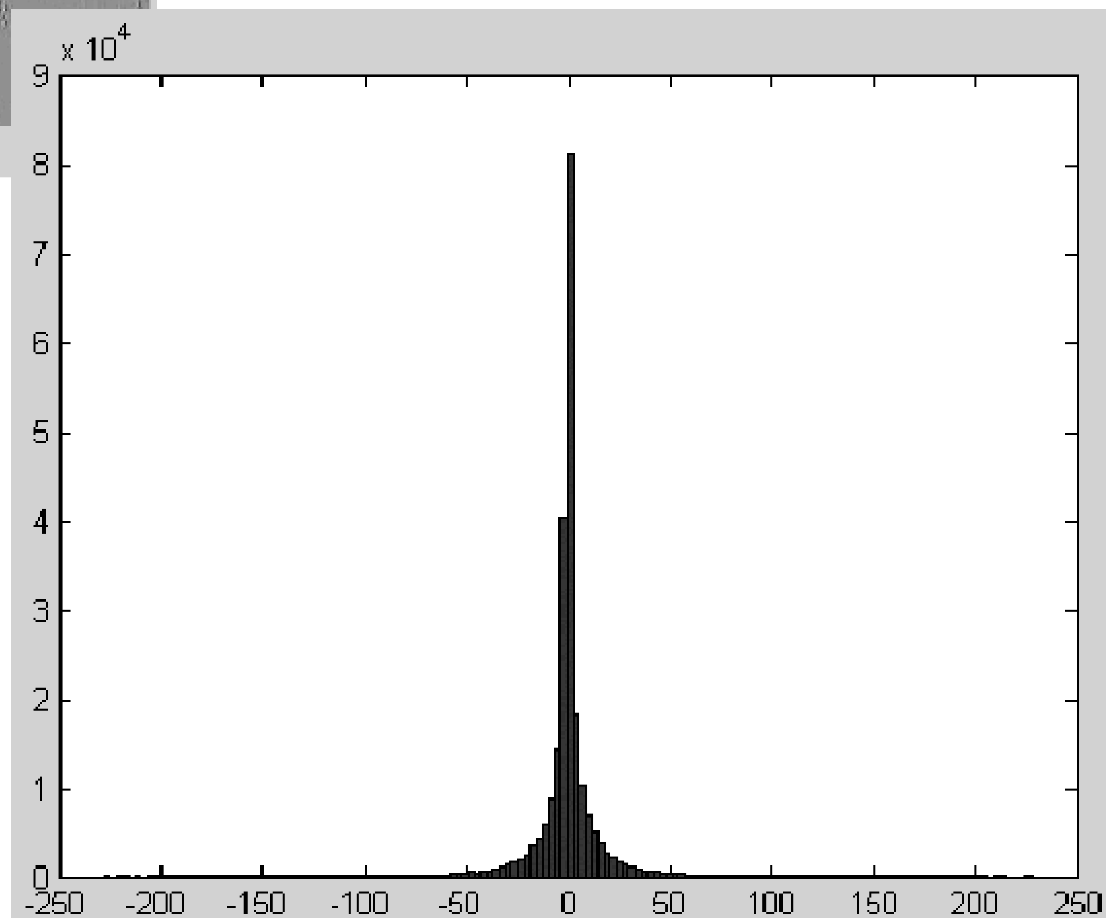
bandpass filtered image



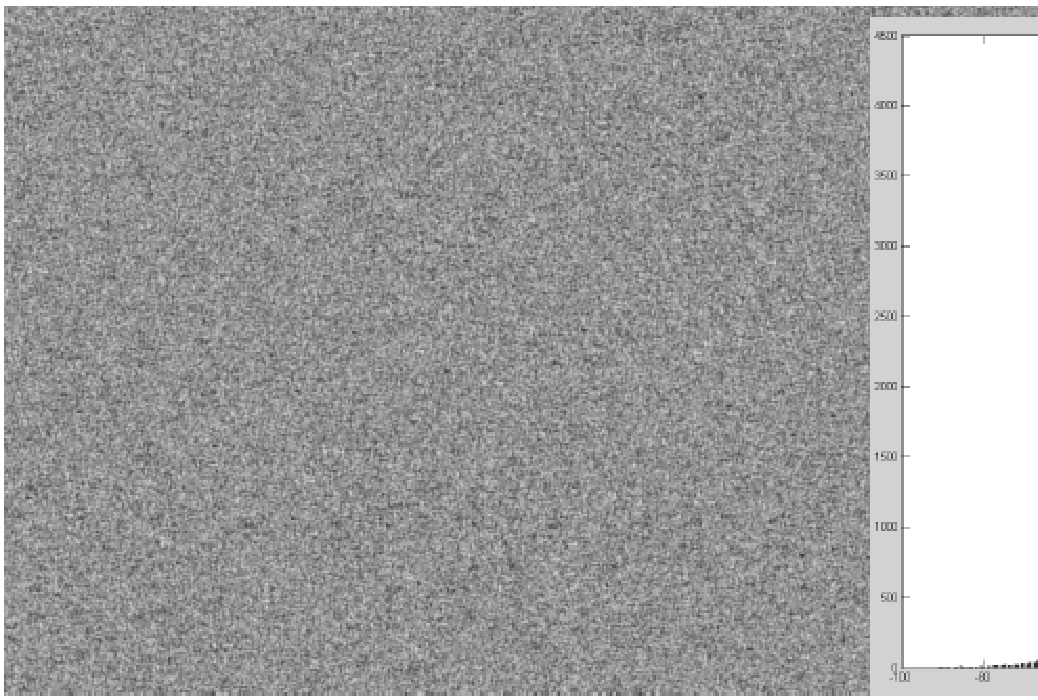
Range [-228, 227]
Dims [394, 598]

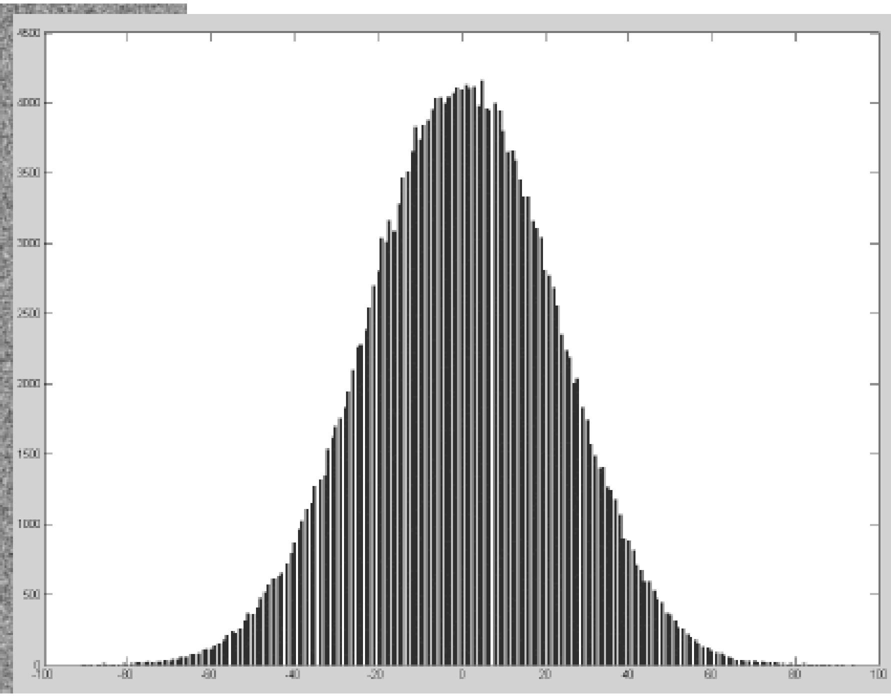
bandpassed representation
image histogram



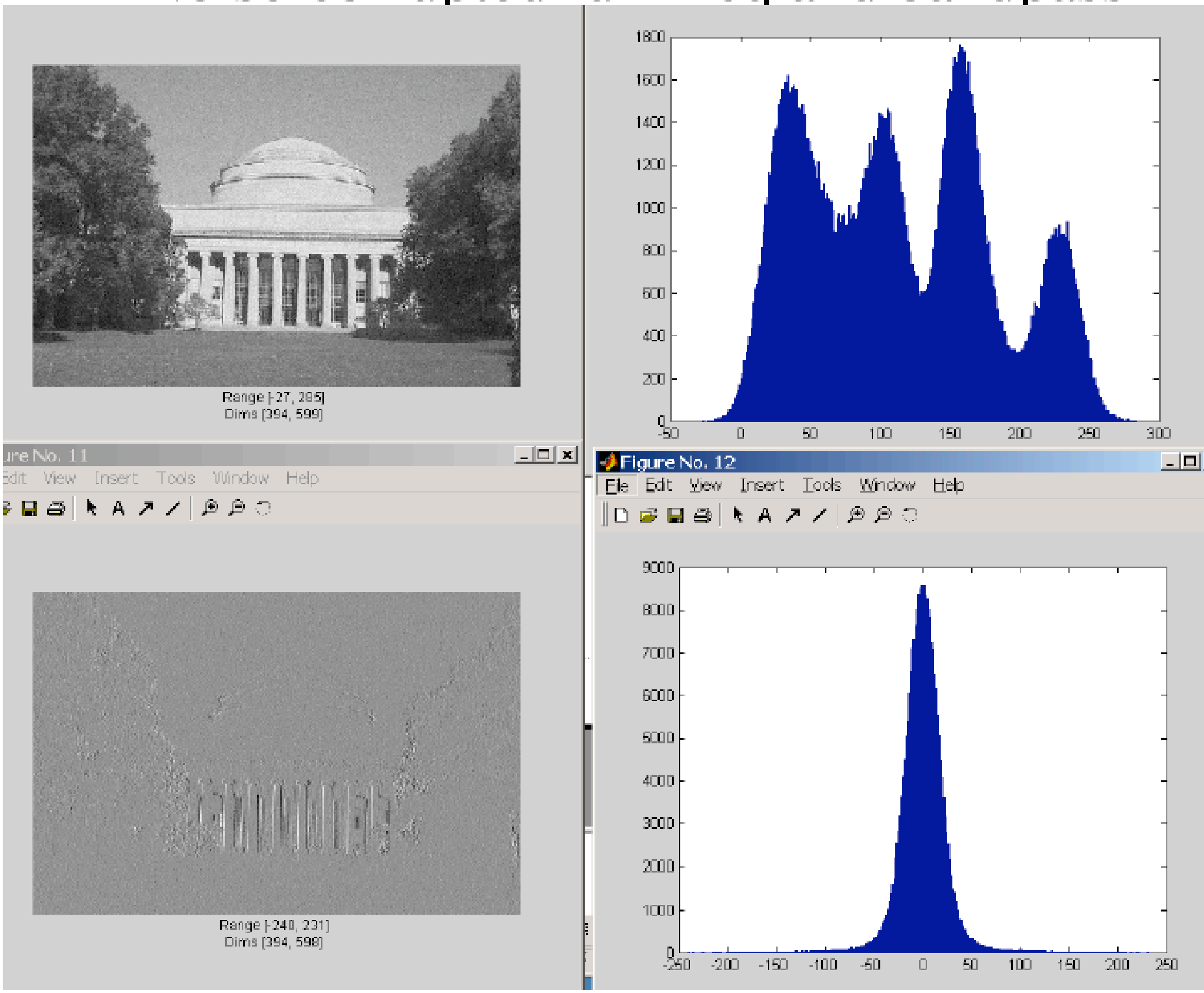


Bandpass domain noise image and histogram





Noise-corrupted full-freq and bandpass images



The Gaussian pyramid

- Smooth with gaussians, because
  - a gaussian*gaussian=another gaussian
- Synthesis
  - smooth and sample
- Analysis
  - take the top image
- Gaussians are low pass filters, so repn is redundant

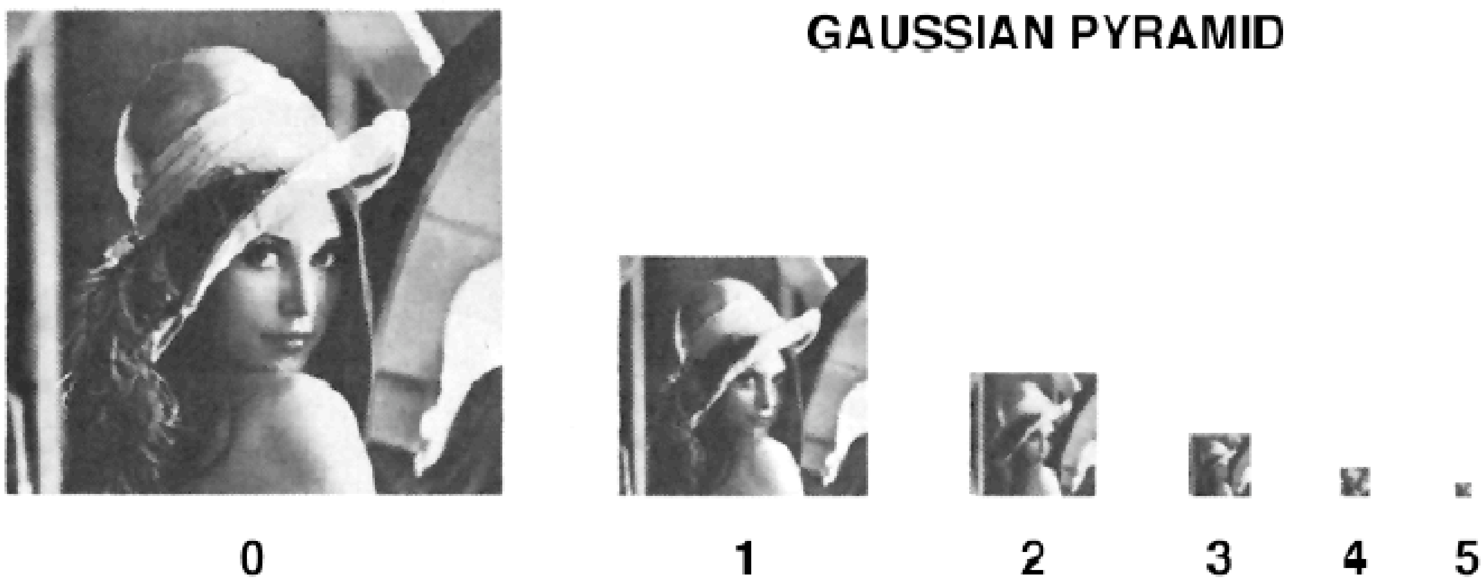
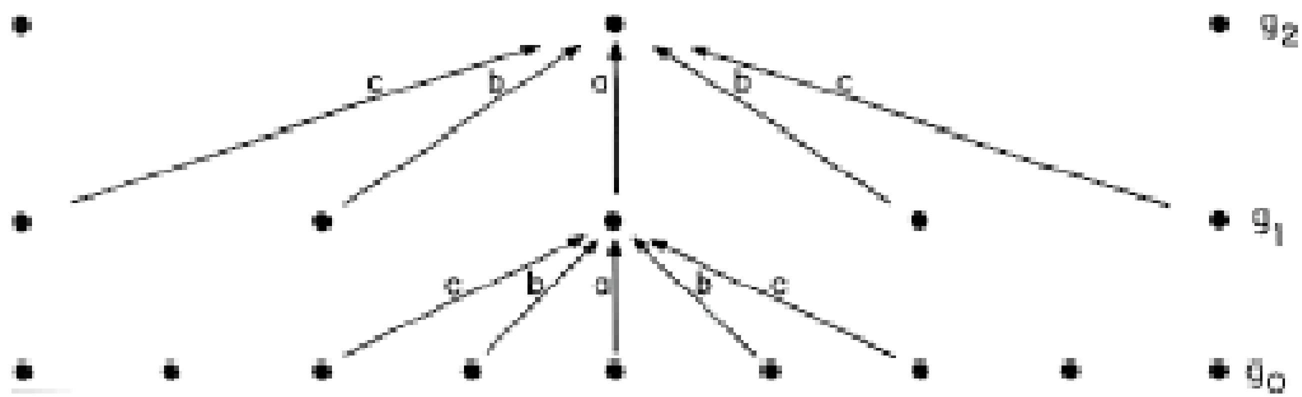


Fig. 4. First six levels of the Gaussian pyramid for the "Lady" image The original image, level 0, meusures 257 by 257 pixels and each higher level array is roughly half the dimensdons of its predecessor. Thus, level 5 measures just 9 by 9 pixels.

512 256 128 64 32 16 8

The computational advantage of pyramids

GAUSSIAN PYRAMID



g_0 = IMAGE

g_L = REDUCE [g_{L-1}]

Fig 1. A one-dimensional graphic representation of the process which generates a Gaussian pyramid Each row of dots represents nodes within a level of the pyramid. The value of each node in the zero level is just the gray level of a corresponding image pixel. The value of each node in a high level is the weighted average of node values in the next lower level. Note that node spacing doubles from level to level, while the same weighting pattern or "generating kernel" is used to generate all levels.

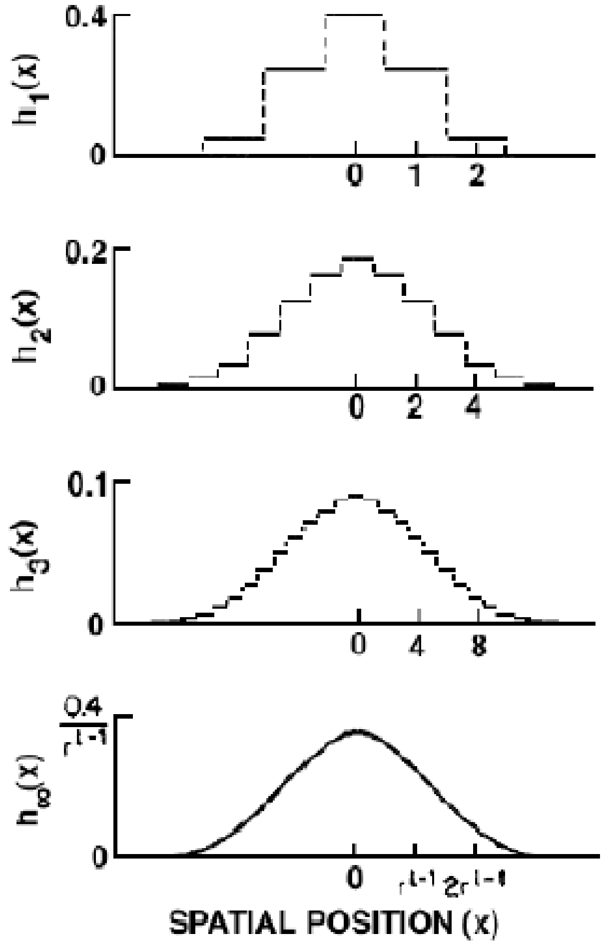
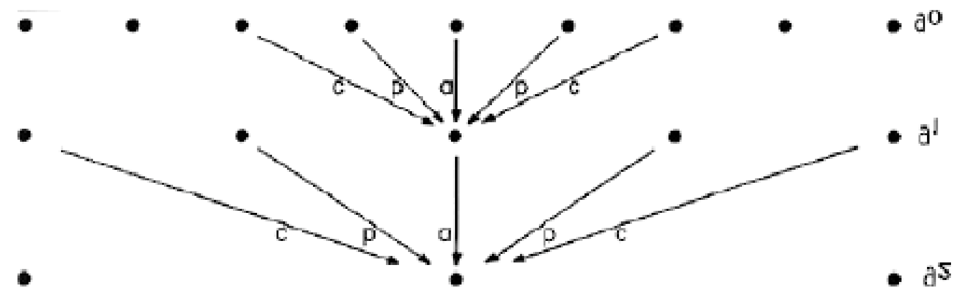


Fig. 2. The equivalent weighting functions $h_l(x)$ for nodes in levels 1, 2, 3, and infinity of the Gaussian pyramid. Note that axis scales have been adjusted by factors of 2 to aid comparison Here the parameter a of the generating kernel is 0.4, and the resulting equivalent weighting functions closely resemble the Gaussian probability density functions.



Simoncelli and Adelson, in “Subband coding”, Kluwer, 1990.

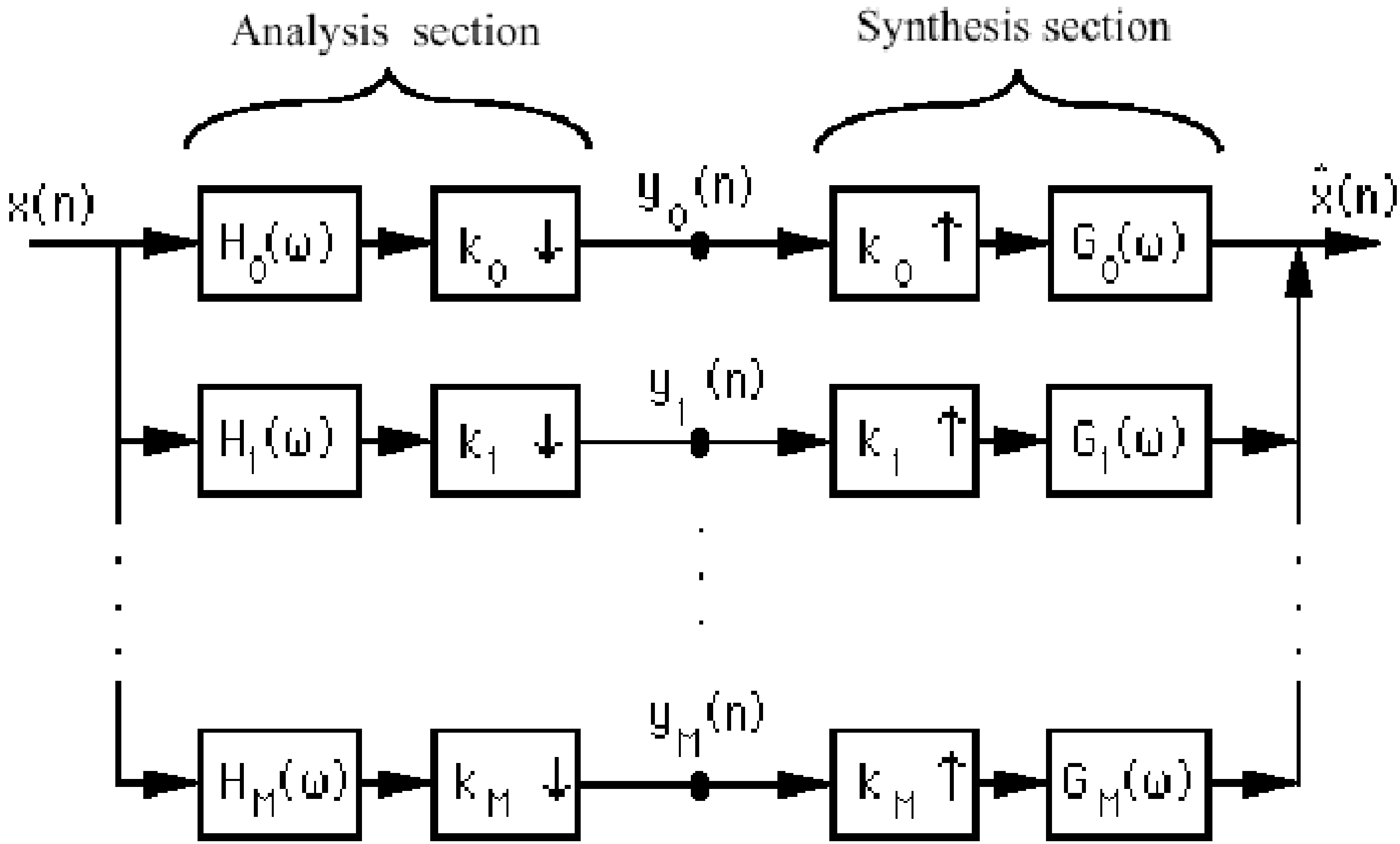


Figure 4.2: An analysis/synthesis filter bank.

Simoncelli and Adelson, in “Subband coding”, Kluwer, 1990.

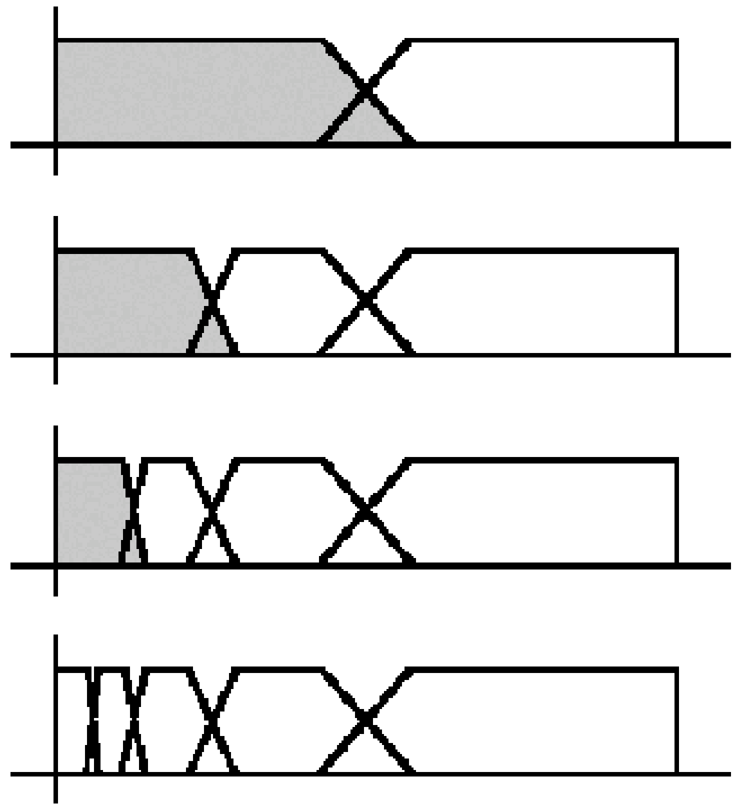
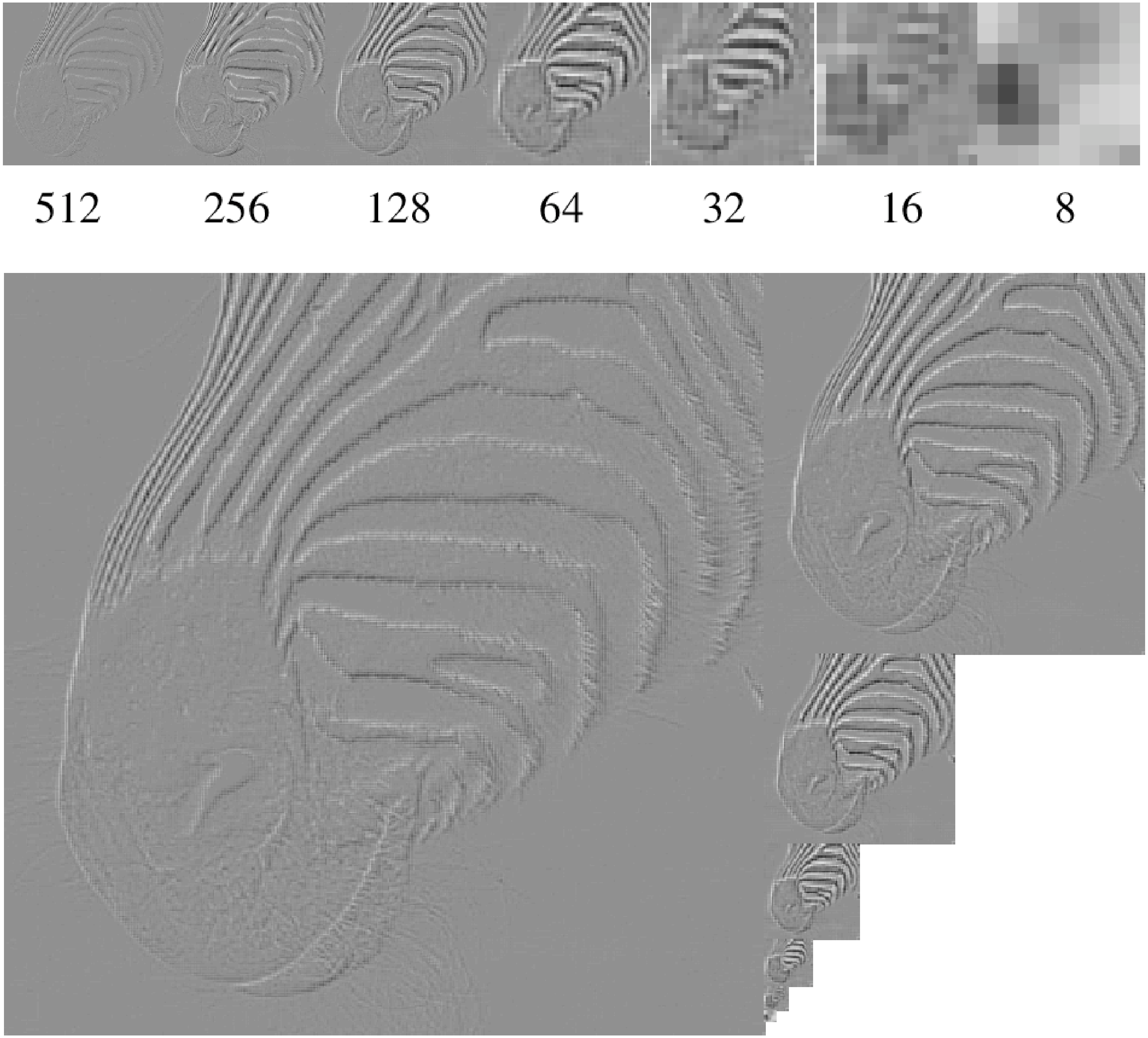


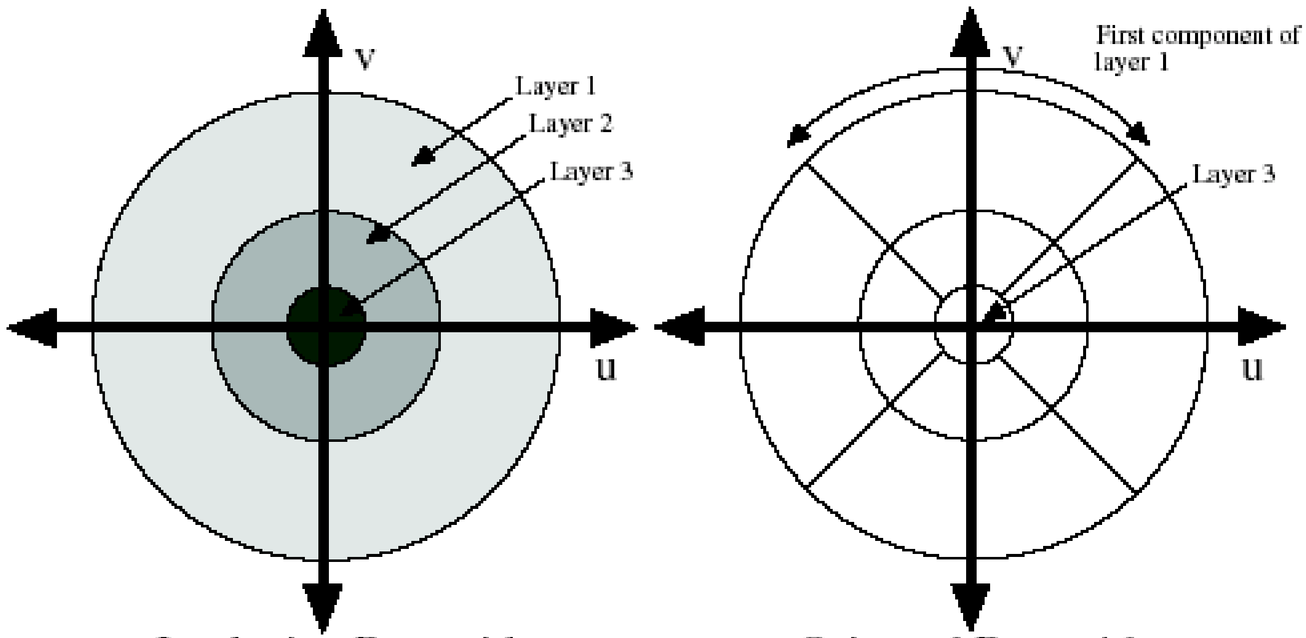
Figure 4.4: Octave band splitting produced by a four-level pyramid cascade of a two-band A/S system. The top picture represents the splitting of the two-band A/S system. Each successive picture shows the effect of re-applying the system to the lowpass subband (indicated in grey) of the previous picture. The bottom picture gives the final four-level partition of the frequency domain. All frequency axes cover the range from 0 to π.

The Laplacian Pyramid

- Synthesis
  - preserve difference between upsampled Gaussian pyramid level and Gaussian pyramid level
  - band pass filter - each level represents spatial frequencies (largely) unrepresented at other levels
- Analysis
  - reconstruct Gaussian pyramid, take top layer

512 256 128 64 32 16 8


512
256
128
64
32
16
8

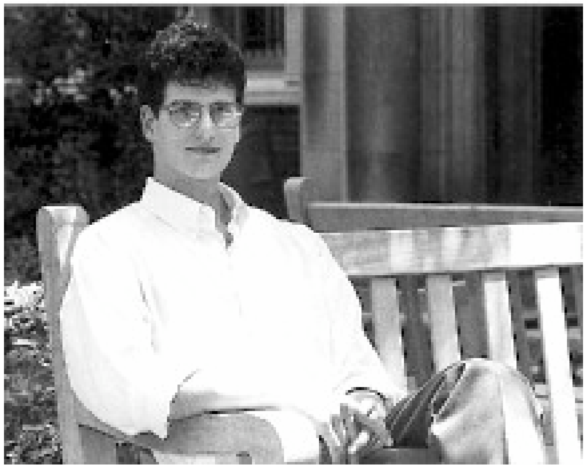


Laplacian Pyramid

Oriented Pyramid

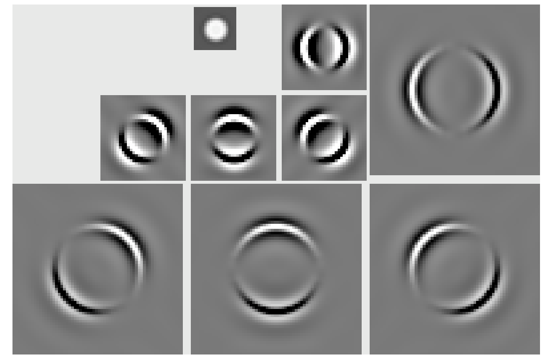
Oriented pyramids

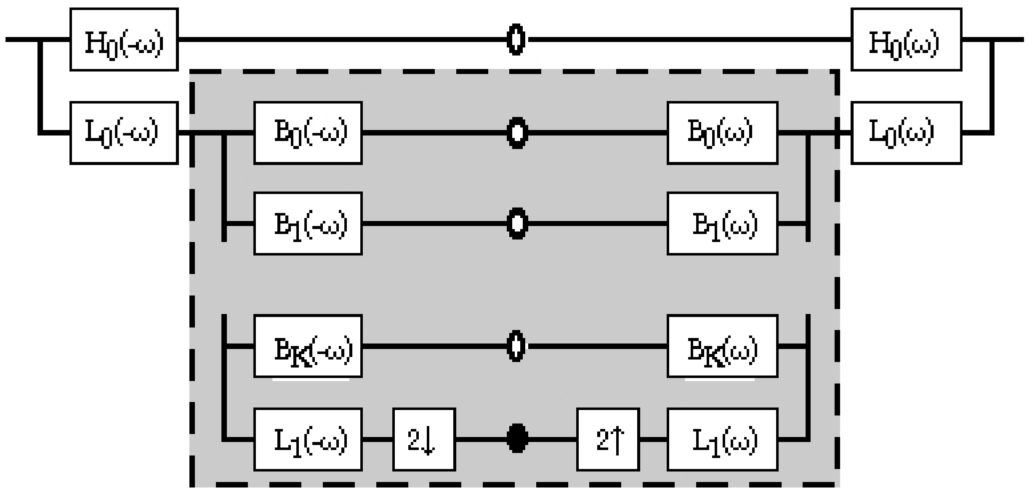
- Laplacian pyramid is orientation independent
- Apply an oriented filter to determine orientations at each layer
  - by clever filter design, we can simplify synthesis
  - this represents image information at a particular scale and orientation



Steerable Pyramids

http://www.cis.upenn.edu/~eero/steerpyr.html





Matlab resources for pyramids (with tutorial)

http://www.cns.nyu.edu/~eero/software.html

Laboratory for Computational Vision

Home | People | Research | Publications | Software

Publicly Available Software Packages

- Texture Analysis/Synthesis - Matlab code is available for analyzing and synthesizing visual textures. README | Contents | ChangeLog | Source code (UNIX/PC, gzip'ed tar file)
- EPWIC - Embedded Progressive Wavelet Image Coder. C source code available.
- **matlabPyrTools** - Matlab source code for multi-scale image processing. Includes tools for building and manipulating Laplacian pyramids, QMF/Wavelets, and steerable pyramids. Data structures are compatible with the Matlab wavelet toolbox, but the convolution code (in C) is faster and has many boundary-handling options. README, Contents, Modification list, UNIX/PC source or Macintosh source.
- The Steerable Pyramid, an (approximately) translation- and rotation-invariant multi-scale image decomposition. MatLab (see above) and C implementations are available.
- Computational Models of cortical neurons. Macintosh program available.
- EPIC - Efficient Pyramid (Wavelet) Image Coder. C source code available.
- OBVIUS [Object-Based Vision & Image Understanding System]: README / ChangeLog / Doc (225k) / Source Code (2.25M).
- CL-SHELL [Gnu Emacs <-> Common Lisp Interface]: README / Change Log / Source Code (119k).

Simoncelli and Adelson, in “Subband coding”, Kluwer, 1990.

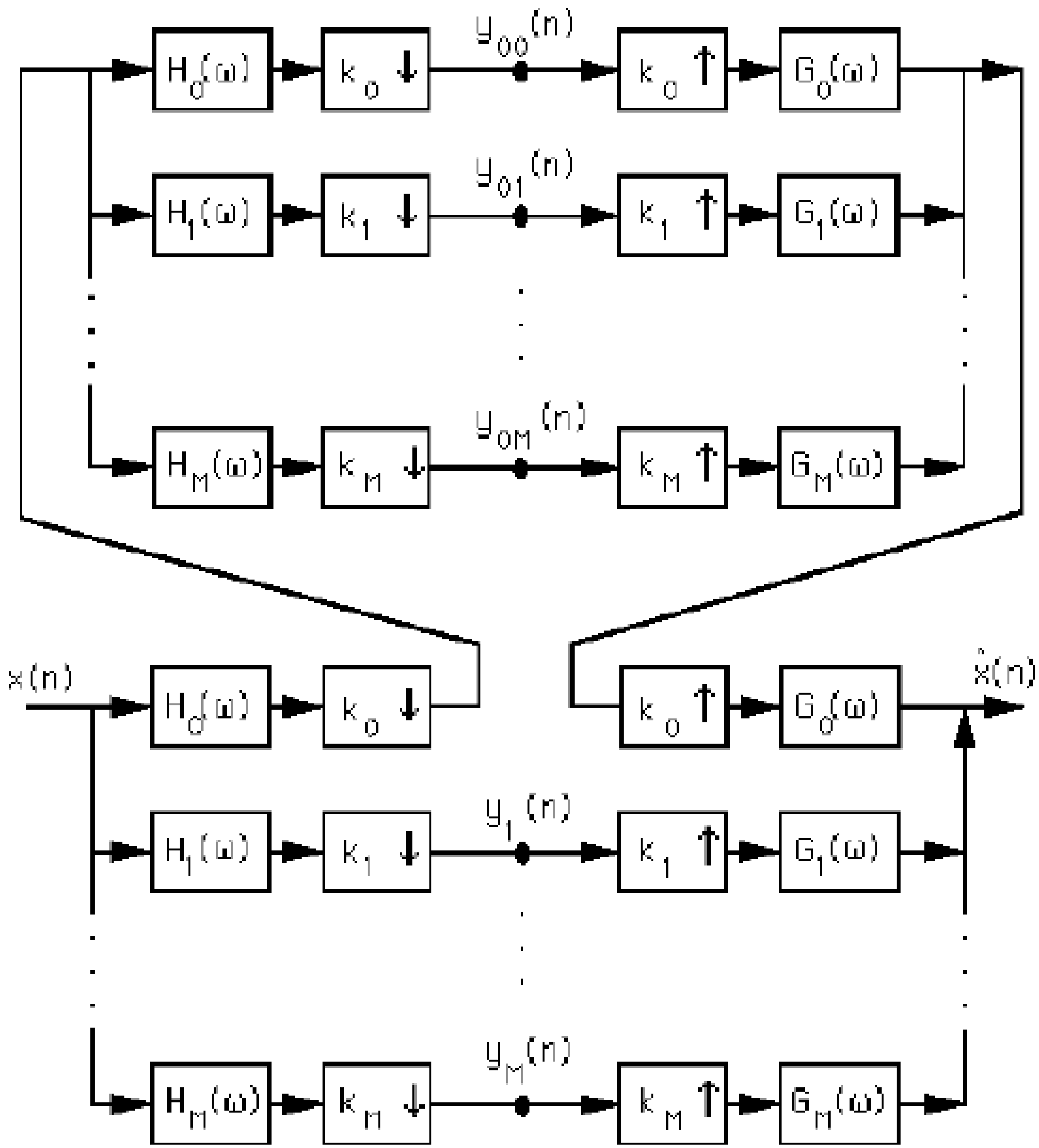
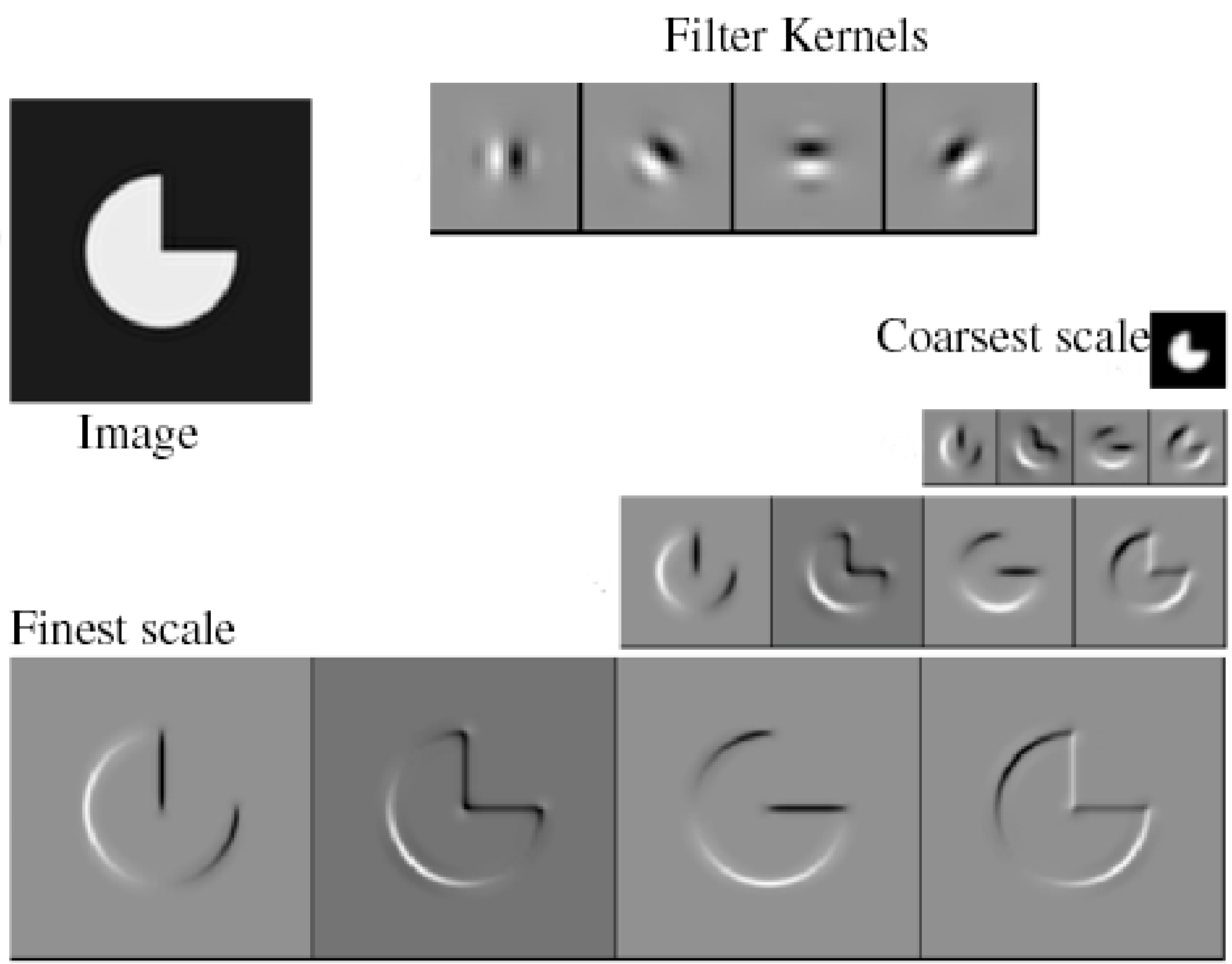
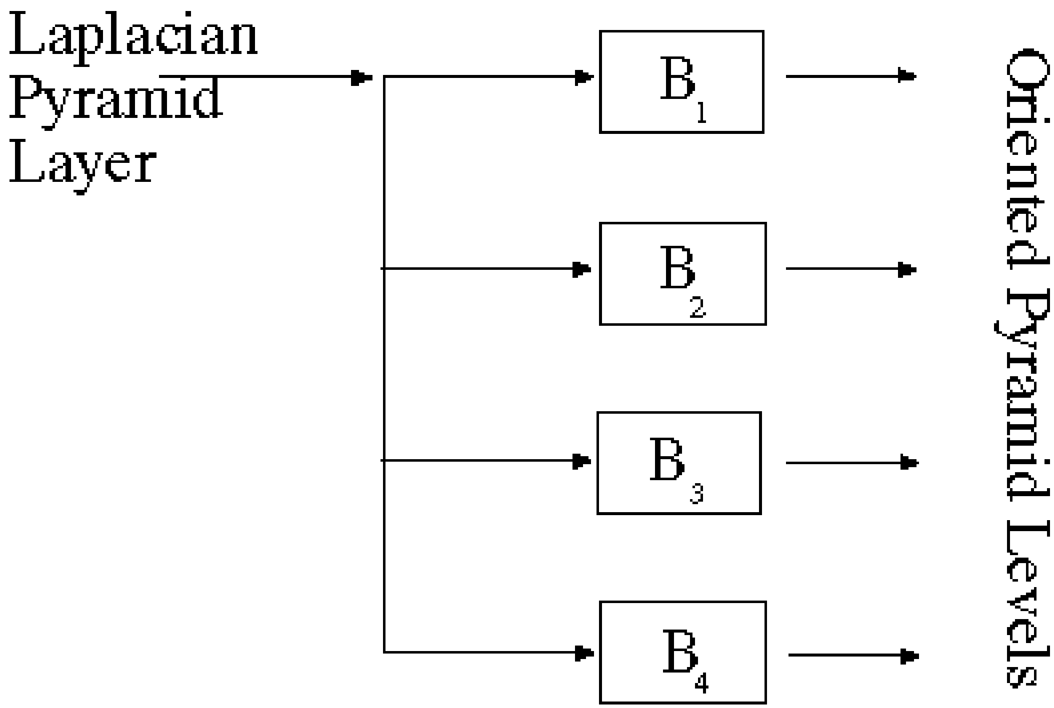


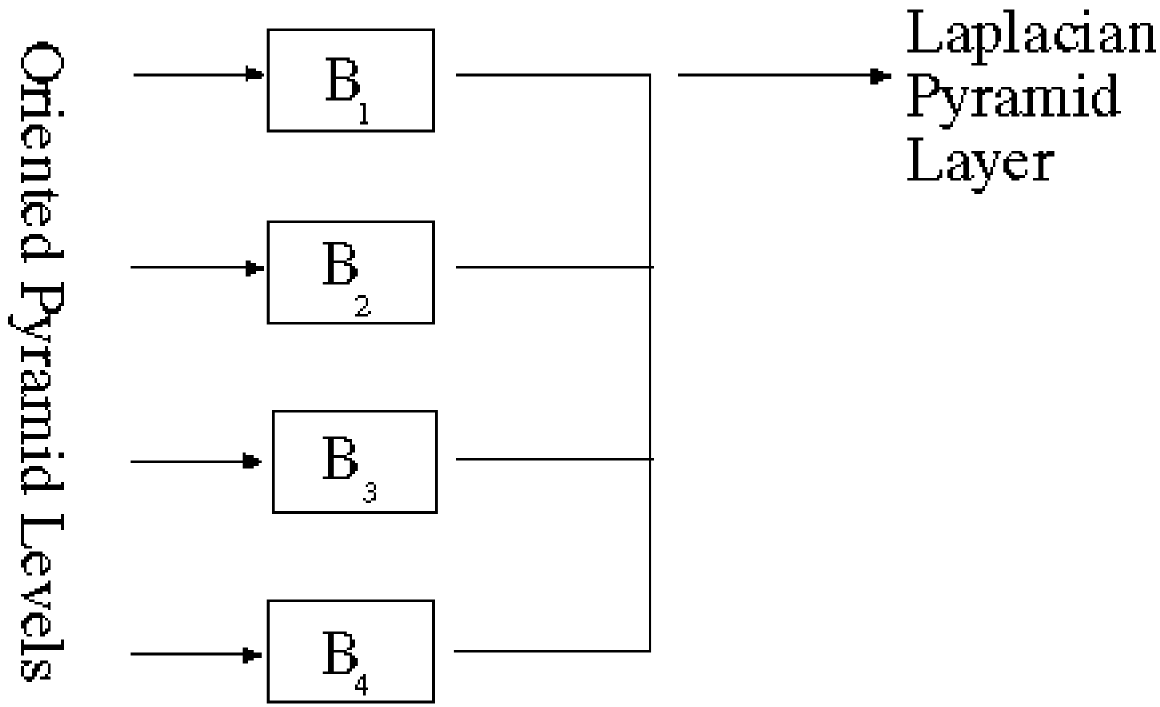
Figure 4.3: A non-uniformly cascaded analysis/synthesis filter bank.



Reprinted from "Shiftable MultiScale Transforms," by Simoncelli et al., IEEE Transactions on Information Theory, 1992, copyright 1992, IEEE


Laplacian
Pyramid
Layer
B_1
B_2
B_3
B_4
Oriented Pyramid Levels

Analysis

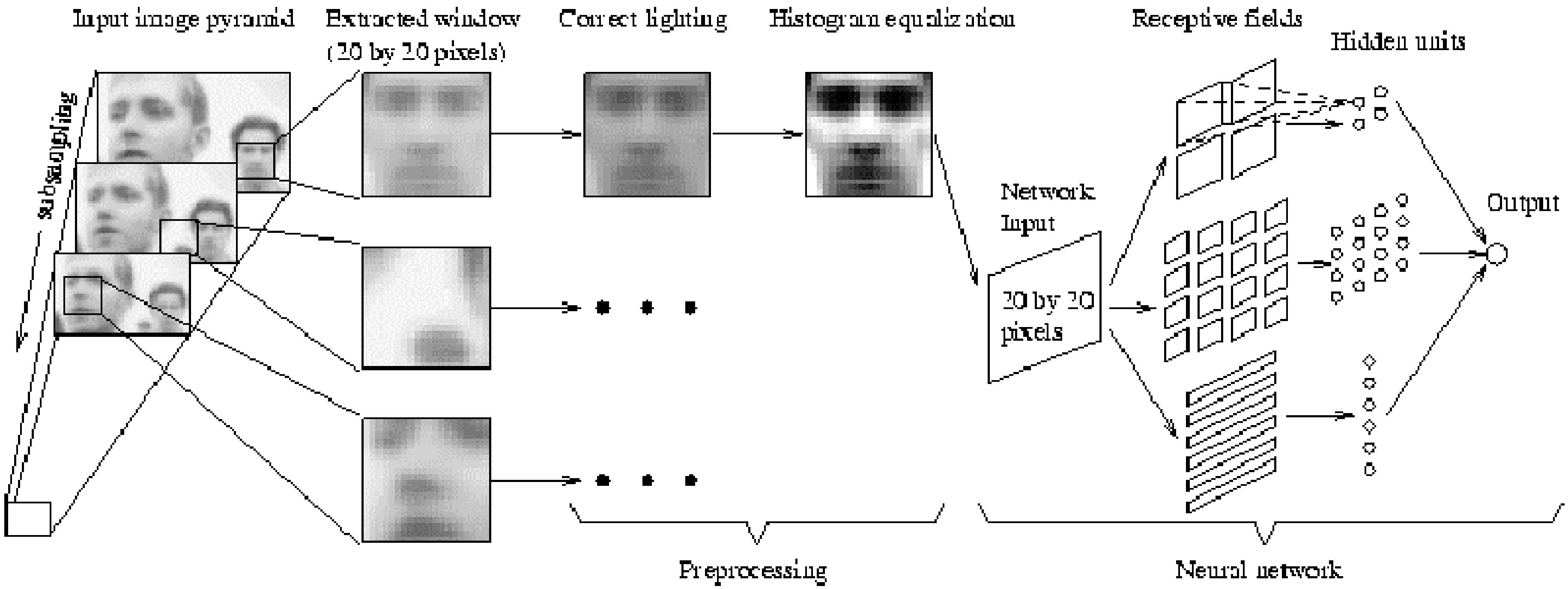

Oriented Pyramid Levels
B1
B2
B3
B4
Laplacian
Pyramid
Layer

synthesis

Final texture representation

- Form an oriented pyramid (or equivalent set of responses to filters at different scales and orientations).
- Square the output
- Take statistics of squared responses
  - e.g. mean of each filter output (are there lots of spots)
  - std of each filter output
  - Histogram of responses
  - mean of one scale conditioned on other scale having a particular range of values (e.g. are the spots in straight rows?)

Example application: CMU face detector



From: http://www.ius.cs.cmu.edu/IUS/har2/har/www/CMU-CS-95-158R/

Texture synthesis

- Use image as a source of probability model
- Choose pixel values by matching neighbourhood, then filling in
- Matching process
  - look at pixel differences
  - count only synthesized pixels

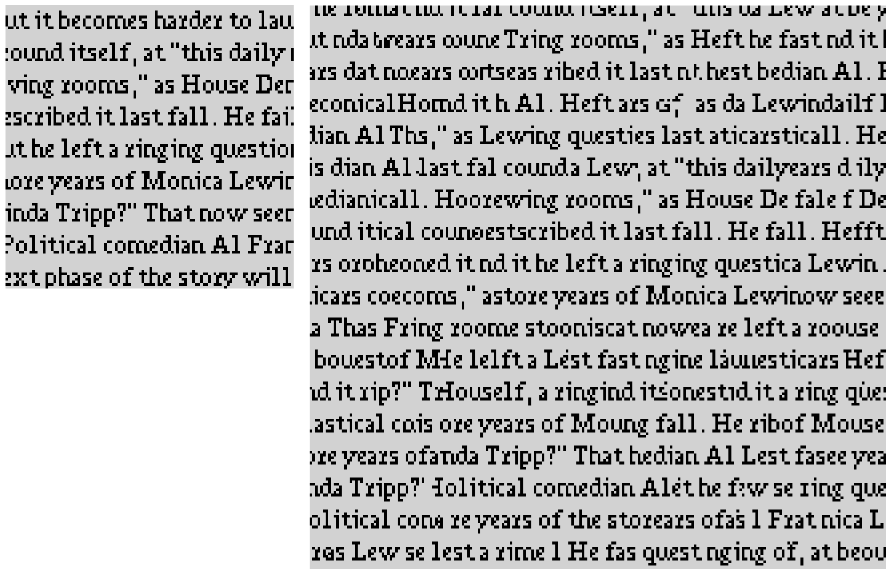
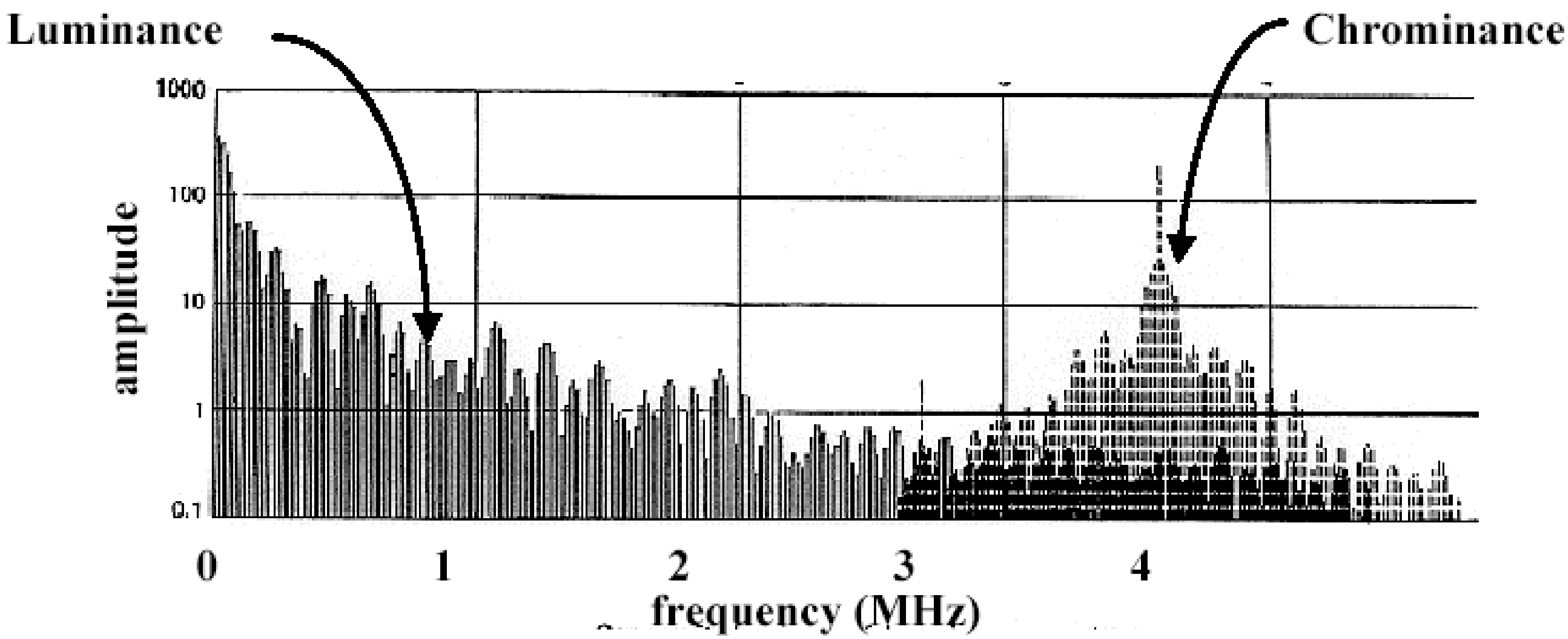


Figure from Texture Synthesis by Non-parametric Sampling, A. Efros and T.K. Leung, Proc. Int. Conf. Computer Vision, 1999 copyright 1999, IEEE

Application to image compression

- (compression is about hiding differences from the true image where you can't see them).

Bandwidth (transmission resources) for the components of the television signal



Understanding image perception allowed NTSC to add color to the black and white television signal (with some, but limited, incompatibility artifacts).

From W. E. Glenn, in Digital Images and Human Vision, MIT Press, edited by Watson, 1993

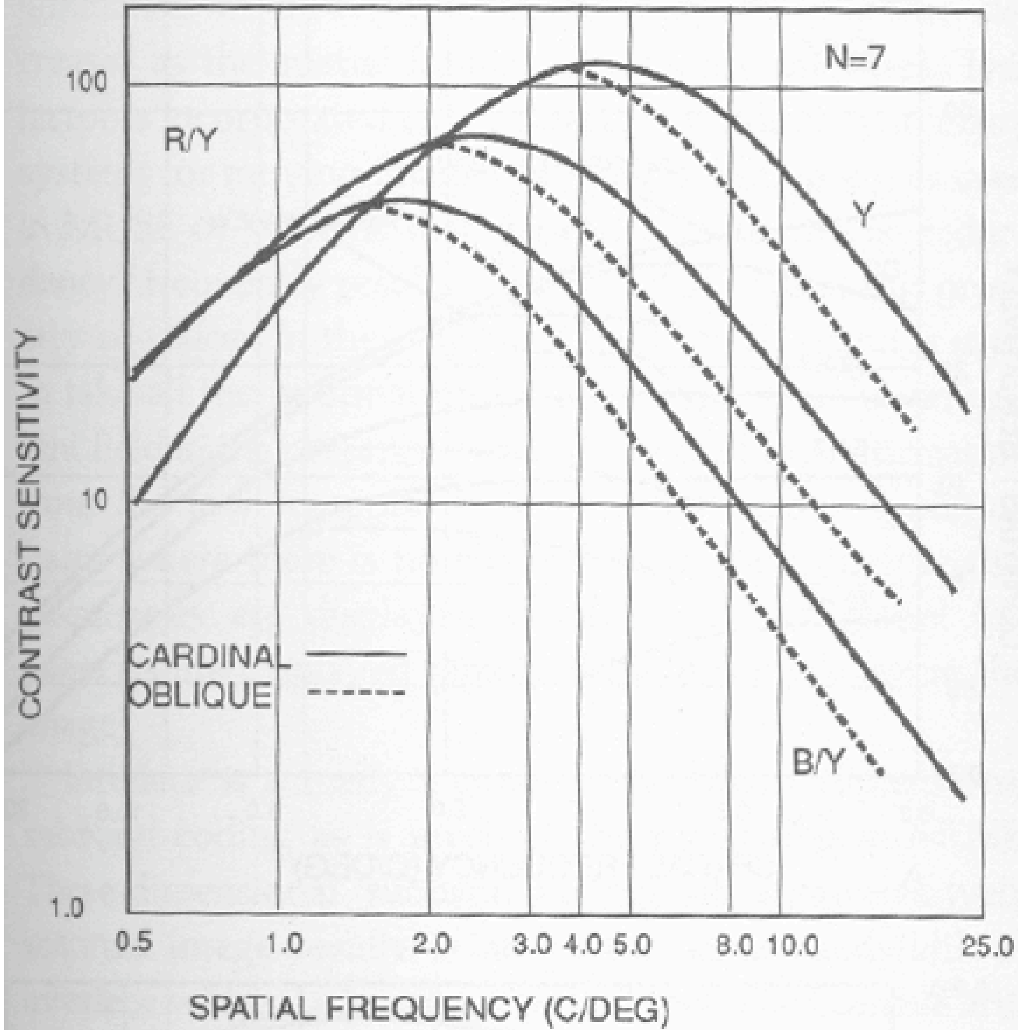


Figure 6.1
Contrast sensitivity threshold functions for static luminance gratings (Y) and isoluminance chromaticity gratings (R/Y,B/Y) averaged over seven observers.

RGB to Lab color space

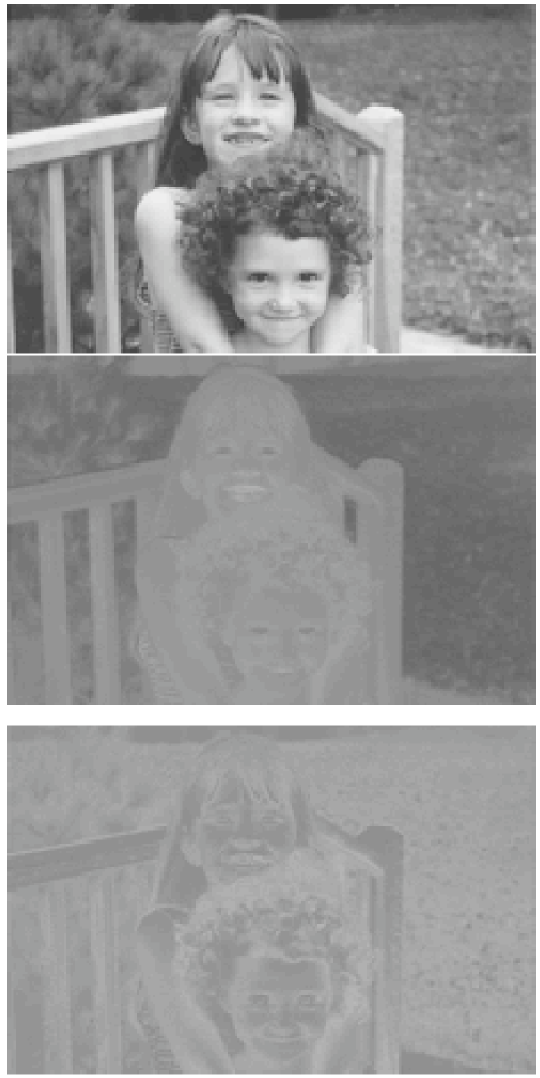
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.189423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} * \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

CIE 1976 L*a*b* is based directly on CIE XYZ and is an attampt to linearize the perceptibility of color differences. The non-linear relations for L*, a*, and b* are intended to mimic the logarithmic response of the eye. Coloring information is referred to the color of the white point of the system, subscript n.

$$L^* = 116 * (Y/Yn)^{1/3} - 16 \quad \text{for } Y/Yn > 0.008856$$
$$L^* = 903.3 * Y/Yn \quad \text{otherwise}$$

$$a^* = 500 * (\ f(X/Xn) - f(Y/Yn)\ )$$
$$b^* = 200 * (\ f(Y/Yn) - f(Z/Zn)\ )$$
$$\text{where } f(t) = t^{1/3} \quad \text{for } t > 0.008856$$
$$f(t) = 7.787 * t + 16/116 \quad \text{otherwise}$$

Lab components



L

a

b

Blurring the L Lab component

original

L

a

b

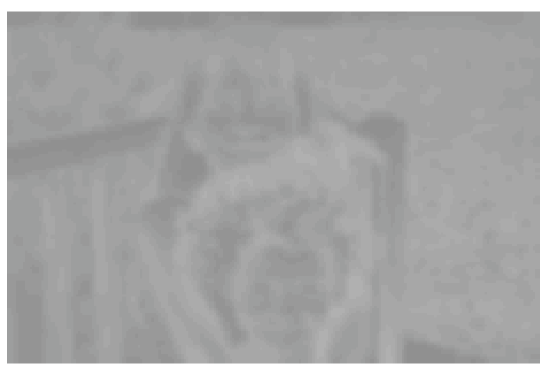
processed

Blurring the b Lab component

original

L

a



b

processed

Compression

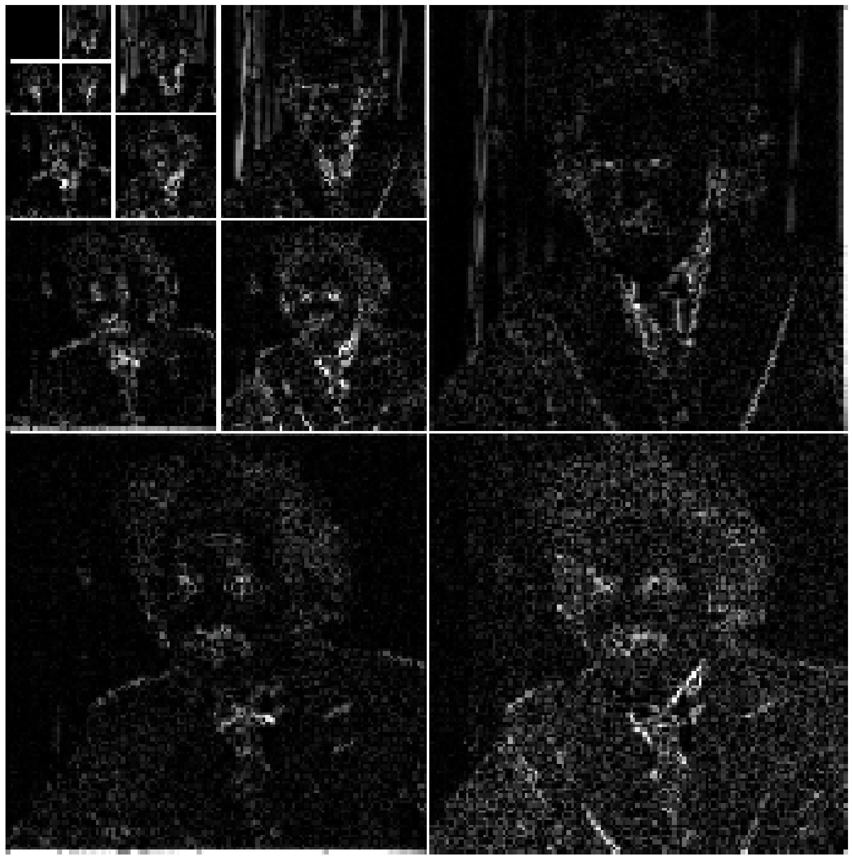
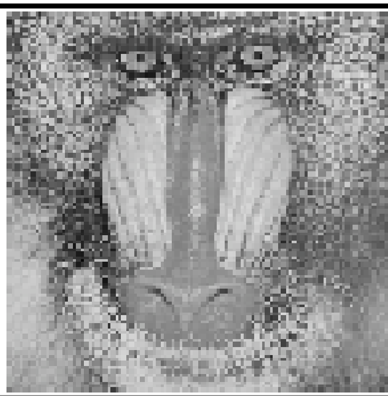
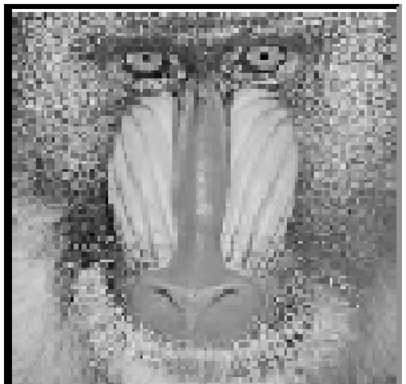
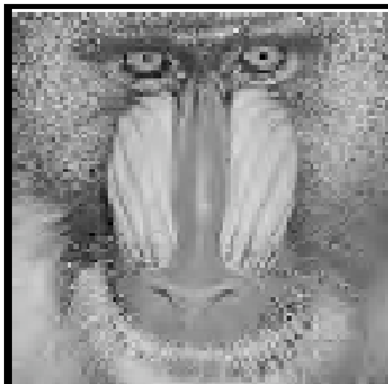


Figure 2. Example coefficient magnitudes of a wavelet decomposition. Shown are absolute values of subband coefficients in a 4-level separable wavelet decomposition of the "Einstein" image. Note that high-magnitude coefficients at adjacent scales tend to be located in the same spatial positions.

Orig 256K (8 bpp)	16K (0.5 bpp)	4K (0.125 bpp)	2K (0.031 bpp)
PSNR	35.77 dB	30.03 dB	27.49 dB
			

Orig 256K (8 bpp)	16K (0.5 bpp)	4K (0.125 bpp)	2K (0.031 bpp)
PSNR	24.92 dB	21.56 dB	20.60 dB
			

Mr. Dupont is a professional wine taster. When given a French wine, he will identify it with probability 0.9 correctly as French, and will mistake it for a Californian wine with probability 0.1.

When given a Californian wine, he will identify it with probability 0.8 correctly as Californian, and will mistake it for a French wine with probability 0.2.

Suppose that Mr. Dupont is given ten unlabelled glasses of wine, three with French and seven with Californian wines. He randomly picks a glass, tries the wine, and solemnly says: "French". What is the probability that the wine he tasted was Californian?

Mr. Dupont is a professional wine taster. When given a French wine, he will identify it with probability 0.9 correctly as French, and will mistake it for a Californian wine with probability 0.1.

When given a Californian wine, he will identify it with probability 0.8 correctly as Californian, and will mistake it for a French wine with probability 0.2.

Suppose that Mr. Dupont is given ten unlabelled glasses of wine, three with French and seven with Californian wines. He randomly picks a glass, tries the wine, and solemnly says: "French". What is the probability that the wine he tasted was Californian?

	Rf	Rc
F	0.9	0.2
C	0.1	0.8

$P(F) = 0.3$; $P(C) = 0.7$;

$$P(C|Rf) = P(Rf|C)\, p(\,C\,)/P(Rf)$$
$$= 0.1*0.7/\square_{w}\, P(Rf\,|w)p(w)$$
$$= 0.1*0.7/(0.9*0.3+0.1*0.7) = 0.21$$
$$= 0.1*0.7/0.34 = 0.21$$

Bayes theorem

$$P(x, y) = P(x|y)\,P(y)$$

so

$$P(x|y)\,P(y) = P(y|x)\,P(x)$$

and

$$P(x|y) = P(y|x)\,P(x)\,/\,P(y)$$

The parameters you want to estimate

What you observe

Likelihood function

Prior probability

Constant w.r.t. parameters x.

"You must choose, but Choose Wisely"

- Given only probabilities, can we minimize the number of errors we make?
- *Given:*

  responses R_i, categories C_i, current category c, data x
- *To Minimize error:*
  - Decide R_i **if** $P(C_i \mid x) > P(C_k \mid x)$ **for all** $i \neq k$

    $P(x \mid C_i)\, P(C_i) > P(x \mid C_k)\, P(C_k)$

    $P(x \mid C_i) / P(x \mid C_k) > P(C_k) / P(C_i)$

    $P(x \mid C_i) / P(x \mid C_k) > T$

Optimal classifications always involve hard boundaries

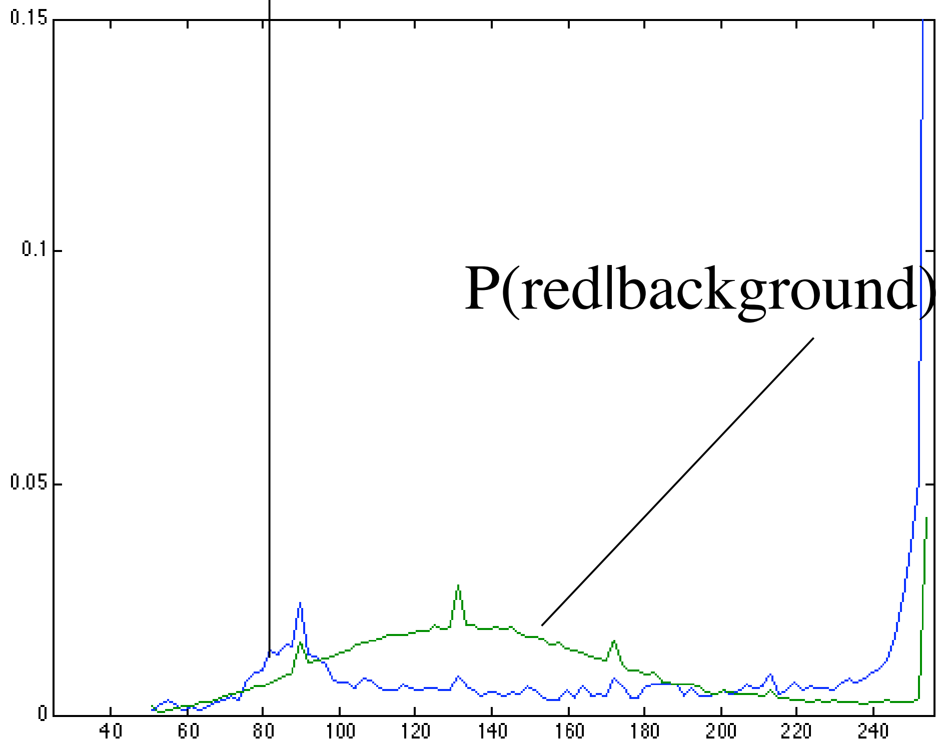
Horse Segmentation

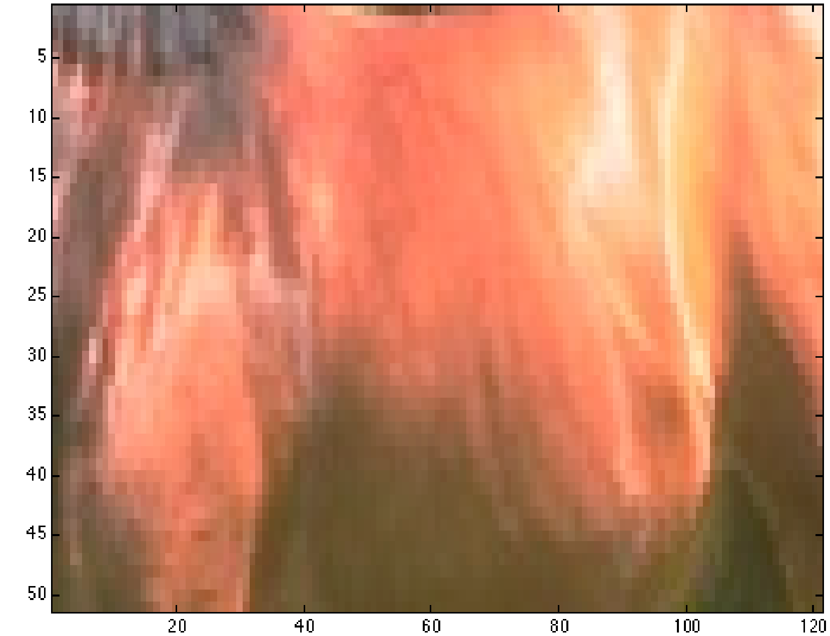
P(horse) = 0.04

P(background) = 0.96

P(red|horse)

P(red|background)





Now evaluate

$$\prod_{j=1:Nmeasurements} p(r_j \mid horse) / p(r_j \mid background)$$