Visual Computing:
Pyramids and Wavelets

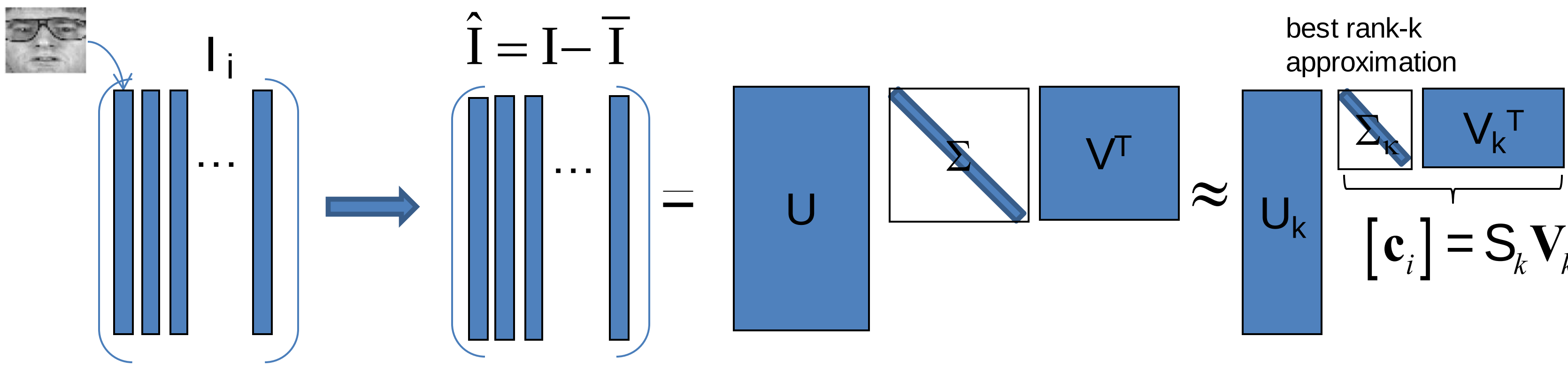
Prof. Marc Pollefeys

ETH

inf Informatik
Computer Science

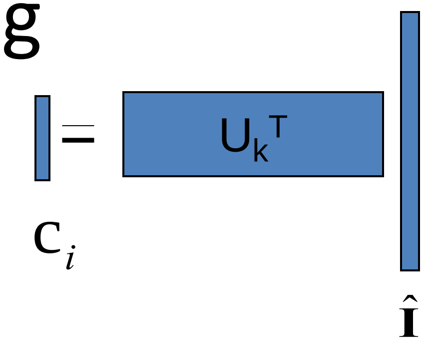
Last lecture

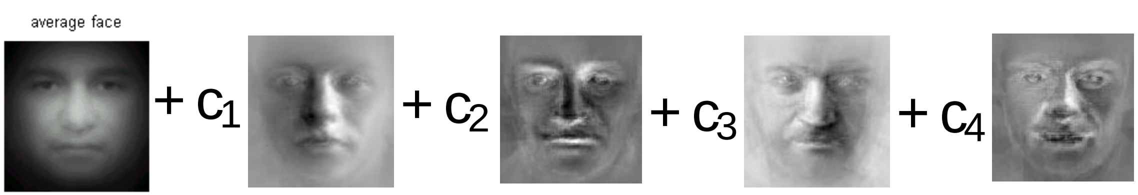
- PCA (or KL transform)



- SSD matching vs. Eigenspace matching

$$\left\| I_i - I \right\| = \left\| \hat{I}_i - \hat{I} \right\| \approx \left\| \mathbf{c}_i - \mathbf{c} \right\| \quad \text{with} \quad \mathbf{c} = \mathbf{U}_k^T \hat{\mathbf{I}}$$

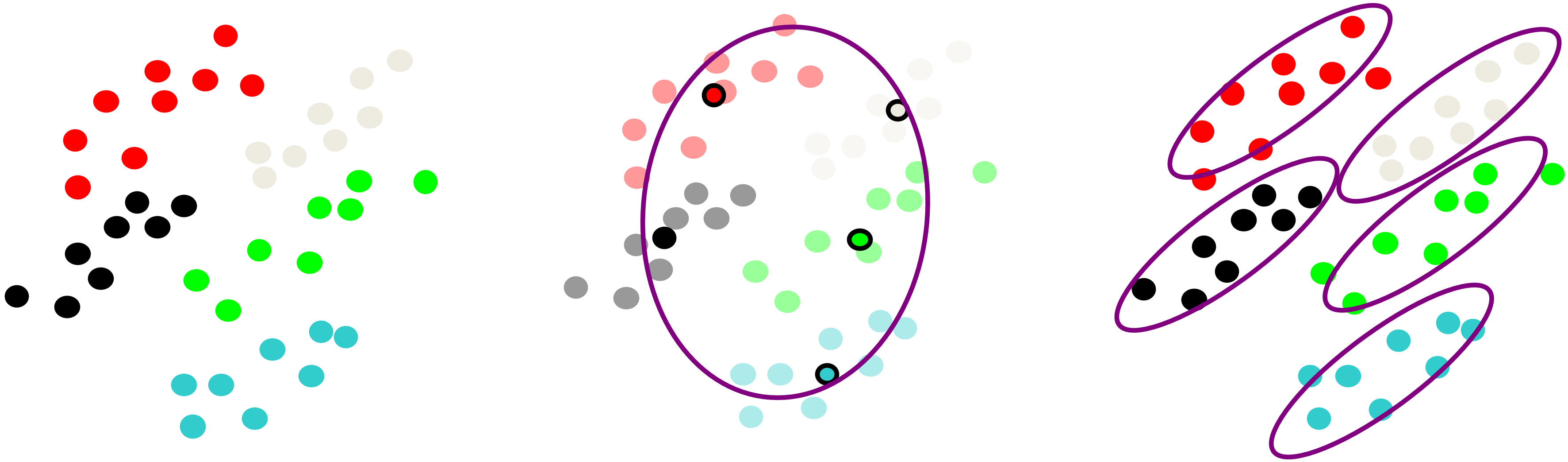




Eigenspace matching will typically work better because only main characteristics are preserved and irrelevant details are discarded

Fisherfaces / LDA (Belhumeur et al. 1997)

Training Data | Between Individual Variance | Within-Individual Variance



KEY IDEAS:

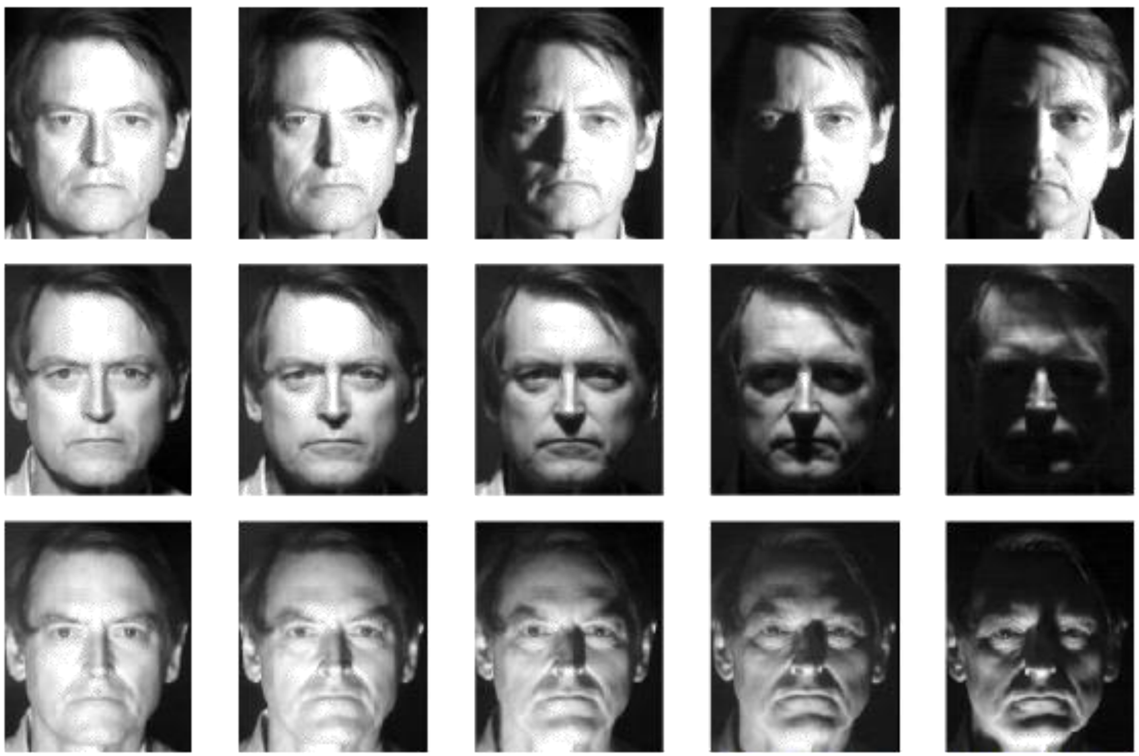
- Find directions where ratio of between:within individual variance are maximized
- Linearly project to basis where dimension with good signal:noise ratio are maximized

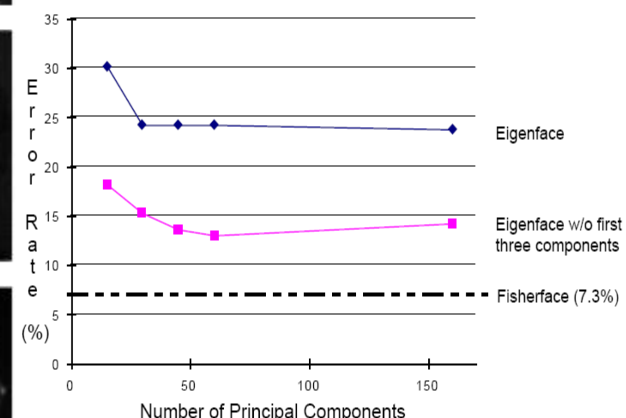
ETH

(slide courtesy of Simon Prince)

Eigenfaces vs. Fisherfaces

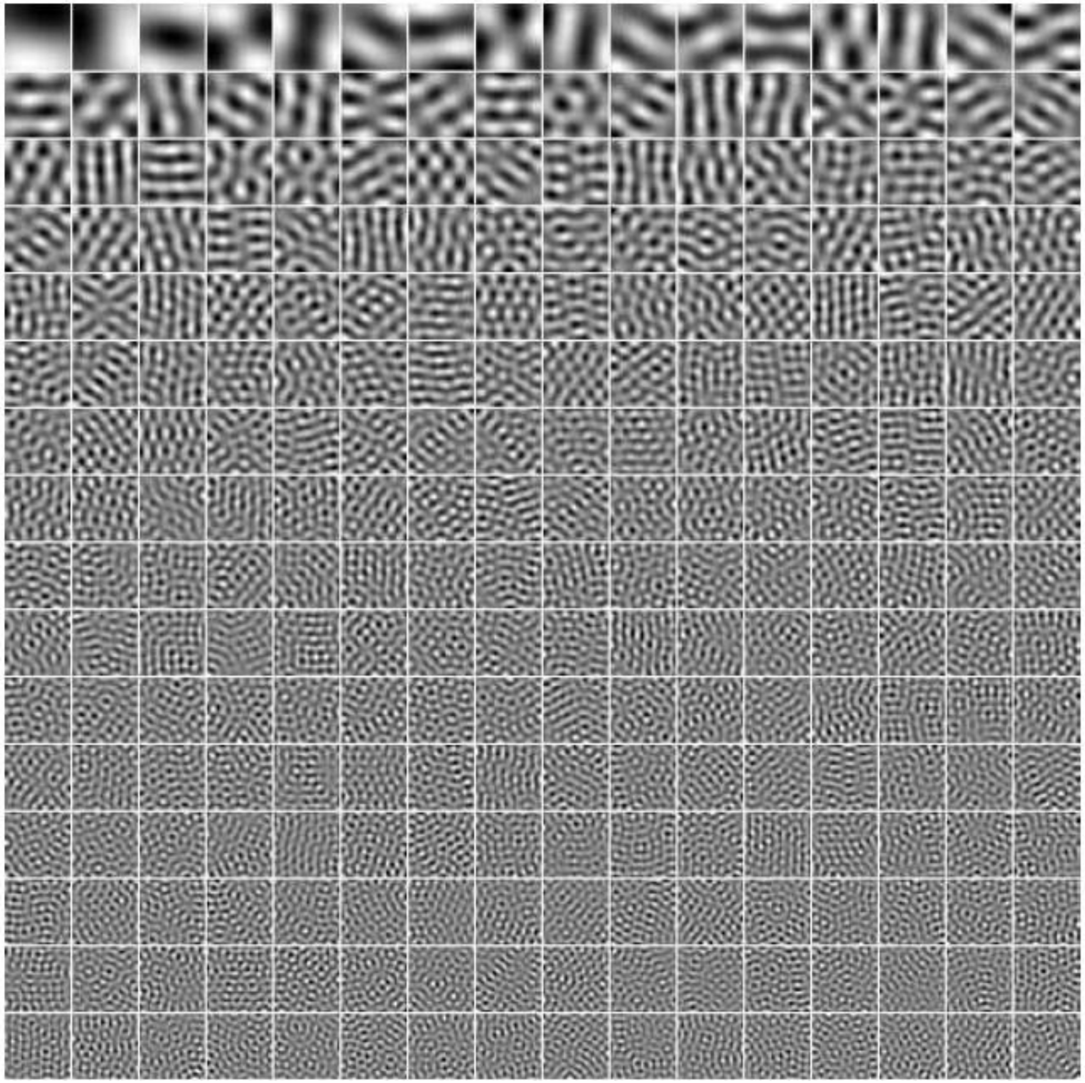
Differences due to varying illumination can be much larger than differences between faces!

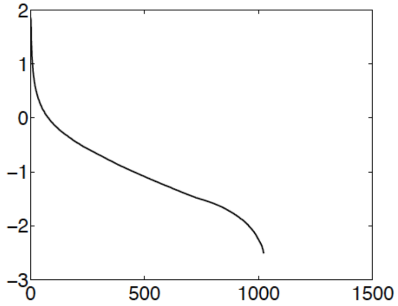




[Belhumeur, Hespanha, Kriegman, 1997]

KLT/PCA on natural image patches





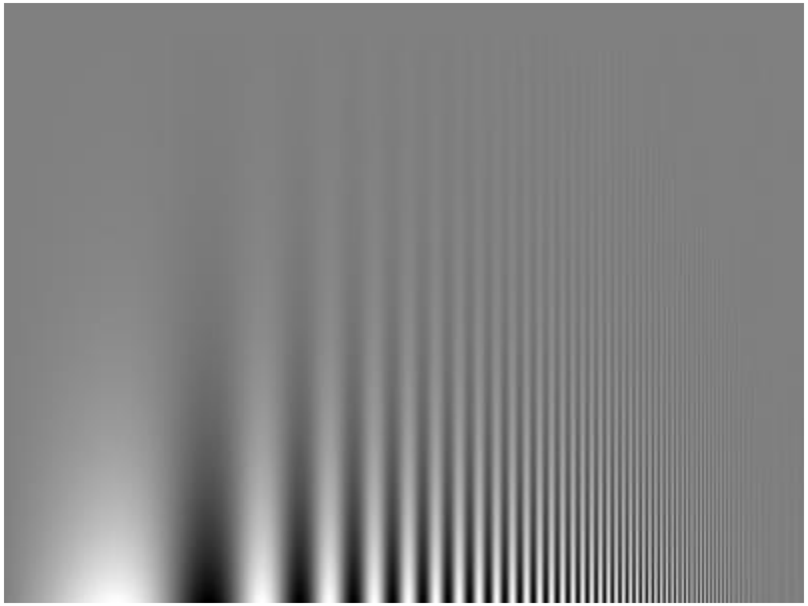
log variance of patches

from Aapo Hyvärinen et al.

JPEG image compression

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes

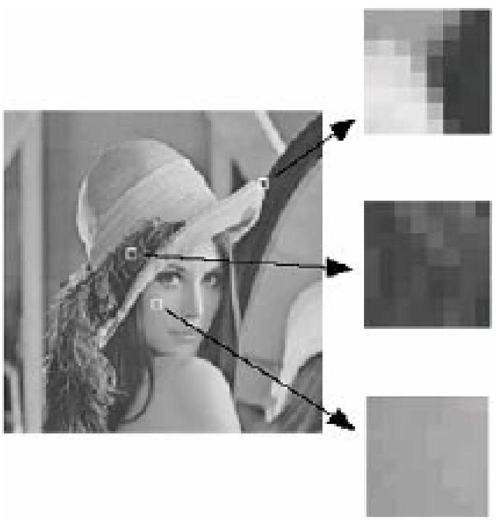
Campbell-Robson contrast sensitivity curve

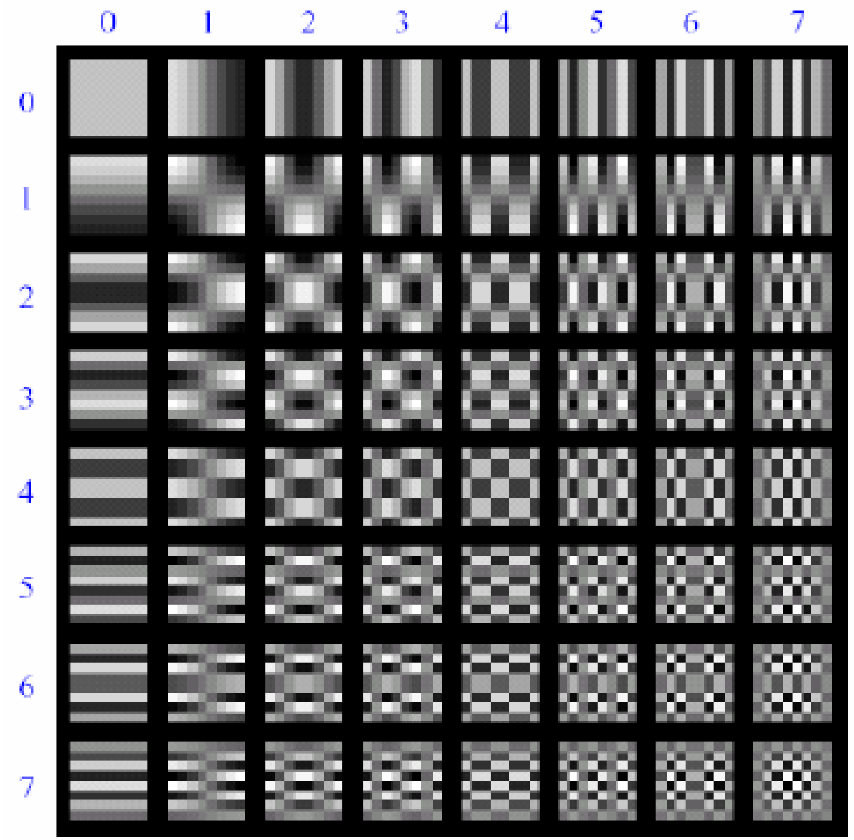


We don't resolve high frequencies too well…

… let's use this to compress images… JPEG!

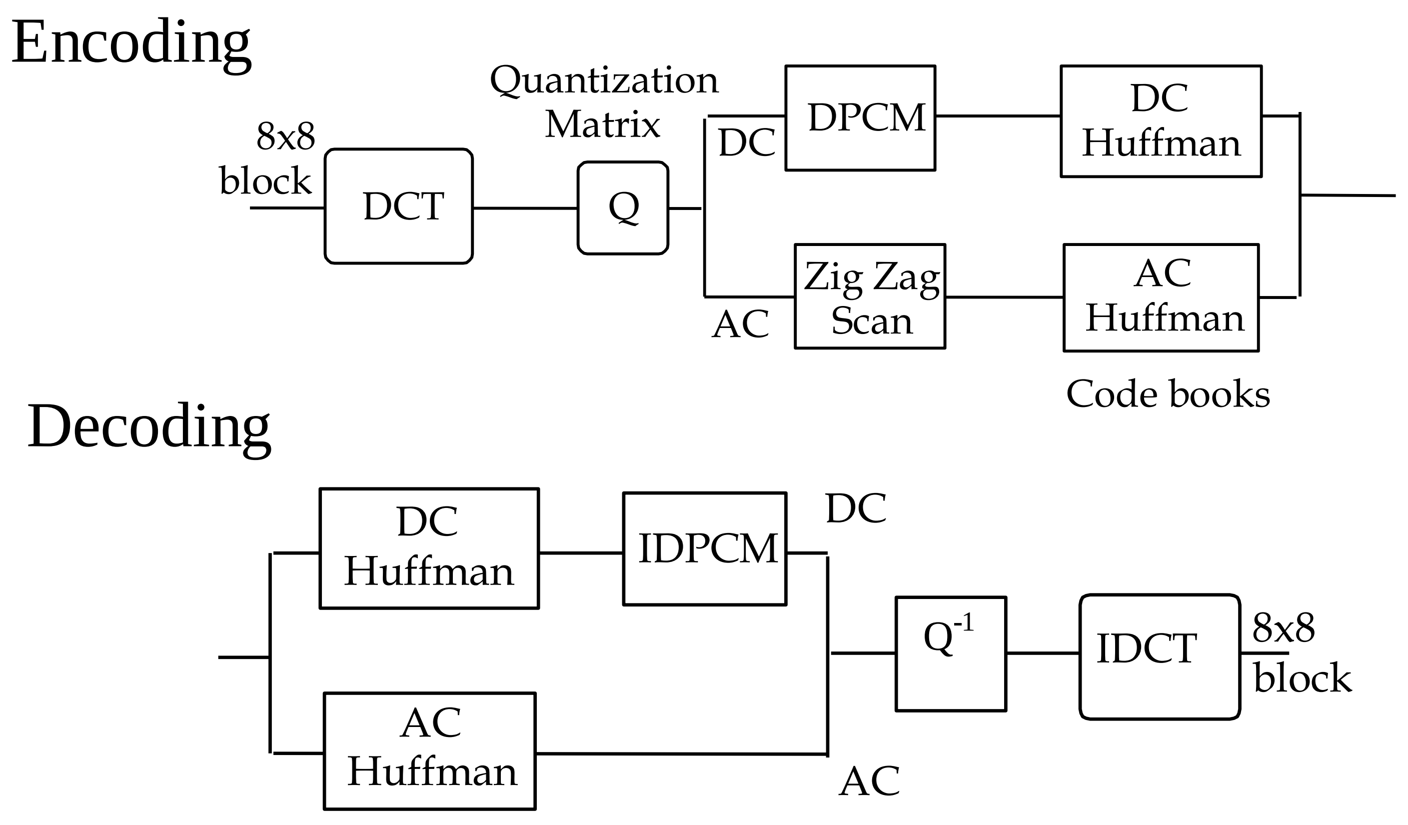
Lossy Image Compression (JPEG)





Block-based Discrete Cosine Transform (DCT)

JPEG Encoding and Decoding



www.jpeg.org

Using DCT in JPEG

A variant of discrete Fourier transform

- Real numbers
- Fast implementation

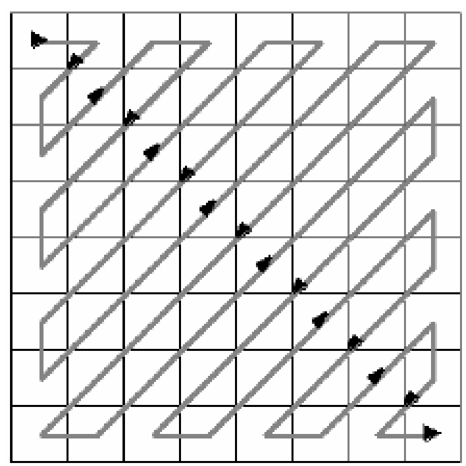
Block size

- small block
  - faster
  - correlation exists between neighboring pixels
- large block
  - better compression in smooth regions

Using DCT in JPEG

The first coefficient $B(0,0)$ is the DC component, the average intensity

The top-left coeffs represent low frequencies, the bottom right – high frequencies



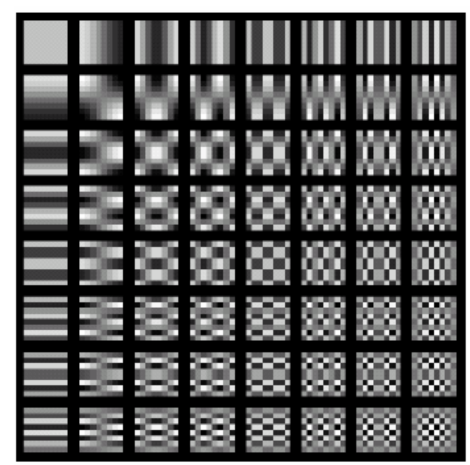


Image compression using DCT

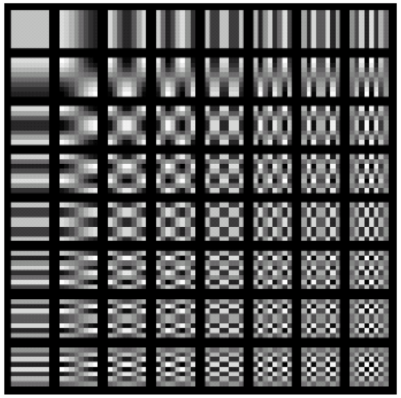
DCT enables image compression by concentrating most image information in the low frequencies

Loose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right

The decoder computes the inverse DCT – IDCT

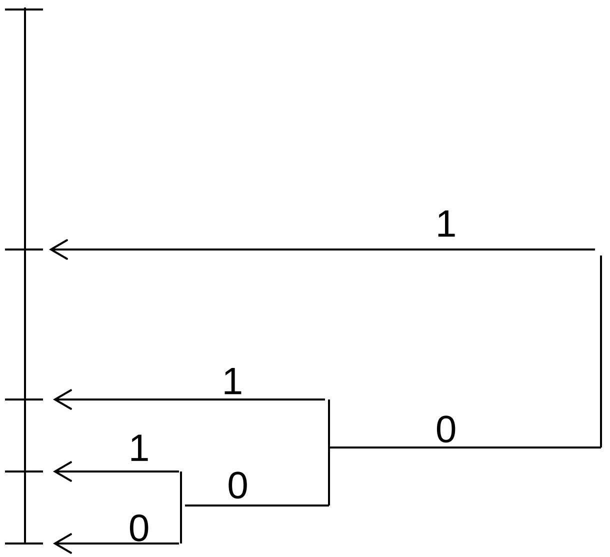
•Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31



Entropy Coding (Huffman code)

Symbol	Prob.	Code	Binary Fraction
Z	0.5	1	0.1
Y	0.25	01	0.01
X	0.125	001	0.001
W	0.125	000	0.000



- The code words, if regarded as a binary fractions, are pointers to the particular interval being coded.
- In Huffman code, the code words point to the base of each interval.
- The average code word length is $H = -\Sigma p(s)\log_2 p(s)$ -> optimal

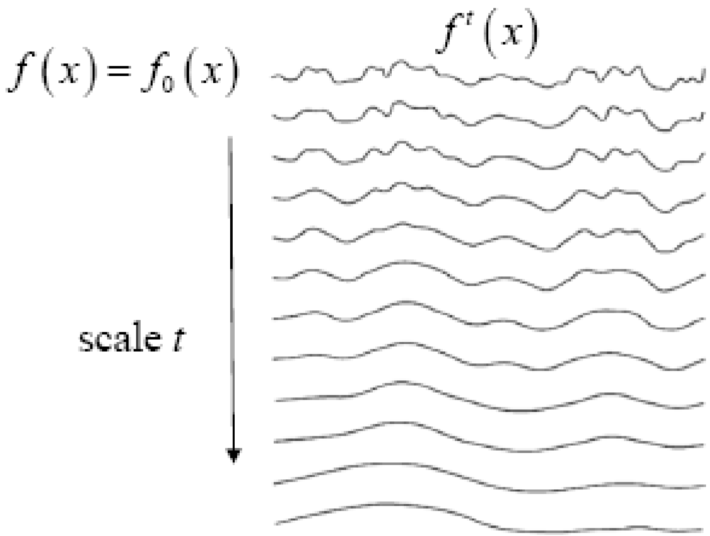
JPEG compression comparison

89k

12k

Scale-space representations

- From an original signal $f(x)$ generate a parametric family of signals $f^t(x)$ where fine-scale information is successively suppressed

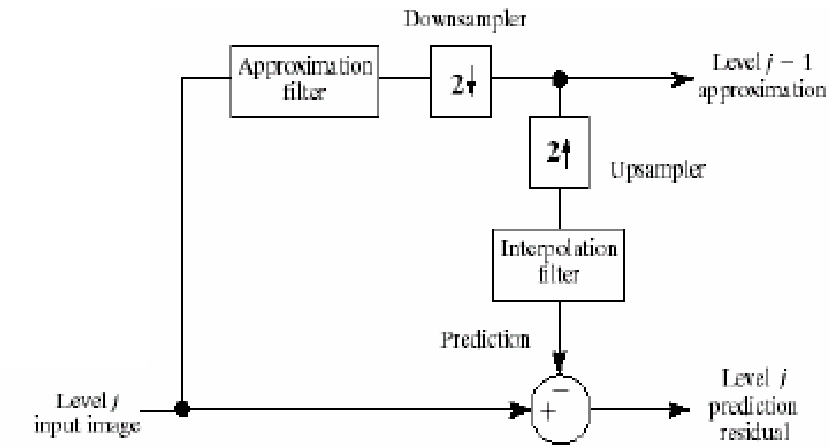


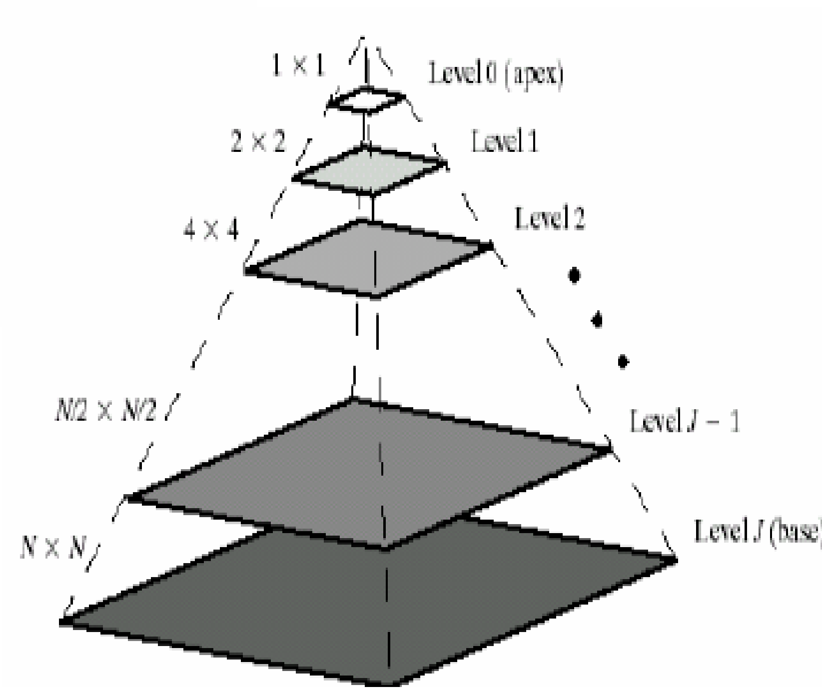
[Witkin 1983]

- Family of signals generated by successive smoothing with a Gaussian filter

- Zero-crossings of 2nd derivative: Fewer features at coarser scales

Image pyramid

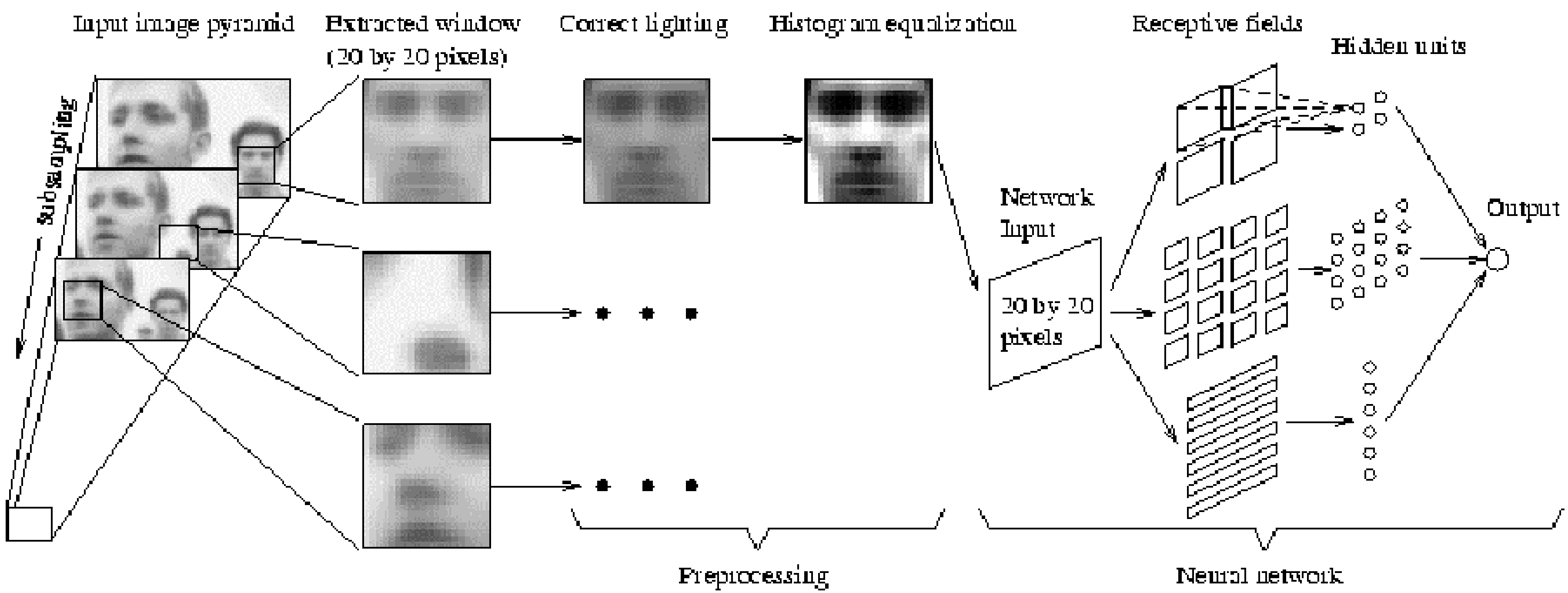




Applications of scaled representations

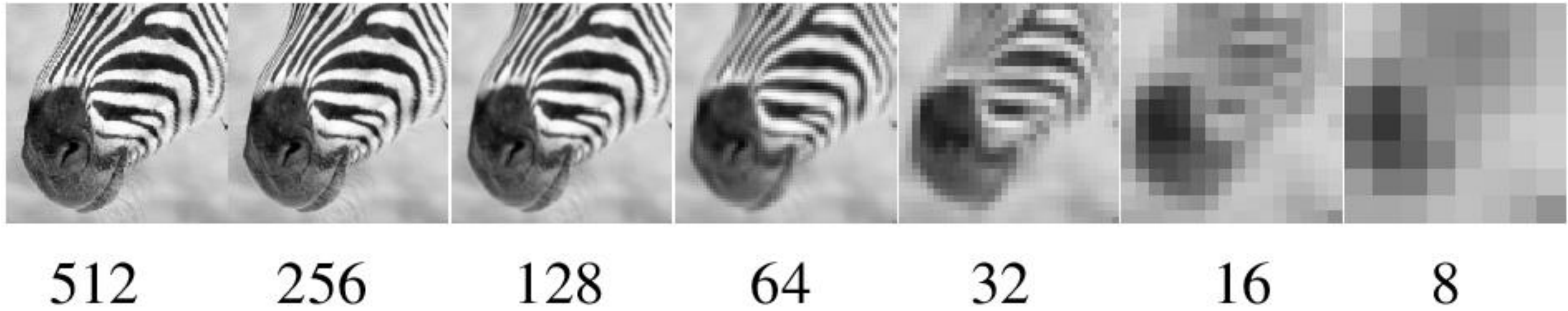
- Search for correspondence
  - look at coarse scales, then refine with finer scales
- Edge tracking
  - a “good” edge at a fine scale has parents at a coarser scale
- Control of detail and computational cost in matching
  - e.g. finding stripes
  - terribly important in texture representation

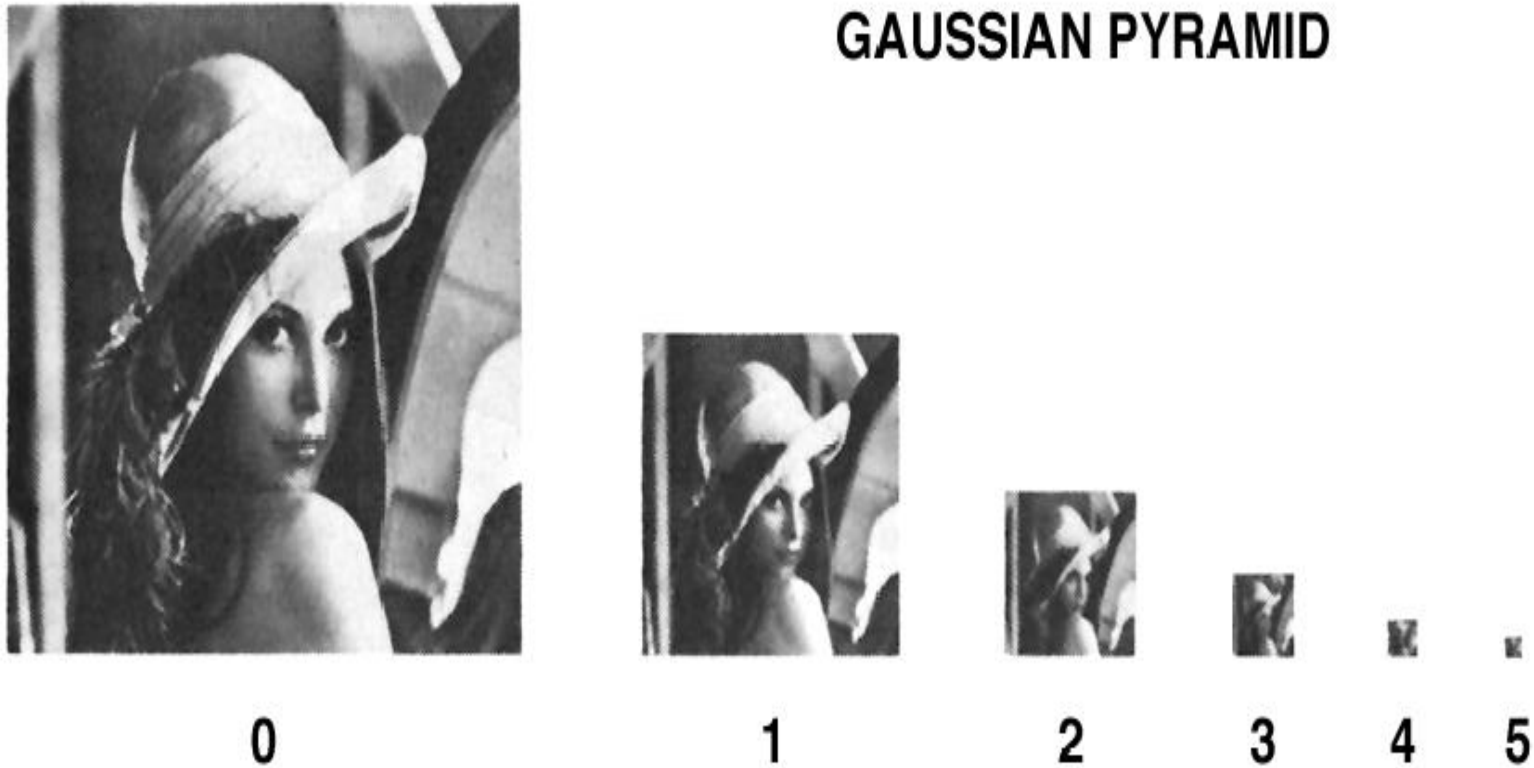
Example: CMU face detection



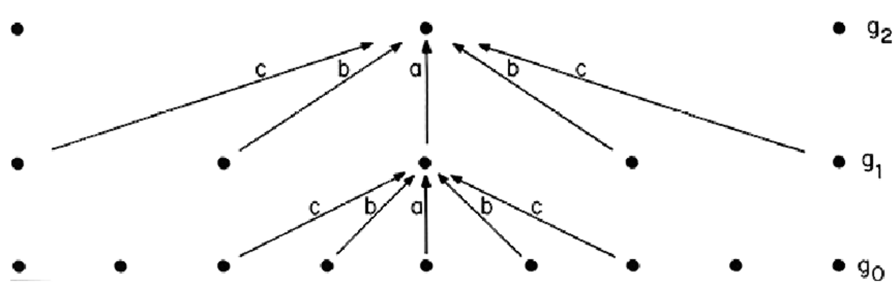
The Gaussian pyramid

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


512
256
128
64
32
16
8



GAUSSIAN PYRAMID



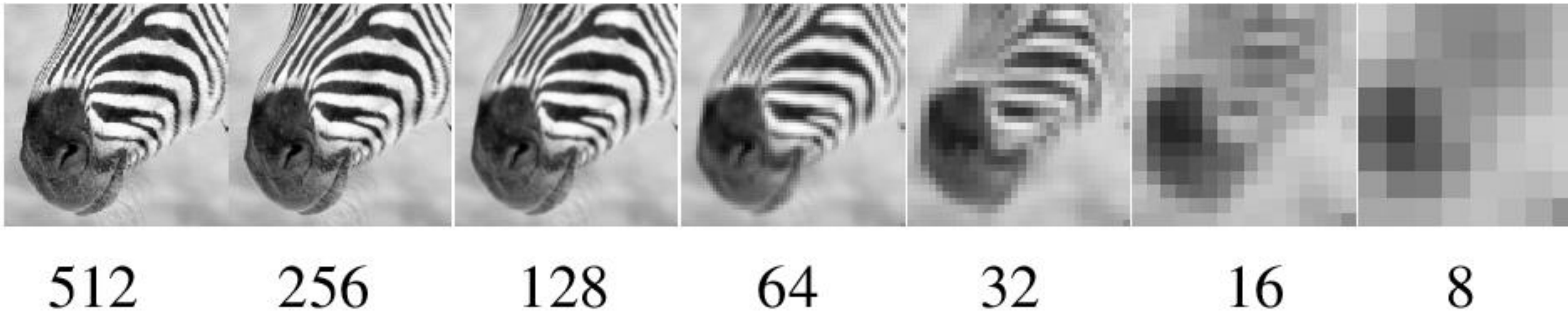
g_0 = IMAGE

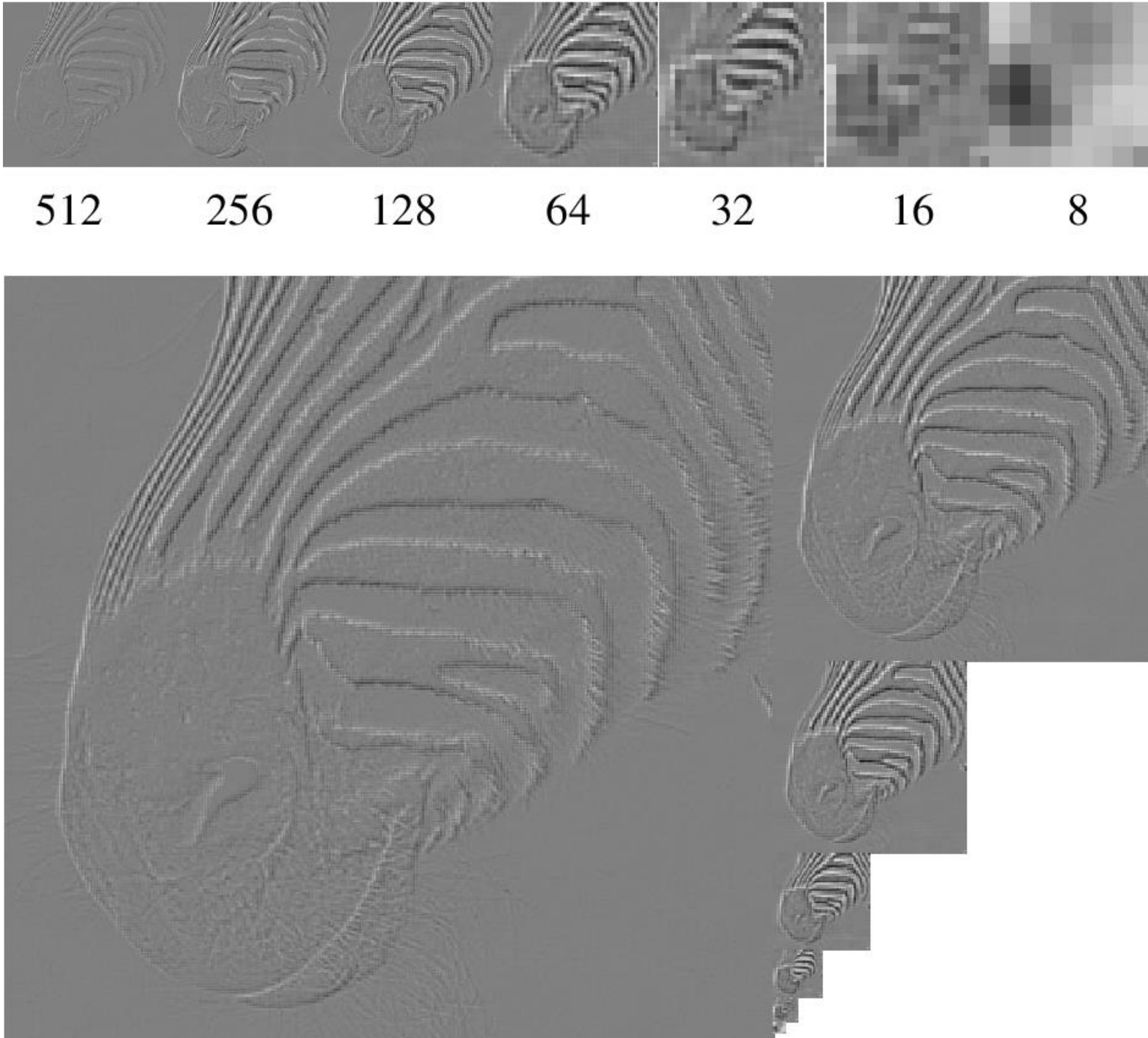
g_L = REDUCE $[g_{L-1}]$

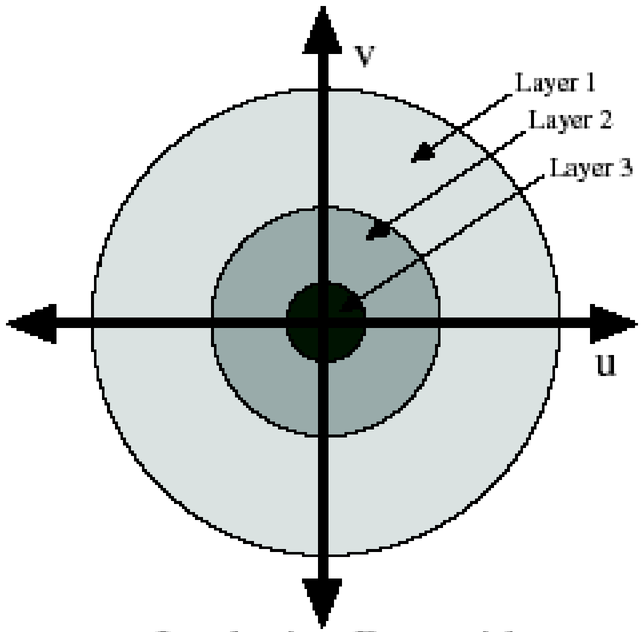
http://web.mit.edu/persci/people/adelson/pub_pdfs/pyramid83.pdf

The Laplacian Pyramid

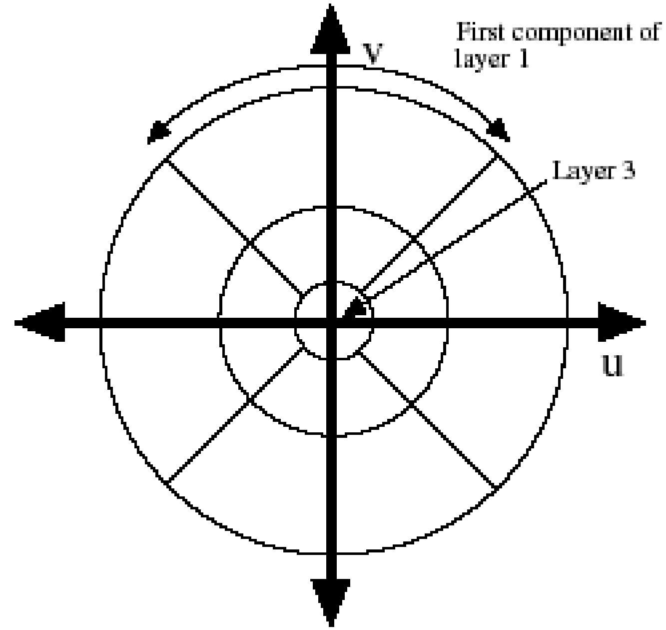
- Synthesis
  - preserve difference between upsampled Gaussian pyramid level and Gaussian pyramid level
  - band pass filter - each level represents spatial frequencies (largely) unrepresented at other levels
- Analysis
  - reconstruct Gaussian pyramid, take top layer


512
256
128
64
32
16
8


512
256
128
64
32
16
8



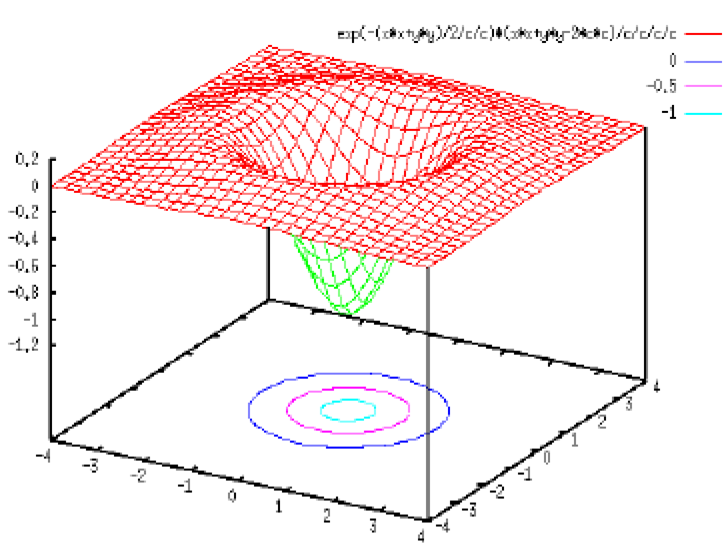
Laplacian Pyramid



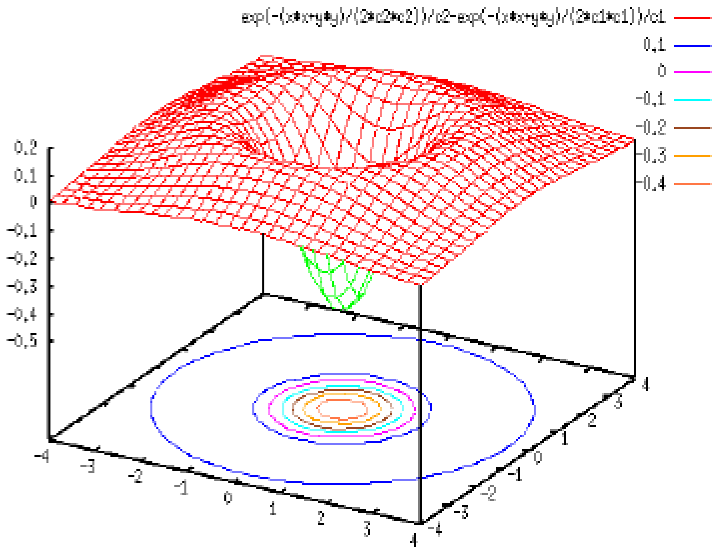
Oriented Pyramid

LoG vs. DoG

Laplacian of Gaussian

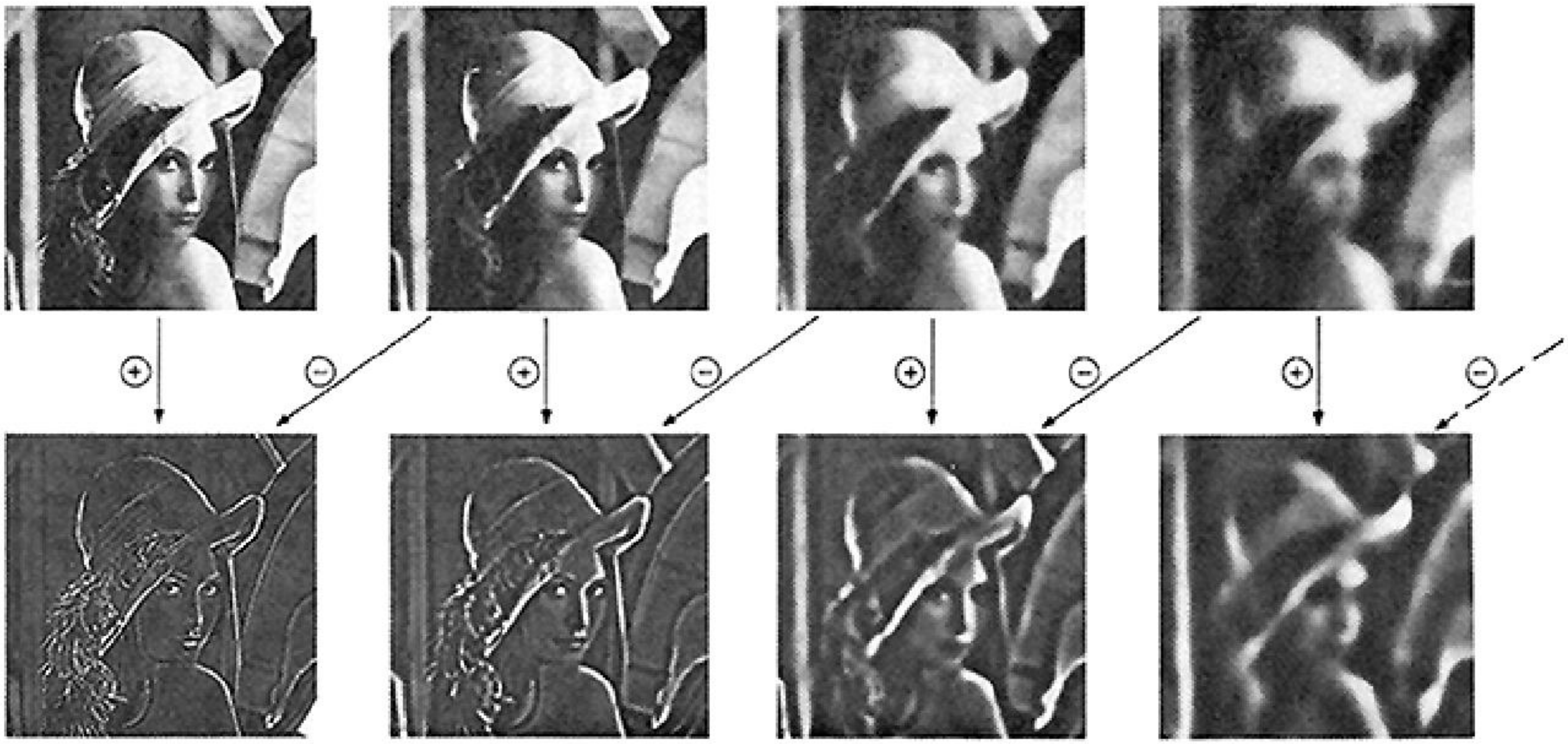


Difference of Gaussians



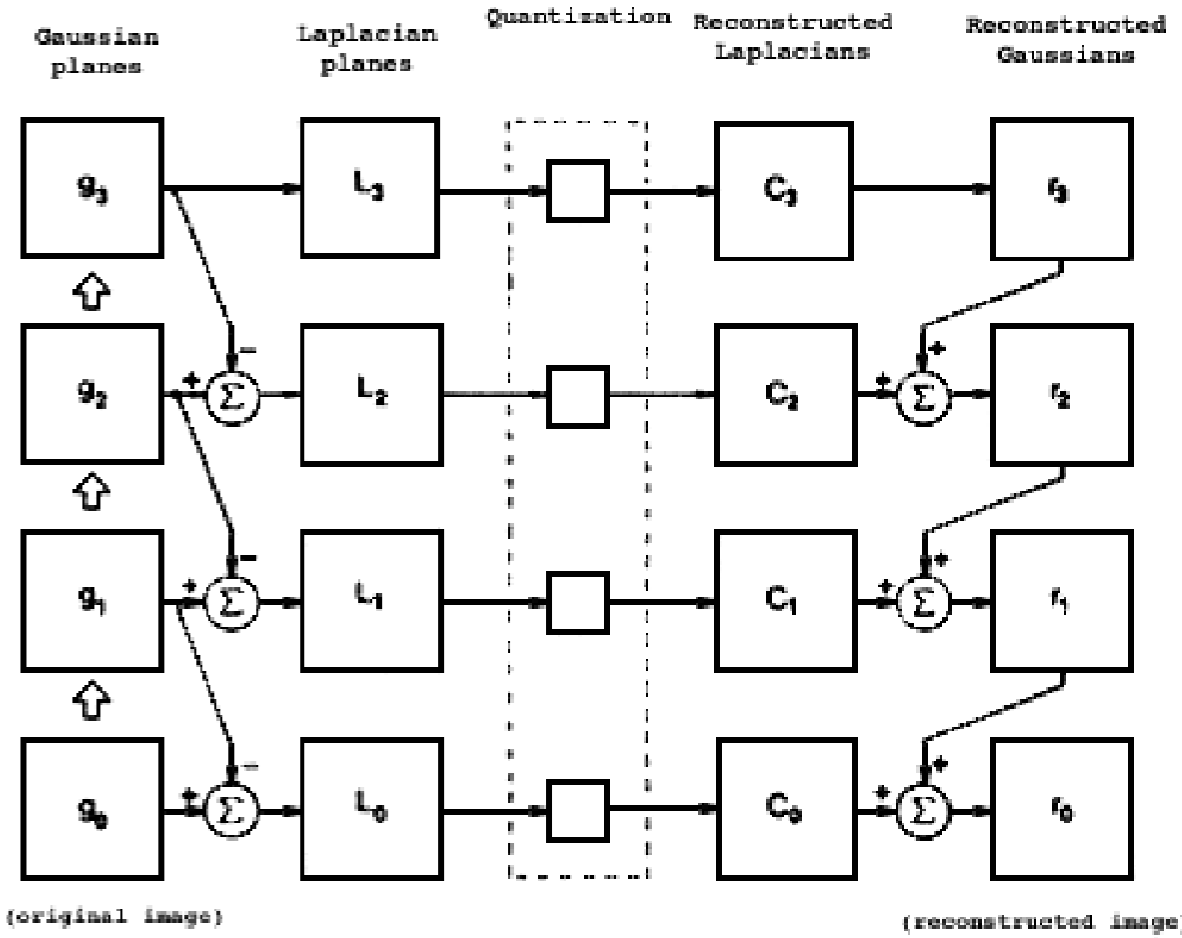
Gaussian Pyramid

Laplacian Pyramid



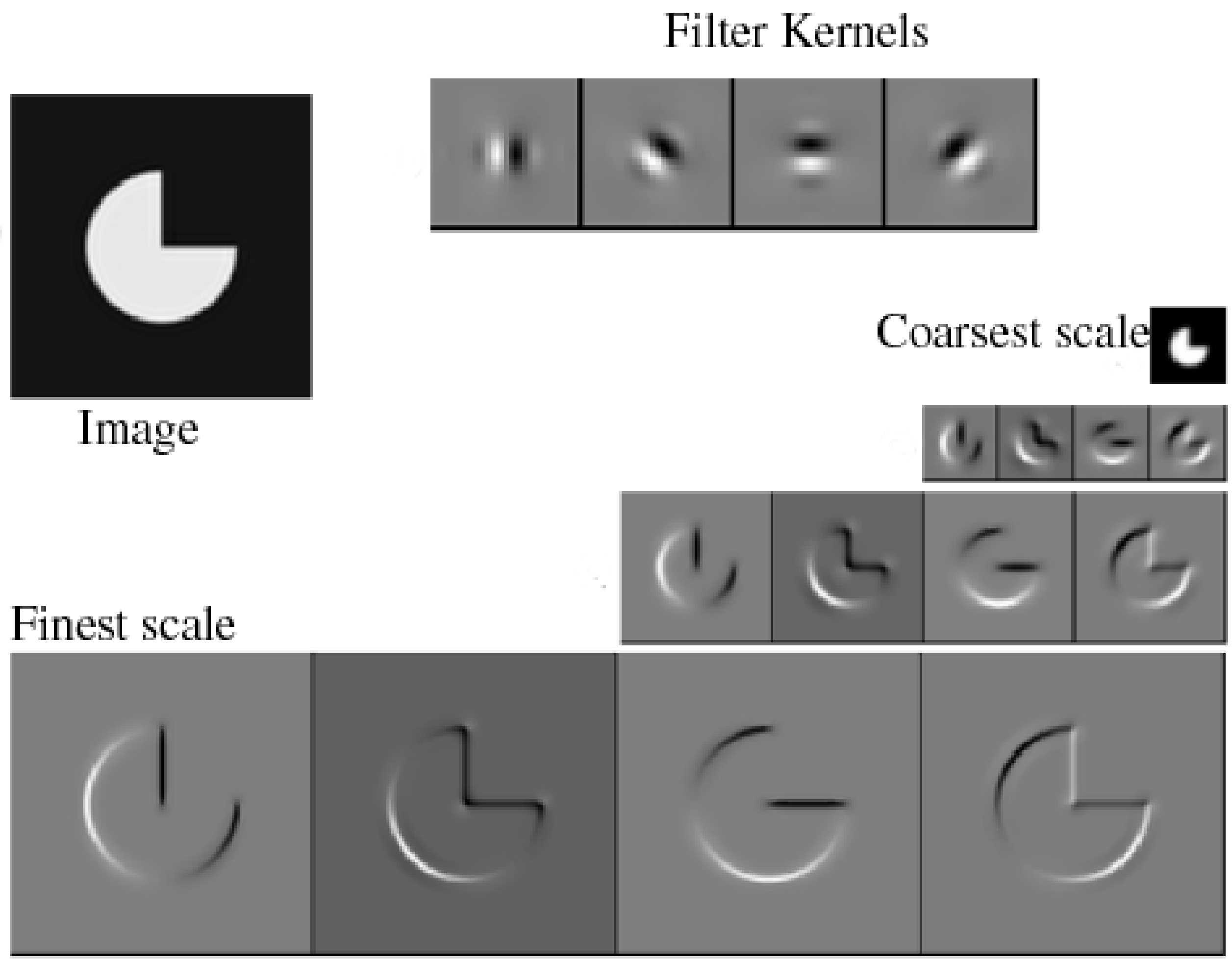
http://web.mit.edu/persci/people/adelson/pub_pdfs/pyramid83.pdf

Application for compression

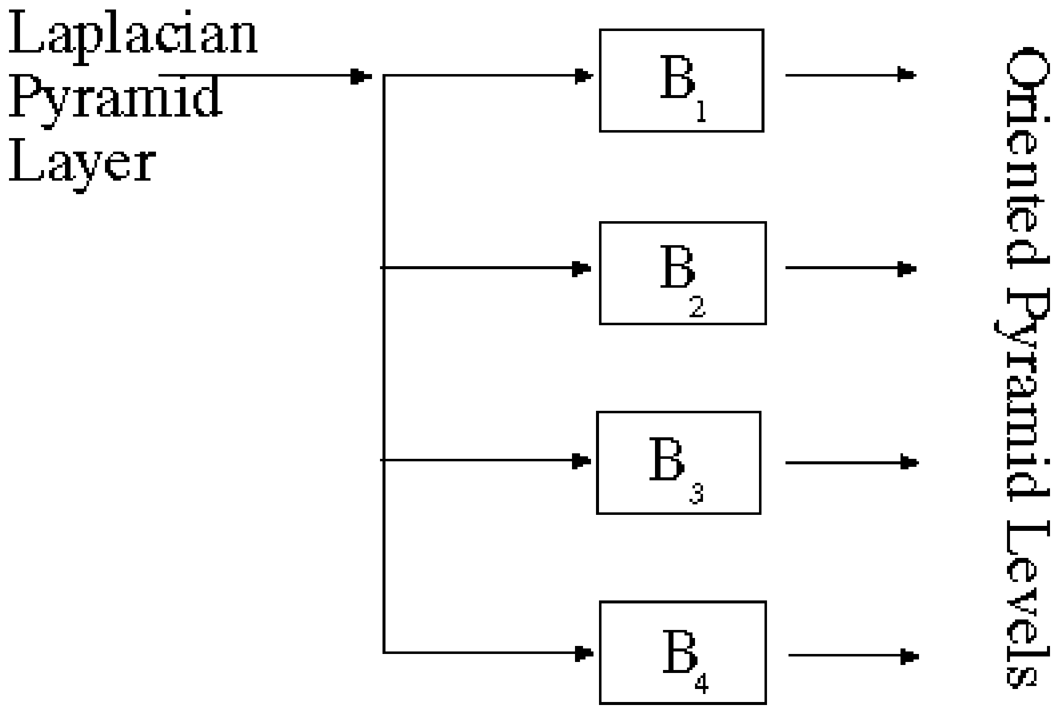


Oriented pyramids

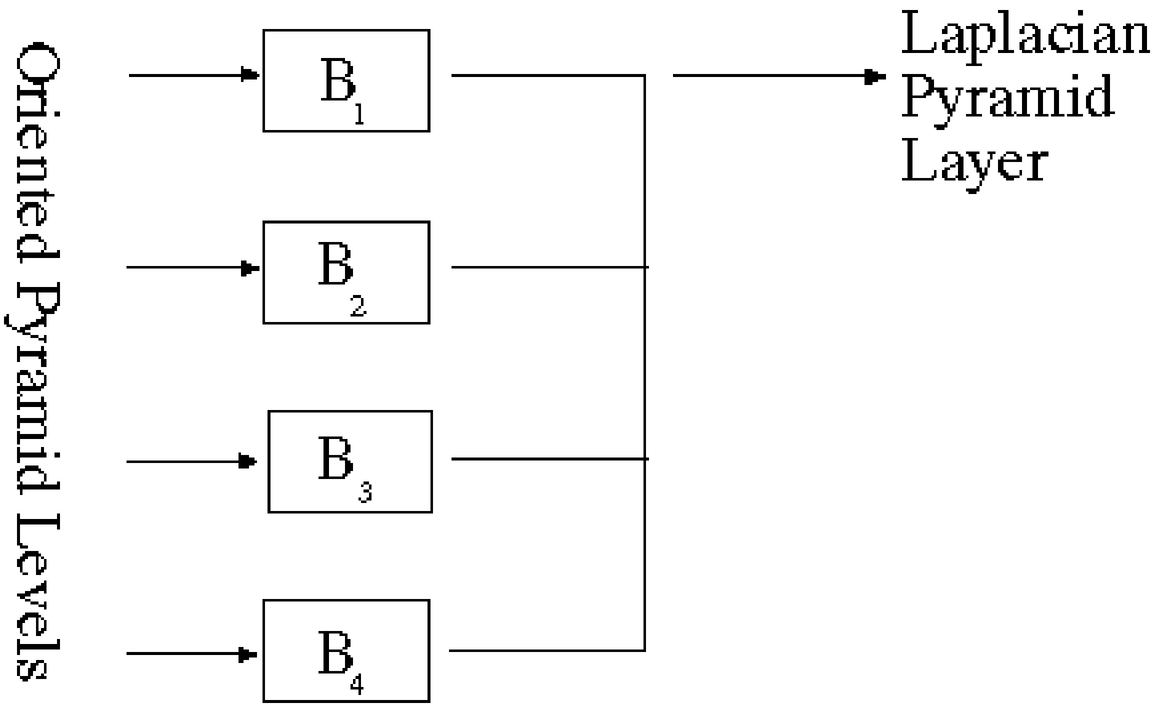
- Laplacian pyramid is orientation independent
- Apply an oriented filter to determine orientations at each layer
  - by clever filter design, we can simplify synthesis
  - this represents image information at a particular scale and orientation



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


Laplacian
Pyramid
Layer
B1
B2
B3
B4
Oriented Pyramid Levels

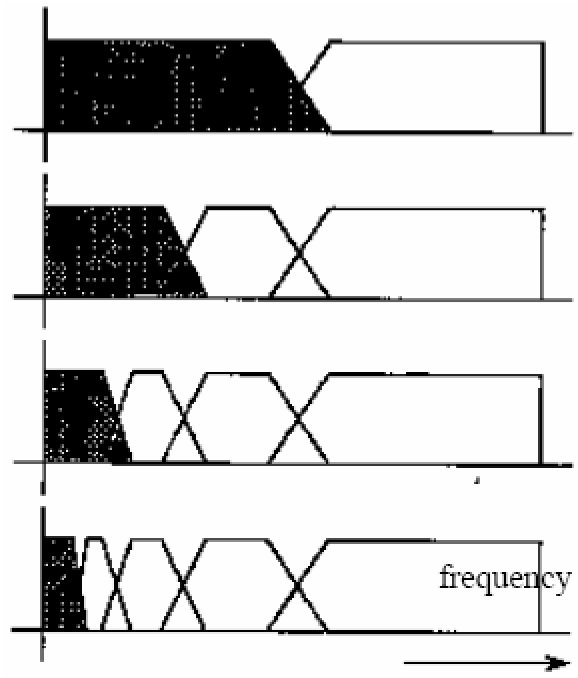
Analysis


Oriented Pyramid Levels
B_1
B_2
B_3
B_4
Laplacian Pyramid Layer

synthesis

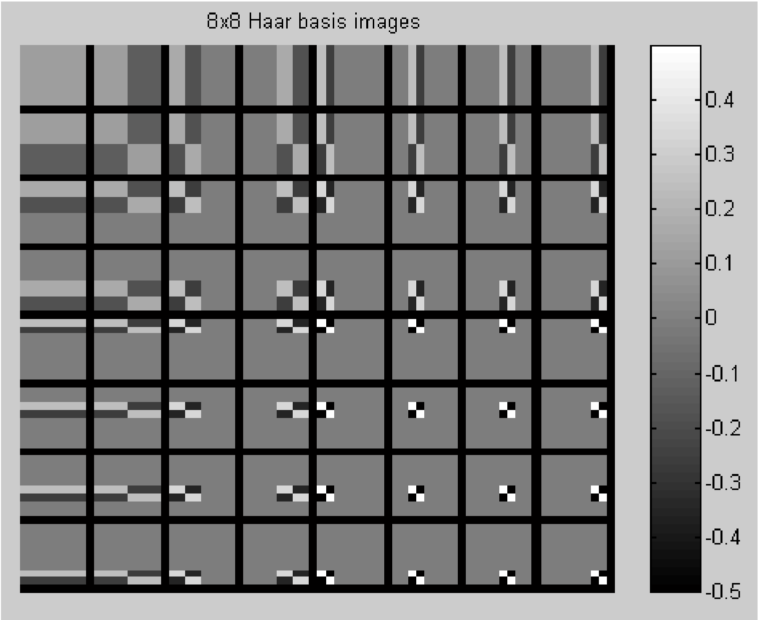
1D Discrete Wavelet Transform

- Recursive application of a two-band filter bank to the lowpass band of the previous stage yields octave band splitting:

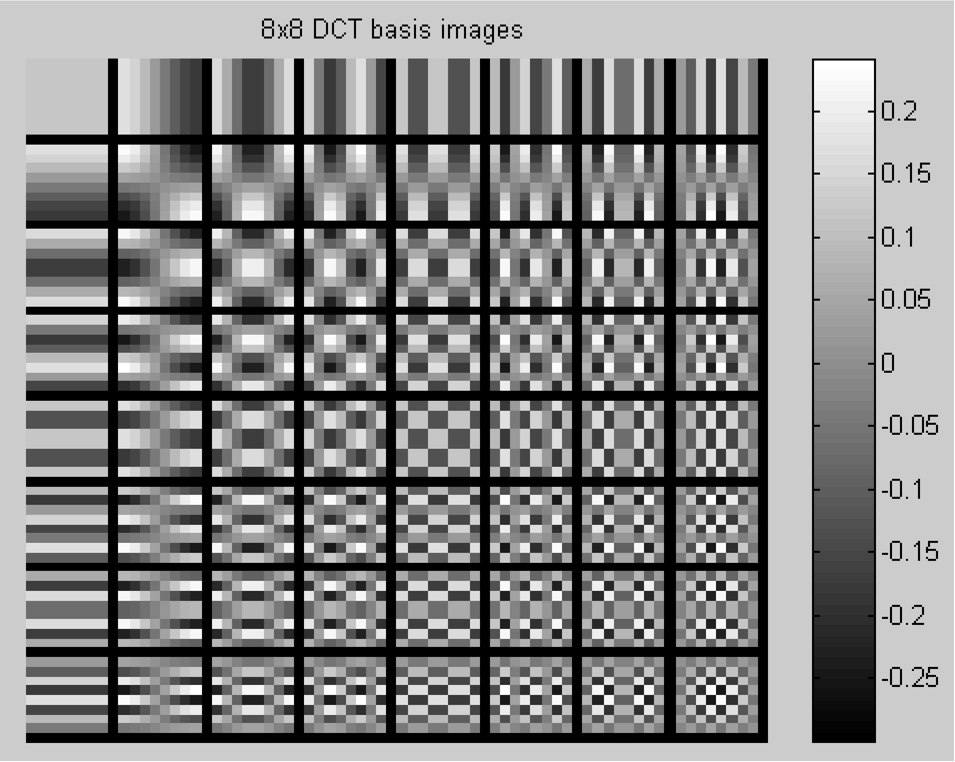


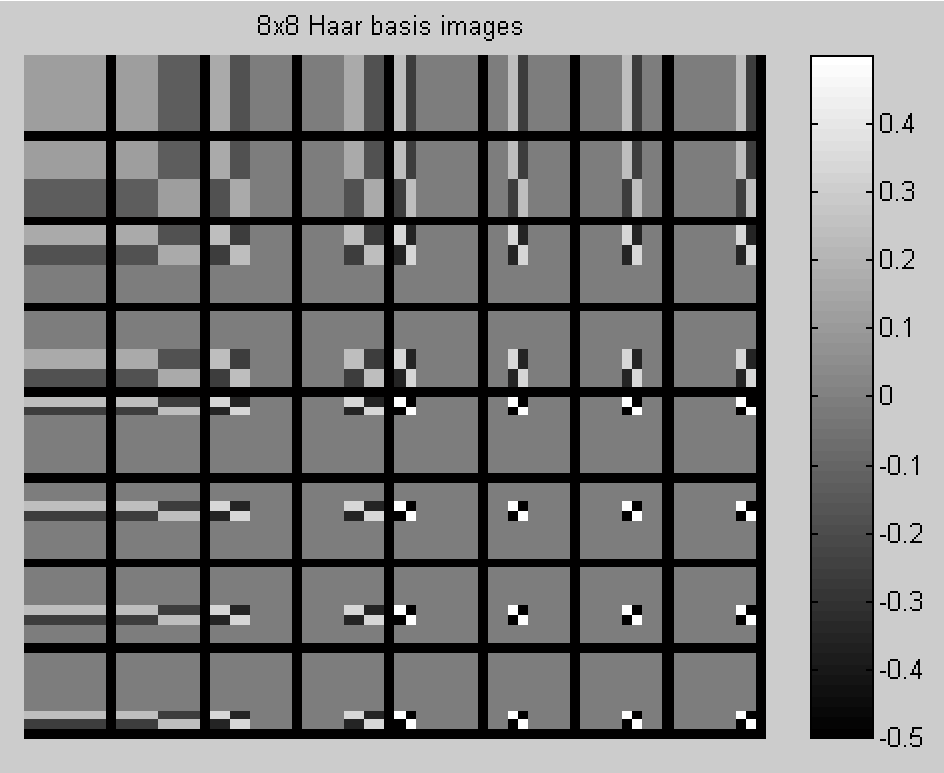
Haar Transform

- Haar transform *H*
  - Sample $h_k(x)$ at {m/N}
    - m = 0, …, N-1
  - Real and orthogonal
  - Transition at each scale *p* is localized according to *q*
- Basis images of 2-D (separable) Haar transform
  - Outer product of two basis vectors



Compare Basis Images of DCT and Haar

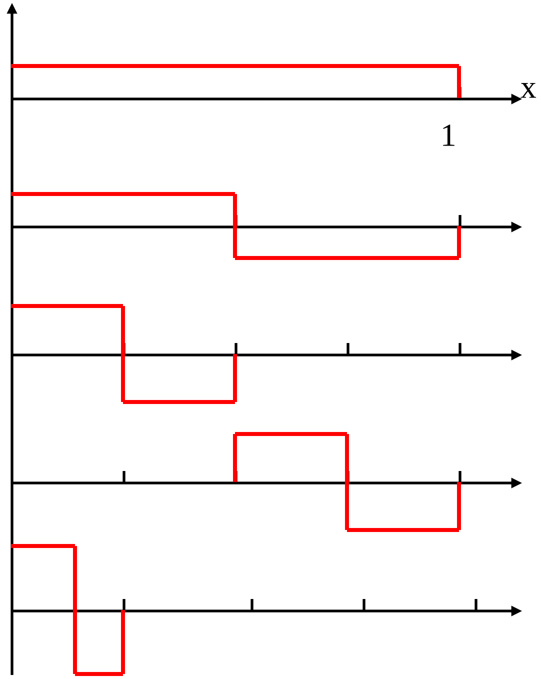


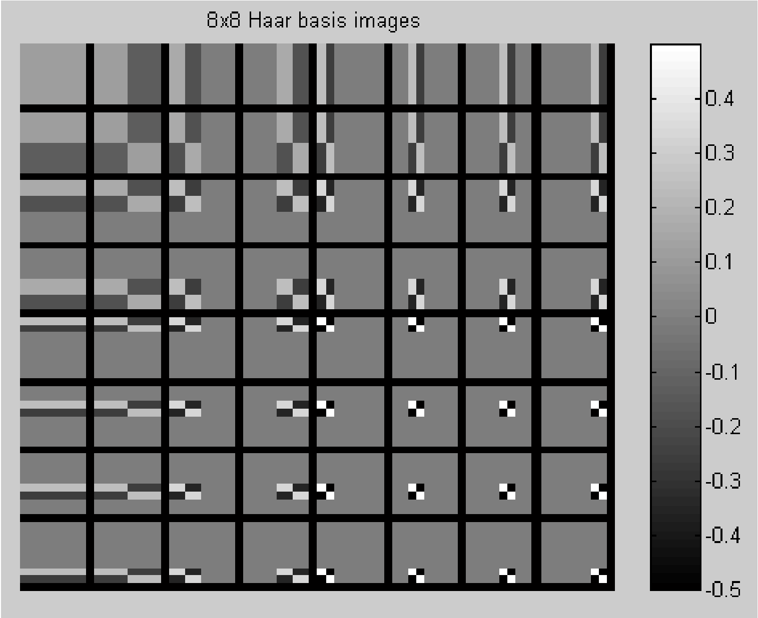


See also: Jain's Fig.5.2 pp136

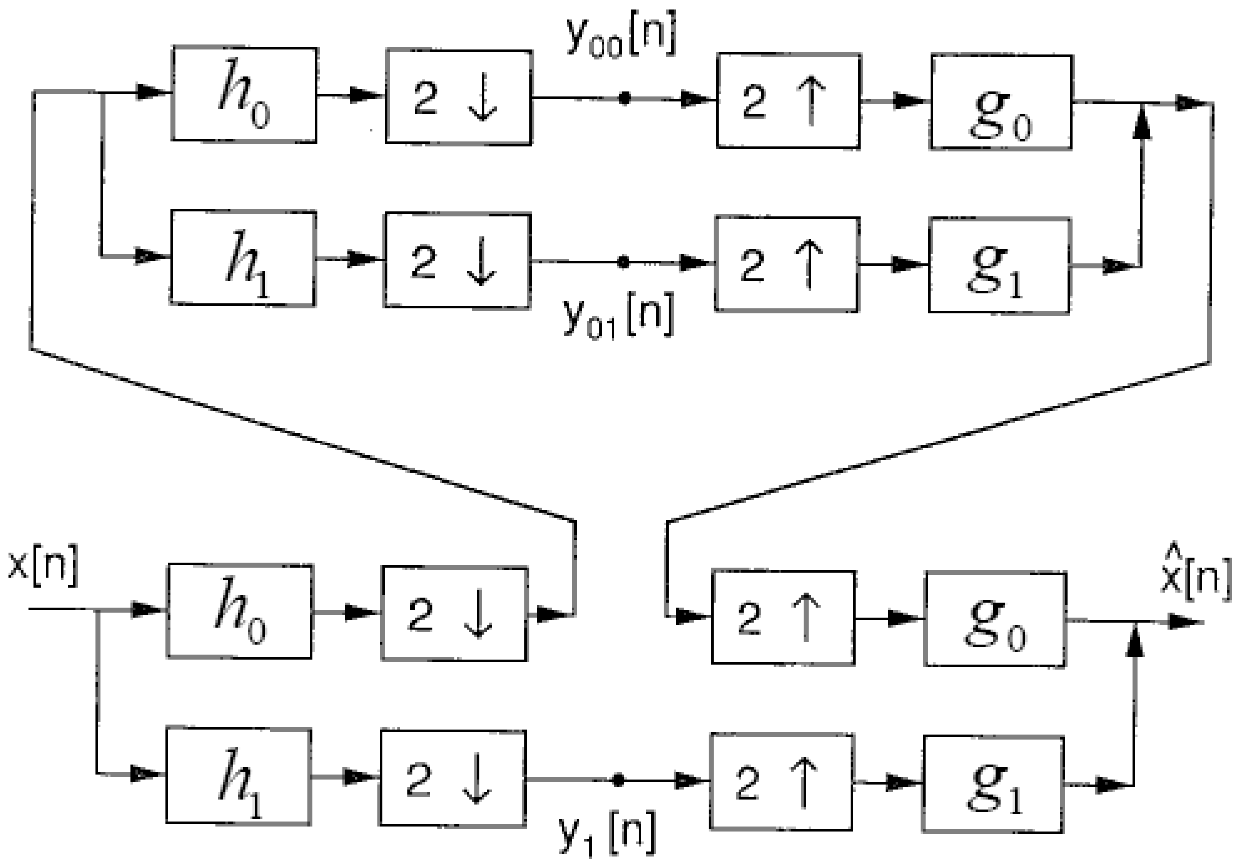
Summary on Haar Transform

- Two major sub-operations
  - Scaling captures info. at different frequencies
  - Translation captures info. at different locations
- Can be represented by filtering and downsampling
- Relatively poor energy compaction

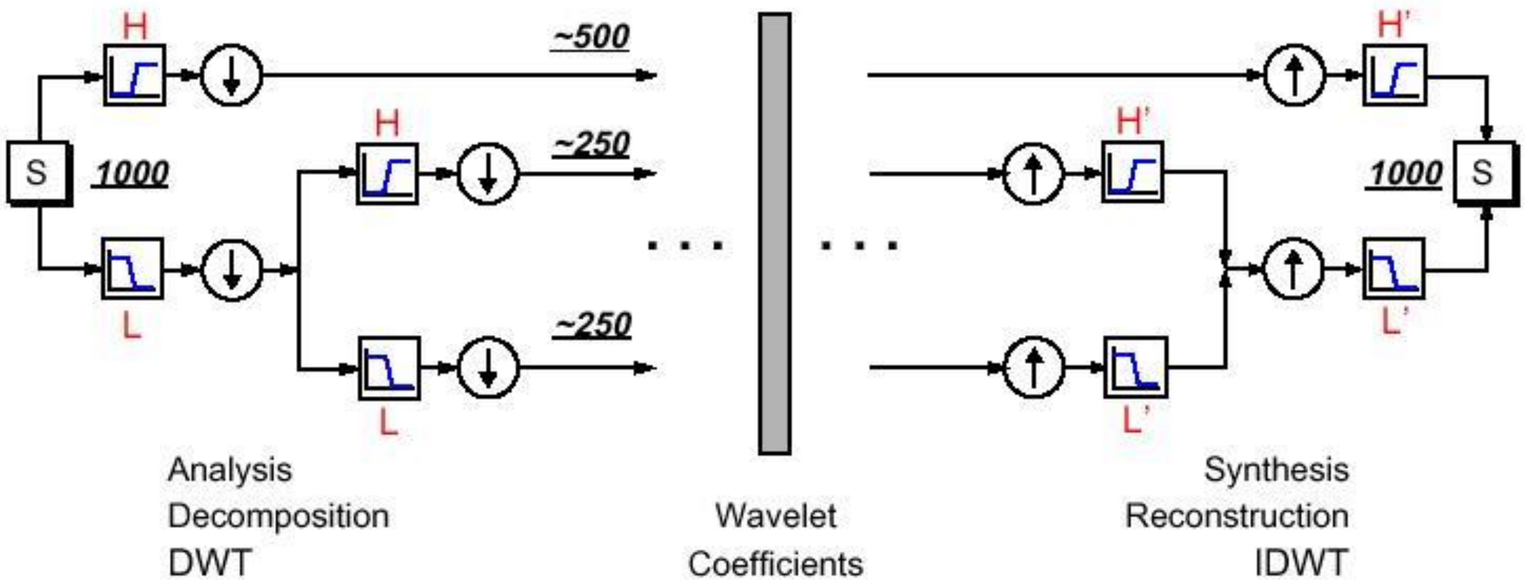




Cascade analysis/synthesis filterbanks



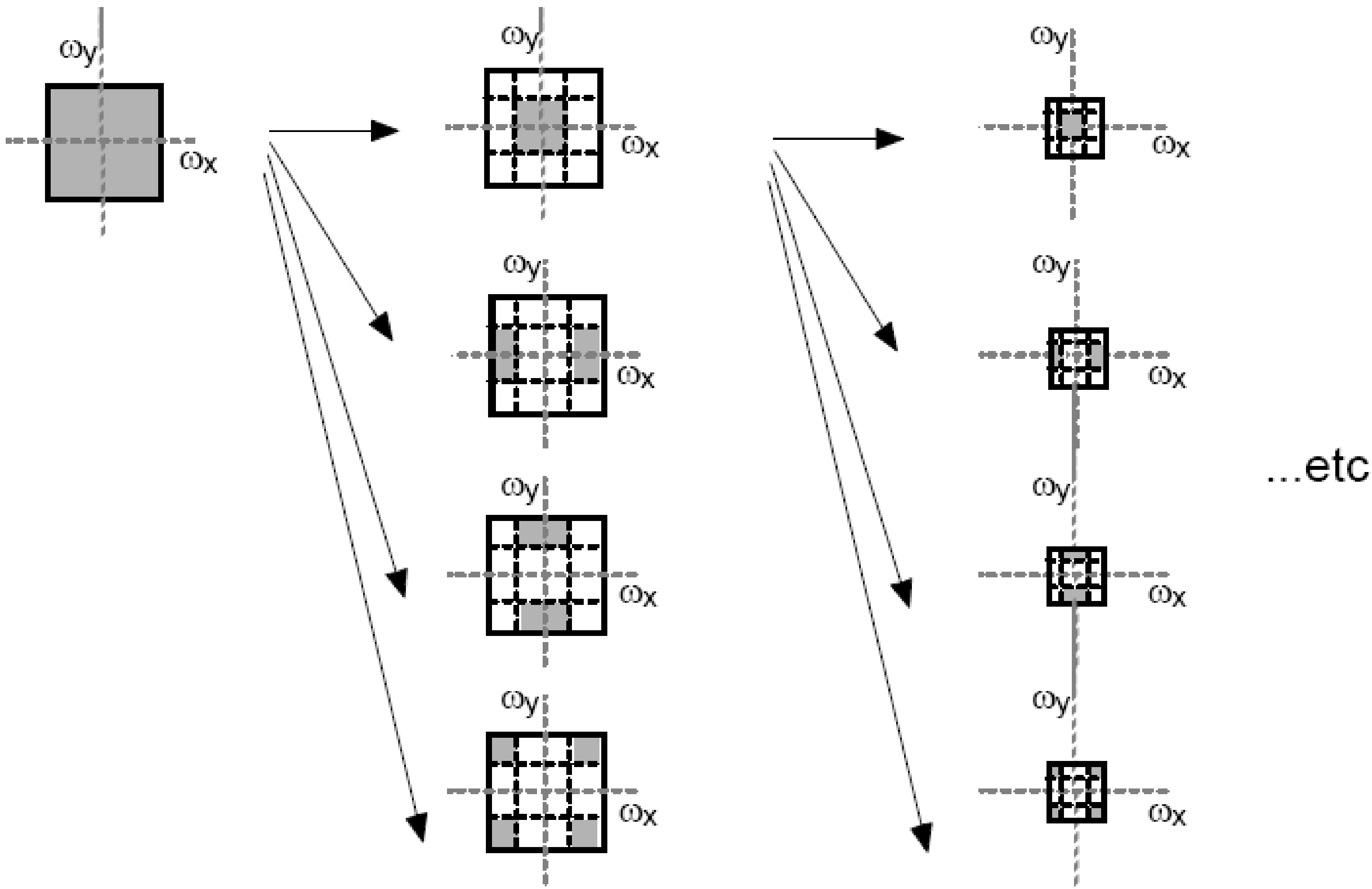
Successive Wavelet/Subband Decomposition

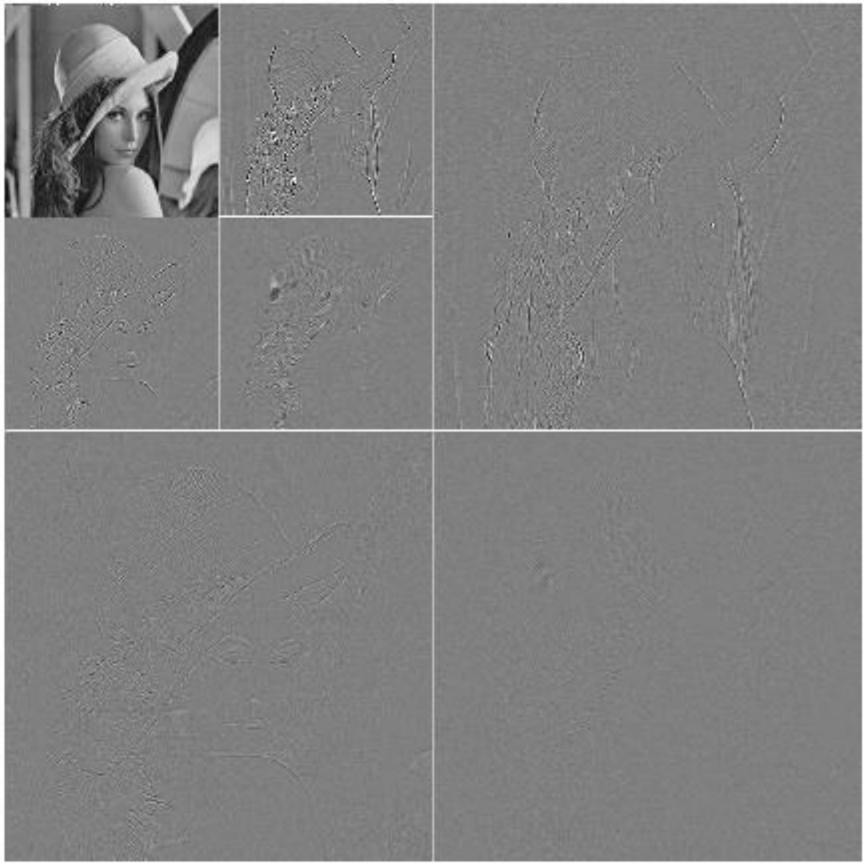


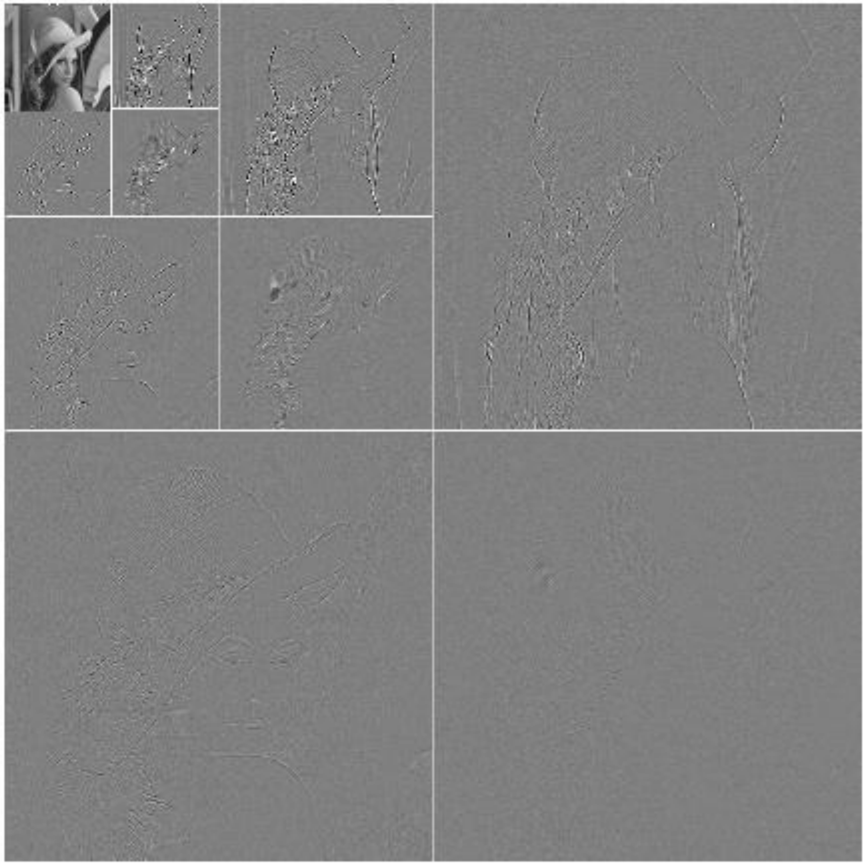
Successive lowpass/highpass filtering and downsampling

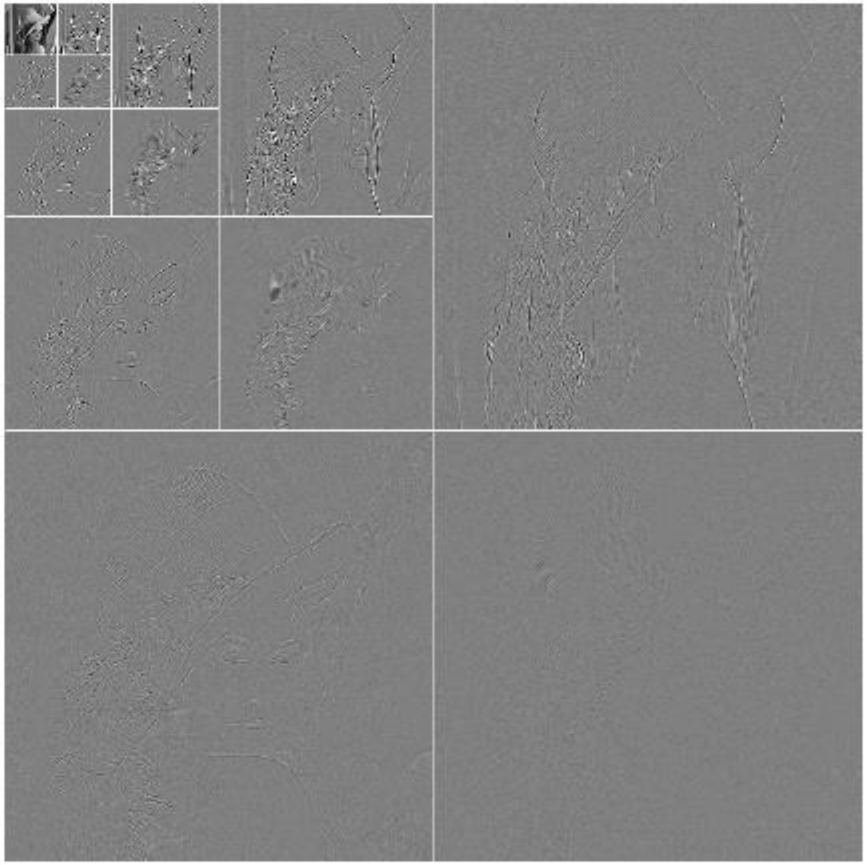
- on different level: capture transitions of different frequency bands
- on the same level: capture transitions at different locations

Figure from Matlab Wavelet Toolbox Documentation


ωy
ωx
...etc







two-filterbanks with perfect reconstruction

- Impulse responses, analysis filters:

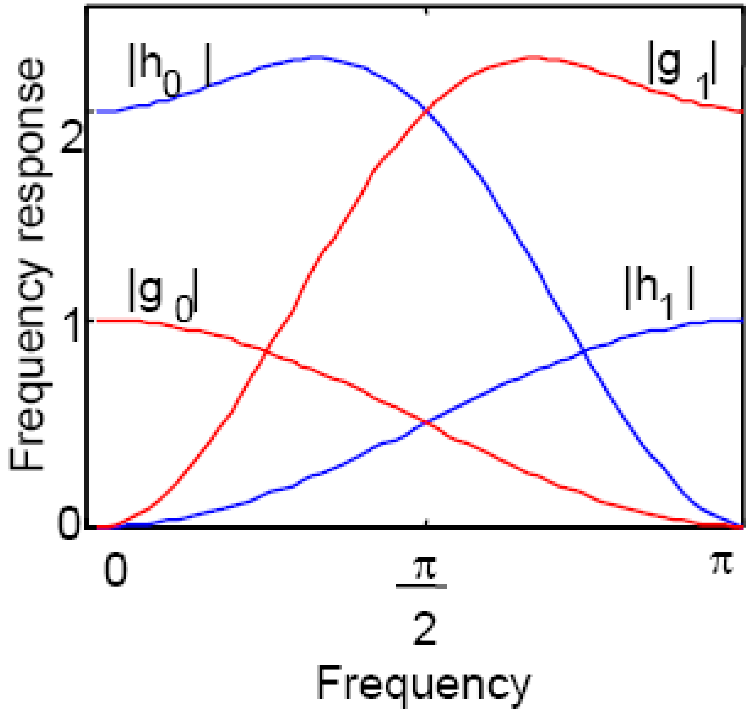
  Lowpass $\left(\frac{-1}{4},\frac{1}{2},\frac{3}{2},\frac{1}{2},\frac{-1}{4}\right)$ highpass $\left(\frac{1}{4},\frac{-1}{2},\frac{1}{4}\right)$

- Impulse responses, synthesis filters

  Lowpass $\left(\frac{1}{4},\frac{1}{2},\frac{1}{4}\right)$ highpass $\left(\frac{1}{4},\frac{1}{2},\frac{-3}{2},\frac{1}{2},\frac{1}{4}\right)$

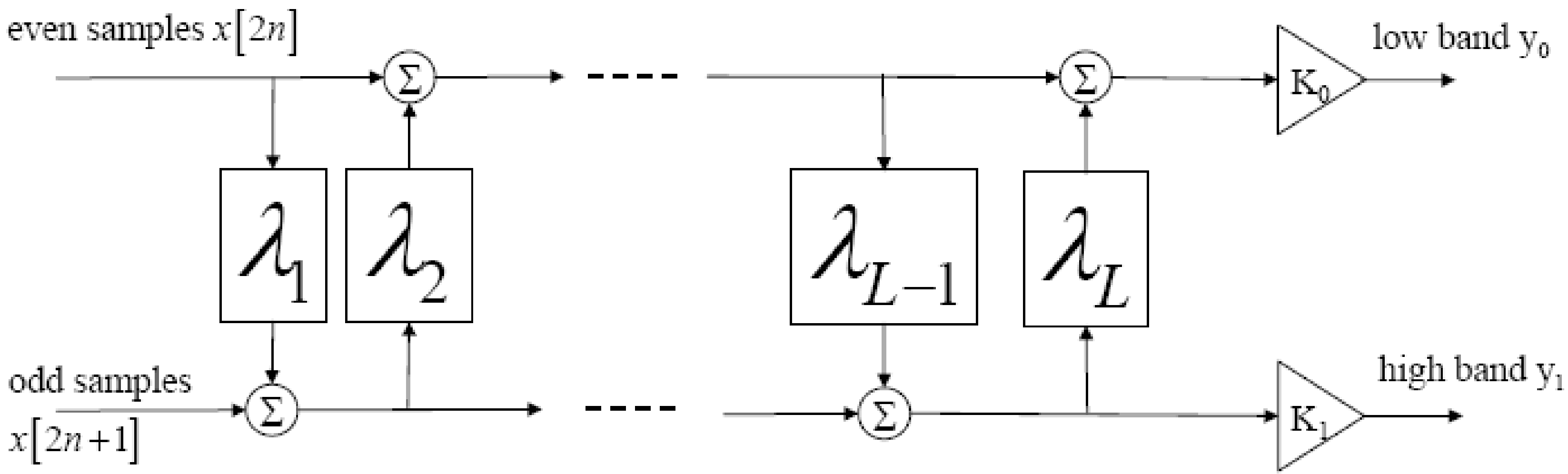
"Biorthogonal 5/3 filters"
"LeGall filters"

- Mandatory in JPEG2000
- Frequency responses:



Lifting

- Analysis filters



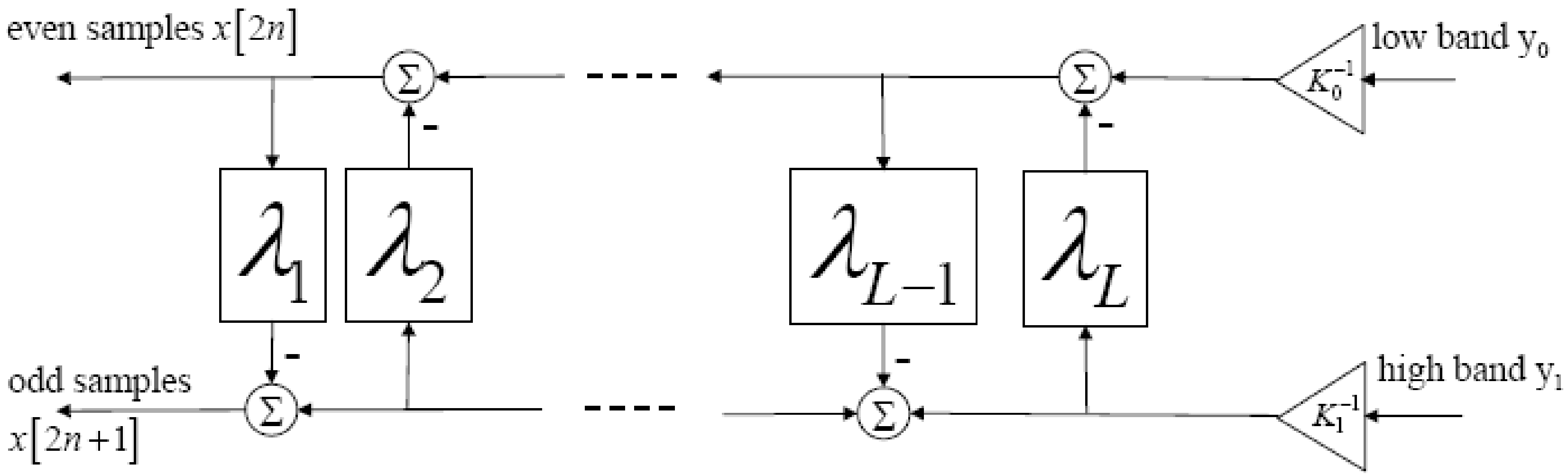
- L "lifting steps"

[Sweldens 1996]

- First step can be interpreted as prediction of odd samples from the even samples

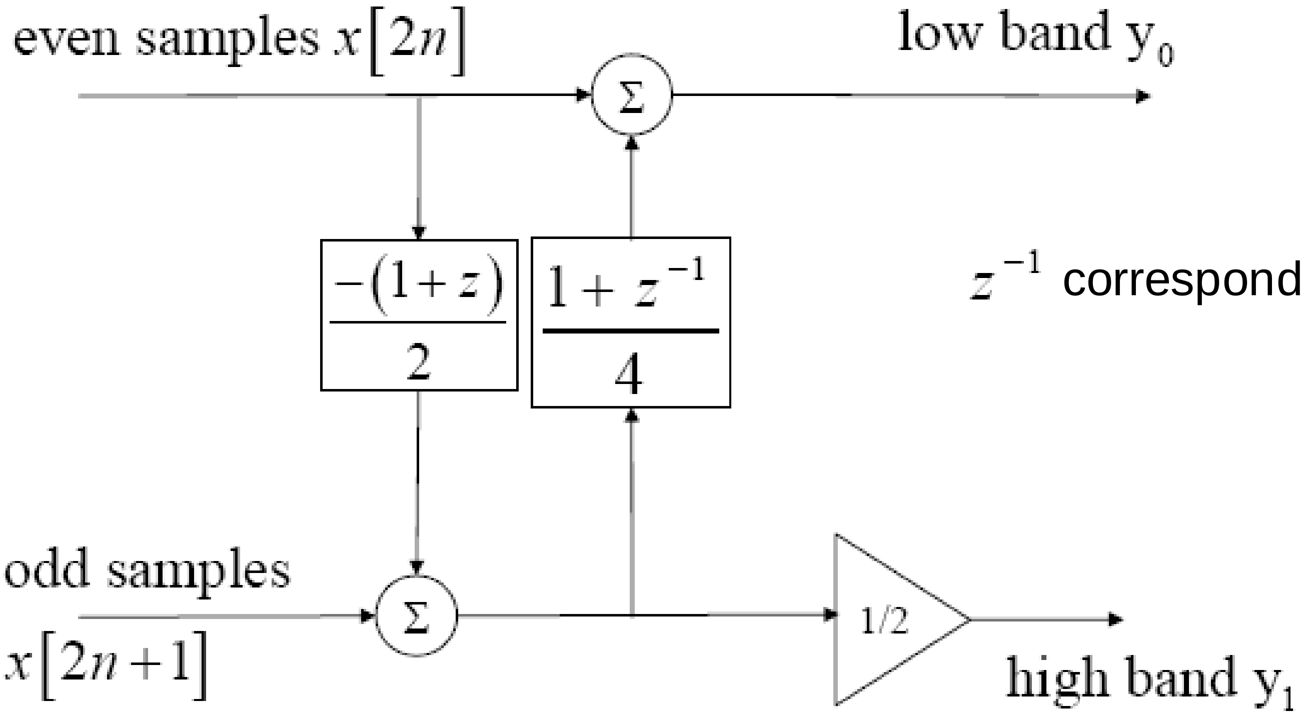
Lifting (cont.)

- Synthesis filters



- Perfect reconstruction (biorthogonality) is directly build into lifting structure
- Powerful for both implementation and filter/wavelet design

Example: Lifting implementation of 5/3 filter



z^{-1} corresponds to delay

Verify by considering response to unit impulse in even and odd input channel.

Operation flow of JPEG2000

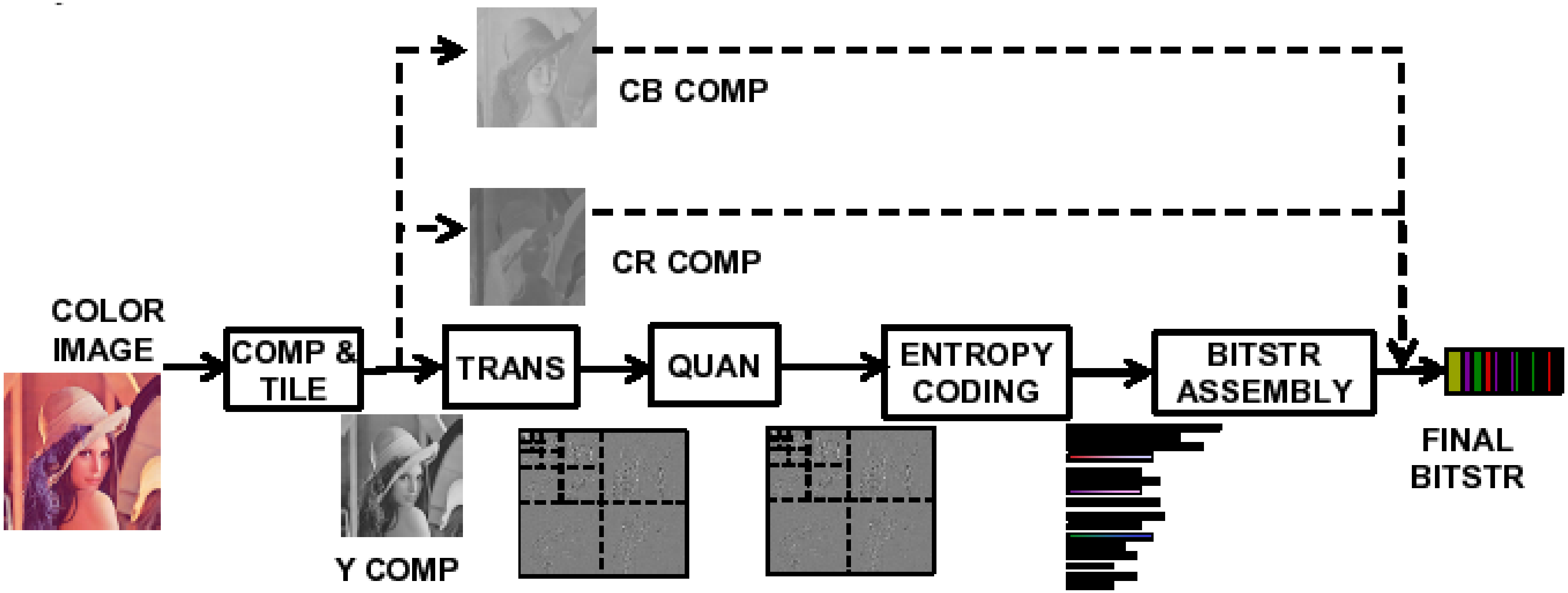


Figure 2 Operation flow of the JPEG 2000 standard.

JPEG vs. JPEG2000

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes

Lenna, 256x256 RGB
JPEG-2000: 4572 bytes

ETH

Examples

JPEG2K vs. JPEG

(a) (b)

Fig. 20. Reconstructed images compressed at 0.125 bpp by means of (a) JPEG and (b) JPEG2000

(a) (b)

Fig. 21. Reconstructed images compressed at 0.25 bpp by means of (a) JPEG and (b) JPEG2000

From Christopoulos (IEEE Trans. on CE 11/00)

Next week:

Optical flow and video compression