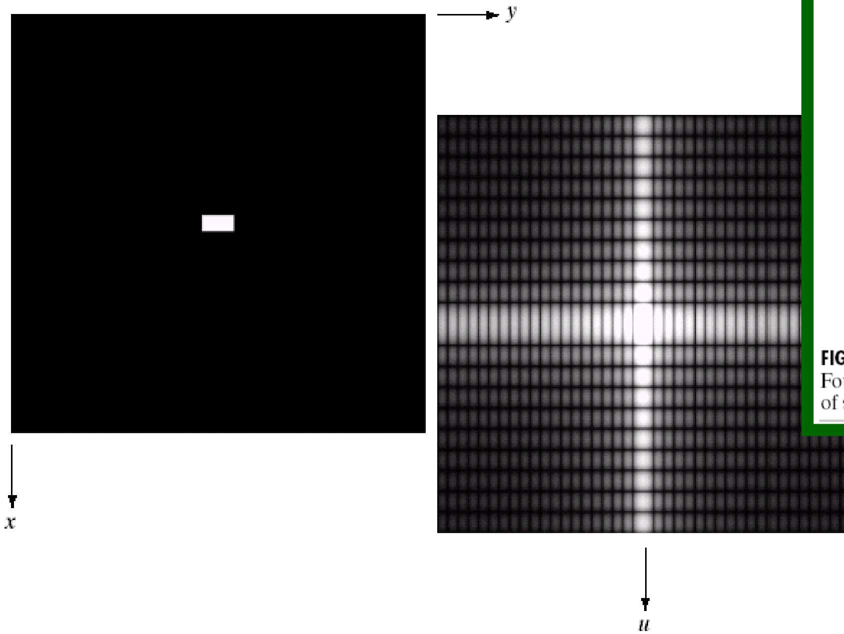
Multiresolution analysis

Multiresolution analysis

Wavelet

Fourier transform for filtering

- Frequency analysis



a b

FIGURE 4.3 (a) Image of a 20 × 40 white rectangle on a black background of size 512 × 512 pixels. (b) Centered Fourier spectrum shown after application of the log transformation given in Eq. (3.2-2). Compare with Fig. 4.2.

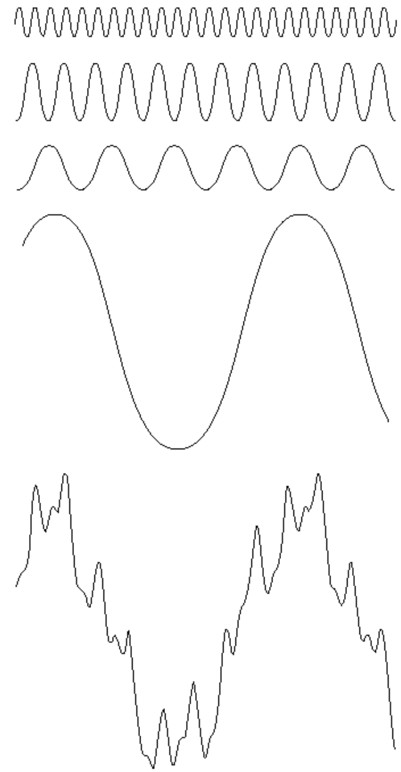
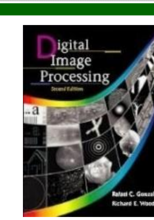
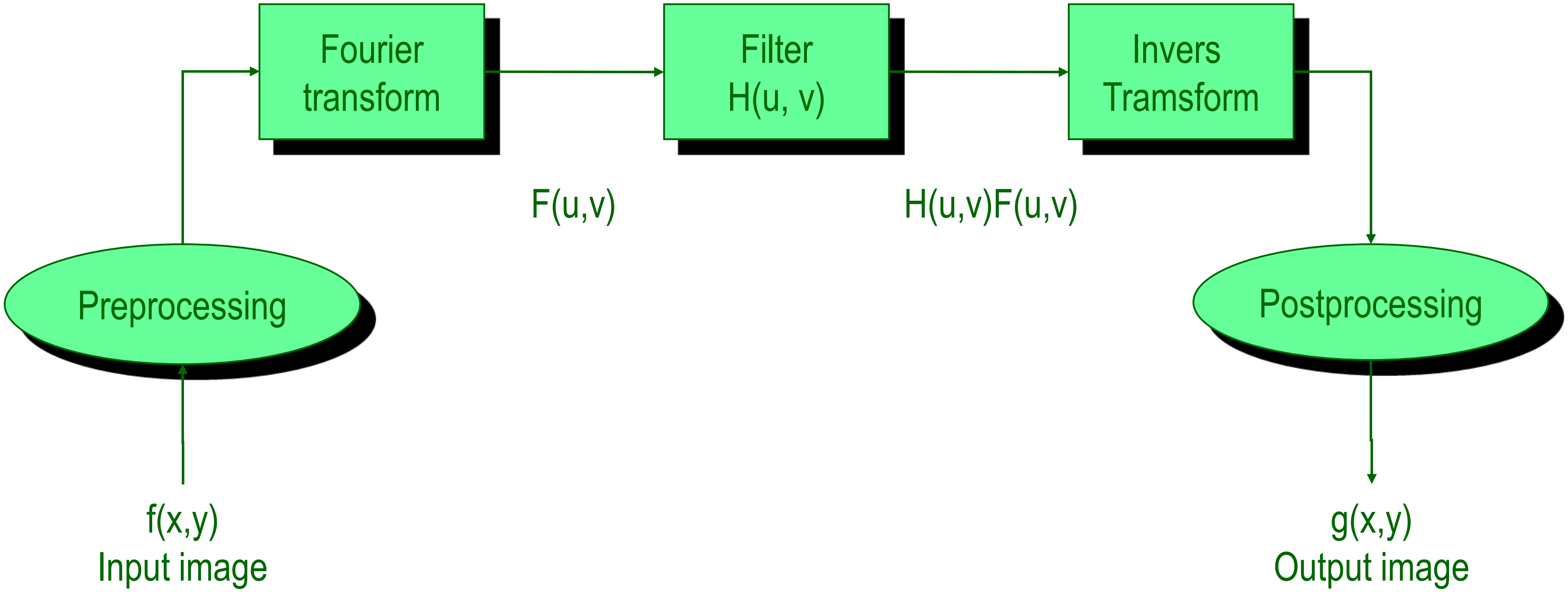


FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.



Fourier transform for filtering

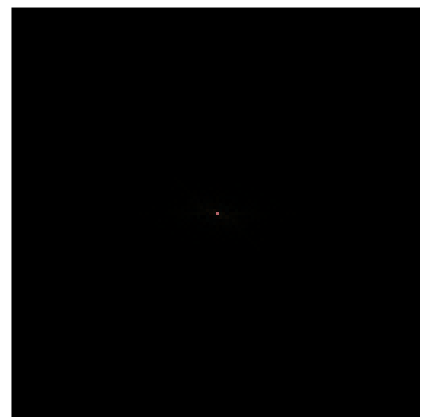


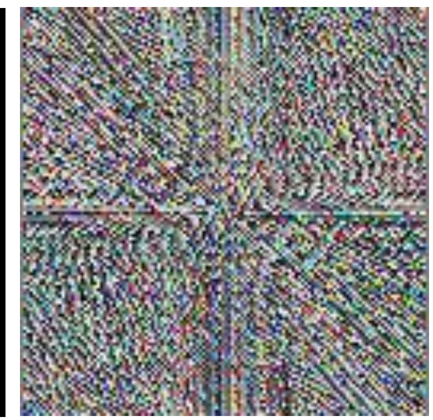
Fourier transform for filtering

```
magick lena.png -fft -ift lena_roundtrip.png
```

Fourier transform for filtering

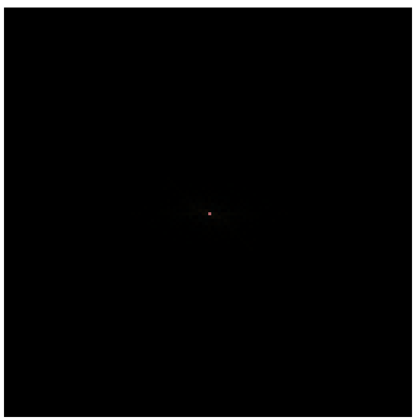
```
magick lena.png -fft +depth +adjoin lena_fft_%d.ng
```

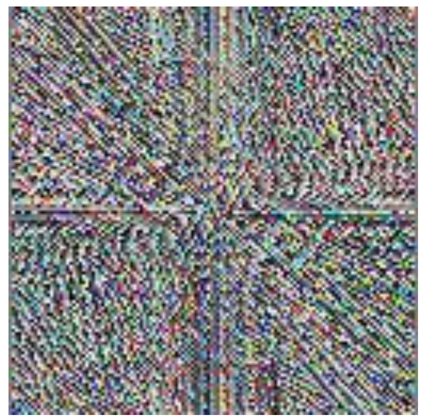




Fourier transform for filtering

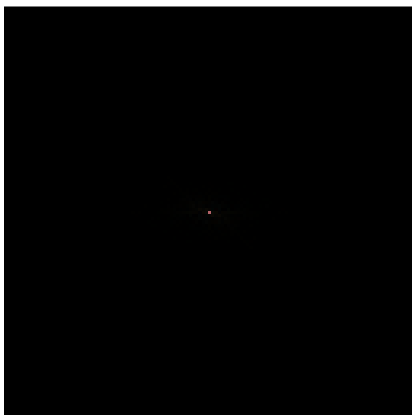
```
magick lena_magnitude.png lena_phase.png \
       -ift lena_restored.png
```

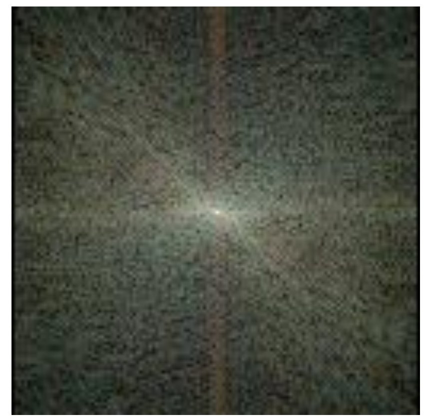




Fourier transform for filtering

```
magick lena_fft_0.png -auto-level \
      -evaluate log 10000 lena_spectrum.png
```





Effects of the DC color (zero frequency)

```
magick lena.png -fft \( -clone 0 \
      -draw "fill tomato color 64,64 point" \) \
      -swap 0 +delete -ift lena_dc_replace.png
```

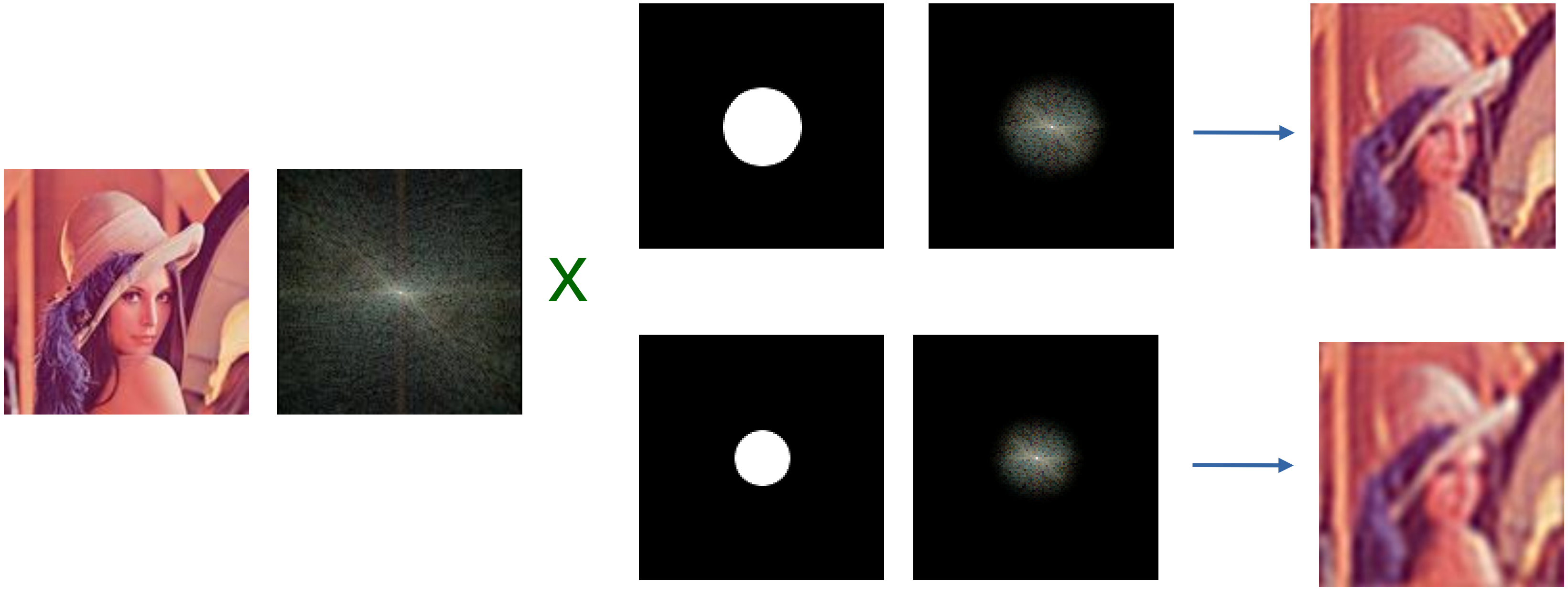
Changing the contrast of an image


```
magick lena.png -fft \( -clone 0 -evaluate pow 0.9 \) \
       -delete 0 +swap -ift lena_plus_contrast.png
magick lena.png -fft \( -clone 0 -evaluate pow 1.1 \) \
       -delete 0 +swap -ift lena_minus_contrast.png
```

Blurring an image - Low pass filtering


```
magick -size 128x128 xc:black -fill white \
      -draw "circle 64,64 44,64" circle_r20.png
magick lena.png -fft \( -clone 0 circle_r20.png \
      -compose multiply -composite \) \( +clone \
      -evaluate log 10000 \
      -write lena_blur_r20_spec.png +delete \) \
      -swap 0 +delete -ift lena_blur_r20.png
magick -size 128x128 xc:black -fill white \
      -draw "circle 64,64 50,64" circle_r14.png
magick lena.png -fft \( -clone 0 circle_r14.png \
      -compose multiply -composite \) \( +clone \
      -evaluate log 10000 \
      -write lena_blur_r14_spec.png +delete \) \
      -swap 0 +delete -ift lena_blur_r14.png
```

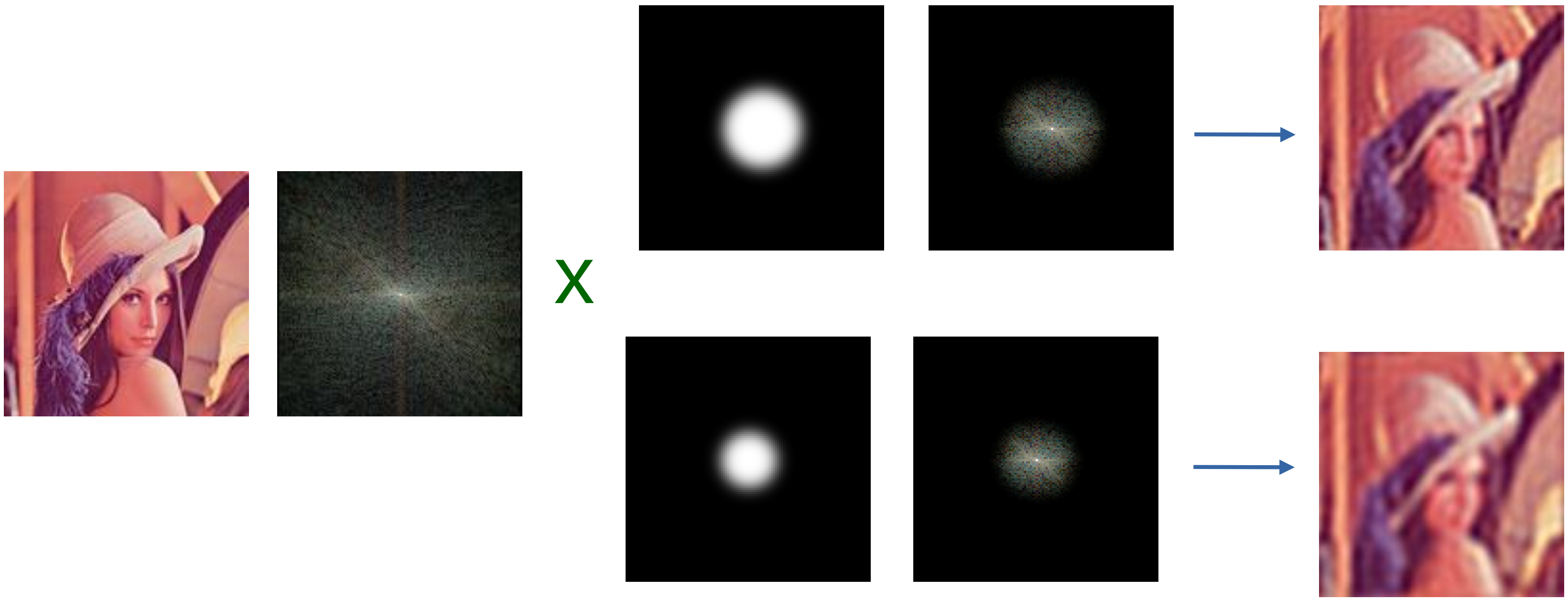
Blurring an image - Low pass filtering



Blurring an image - Low pass filtering


```
magick circle_r20.png -blur 0x4 -auto-level \
       gaussian_r20.png
magick lena.png -fft \( -clone 0 gaussian_r20.png \
       -compose multiply -composite \) \( +clone \
       -evaluate log 10000 -write lena_gblur_r20_spec.png\
       +delete \) -swap 0 +delete -ift lena_gblur_r20.png
magick circle_r14.png -blur 0x4 -auto-level \
       gaussian_r14.png
magick lena.png -fft \( -clone 0 gaussian_r14.png \
       -compose multiply -composite \) \( +clone \
       -evaluate log 10000 \
       -write lena_gblur_r14_spec.png +delete \) \
       -swap 0 +delete -ift lena_gblur_r14.png
```

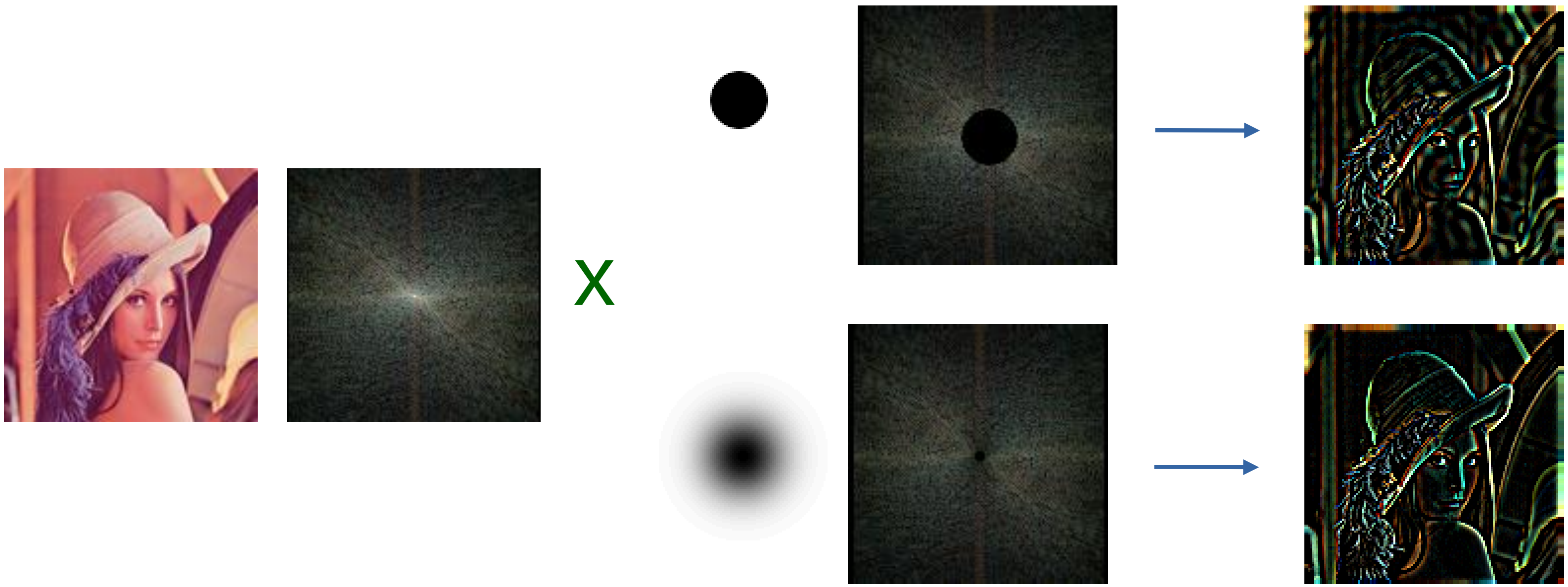
Blurring an image - Low pass filtering



Detecting edges - High pass filtering


```
magick circle_r14.png -negate circle_r14i.png
magick lena.png -fft \( -clone 0 circle_r14i.png \
       -compose multiply -composite \) \( +clone \
       -evaluate log 10000 -write lena_edge_r14_spec.png \
       +delete \) -delete 0 +swap -ift \
       -normalize lena_edge_r14.png
magick -size 128x128 xc: -draw "point 64,64" -blur 0x14 \
       -auto-level gaussian_s14i.png
magick lena.png -fft \( -clone 0 gaussian_s14i.png  \
       -compose multiply -composite \) \
       \( +clone -evaluate log 10000 \
       -write lena_edge_s14_spec.png +delete \) \
       -delete 0 +swap -ift -normalize lena_edge_s14.png
```

Detecting edges - High pass filteringg

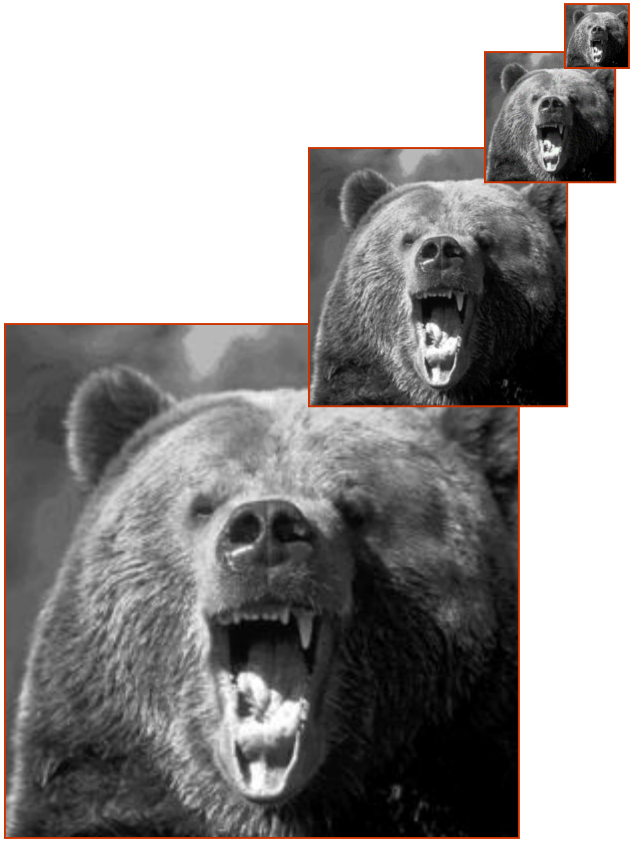


Sharpening an image

```
magick lena.png -fft \( -size 128x128 xc: \
      -draw "point 64,64" -blur 0x14 -auto-level \
      -clone 0 -compose multiply -composite \) \
      -delete 0 +swap -ift lena.png -compose blend
      -set option:compose:args 100x100 \
      -composite lena_sharp14.png
```

An image pyramid

- An image pyramid is a set of images of decreasing resolution
- Normally the original (base) image is a square image of size power of 2
- The various levels of the pyramid (successive approximations) are versions of half (linear) size, the number of levels depends on the power of 2
- The top level of the pyramid (apex) is size 2^0 (but often the pyramid is truncated earlier)
- The memory required to memorize a pyramid differs slightly from the starting image



$$N^2\left(1+\frac{1}{(4)^1}+\frac{1}{(4)^2}+\ldots+\frac{1}{(4)^k}\right)<\frac{4}{3}N^2$$

Gaussian pyramid

- The levels are obtained through subsampling (one pixel every two is kept) normally preceded by a low-pass filter (Gaussian filter)
- In this case we talk about a Gaussian pyramid
- Normally the filters used have the properties:
  - they can be broken down (first one-dimensional filtering is performed by rows and then by columns)
  - the sum of the even weights is equal to that of the odd weights (equal to ½)
  - the size is odd
- Examples [¼, ½, ¼]

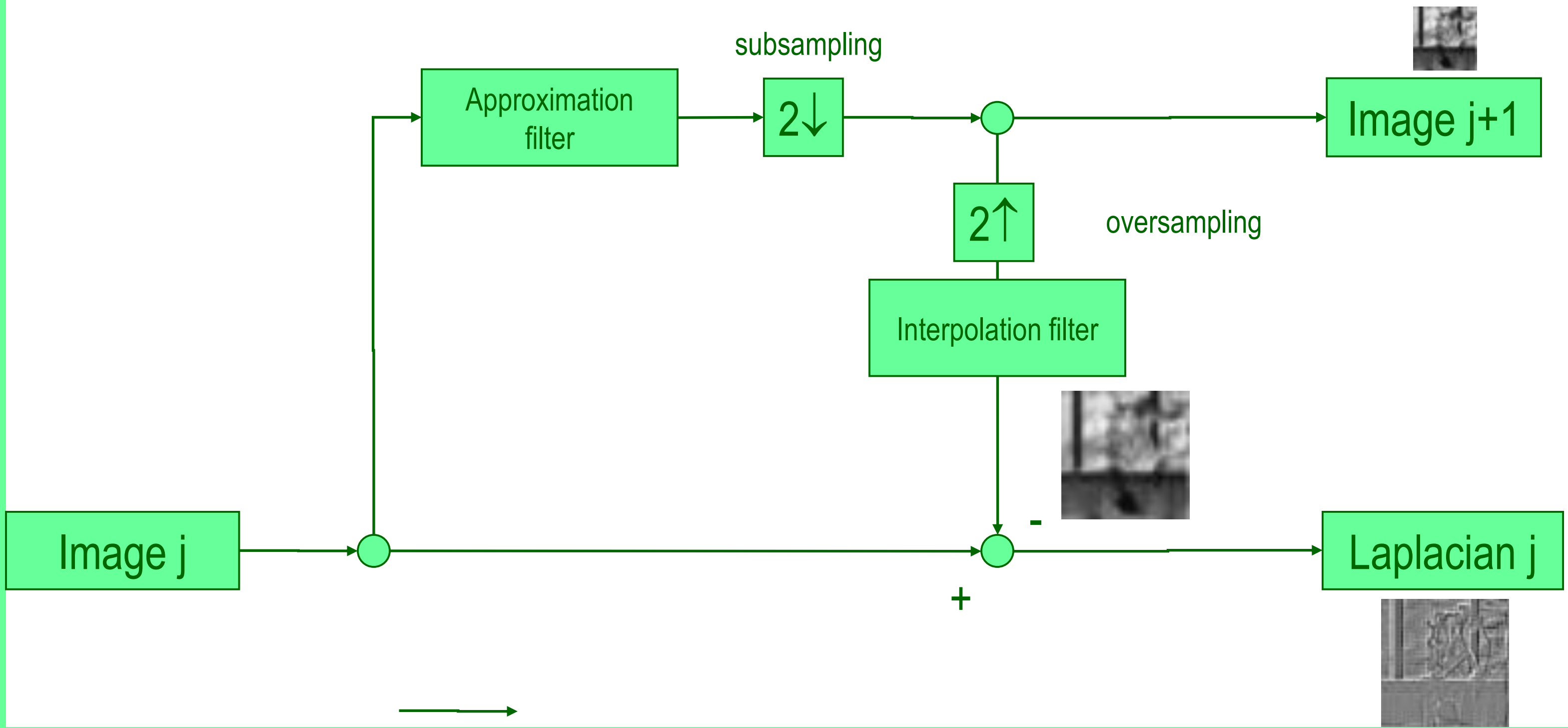
  [1/20, 5/20, 8/20, 5/20, 1/20] = [1/20, 1/4, 2/5, 1/4, 1/20]

  [1/16, 4/16, 6/16, 4/16, 1/16]

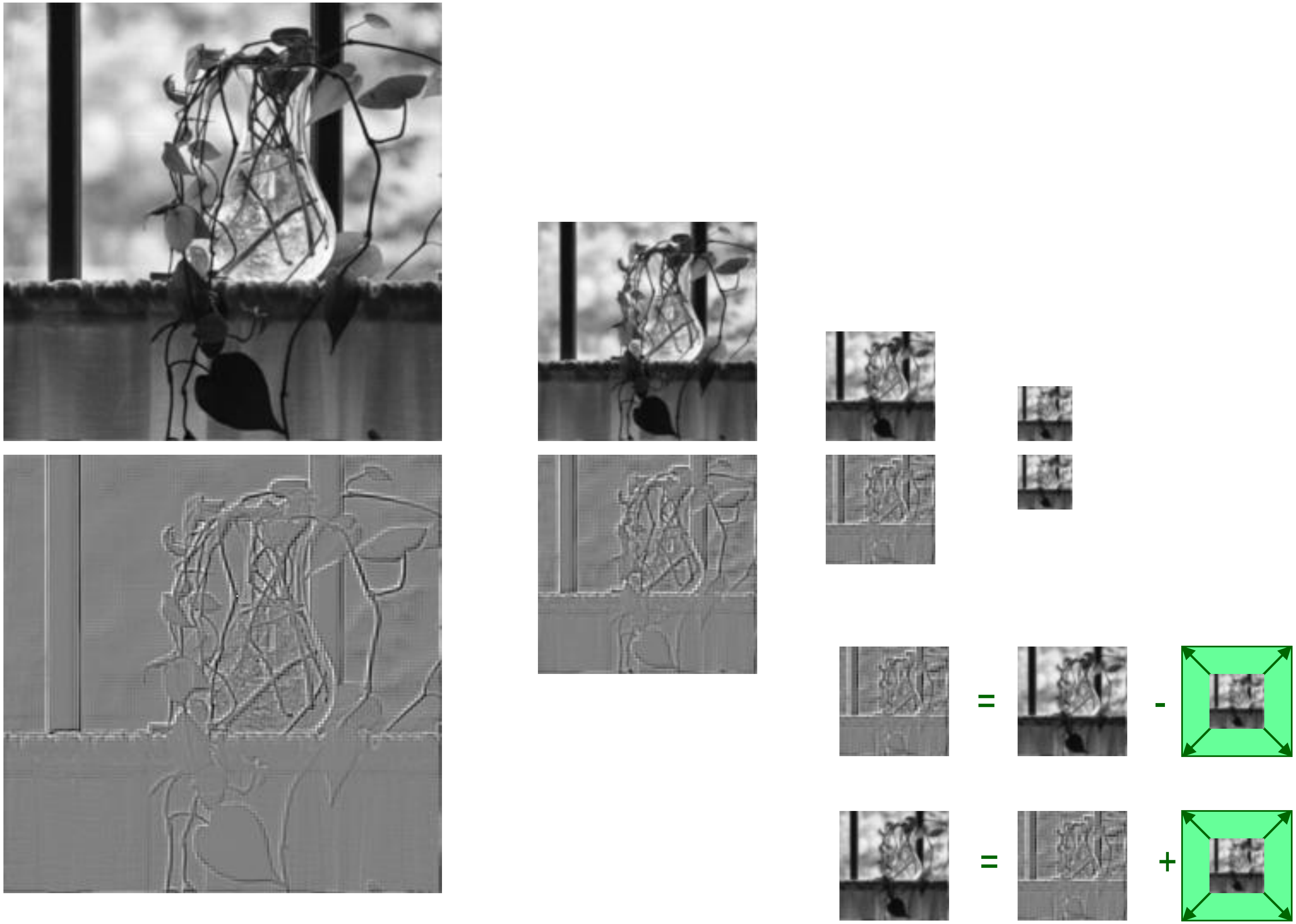
  Larger filters are expensive in terms of computation time

Laplacian pyramid

- It is the set of images that contain information relating to decreasing frequency bands (and the apex of a Gaussian image)
- Each level is obtained from the difference between the image of level j of a Gaussian pyramid and its approximation obtained from level j+1
- In theory, it is possible to reconstruct the original (not stored) image



Laplacian pyramid



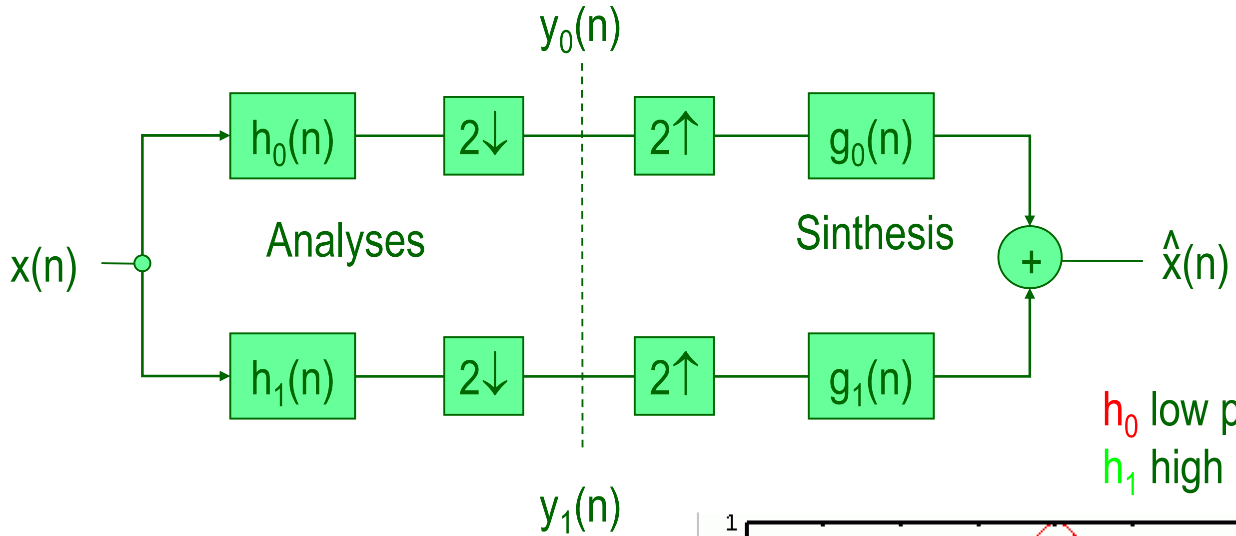
The wavelet transform

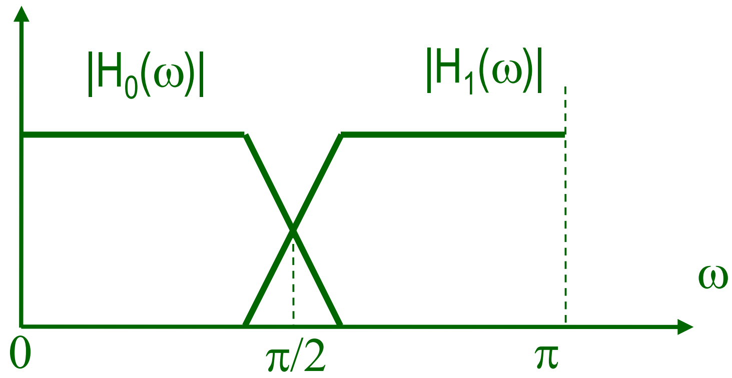
- The wavelet transform is capable of offering high compression efficiency and, at the same time, spatial localization and variable resolution analysis capabilities that are impossible with traditional transforms
- The traditional approach to the analysis of a 1-D f(t) signal involves its representation through a weighted sum of basis functions:

$$f(t) = \sum_i c_i \psi_i(t)$$

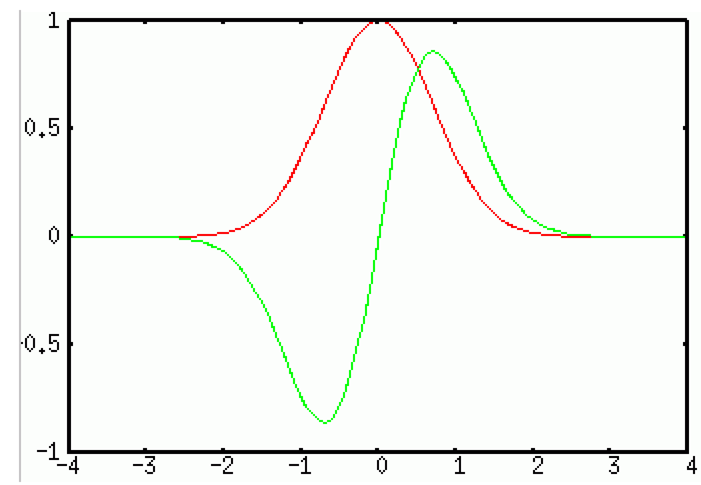
- The basic functions are fixed, once and for all, by the type of transformation used, so all the information on the signal is contained in the weights, which are the coefficients of the transformation itself
- For example, basis functions are complex trigonometric functions in the case of the Fourier transform, real trigonometric ones for the DCT, etc.

The wavelet transform

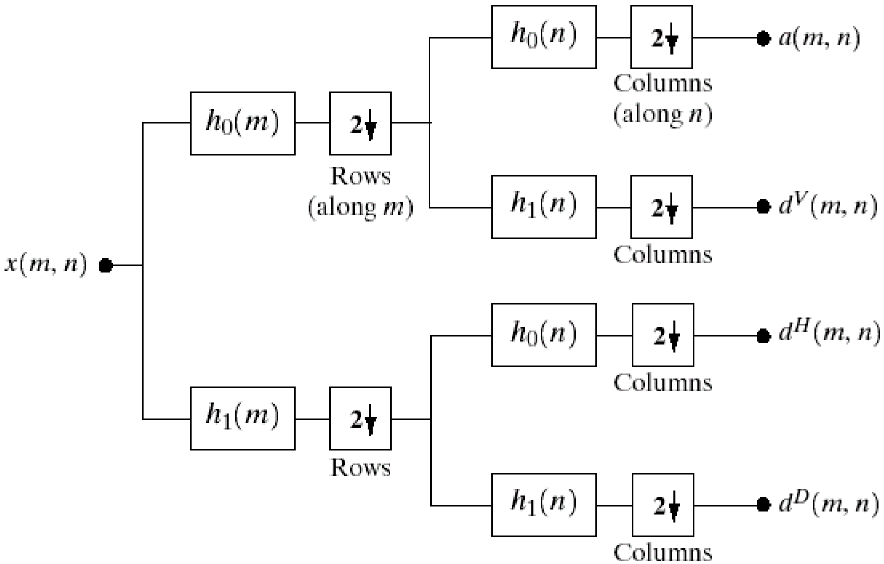


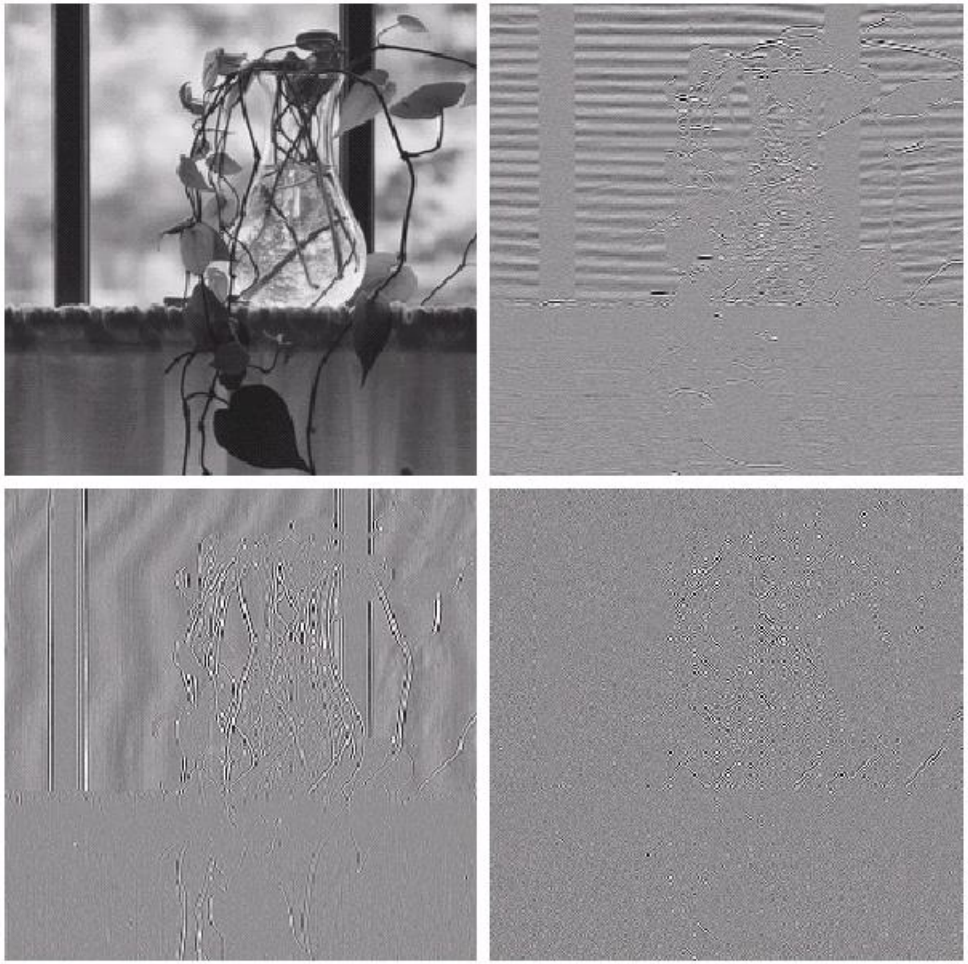


h_0 low pass filter
h_1 high pass filter

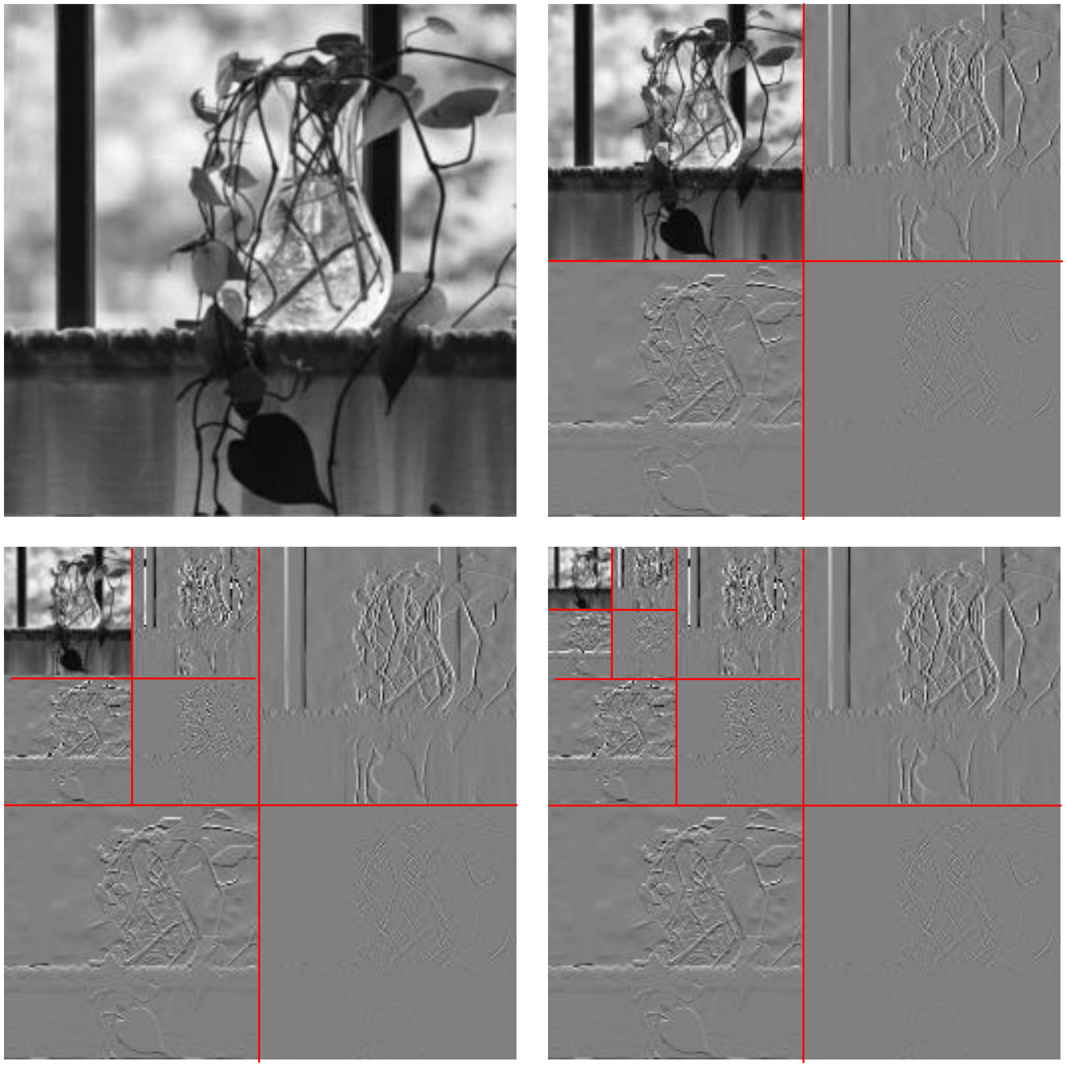


The wavelet transform

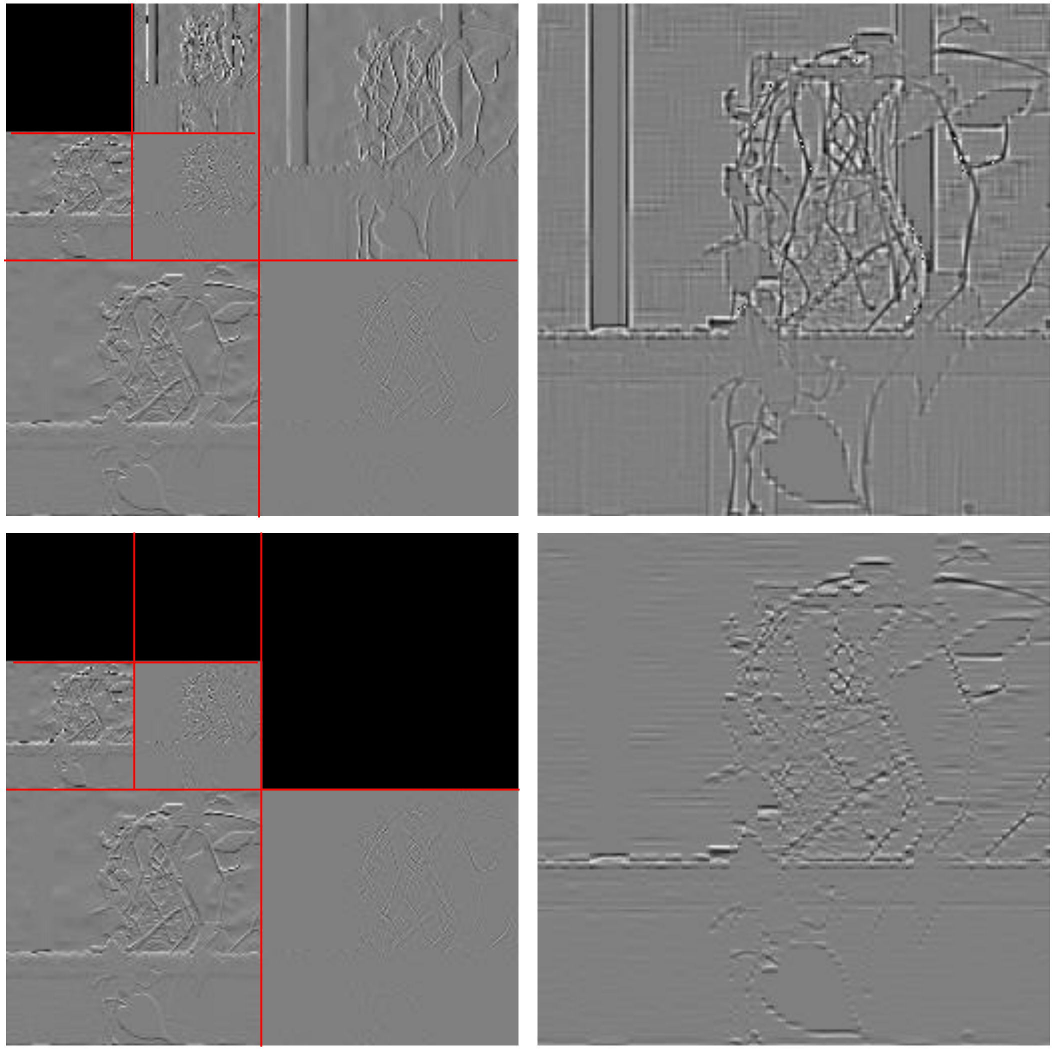




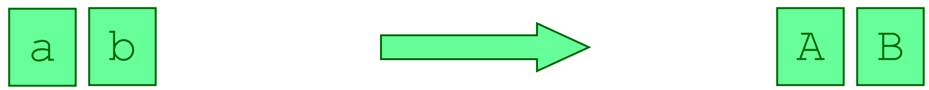
The wavelet transform



Example: Edge detection



Haar transform



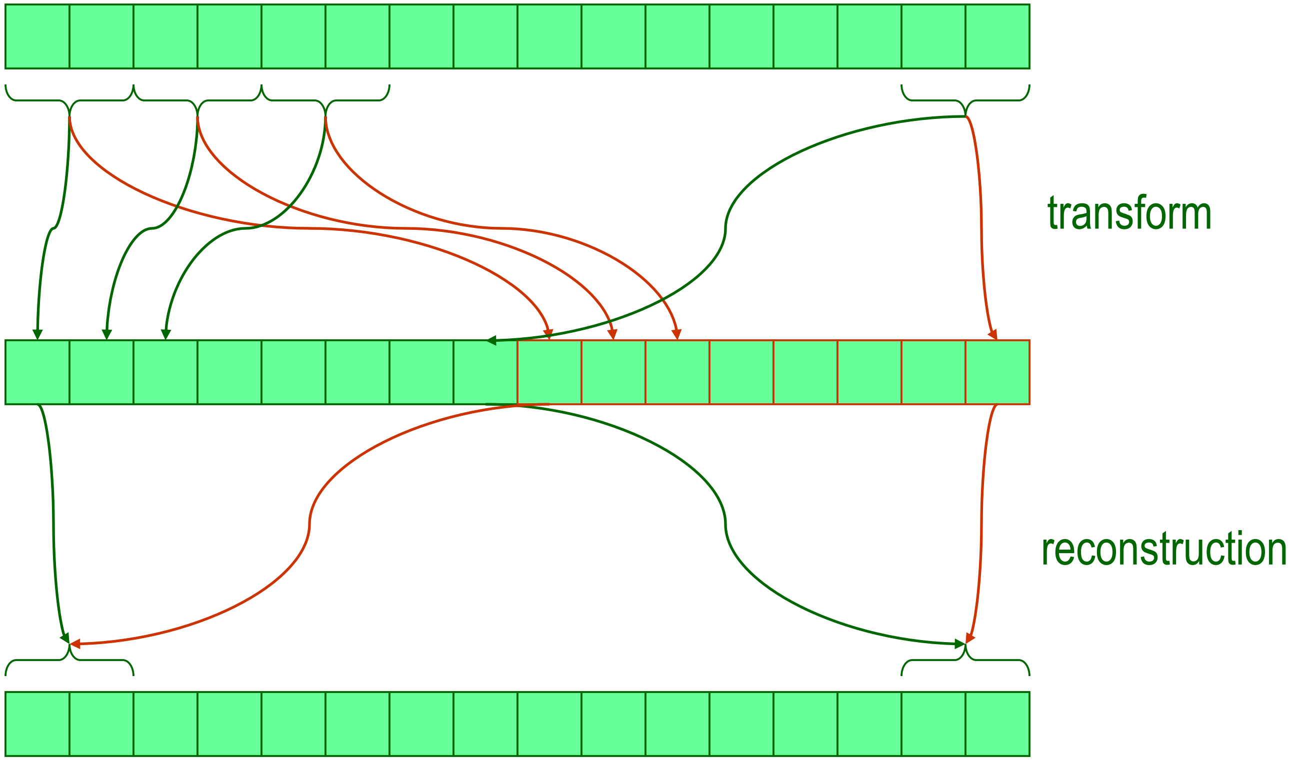
A=(a+b)/2 (it is an average, low pass filter))

B=(b-a)/2 (it is a differential filter, a high pass filter)

Reconstruction a=A-B b=A+B exactly the starting values

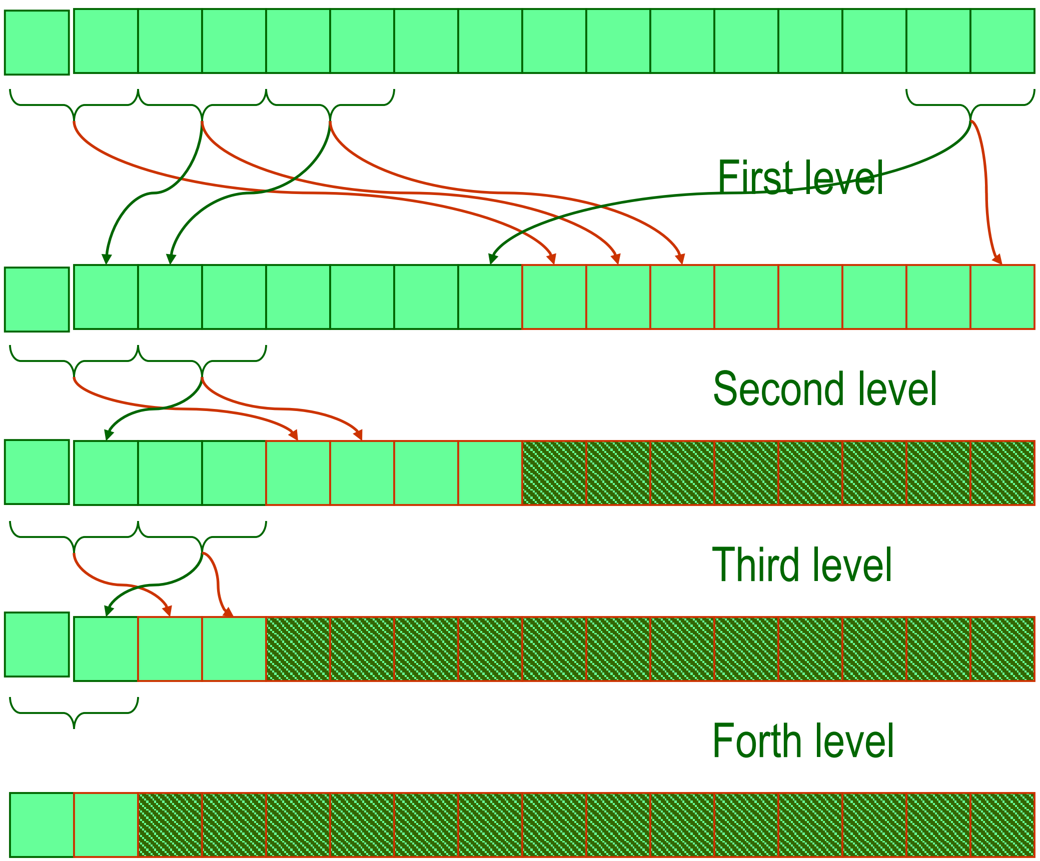
Haar transform

Multiresolution



The number of operations is proportional to the number of samples
No memory is needed other than the initial one (number of samples)

Haar transform



$$N + \frac{N}{2} + \frac{N}{4} + \ldots + 2 < 2N$$

In two dimensions the number of operations doubles

Haar transform