Image pyramids and their applications

6.882

Bill Freeman and Fredo Durand

Feb. 28, 2006

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

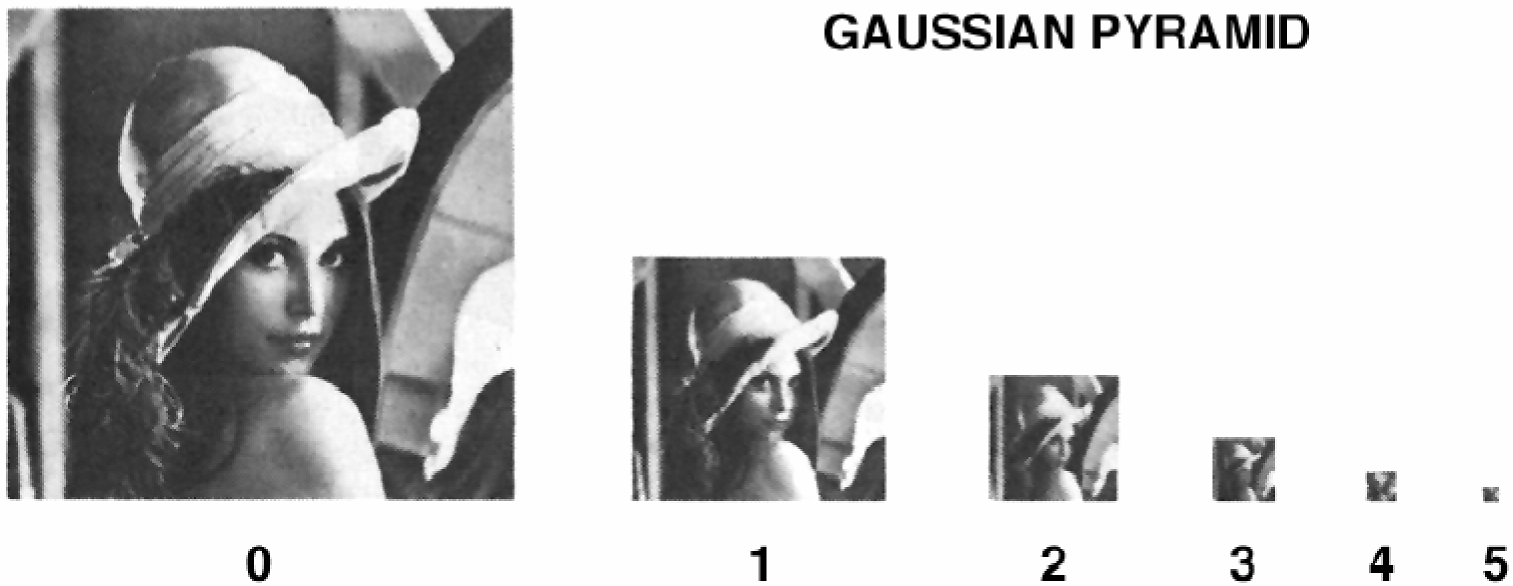


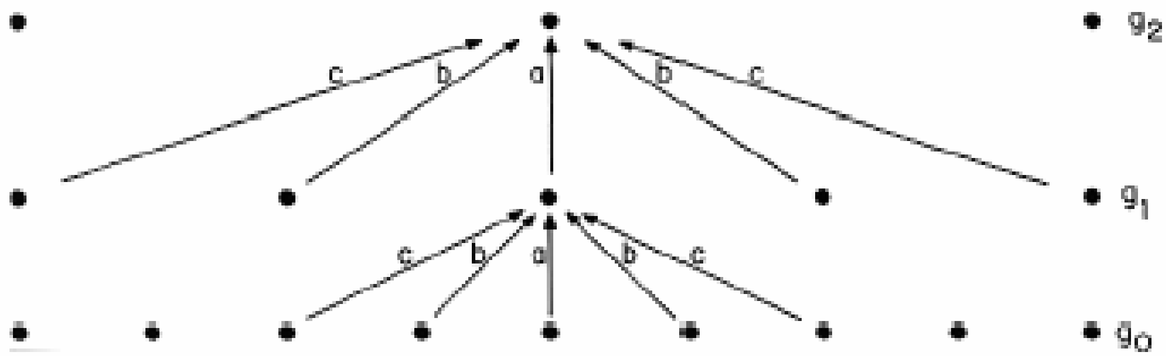
Fig. 4. First six levels of the Gaussian pyramid for the "Lady" image The original image, level 0, meusures 257 by 257 pixels and each higher level array is roughly half the dimensdons of its predecessor. Thus, level 5 measures just 9 by 9 pixels.

http://www-bcs.mit.edu/people/adelson/pub_pdfs/pyramid83.pdf

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983

The computational advantage of pyramids

GAUSSIAN PYRAMID



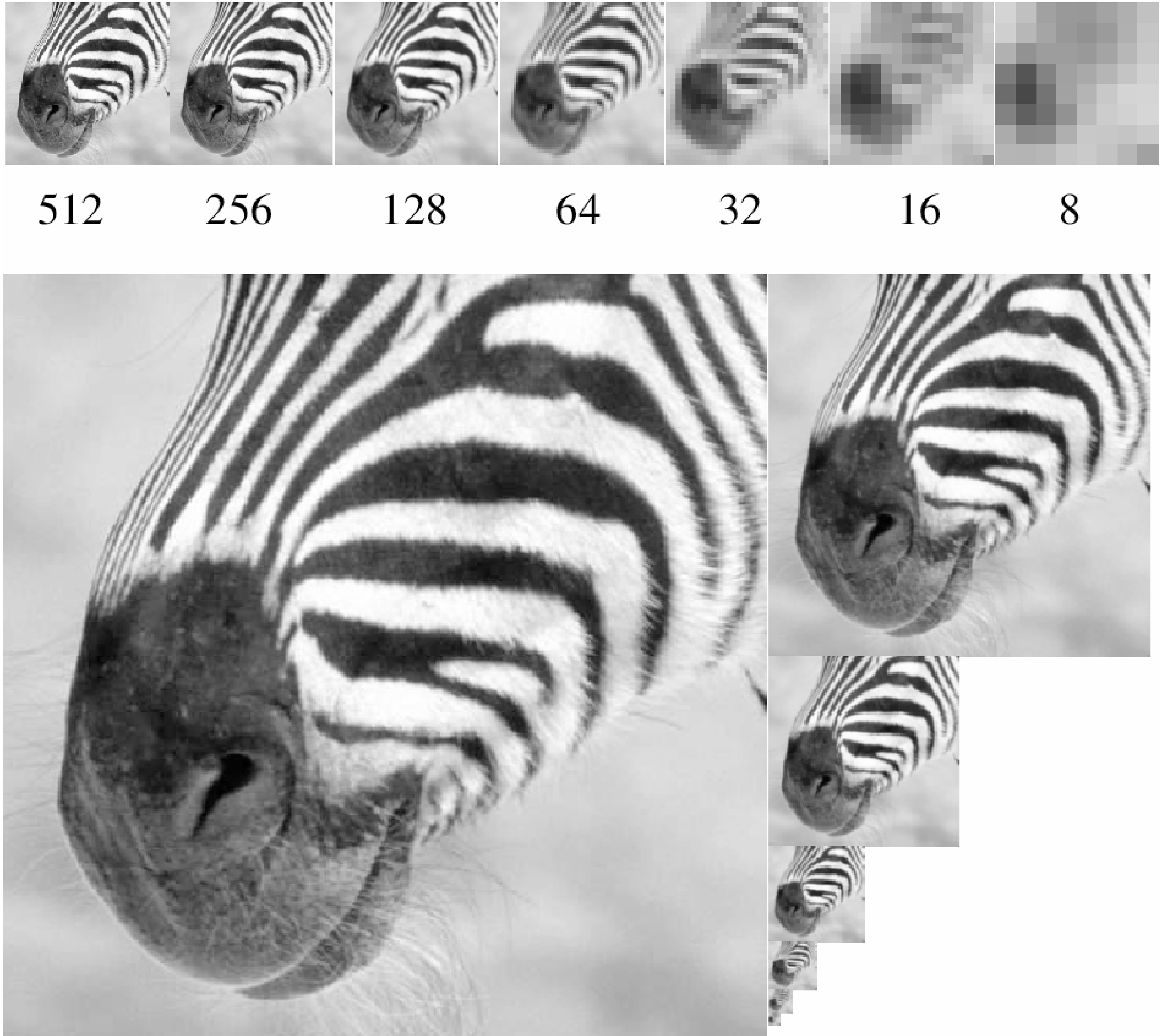
g_0 = IMAGE

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.

http://www-bcs.mit.edu/people/adelson/pub_pdfs/pyramid83.pdf

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983


512
256
128
64
32
16
8

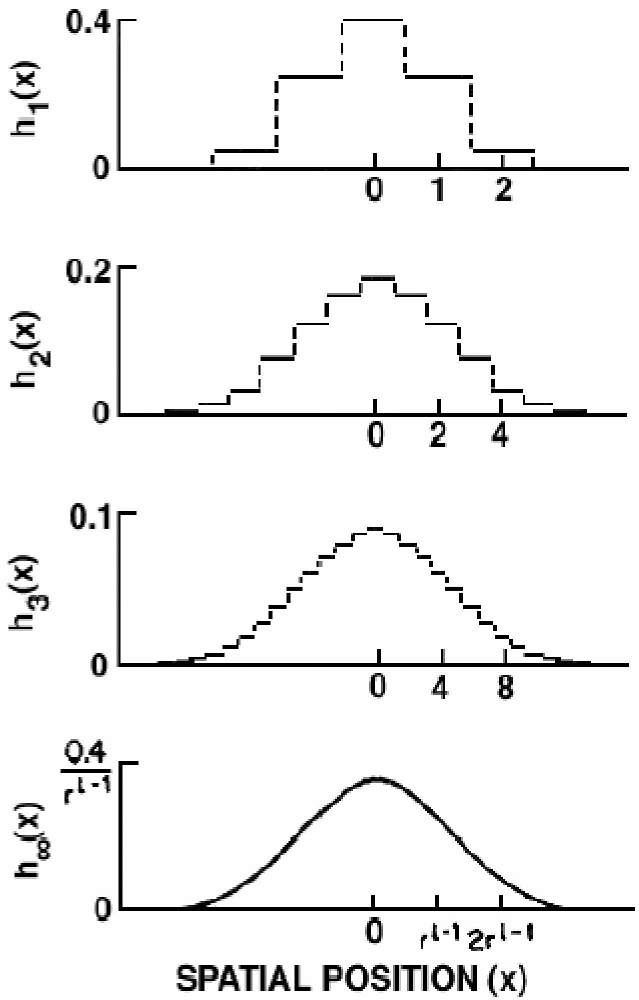
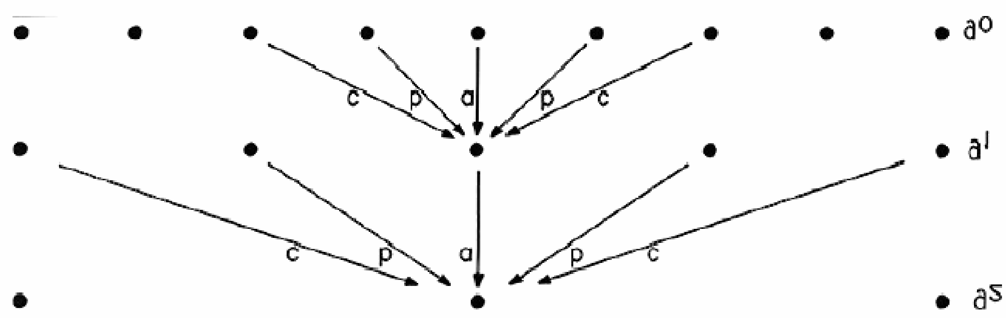


Fig. 2. The equivalent weighting functions $h_l(x)$ for nodes in levels 1, 2, 3, and infinity of the Gaussian pyramid. Note that axis scales have been adjusted by factors of 2 to aid comparison Here the parameter a of the generating kernel is 0.4, and the resulting equivalent weighting functions closely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



http://www-bcs.mit.edu/people/adelson/pub_pdfs/pyramid83.pdf

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983

Convolution and subsampling as a matrix multiply (1-d case)

U1 =

1	4	6	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	4	6	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	4	6	4	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	4	6	4	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	4	6	4	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	4	6	4	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	4	6	4	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	6	4	1	0

Next pyramid level

U2 =

1	4	6	4	1	0	0	0
0	0	1	4	6	4	1	0
0	0	0	0	1	4	6	4
0	0	0	0	0	0	1	4

b * a, the combined effect of the two pyramid levels

```
>> U2 * U1

ans =

     1     4    10    20    31    40    44    40    31    20    10     4     1     0     0     0     0     0     0     0
     0     0     0     0     1     4    10    20    31    40    44    40    31    20    10     4     1     0     0     0
     0     0     0     0     0     0     0     0     1     4    10    20    31    40    44    40    30    16     4     0
     0     0     0     0     0     0     0     0     0     0     0     0     1     4    10    20    25    16     4     0
```

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

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

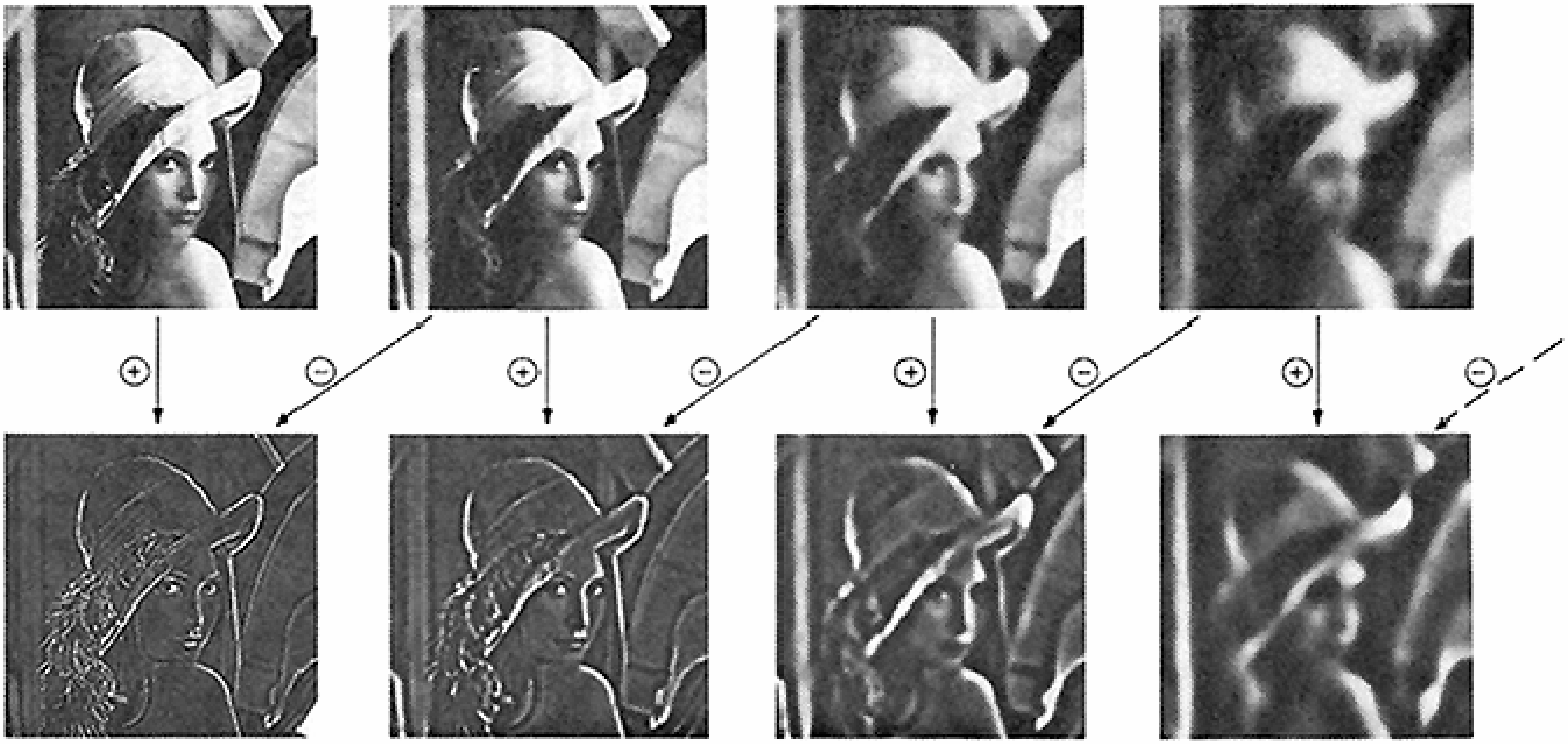
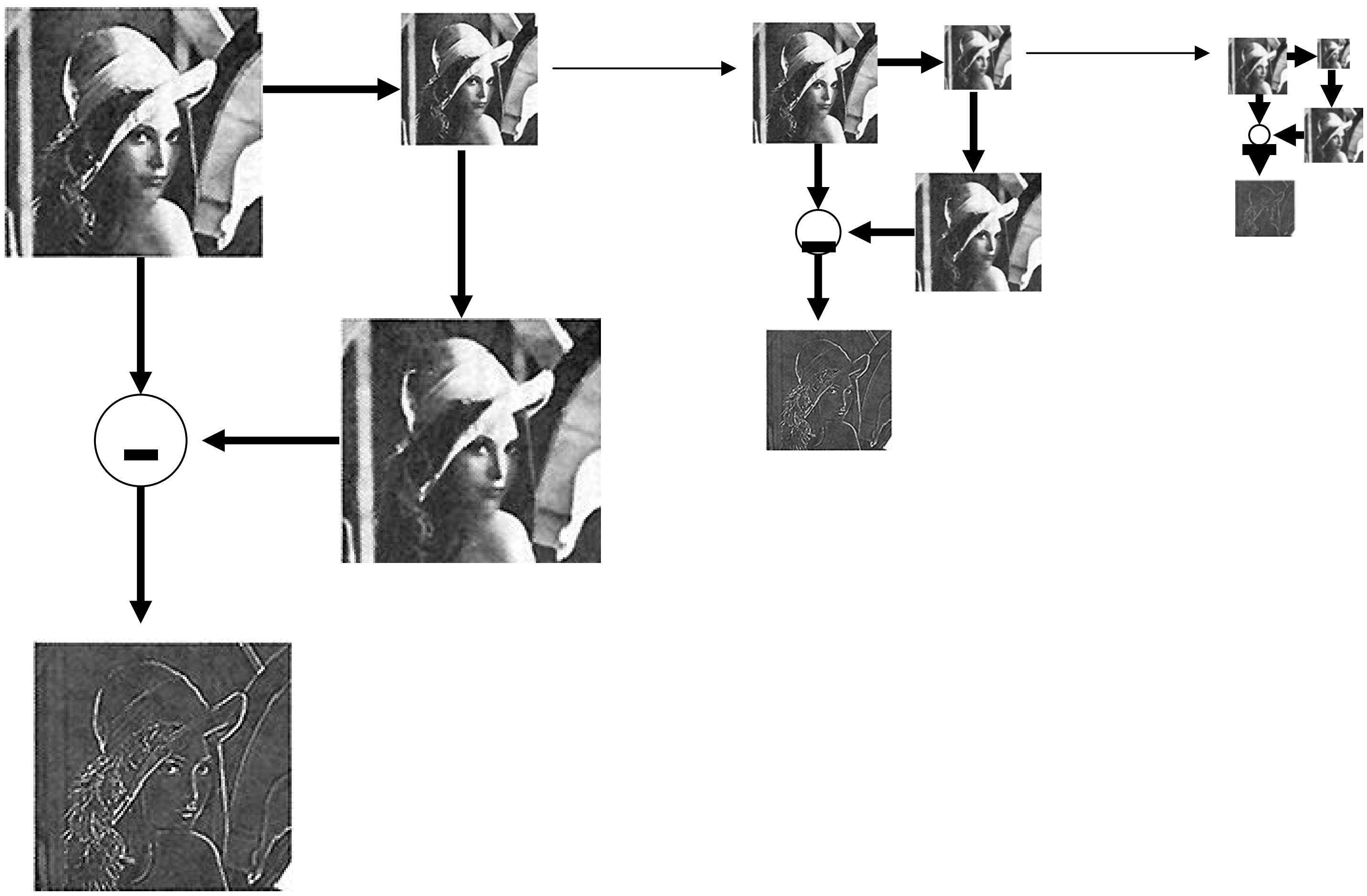


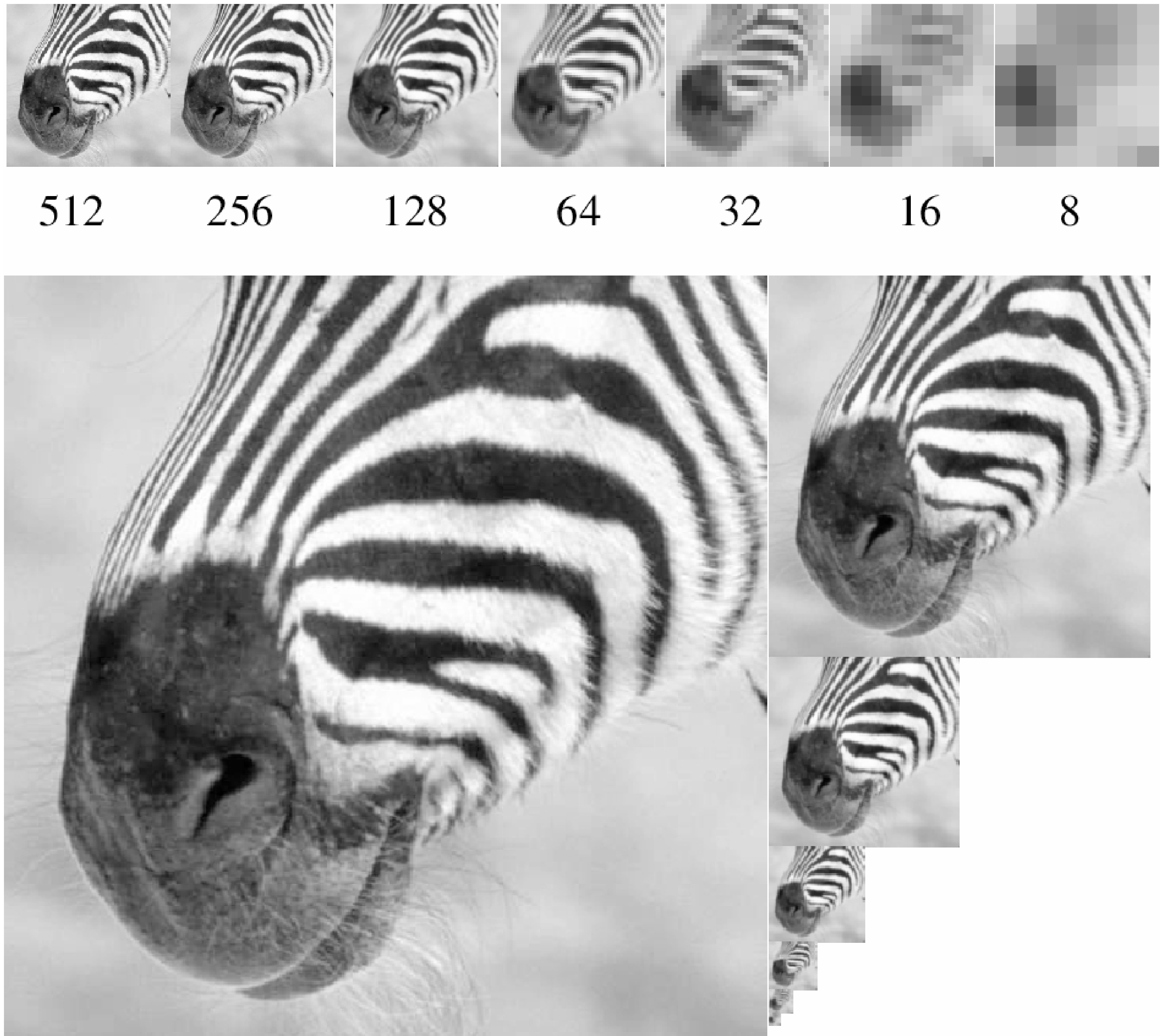
Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

http://www-bcs.mit.edu/people/adelson/pub_pdfs/pyramid83.pdf

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983

Laplacian pyramid algorithm




512
256
128
64
32
16
8

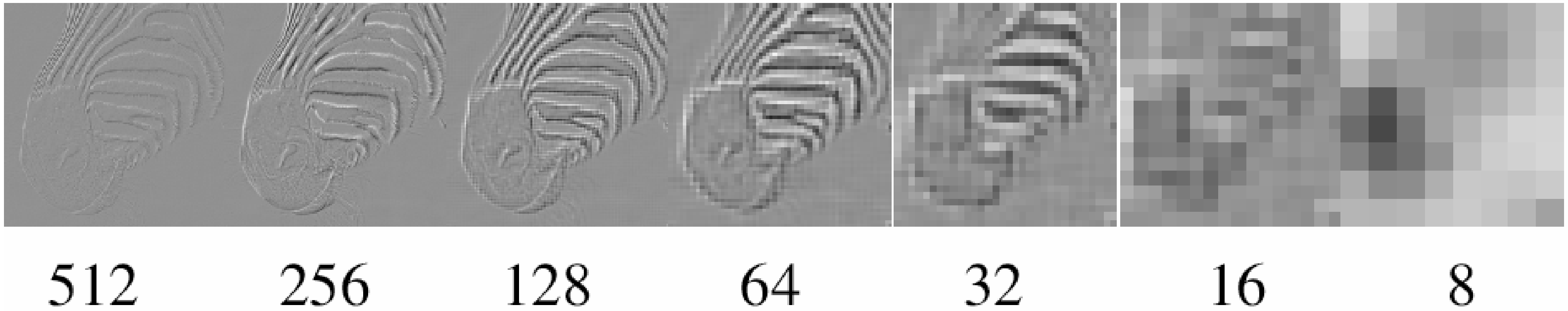

512
256
128
64
32
16
8

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

What is a good representation for image analysis?

(Goldilocks and the three representations)

- Fourier transform domain tells you “what” (textural properties), but not “where”. In space, this representation is too spread out.
- Pixel domain representation tells you “where” (pixel location), but not “what”. In space, this representation is too localized
- Want an image representation that gives you a local description of image events—what is happening where. That representation might be “just right”.

Wavelets/QMF's

transformed image

$$\vec{F} = U\vec{f}$$

Vectorized image

Fourier transform, or
Wavelet transform, or
Steerable pyramid transform

The simplest wavelet transform: the Haar transform

$$U = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

The inverse transform for the Haar wavelet

```
>> inv(U)

ans =

    0.5000    0.5000
    0.5000   -0.5000
```

Apply this over multiple spatial positions

$$U = \begin{matrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{matrix}$$

The high frequencies

U =

$$
\begin{matrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{1} & \mathbf{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
\mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{-1} & \mathbf{0} & \mathbf{0} \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{-1}
\end{matrix}
$$

The low frequencies

U =

1	**1**	**0**	**0**	**0**	**0**	**0**	**0**
1	-1	0	0	0	0	0	0
0	**0**	**1**	**1**	**0**	**0**	**0**	**0**
0	0	1	-1	0	0	0	0
0	**0**	**0**	**0**	**1**	**1**	**0**	**0**
0	0	0	0	1	-1	0	0
0	**0**	**0**	**0**	**0**	**0**	**1**	**1**
0	0	0	0	0	0	1	-1

The inverse transform

```
>> inv(U)

ans =

    0.5000    0.5000         0         0         0         0         0         0
    0.5000   -0.5000         0         0         0         0         0         0
         0         0    0.5000    0.5000         0         0         0         0
         0         0    0.5000   -0.5000         0         0         0         0
         0         0         0         0    0.5000    0.5000         0         0
         0         0         0         0    0.5000   -0.5000         0         0
         0         0         0         0         0         0    0.5000    0.5000
         0         0         0         0         0         0    0.5000   -0.5000
```

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

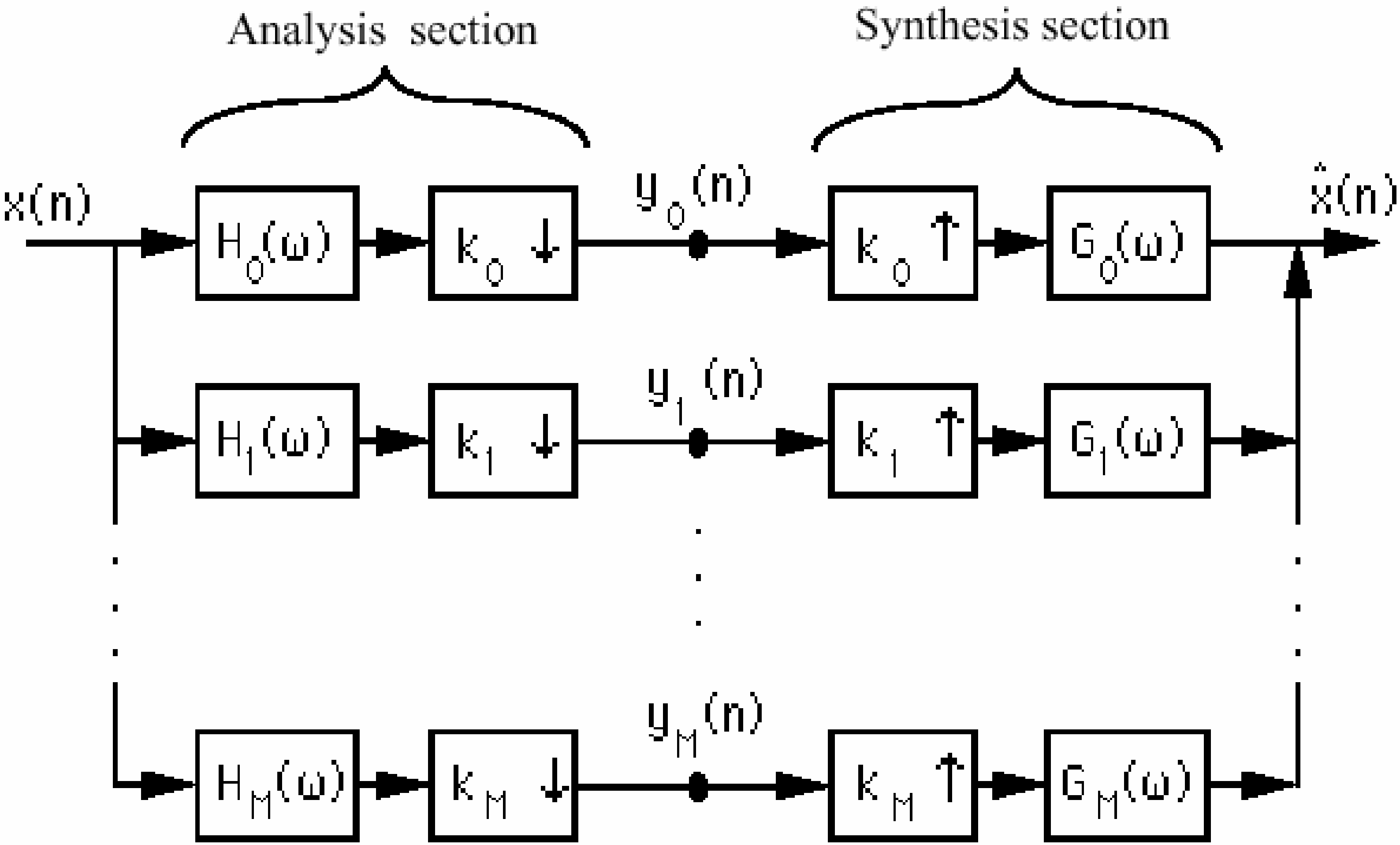


Figure 4.2: An analysis/synthesis filter bank.

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

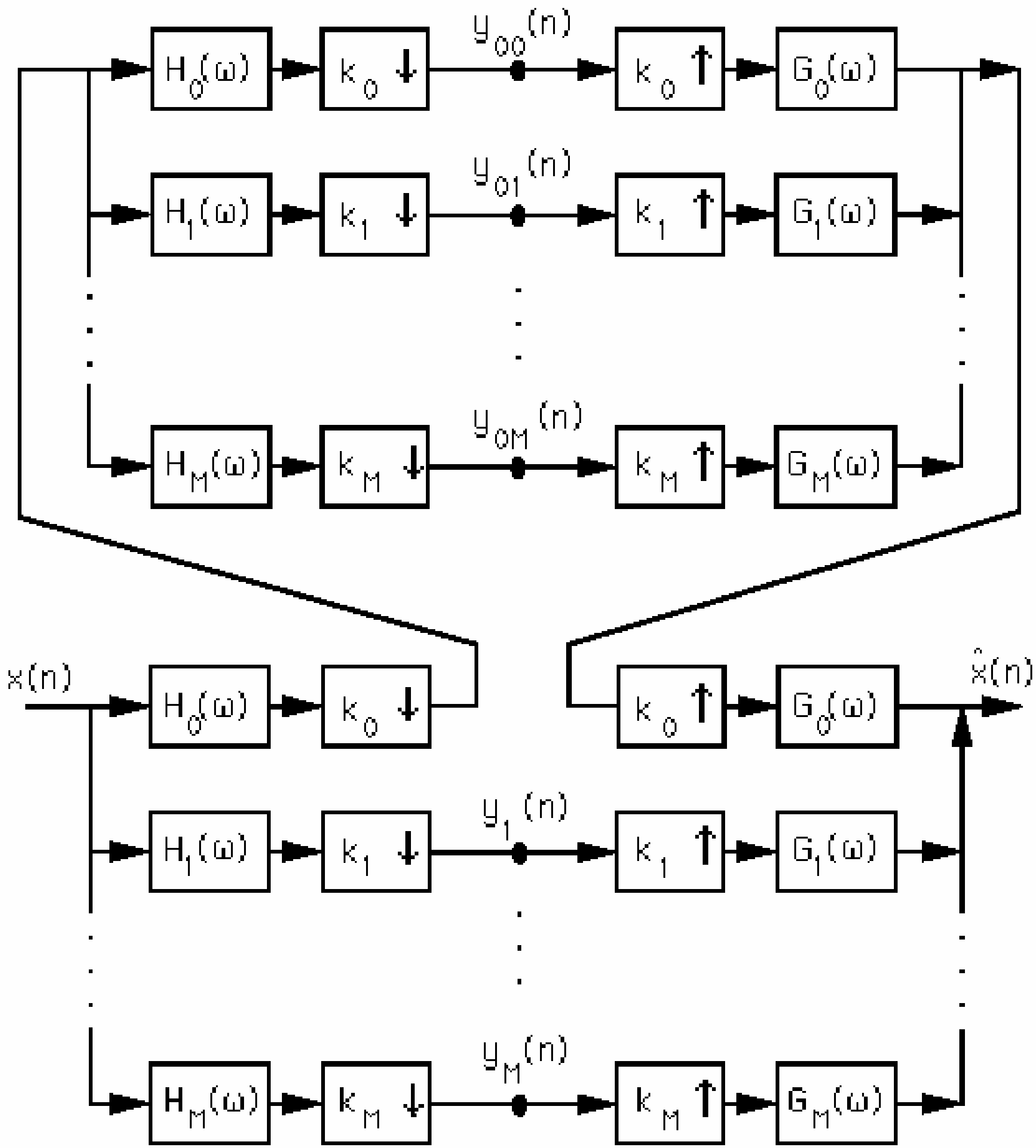


Figure 4.3: A non-uniformly cascaded analysis/synthesis filter bank.

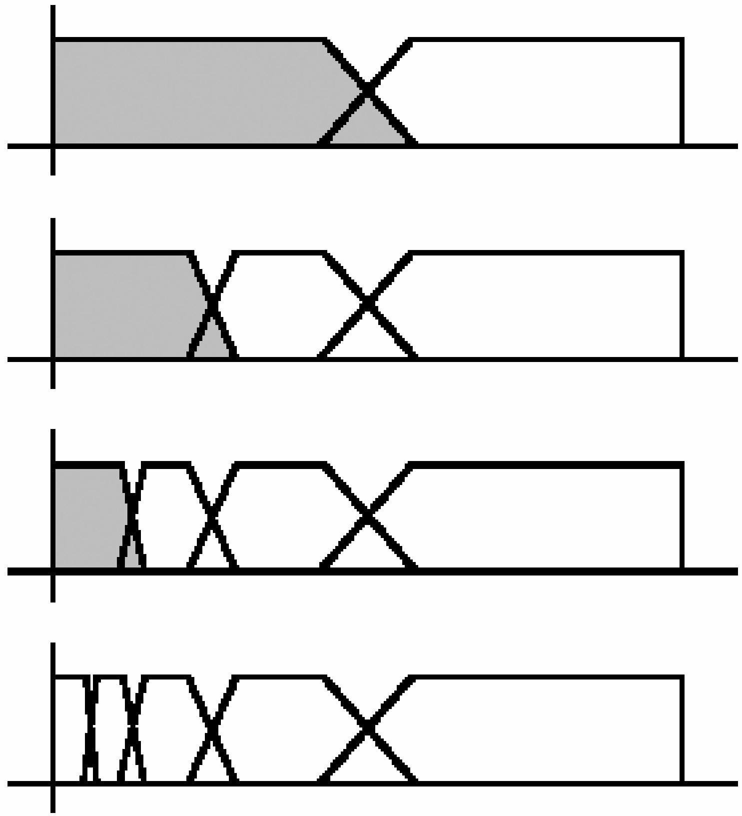


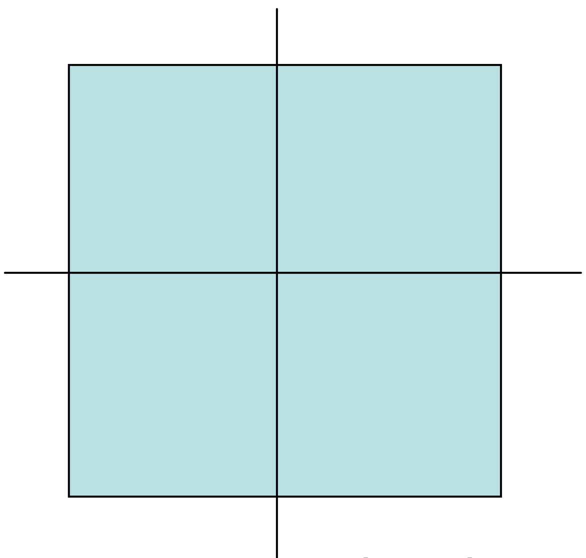
Figure 4.4: Octave band splitting produced by a four-level pyramid cascade of a two-band A/S system. The top picture represents the splitting of the two-band A/S system. Each successive picture shows the effect of re-applying the system to the lowpass subband (indicated in grey) of the previous picture.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

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

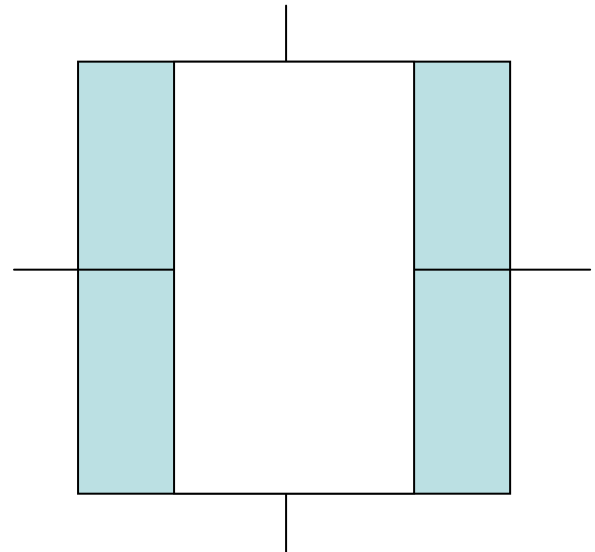
n	QMF-5	QMF-9	QMF-13
0	0.8593118	0.7973934	0.7737113
1	0.3535534	0.41472545	0.42995453
2	-0.0761025	-0.073386624	-0.057827797
3		-0.060944743	-0.09800052
4		0.02807382	0.039045125
5			0.021651438
6			-0.014556438

Table 4.1: Odd-length QMF kernels. Half of the impulse response sample values are shown for each of the normalized lowpass QMF filters (All filters are symmetric about $n = 0$). The appropriate highpass filters are obtained by delaying by one sample and multiplying with the sequence $(-1)^n$.

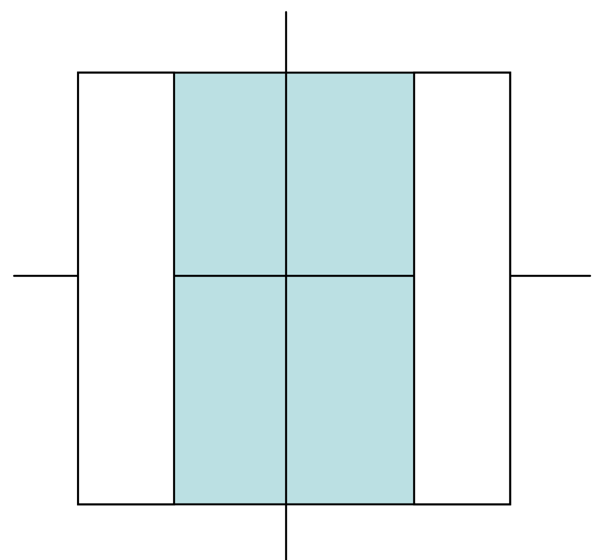
Now, in 2 dimensions…



Frequency domain

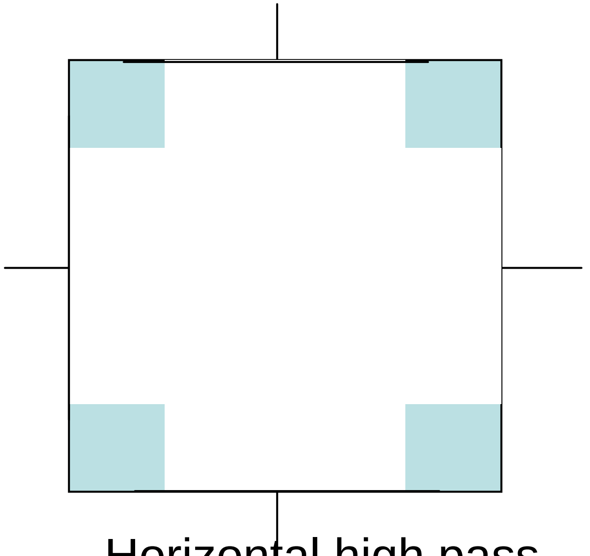


Horizontal high pass

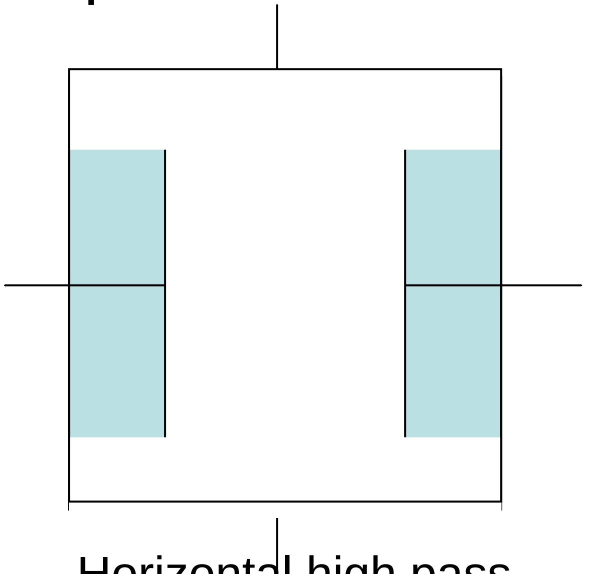


Horizontal low pass

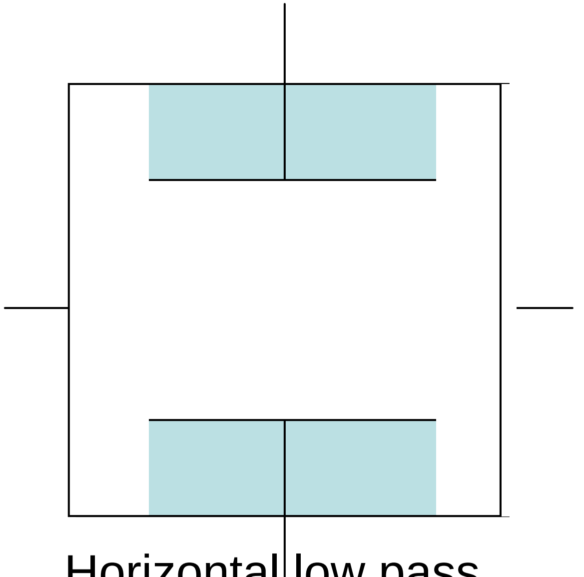
Apply the wavelet transform separable in both dimensions



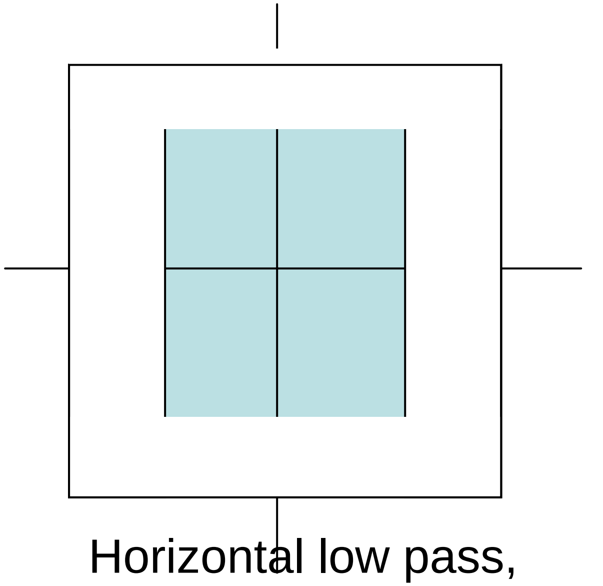
Horizontal high pass,
vertical high pass



Horizontal high pass,
vertical low-pass



Horizontal low pass,
vertical high-pass



Horizontal low pass,
Vertical low-pass

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

To create 2-d filters, apply the 1-d filters separably in the two spatial dimensions

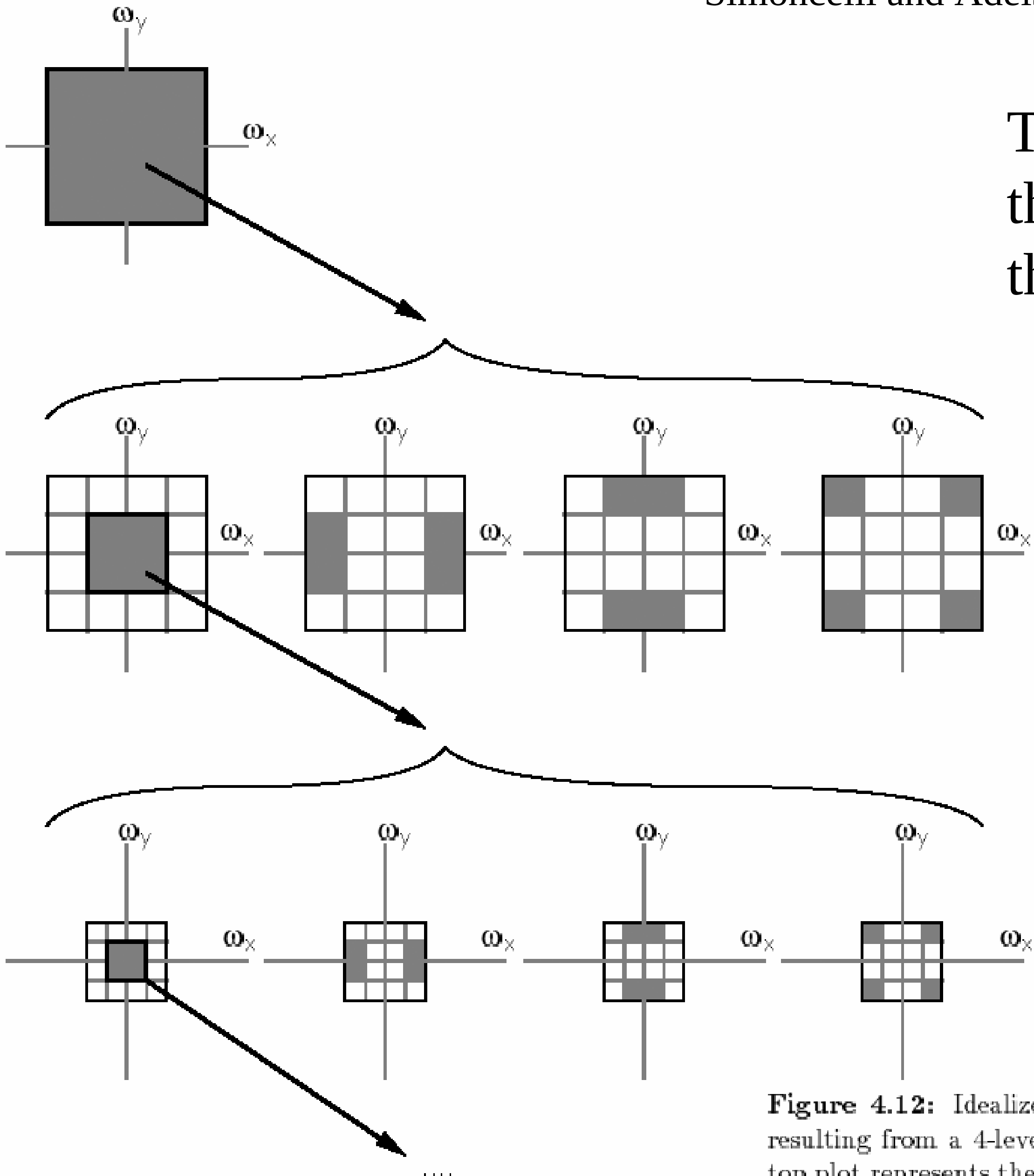
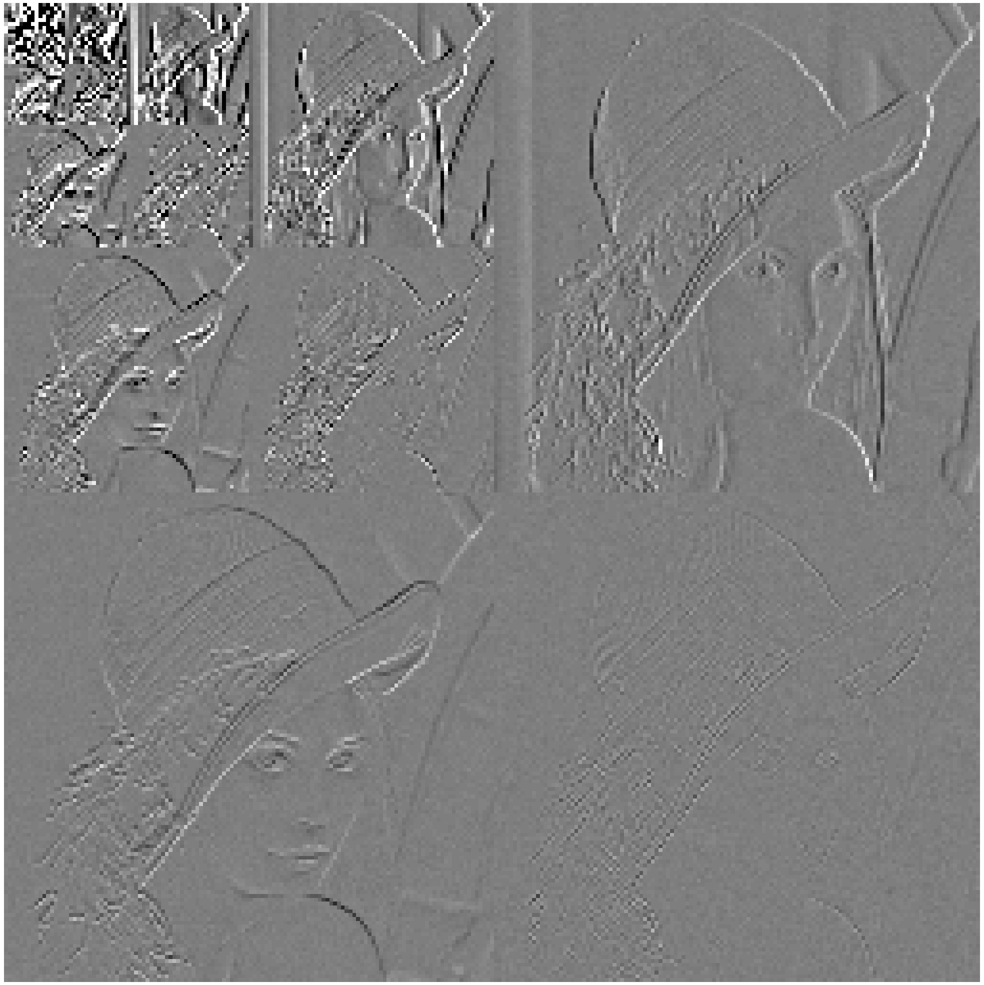


Figure 4.12: Idealized diagram of the partition of the frequency plane resulting from a 4-level pyramid cascade of separable 2-band filters. The top plot represents the frequency spectrum of the original image, with axes ranging from $-\pi$ to π. This is divided into four subbands at the next level. On each subsequent level, the lowpass subband (outlined in bold) is subdivided further.

Wavelet/QMF representation



Good and bad features of wavelet/QMF filters

- Bad:
  - Aliased subbands
  - Non-oriented diagonal subband
- Good:
  - Not overcomplete (so same number of coefficients as image pixels).
  - Good for image compression (JPEG 2000)

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

Steerable filters

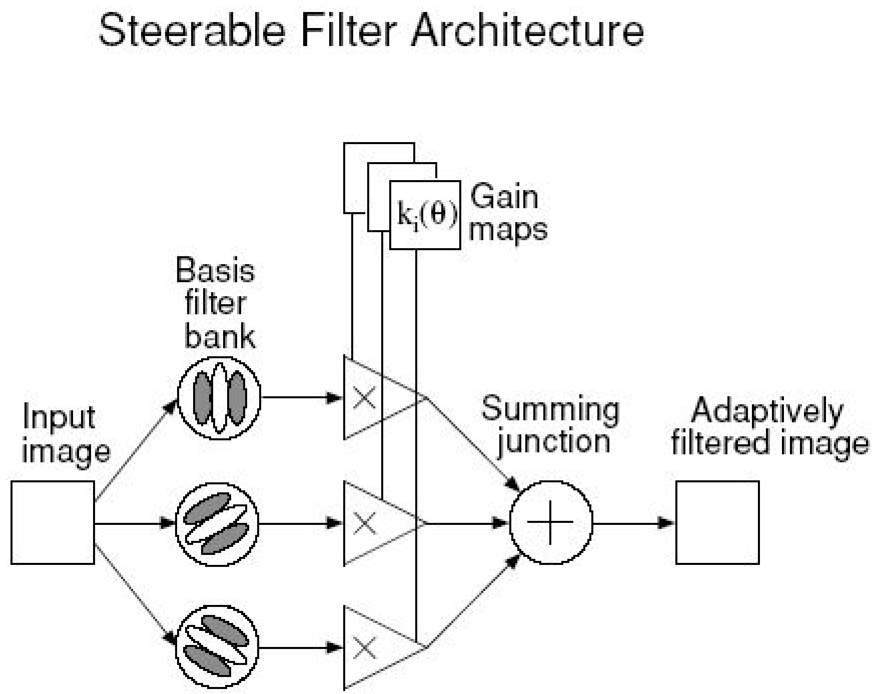


Figure 2-3: Steerable filter system block diagram. A bank of dedicated filters process the image. Their outputs are multiplied by a set of gain maps which adaptively control the orientation of the synthesized filter.

http://people.csail.mit.edu/billf/freemanThesis.pdf

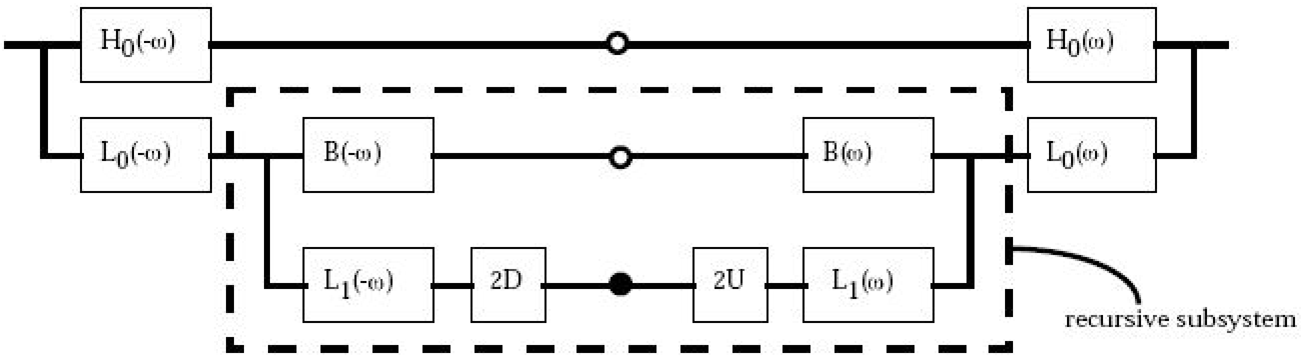


Figure 2: System diagram for the radial portion of the steerable pyramid, illustrating the filtering and sampling operations, and the recursive construction. Boxes containing "2D" and "2U" correspond to downsampling and upsampling by a factor of 2. All other boxes correspond to standard 2D convolution. The circles correspond to the transform coefficients. The recursive construction of a pyramid is achieved by inserting a copy of the diagram contents enclosed by the dashed rectangle at the location of the *solid* circle (i.e., the lowpass branch).

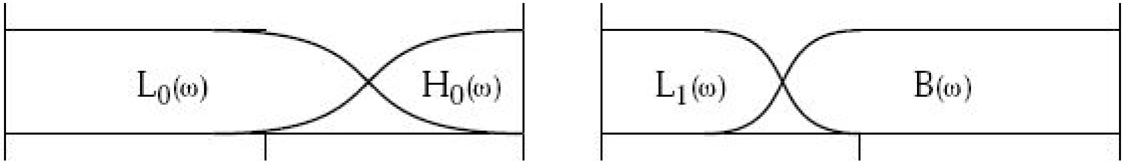
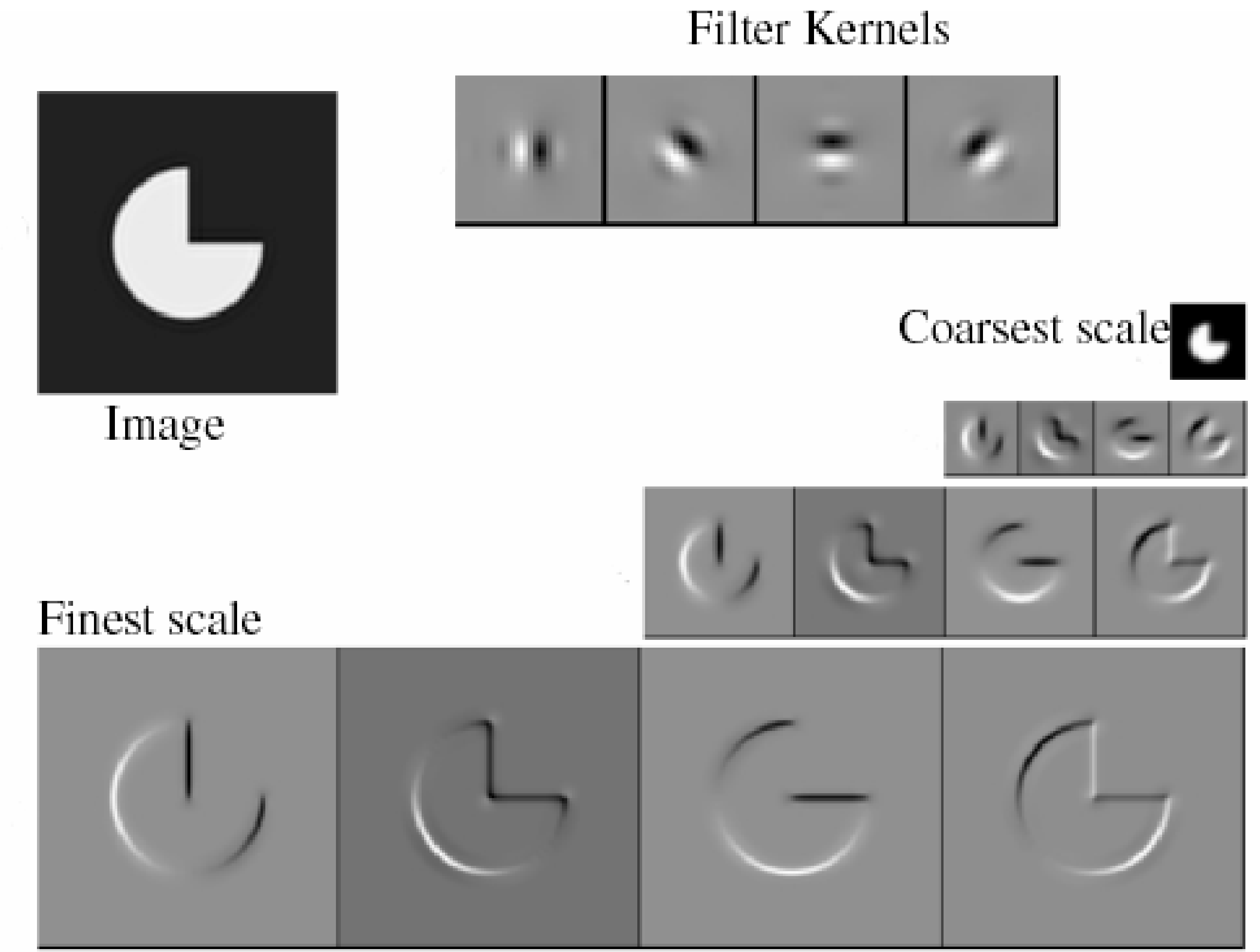


Figure 3: Idealized depiction of filters satisfying the constraints of the block diagram in figure 2. Plots show Fourier spectra over the range $[0, \pi]$.

http://www.merl.com/reports/docs/TR95-15.pdf



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

Non-oriented steerable pyramid

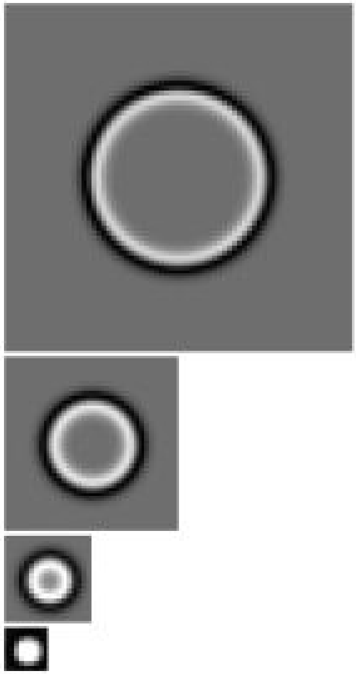


Figure 4: A 3-level $k = 1$ (non-oriented) steerable pyramid. Shown are the bandpass images and the final lowpass image.

http://www.merl.com/reports/docs/TR95-15.pdf

3-orientation steerable pyramid

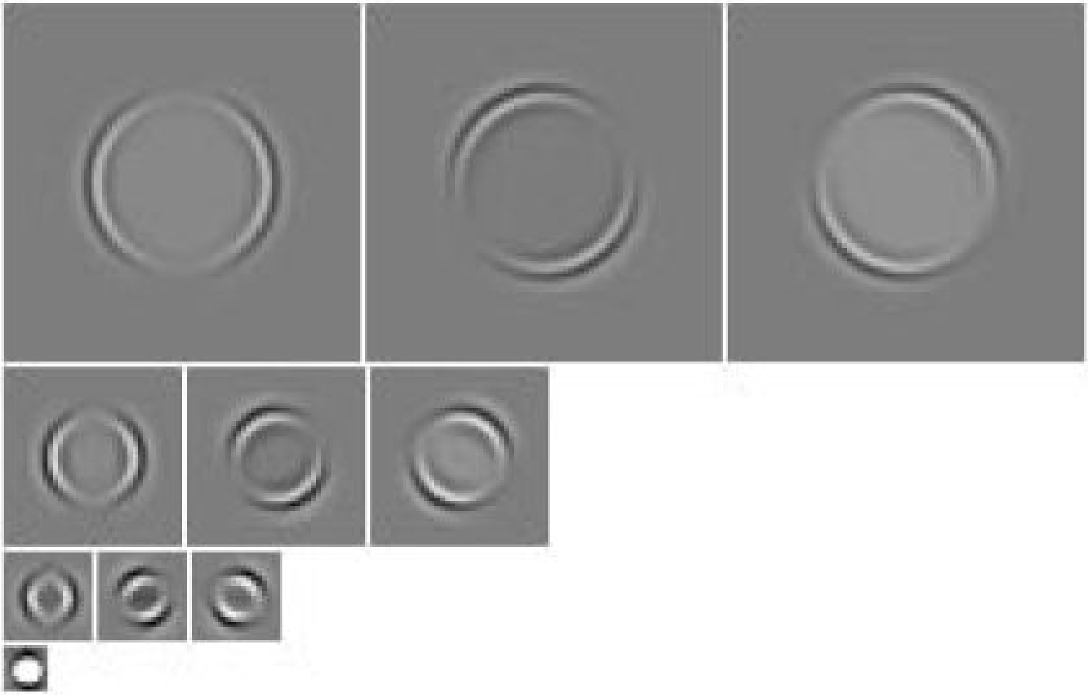


Figure 5: A 3-level $k = 3$ (second derivative) steerable pyramid. Shown are the three band-pass images at each scale and the final lowpass image.

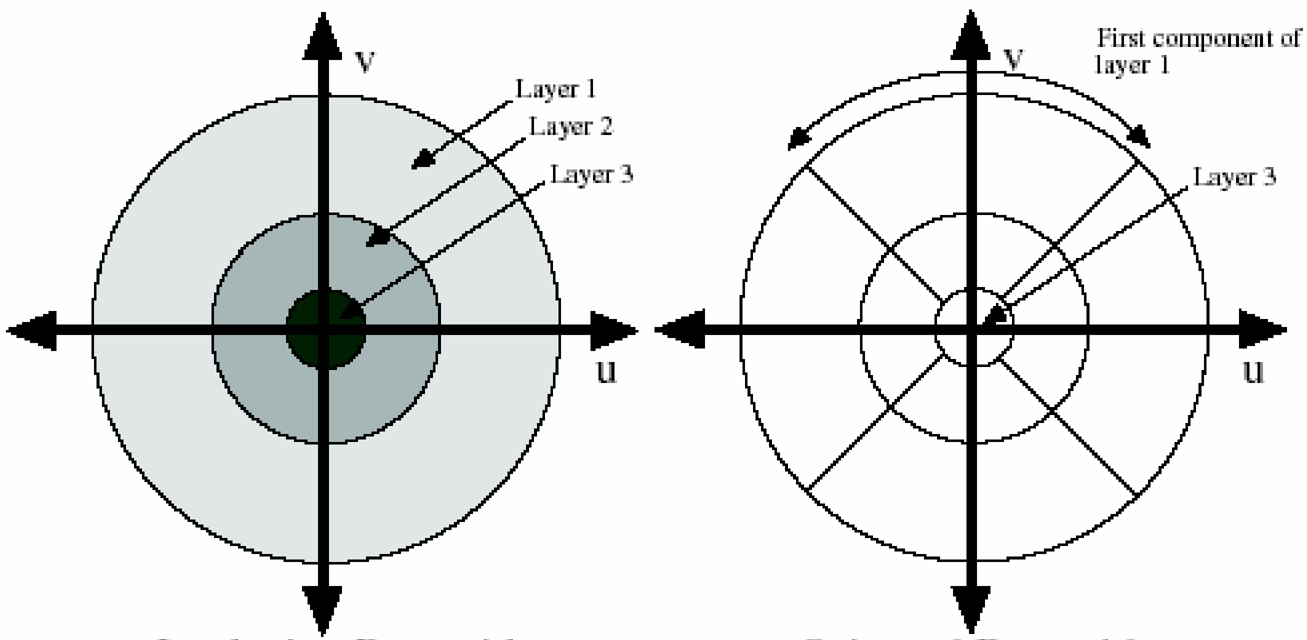
http://www.merl.com/reports/docs/TR95-15.pdf

Steerable pyramids

- Good:
  - Oriented subbands
  - Non-aliased subbands
  - Steerable filters
- Bad:
  - Overcomplete
  - Have one high frequency residual subband, required in order to form a circular region of analysis in frequency from a square region of support in frequency.

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



Laplacian Pyramid   Oriented Pyramid

	Laplacian Pyramid	Dyadic QMF/Wavelet	Steerable Pyramid
self-inverting (tight frame)	no	yes	yes
overcompleteness	$4/3$	1	$4k/3$
aliasing in subbands	perhaps	yes	no
rotated orientation bands	no	only on hex lattice [9]	yes

Table 1: Properties of the Steerable Pyramid relative to two other well-known multi-scale representations.

http://www.cns.nyu.edu/ftp/eero/simoncelli95b.pdf

Simoncelli and Freeman, ICIP 1995

But we need to get rid of the corner regions before starting the recursive circular filtering

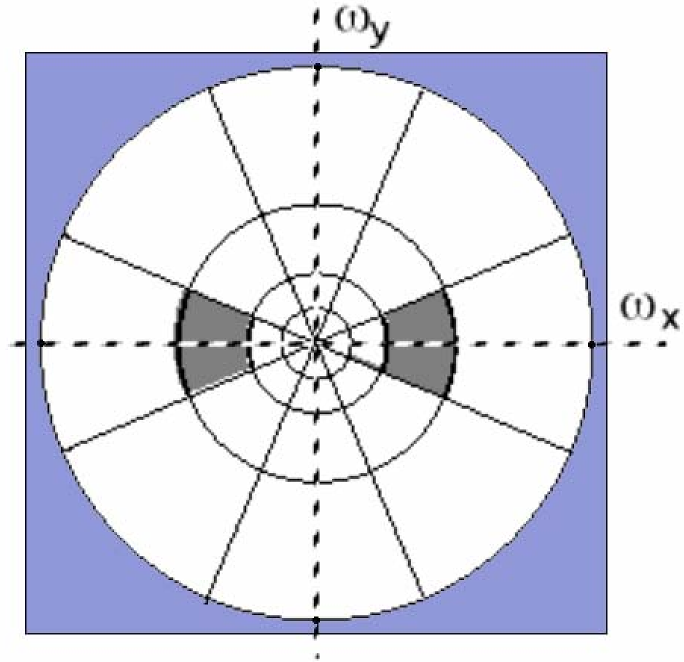


Figure 1. Idealized illustration of the spectral decomposition performed by a steerable pyramid with $k = 4$. Frequency axes range from $-\pi$ to π. The basis functions are related by translations, dilations and *rotations* (except for the initial highpass subband and the final lowpass subband). For example, the shaded region corresponds to the spectral support of a single (vertically-oriented) subband.

http://www.cns.nyu.edu/ftp/eero/simoncelli95b.pdf

Simoncelli and Freeman, ICIP 1995

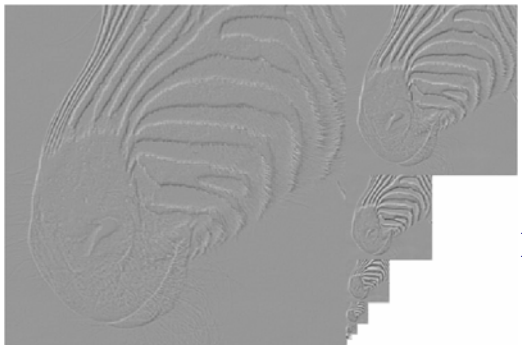
- Summary of pyramid representations

Image pyramids

- Gaussian

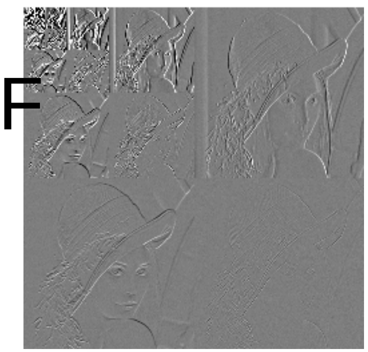
Progressively blurred and subsampled versions of the image. Adds scale invariance to fixed-size algorithms.

- Laplacian



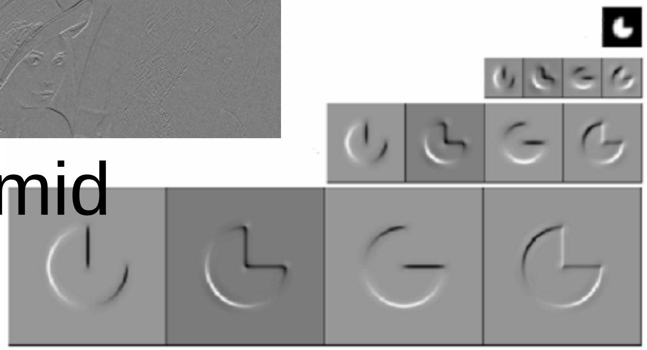
Shows the information added in Gaussian pyramid at each spatial scale. Useful for noise reduction & coding.

- Wavelet/QMF



Bandpassed representation, complete, but with aliasing and some non-oriented subbands.

- Steerable pyramid



Shows components at each scale and orientation separately. Non-aliased subbands. Good for texture and feature analysis.

Schematic pictures of each matrix transform

Shown for 1-d images

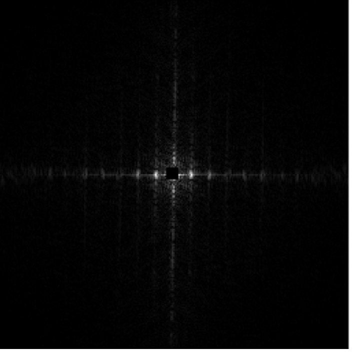
The matrices for 2-d images are the same idea, but more complicated, to account for vertical, as well as horizontal, neighbor relationships.

transformed image

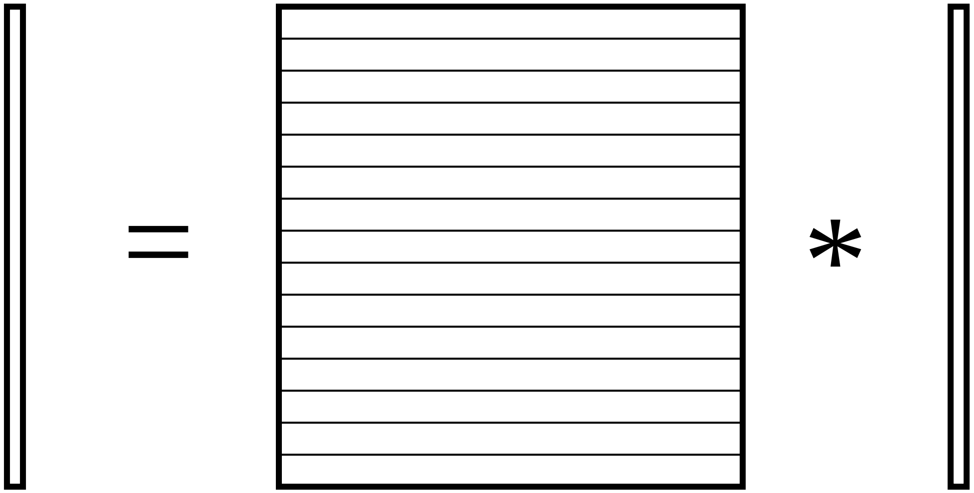
$$\vec{F} = U\vec{f}$$

Vectorized image

Fourier transform, or
Wavelet transform, or
Steerable pyramid transform



Fourier transform

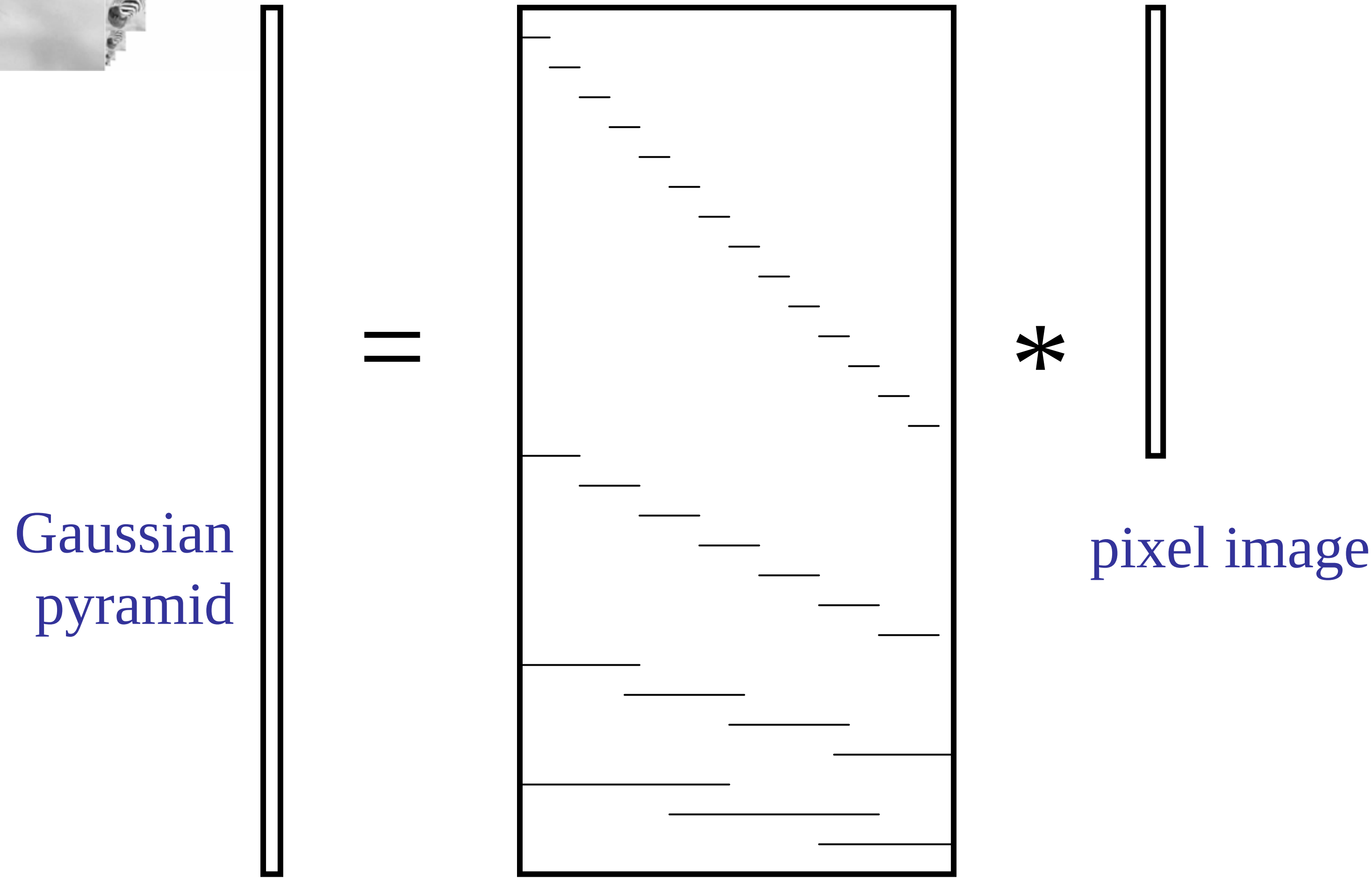


Fourier transform

Fourier bases are global: each transform coefficient depends on all pixel locations.

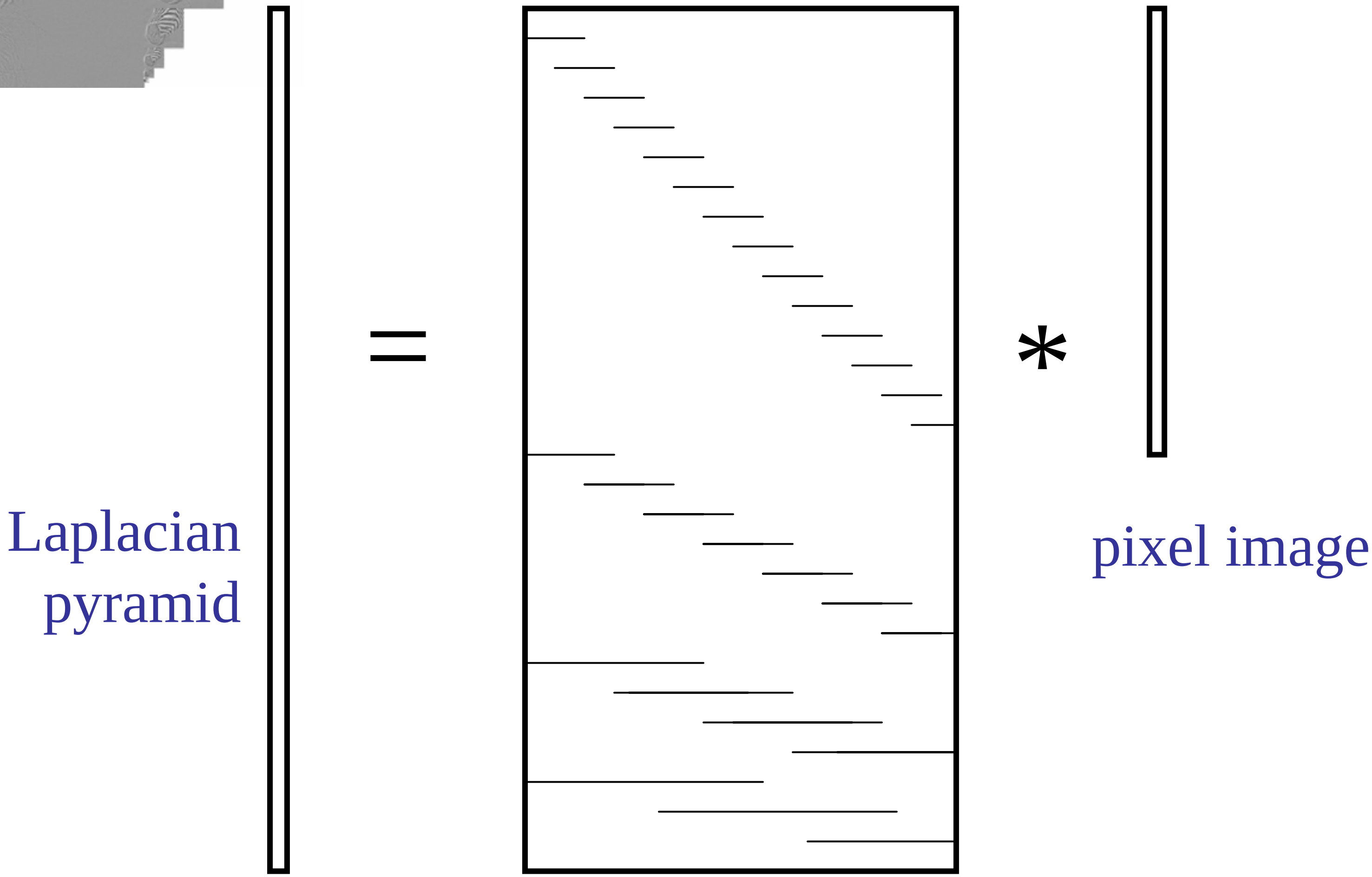
pixel domain image

Gaussian pyramid

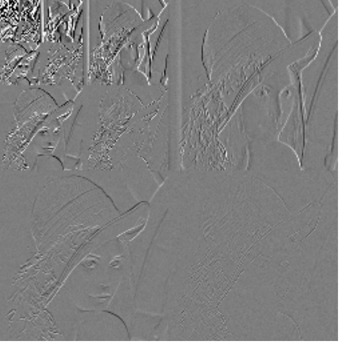


Overcomplete representation.
Low-pass filters, sampled appropriately for their blur.

Laplacian pyramid

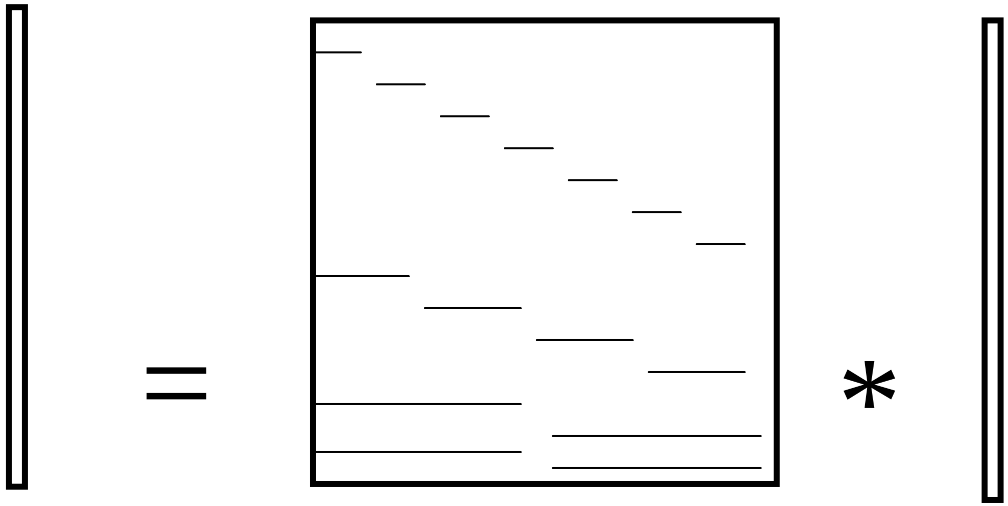


Overcomplete representation.
Transformed pixels represent bandpassed image information.



Wavelet (QMF) transform

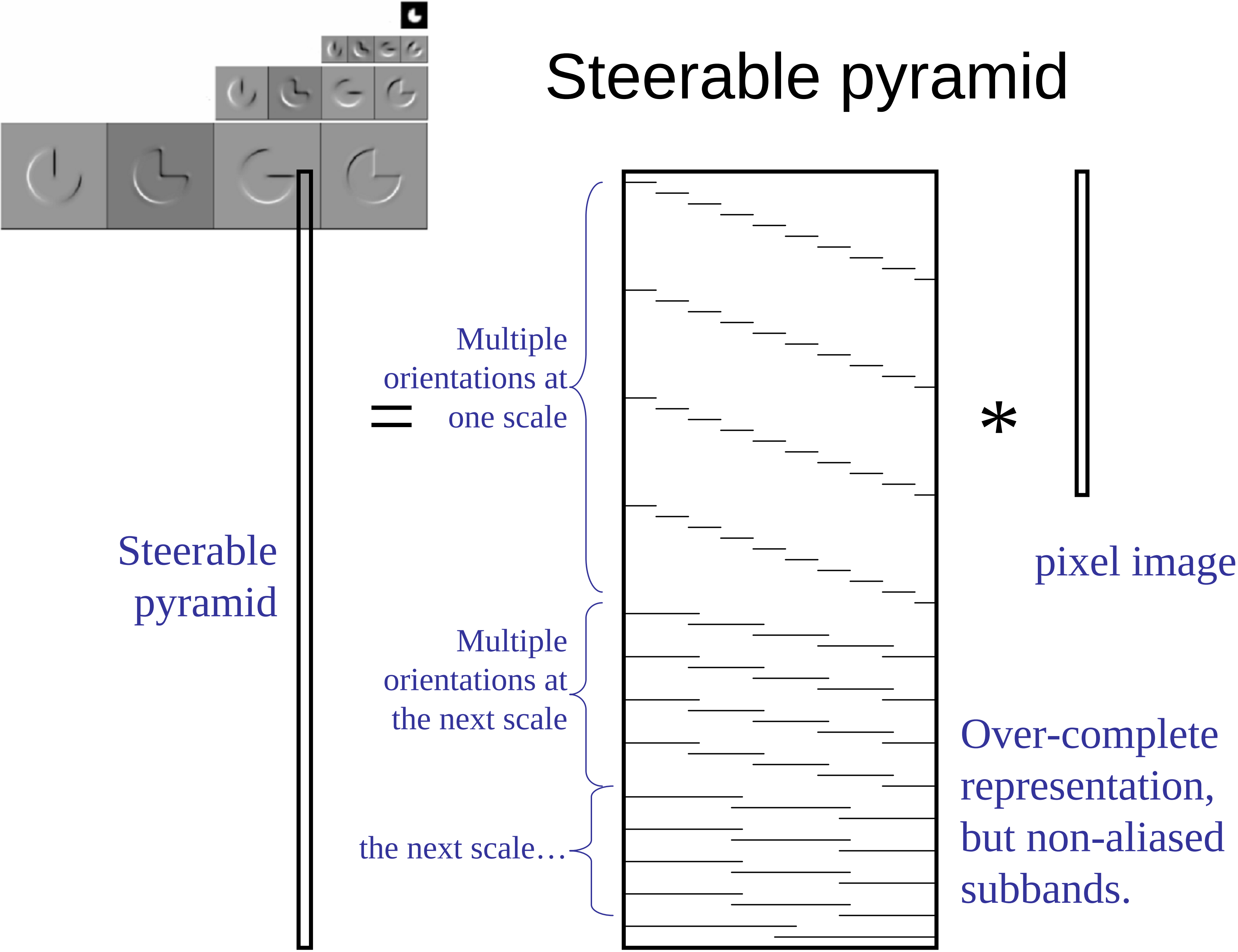
Wavelet pyramid



Ortho-normal transform (like Fourier transform), but with localized basis functions.

pixel image

Steerable pyramid



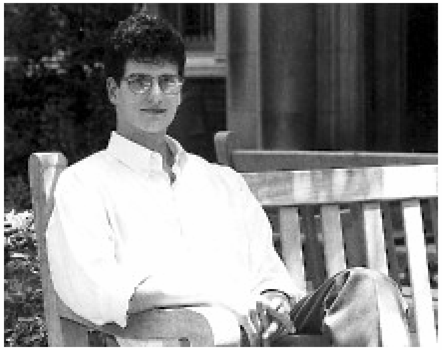
Matlab resources for pyramids (with tutorial)

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

Eero P. Simoncelli

Associate Investigator,
Howard Hughes Medical Institute

Associate Professor,
Neural Science and Mathematics,
New York University



Matlab resources for pyramids (with tutorial)

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

lcv

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).

Why use these representations?

- Handle real-world size variations with a constant-size vision algorithm.
- Remove noise
- Analyze texture
- Recognize objects
- Label image features

E. H. Adelson | C. H. Anderson | J. R. Bergen | P. J. Burt | J. M. Ogden

Pyramid methods in image processing

The image pyramid offers a flexible, convenient multiresolution format that mirrors the multiple scales of processing in the human visual system.

http://web.mit.edu/persci/people/adelson/pub_pdfs/RCA84.pdf

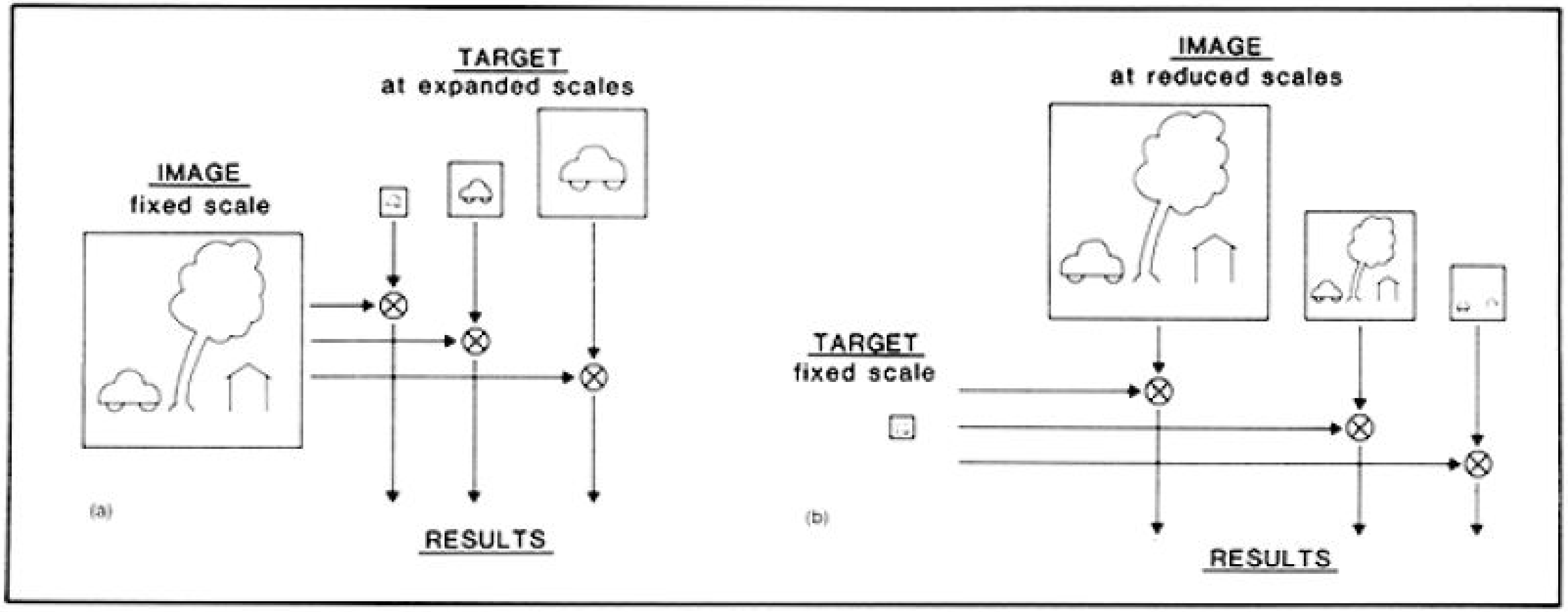


Fig. 1. Two methods of searching for a target pattern over many scales. In the first approach, (a), copies of the target pattern are constructed at several expanded scales, and each is convolved with the original image. In the second approach, (b), a single copy of the target is convolved with copies of the image reduced in scale. The target should be just large enough to resolve critical details The two approaches should give equivalent results, but the second is more efficient by the fourth power of the scale factor (image convolutions are represented by 'O').

http://web.mit.edu/persci/people/adelson/pub_pdfs/RCA84.pdf

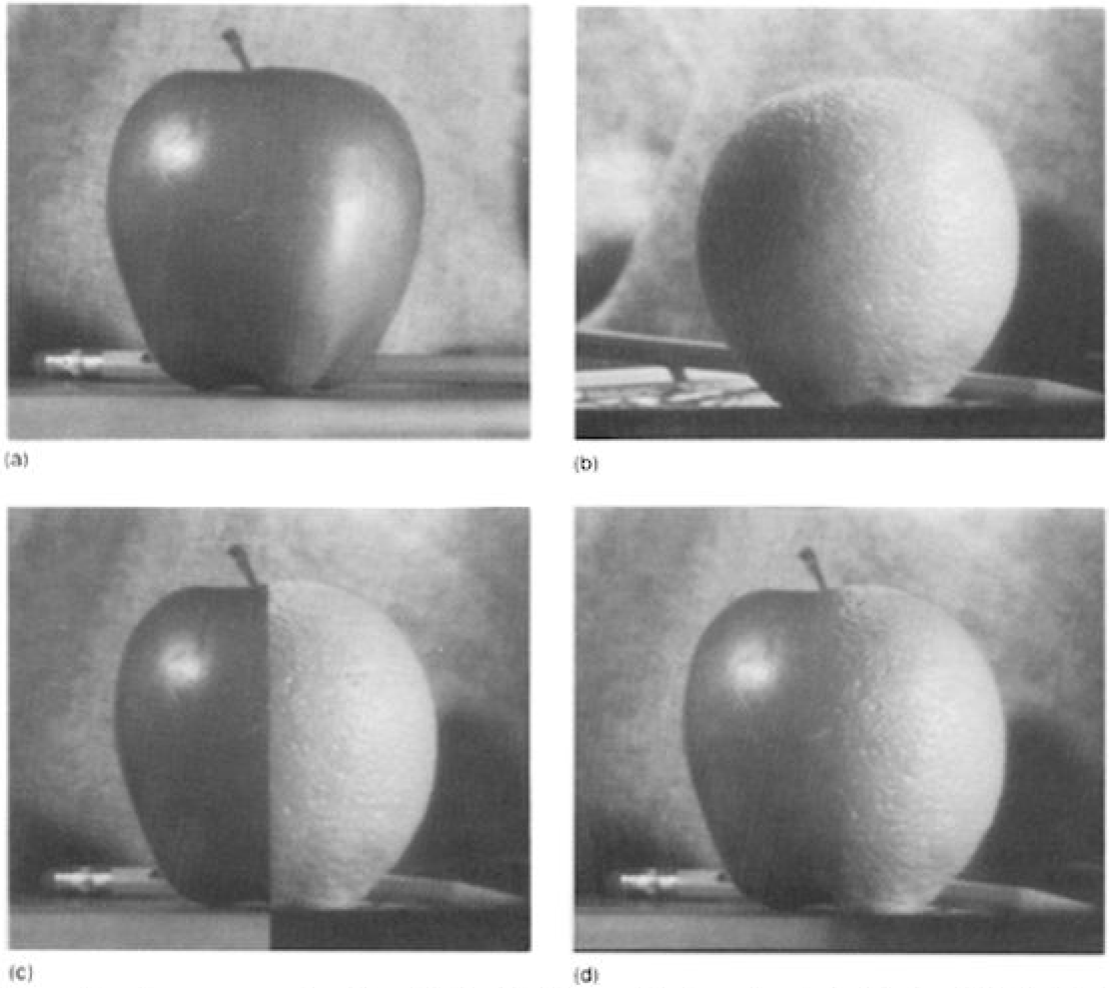
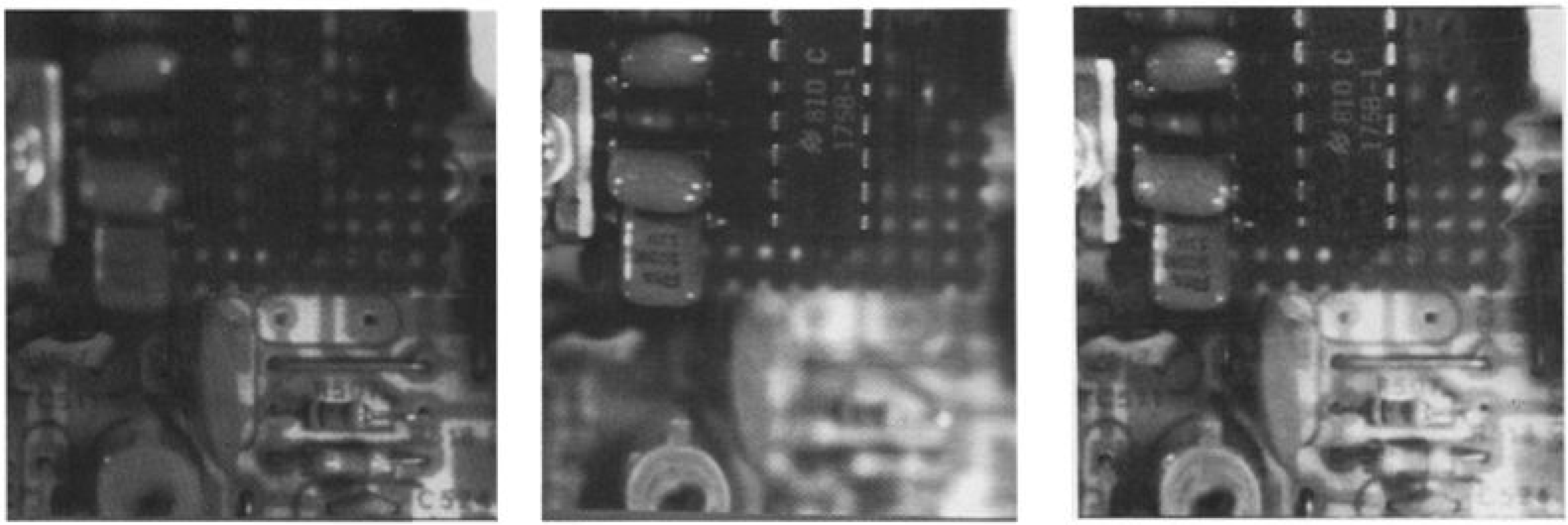


Fig. 10. *Image mosaics. The left half of image (a) is catinated with the right half of image (b) to give the mosaic in (c). Note that the boundary between regions is clearly visible. The mosaic in (d) was obtained by combining images separately in each spatial frequency band of their pyramid representations then expanding and summing these bandpass mosaics.*

http://web.mit.edu/persci/people/adelson/pub_pdfs/RCA84.pdf

Very early computational approach to creating large depth-of-field



(a) (b) (c)

Fig. 9. *Multifocus composite image. The original images with limited depth of field are shown in (a) and (b). These are combined digitally to give the image will an extended depth of field in (c).*

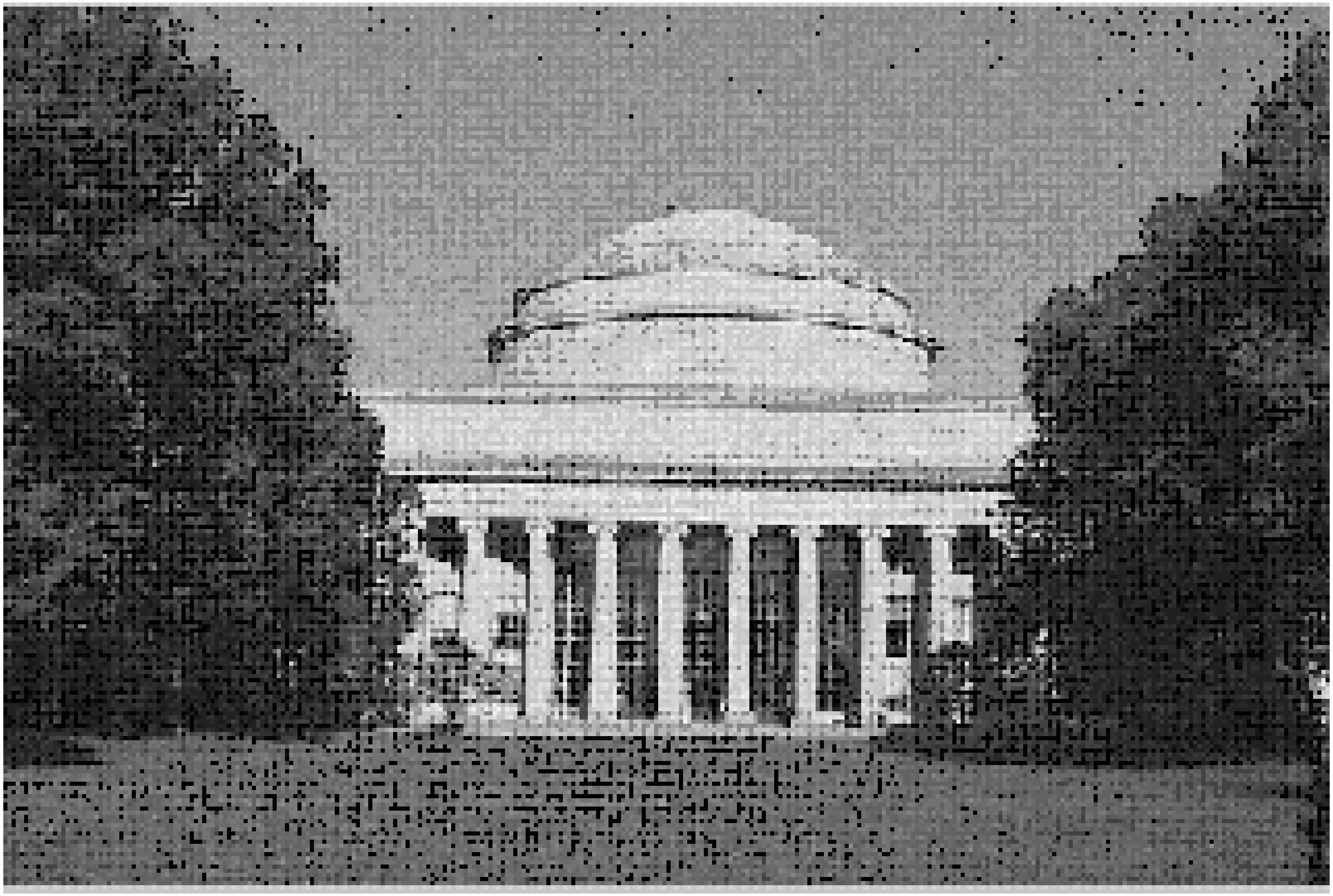
summe
Note th
values

http://web.mit.edu/persci/people/adelson/pub_pdfs/RCA84.pdf

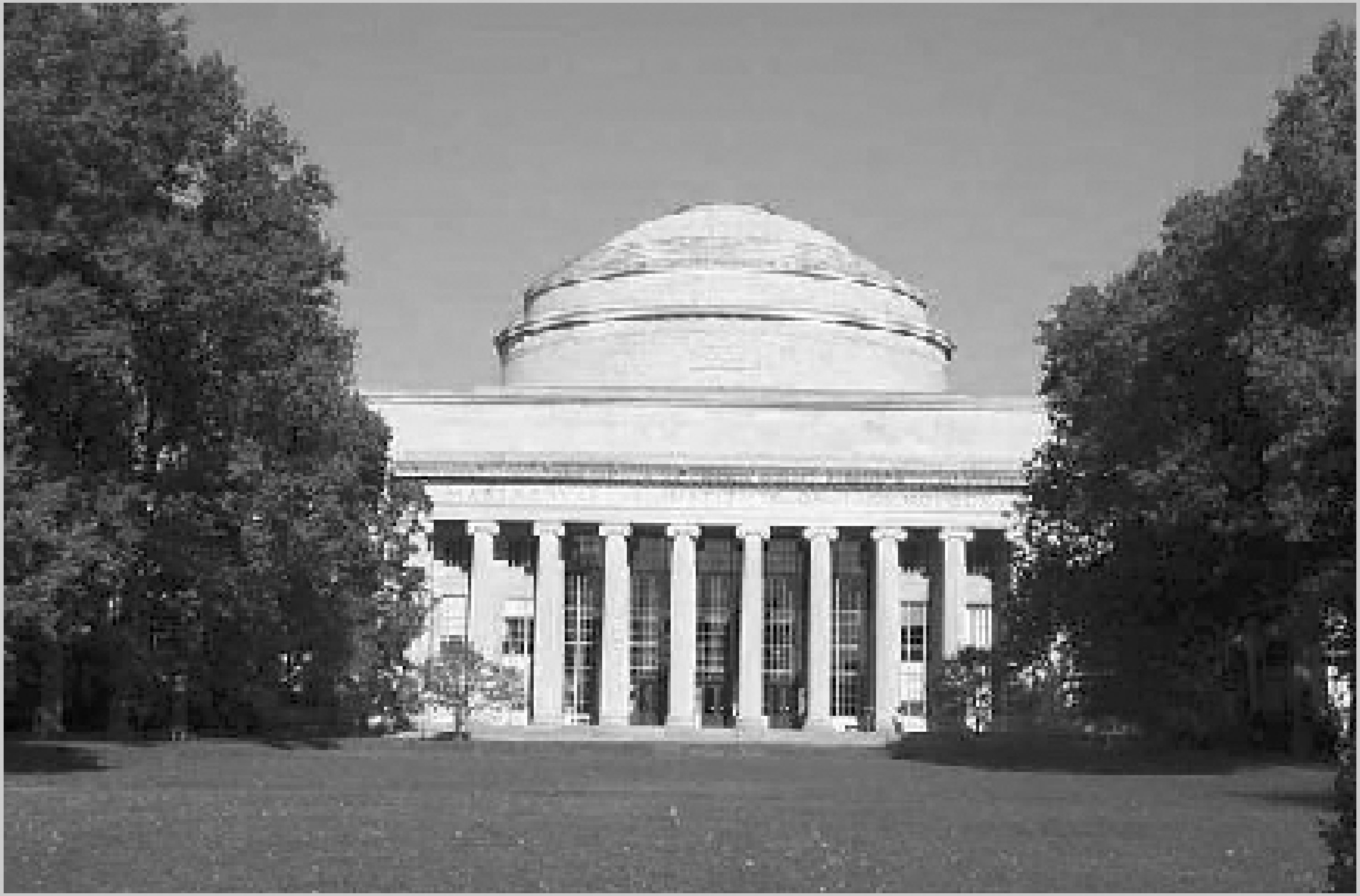
An application of image pyramids: noise removal

Image statistics (or, mathematically, how can you tell image from noise?)

Noisy image

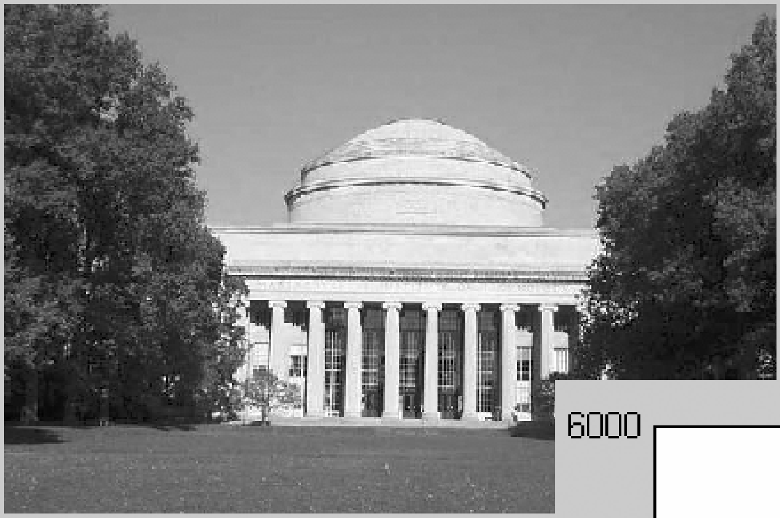


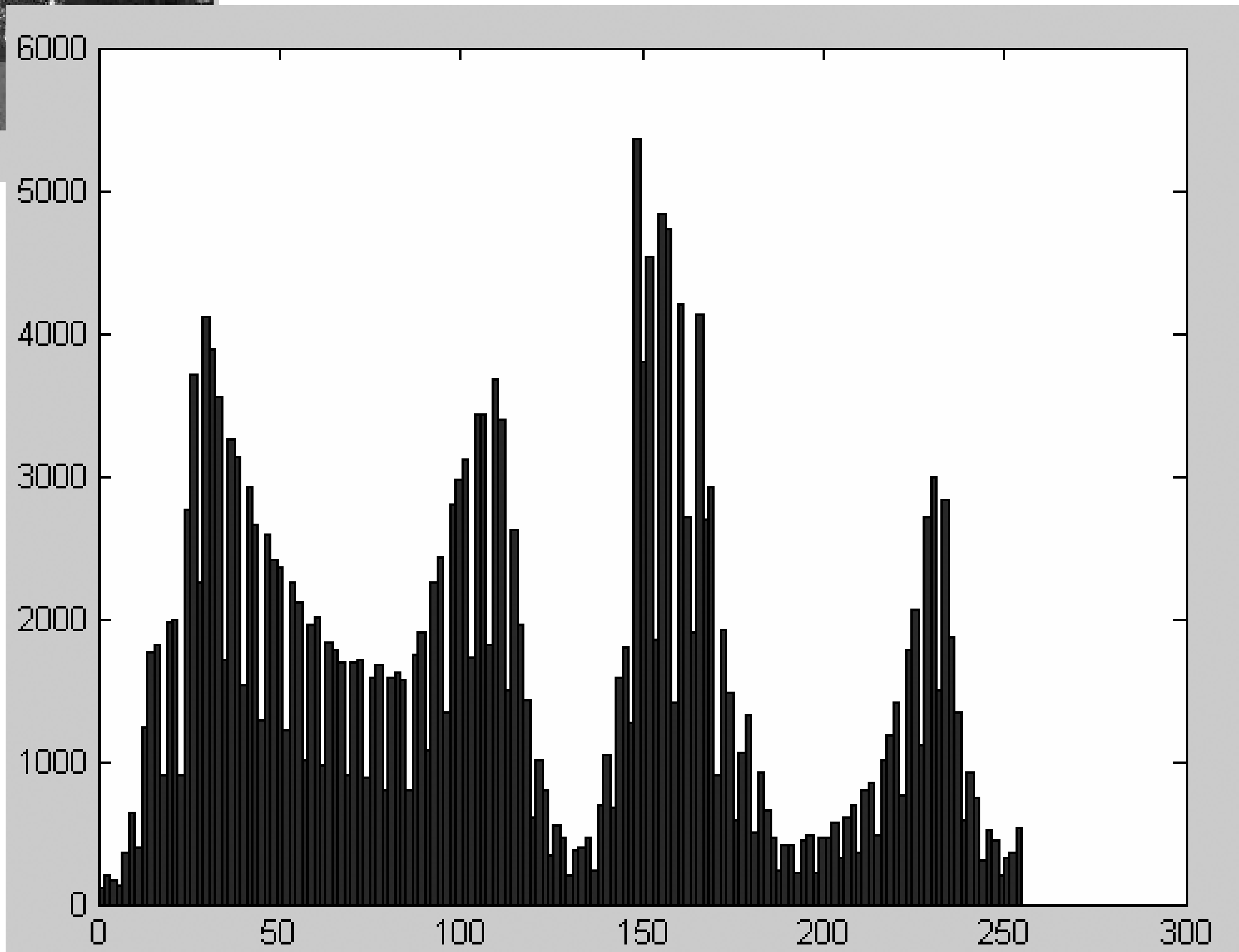
Clean image



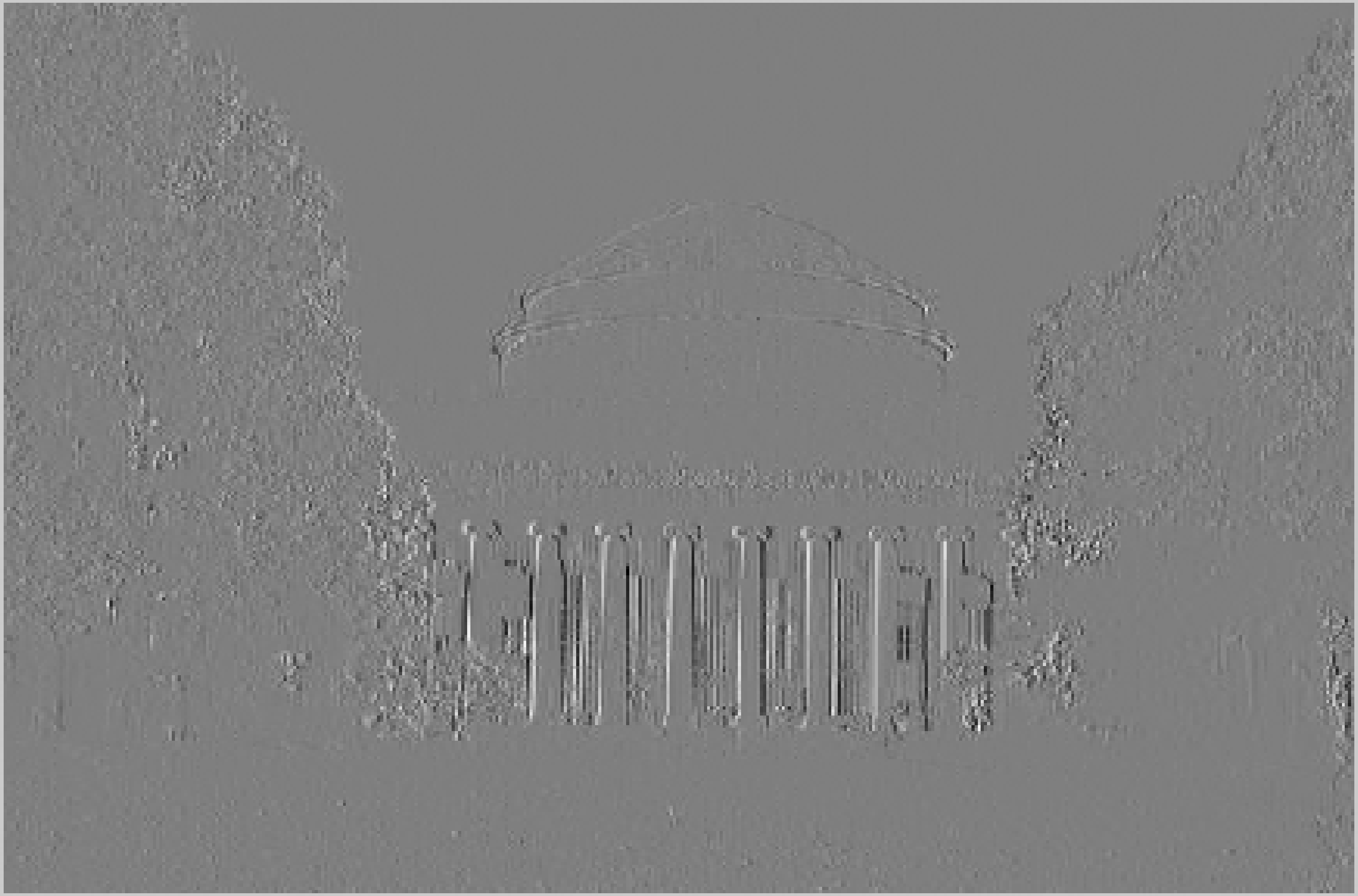
Range [0, 255]
Dims [394, 599]

Pixel representation image histogram

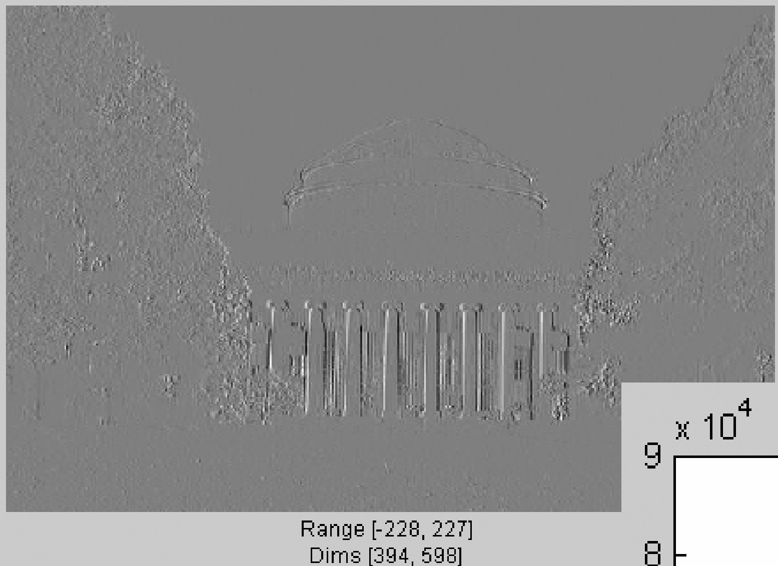




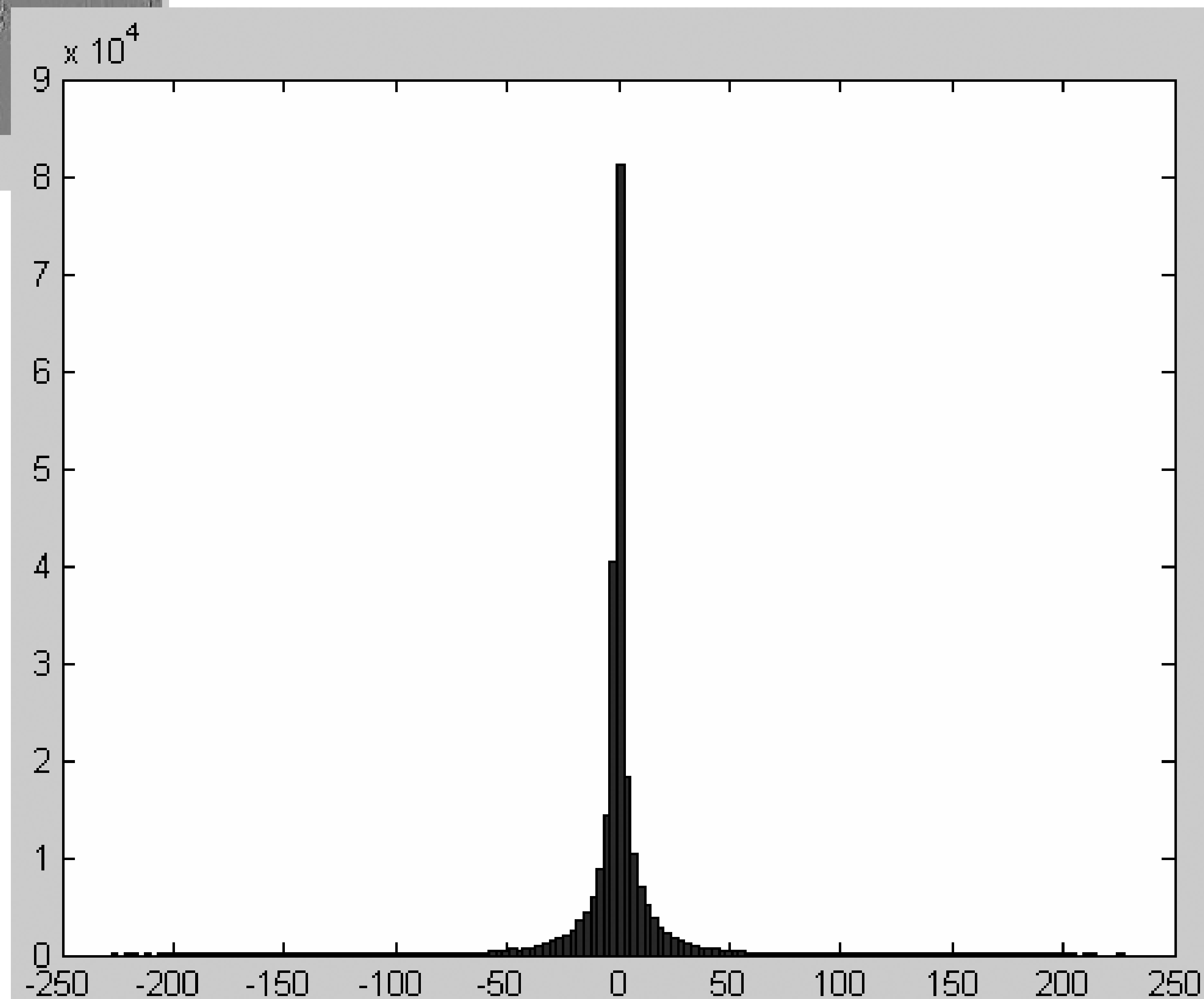
bandpass filtered image



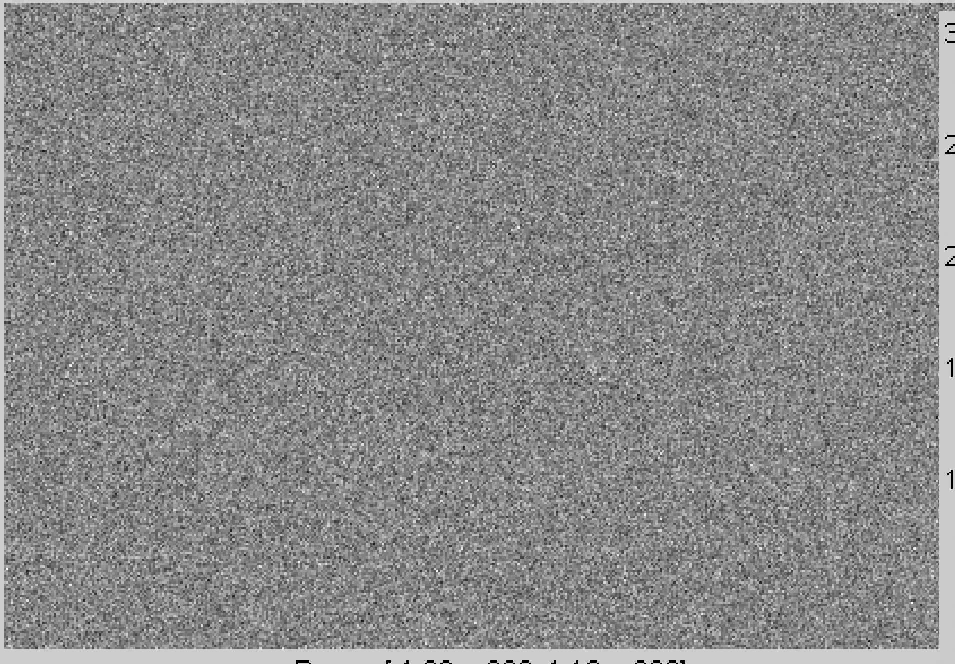
Range [-228, 227]
Dims [394, 598]



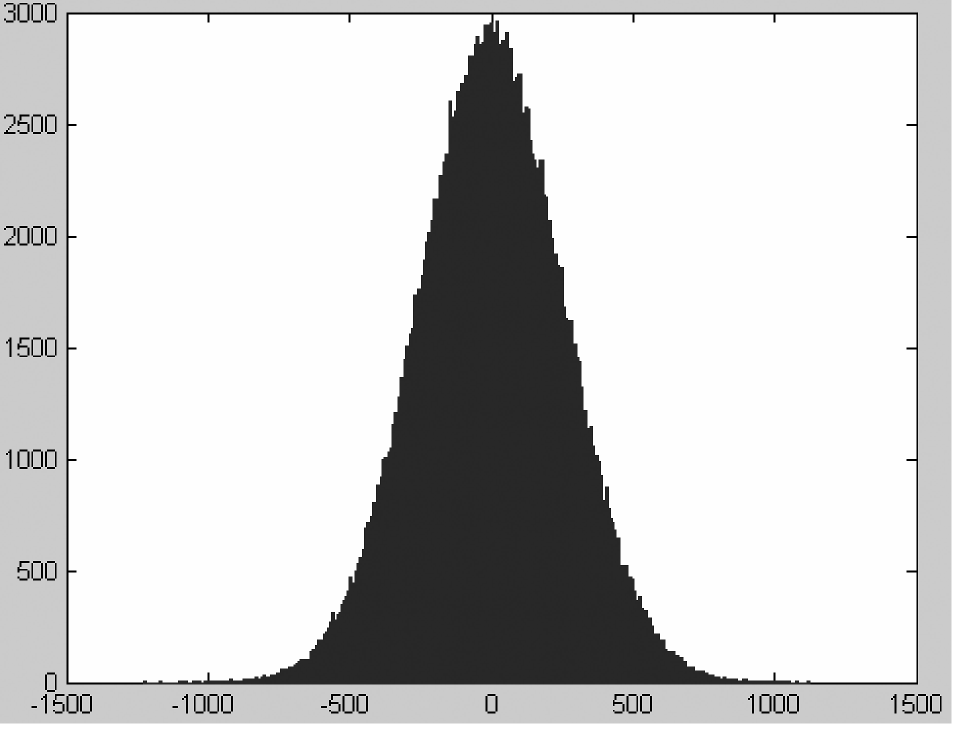
bandpassed representation
image histogram



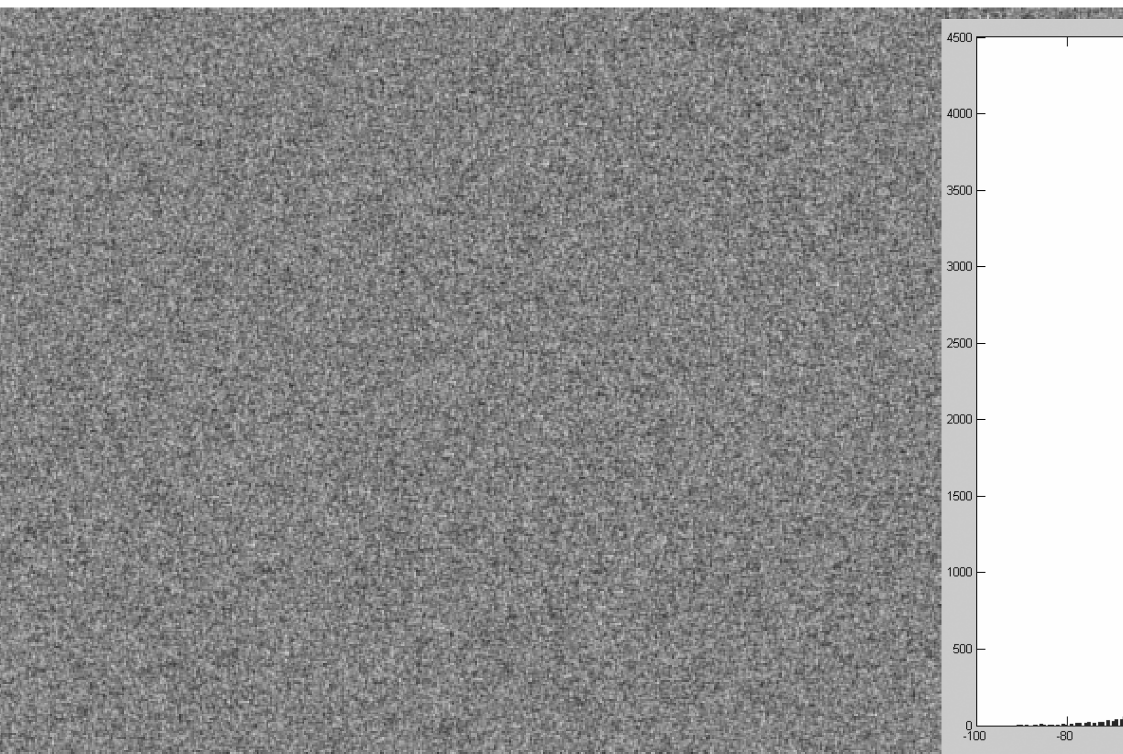
Pixel domain noise image and histogram

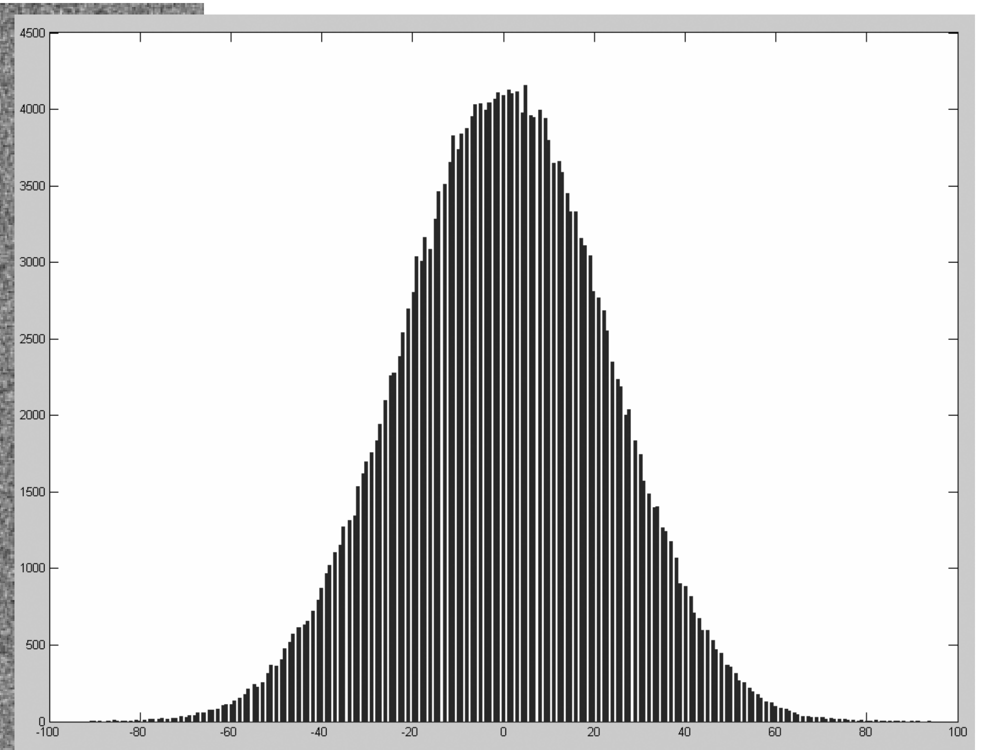


Range [-1.23e+003, 1.12e+003]
Dims [394, 599]

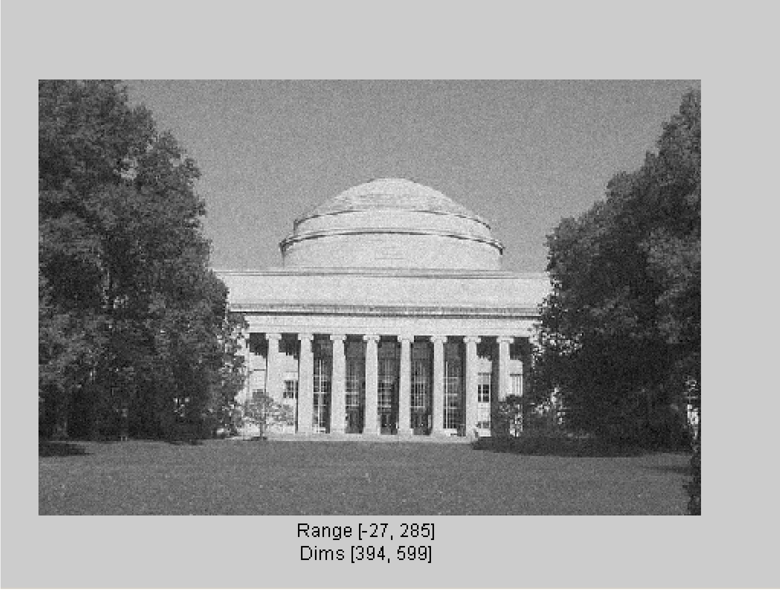


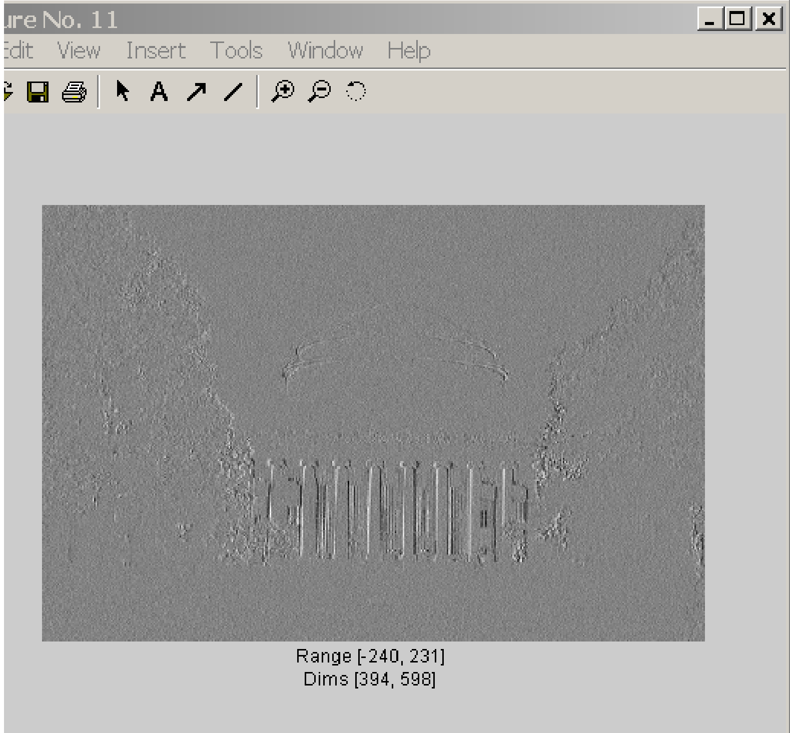
Bandpass domain noise image and histogram

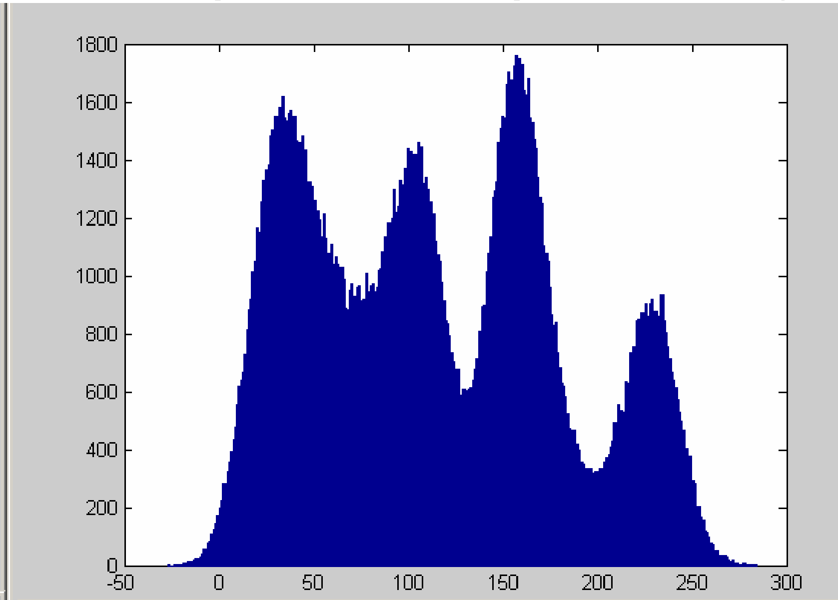


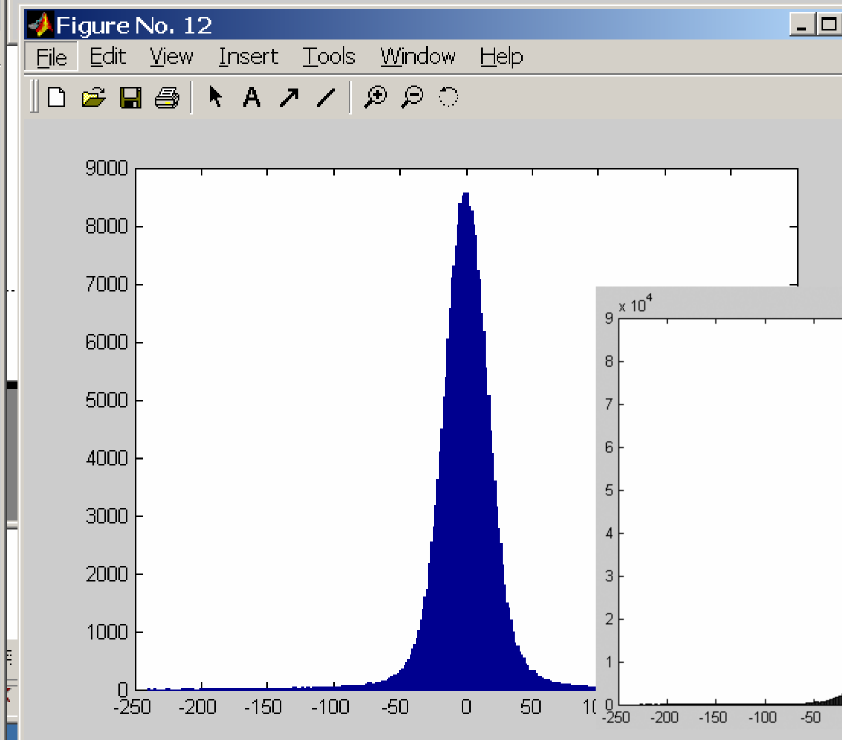


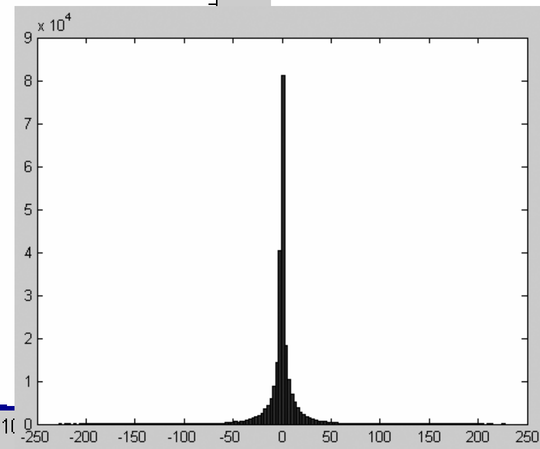
Noise-corrupted full-freq and bandpass images











But want the bandpass image histogram to look like this

Bayes theorem

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

By definition of conditional probability

so

Using that twice

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

and

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

The parameters you want to estimate — x in $P(x|y)$

What you observe — y in $P(x|y)$

Likelihood function — $P(y|x)$

Prior probability — $P(x)$

Constant w.r.t. parameters x. — $P(y)$

Bayesian MAP estimator for clean bandpass coefficient values

Let x = bandpassed image value before adding noise.

Let y = noise-corrupted observation.

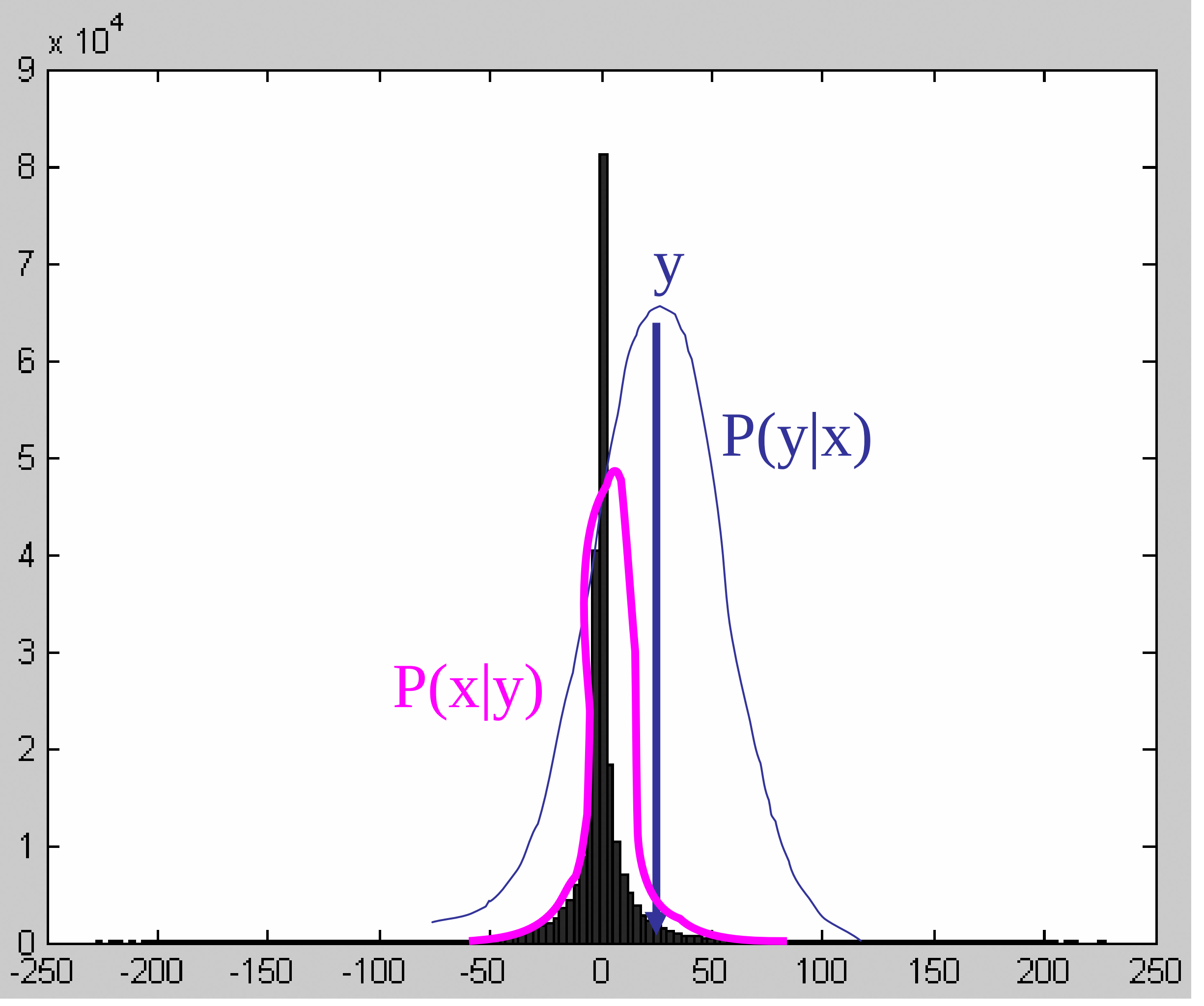
By Bayes theorem

P(x|y) = k P(y|x) P(x)

P(x)

P(y|x)

P(x|y)

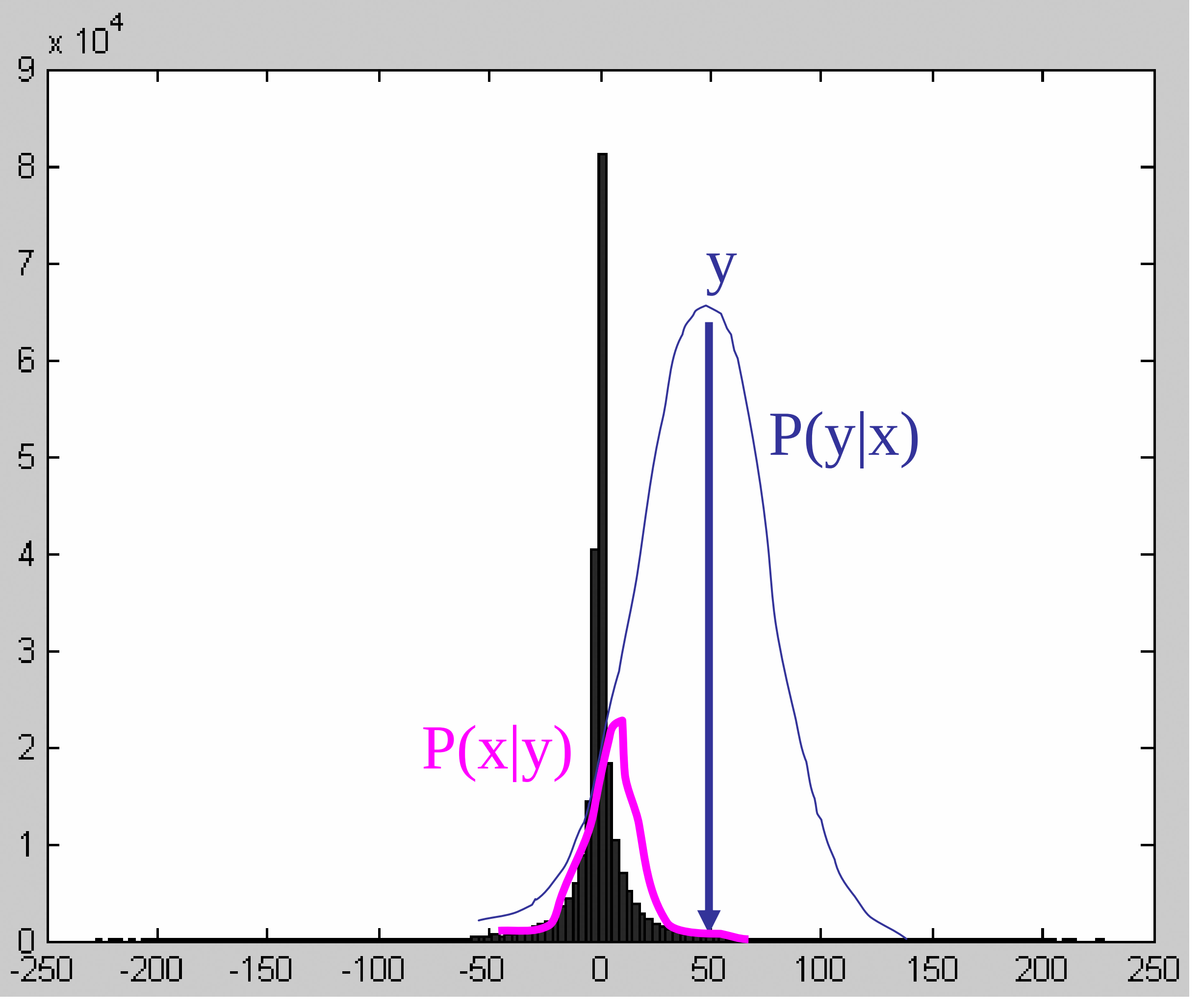


Bayesian MAP estimator

Let x = bandpassed image value before adding noise.
Let y = noise-corrupted observation.

By Bayes theorem

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

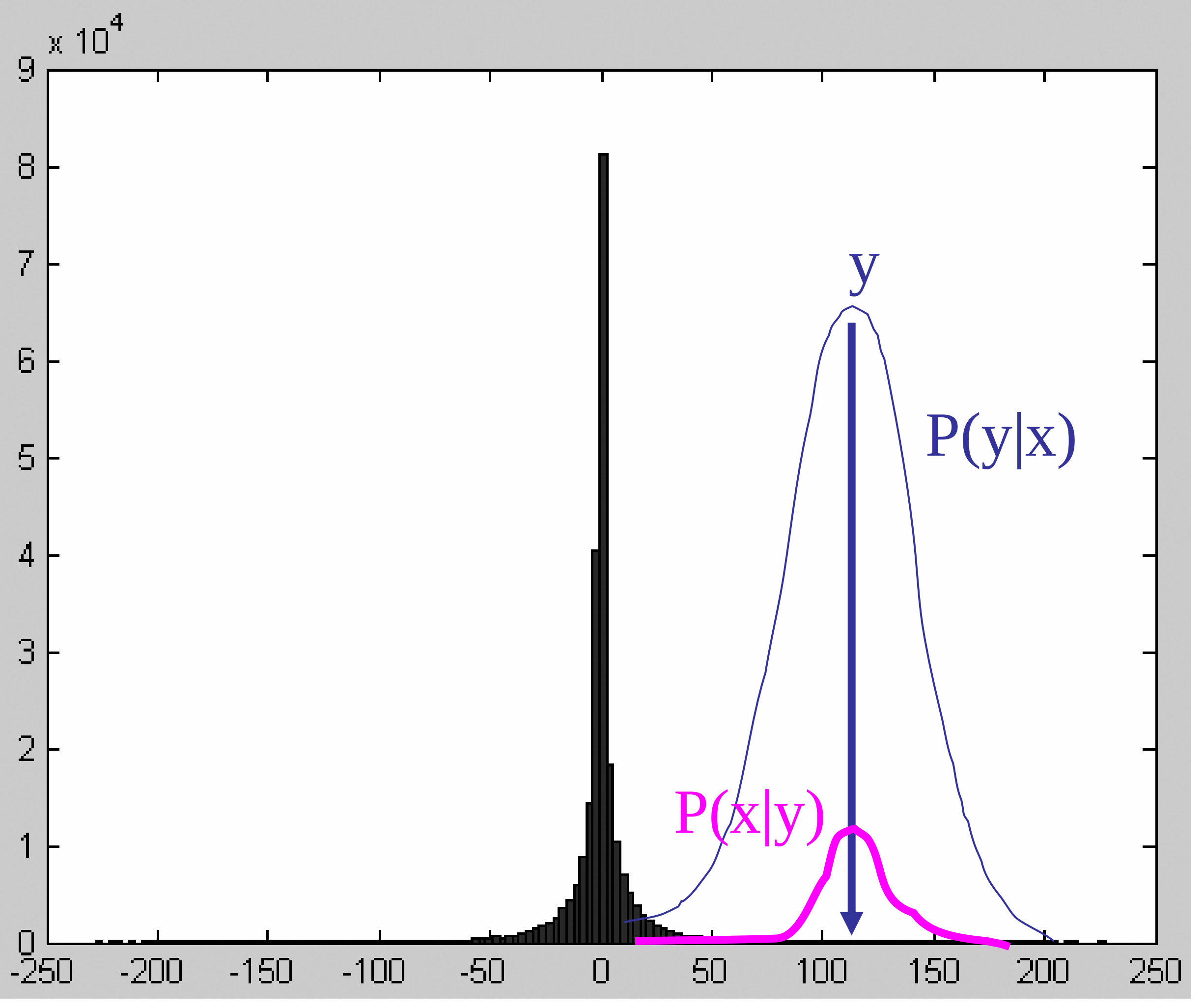


Bayesian MAP estimator

Let x = bandpassed image value before adding noise.
Let y = noise-corrupted observation.

By Bayes theorem

P(x|y) = k P(y|x) P(x)



MAP estimate, $\hat{x}$, as function of observed coefficient value, y

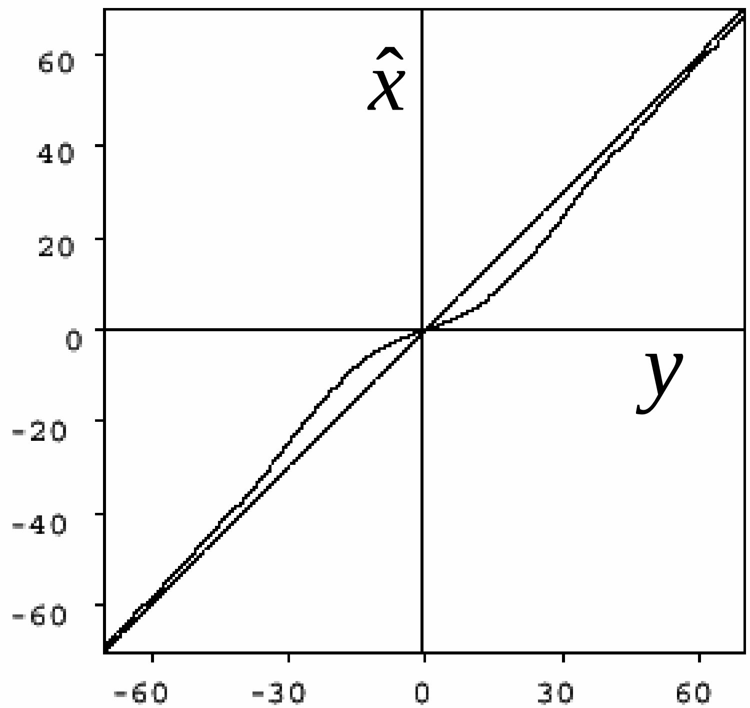


Figure 2: Bayesian estimator (symmetrized) for the signal and noise histograms shown in figure 1. Superimposed on the plot is a straight line indicating the identity function.

http://www-bcs.mit.edu/people/adelson/pub_pdfs/simoncelli_noise.pdf **Simoncelli and Adelson, Noise Removal via Bayesian Wavelet Coring**

Noise removal results

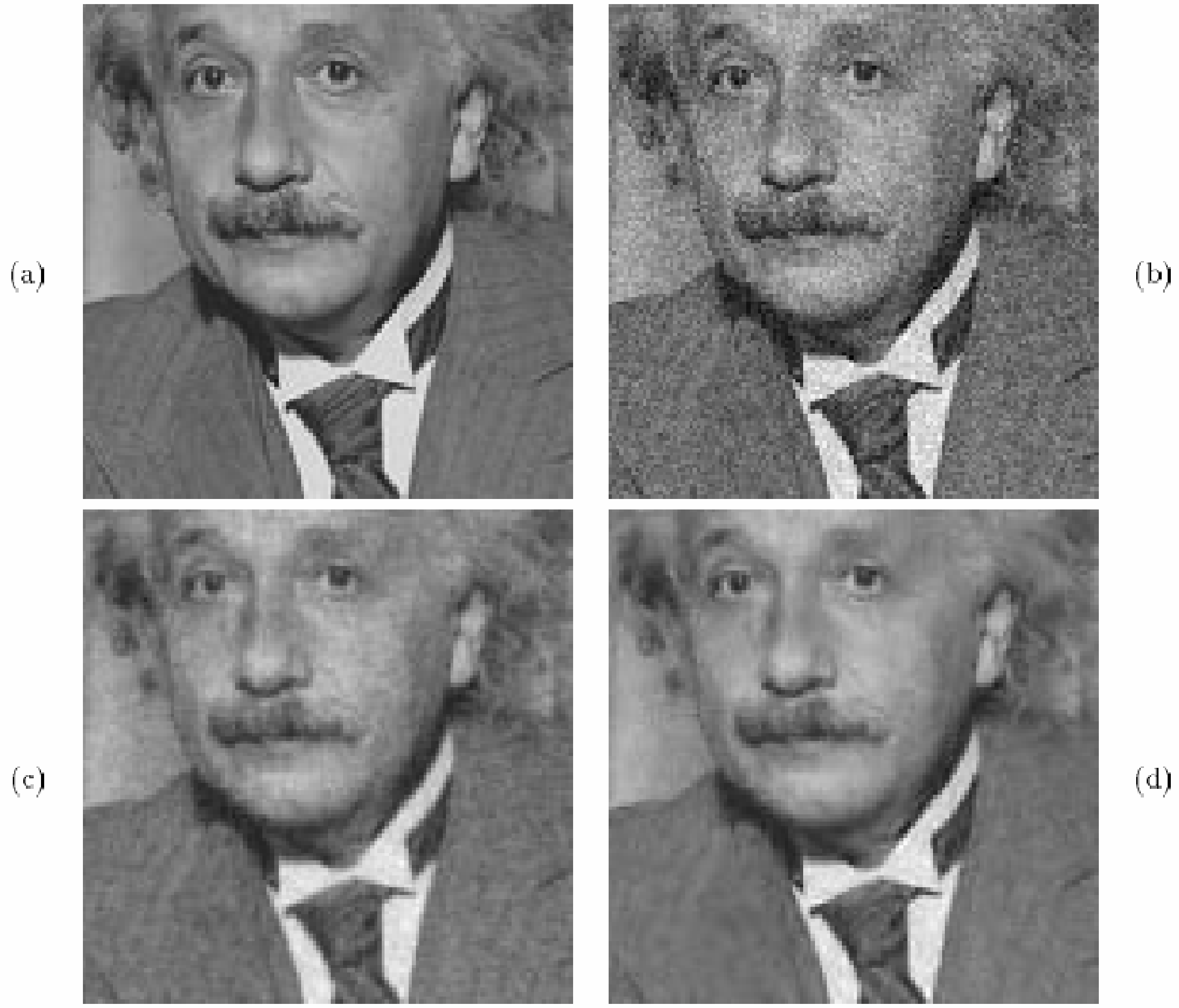


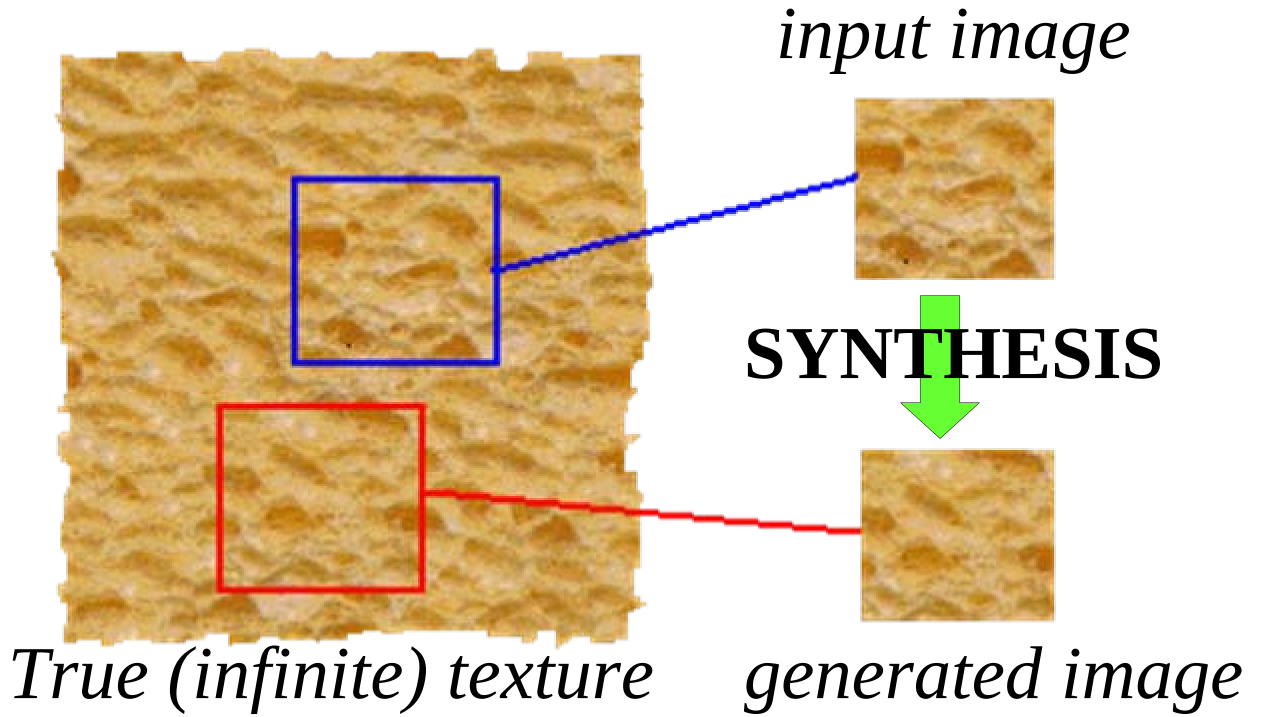
Figure 4: Noise reduction example. (a) Original image (cropped). (b) Image contaminated with additive Gaussian white noise (SNR = 9.00dB). (c) Image restored using (semi-blind) Wiener filter (SNR = 11.88dB). (d) Image restored using (semi-blind) Bayesian estimator (SNR = 13.82dB).

Simoncelli and Adelson, Noise Removal via Bayesian Wavelet Coring

http://www-bcs.mit.edu/people/adelson/pub_pdfs/simoncelli_noise.pdf

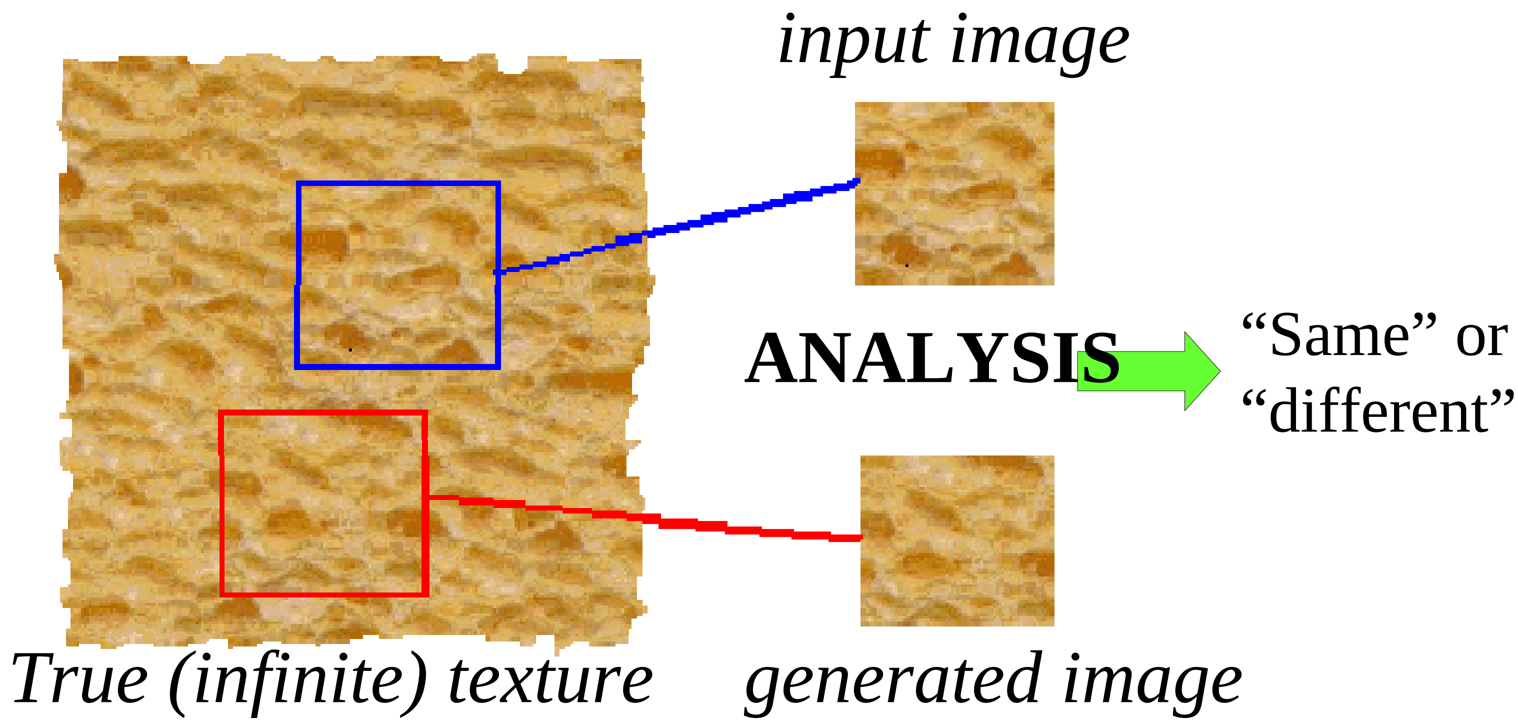
Image texture

The Goal of Texture Synthesis



- Given a finite sample of some texture, the goal is to synthesize other samples from that same texture
  - The sample needs to be "large enough“

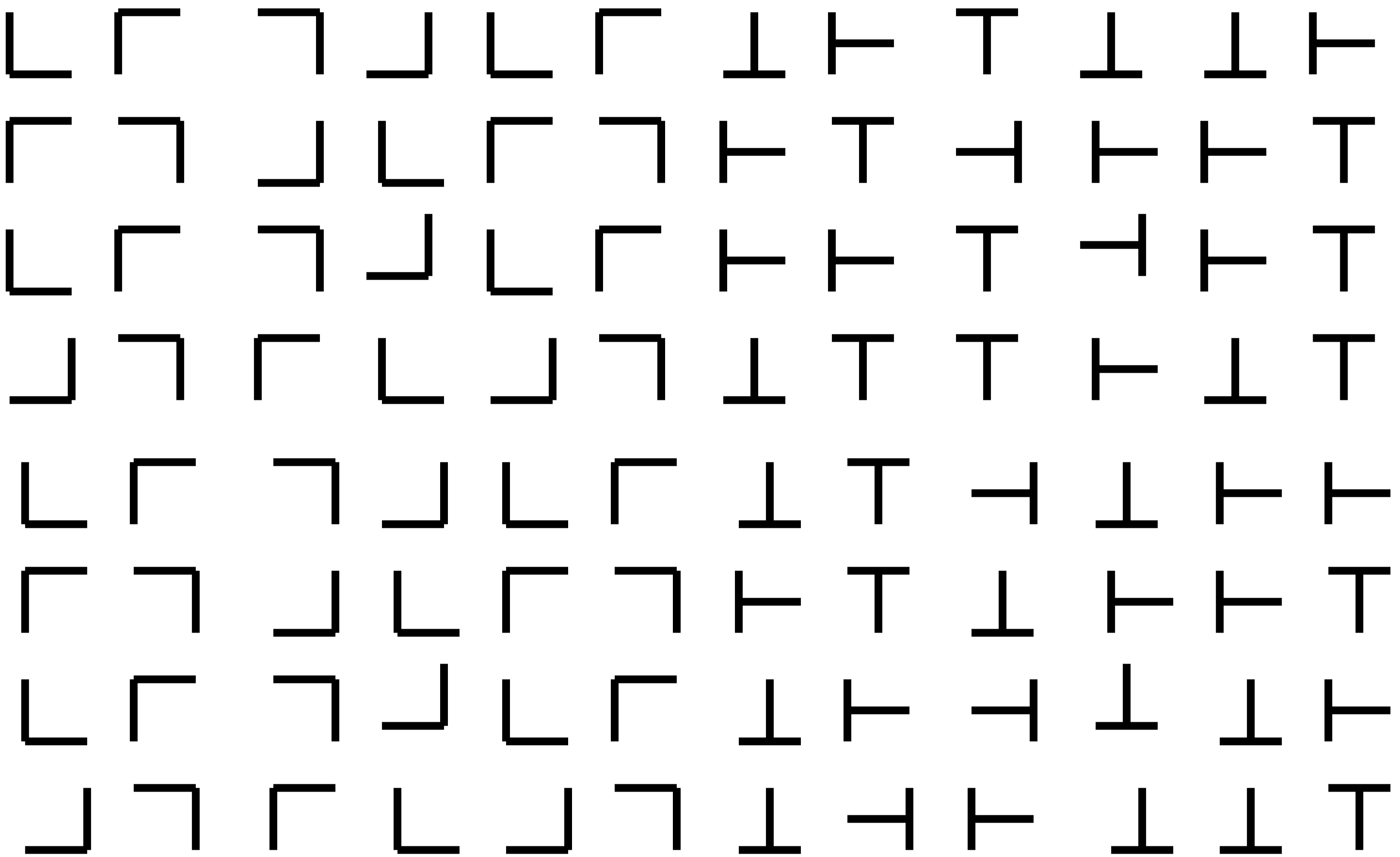
The Goal of Texture Analysis



Compare textures and decide if they’re made of the same “stuff”.

Pre-attentive texture discrimination

Pre-attentive texture discrimination

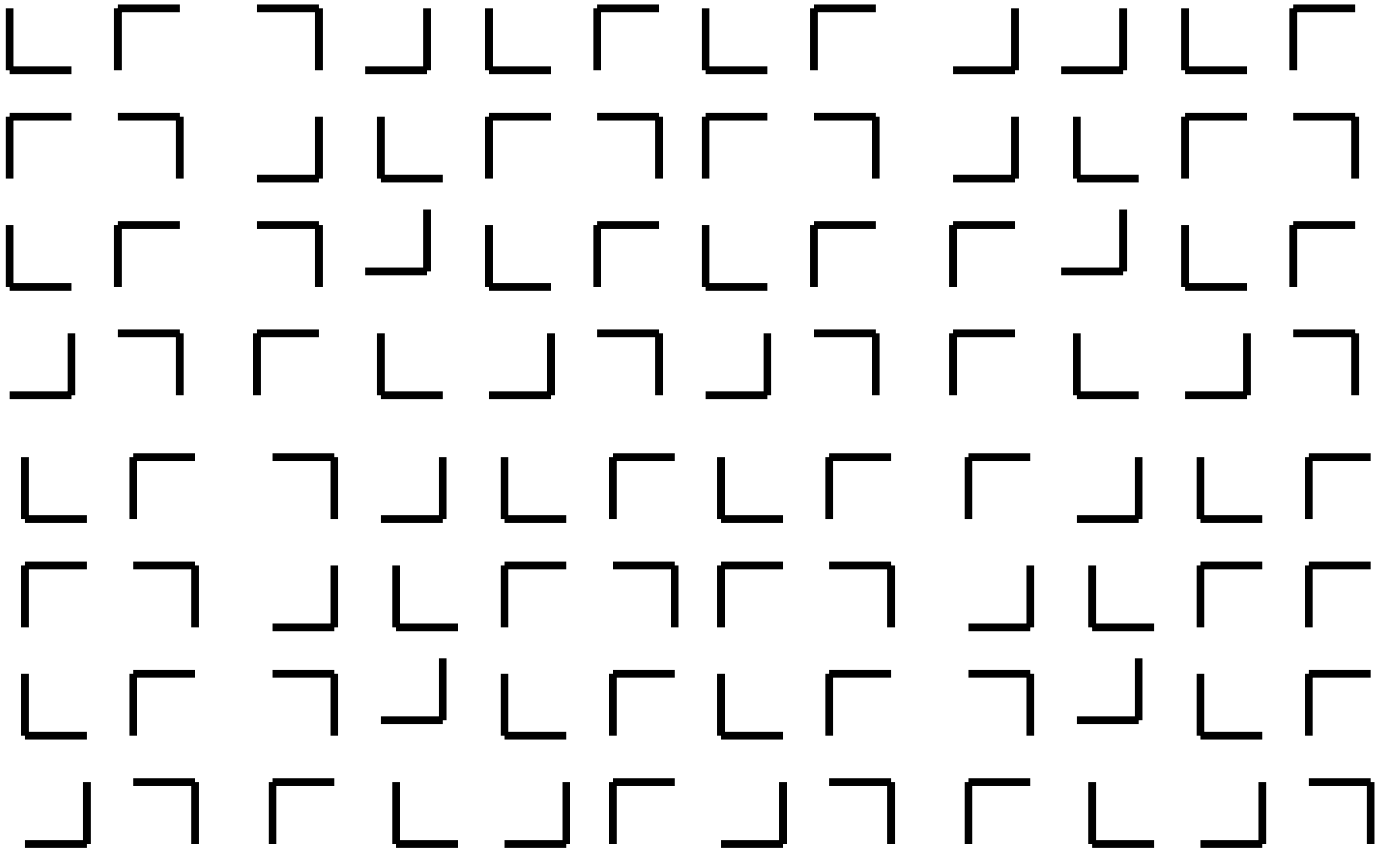


Pre-attentive texture discrimination

Same or different textures?

Pre-attentive texture discrimination

Pre-attentive texture discrimination

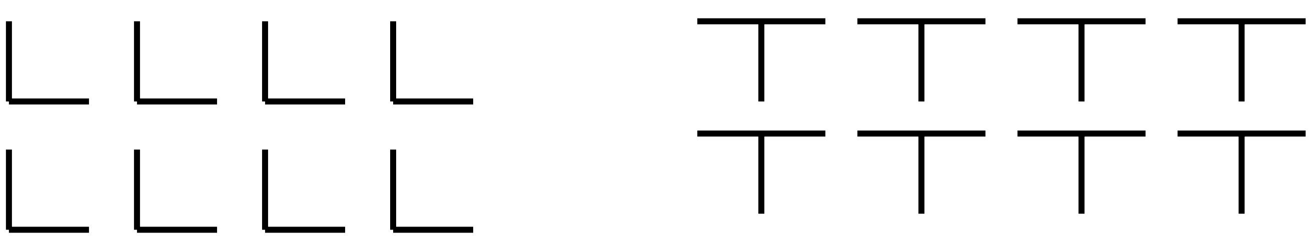


Pre-attentive texture discrimination

Same or different textures?

Julesz

- Textons: analyze the texture in terms of statistical relationships between fundamental texture elements, called “textons”.
- It generally required a human to look at the texture in order to decide what those fundamental units were...



Influential paper:

Early vision and texture perception

James R. Bergen* & Edward H. Adelson**

* SRI David Sarnoff Research Center, Princeton,
New Jersey 08540, USA
** Media Lab and Department of Brain and Cognitive Science,
Massachusetts Institute of Technology, Cambridge,
Massachusetts 02139, USA

Learn: use filters.

Bergen and Adelson, Nature 1988

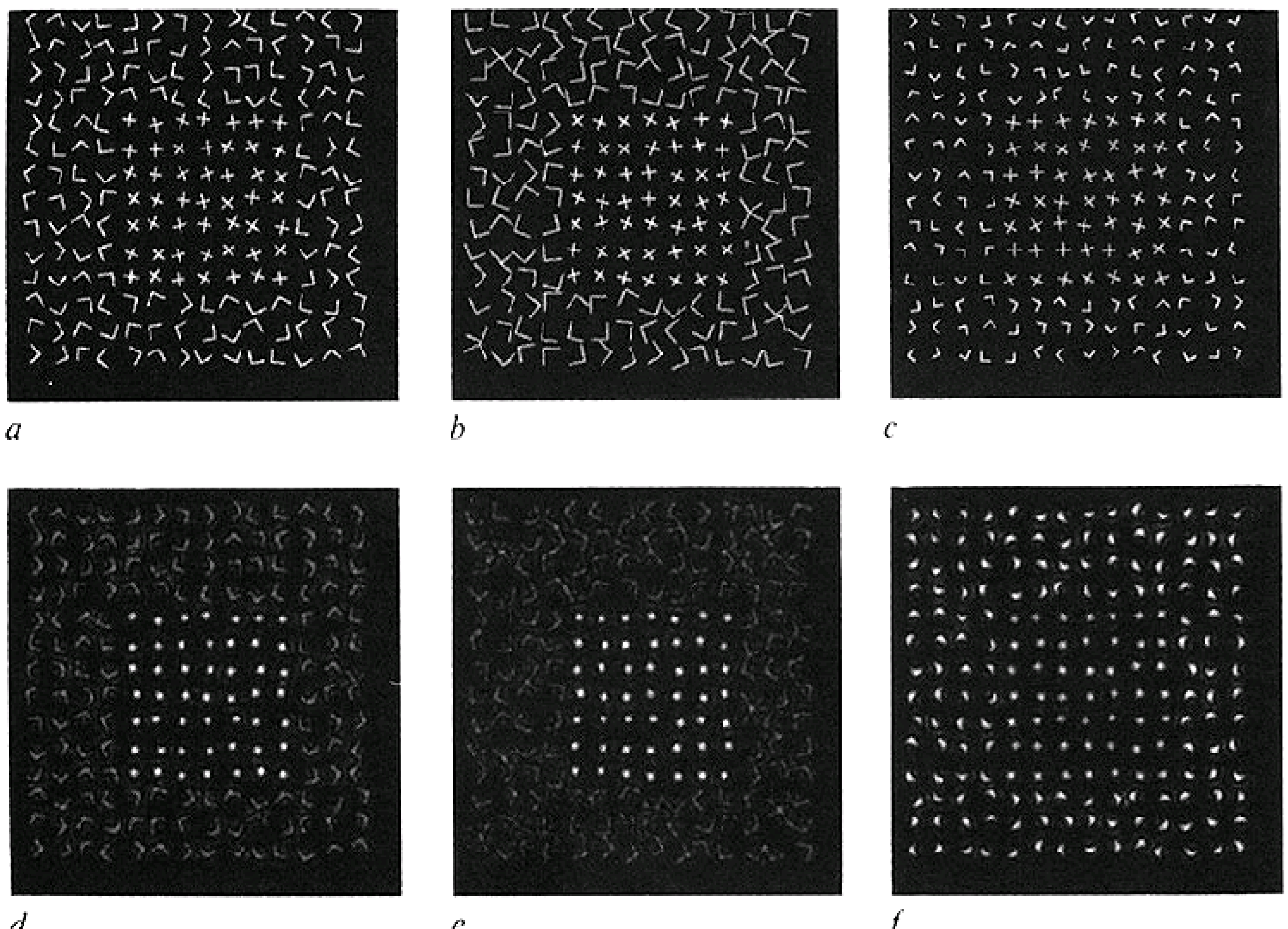
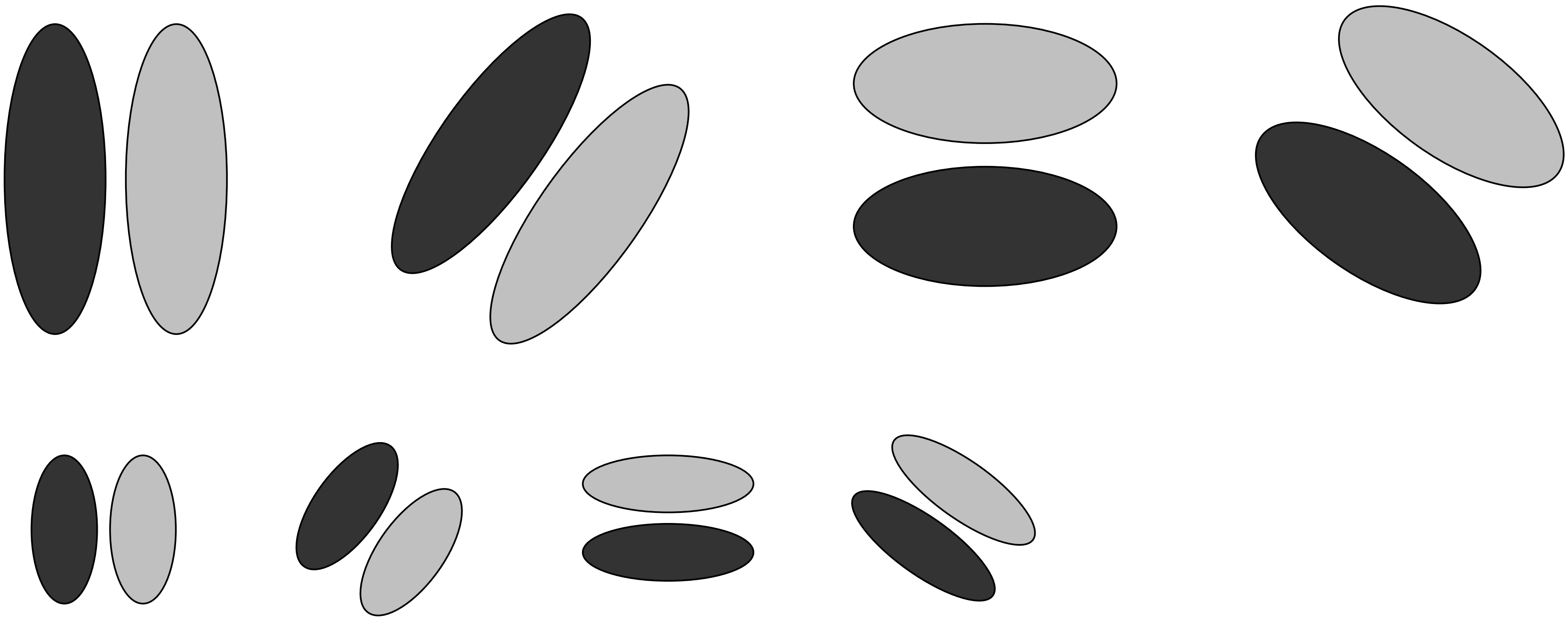


Fig. 1 *Top row*, Textures consisting of Xs within a texture composed of Ls. The micropatterns are placed at random orientations on a randomly perturbed lattice. *a*, The bars of the Xs have the same length as the bars of the Ls. *b*, The bars of the Ls have been lengthened by 25%, and the intensity adjusted for the same mean luminance. Discriminabitity is enhanced. *c*, The bars of the Ls have been shortened by 25%, and the intensity adjusted for the same mean luminance. Discriminabitity is impaired. *Bottom row*: the responses of a size-tuned mechanism *d*, response to image *a*; *e*, response to image *b*; *f*; response to image *c*.

Learn: use lots of filters, multi-ori&scale.

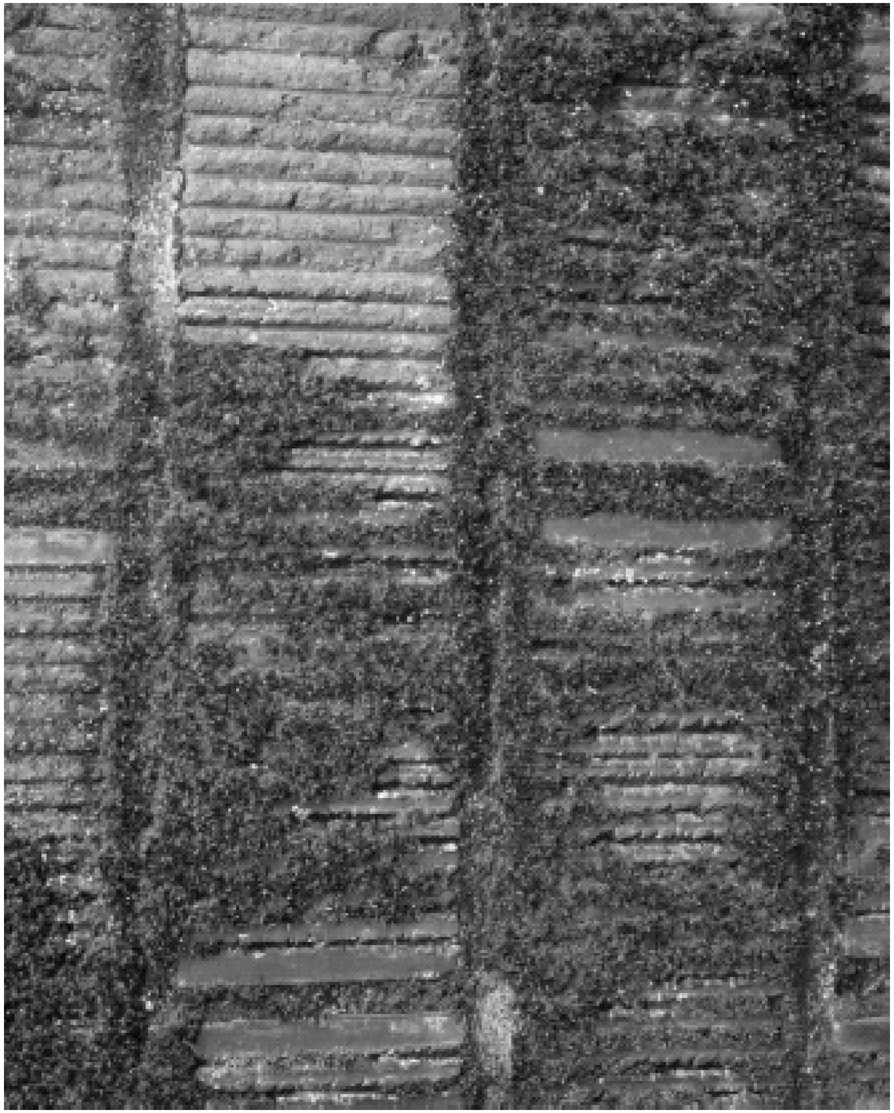
Malik and Perona

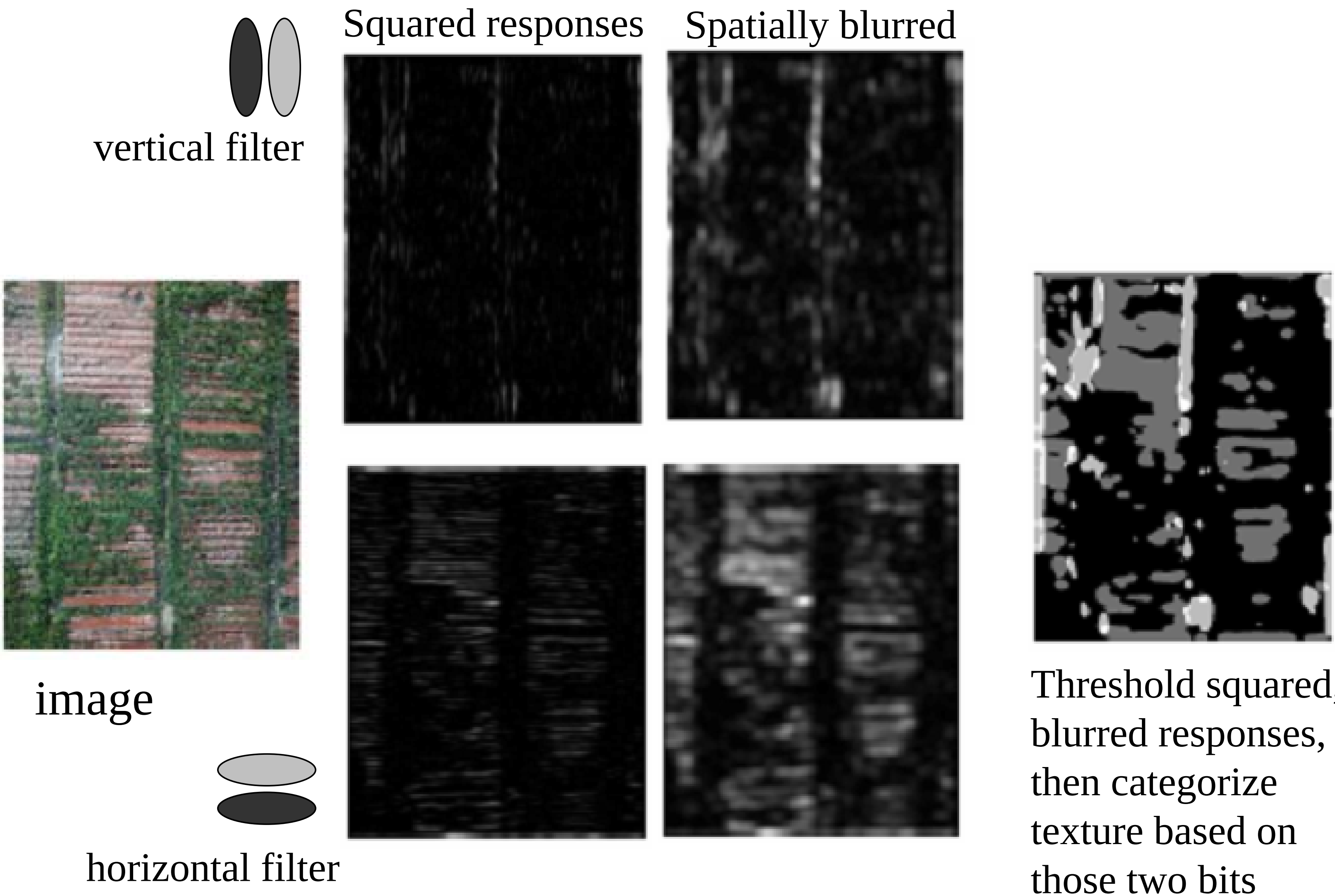


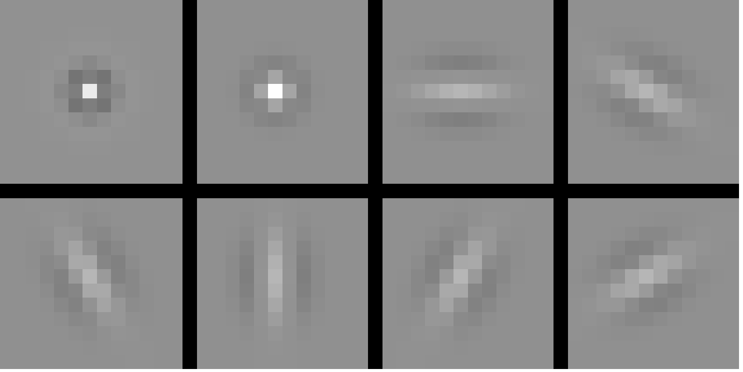
Malik J, Perona P. Preattentive texture discrimination with early vision mechanisms. J OPT SOC AM A 7: (5) 923-932 MAY 1990

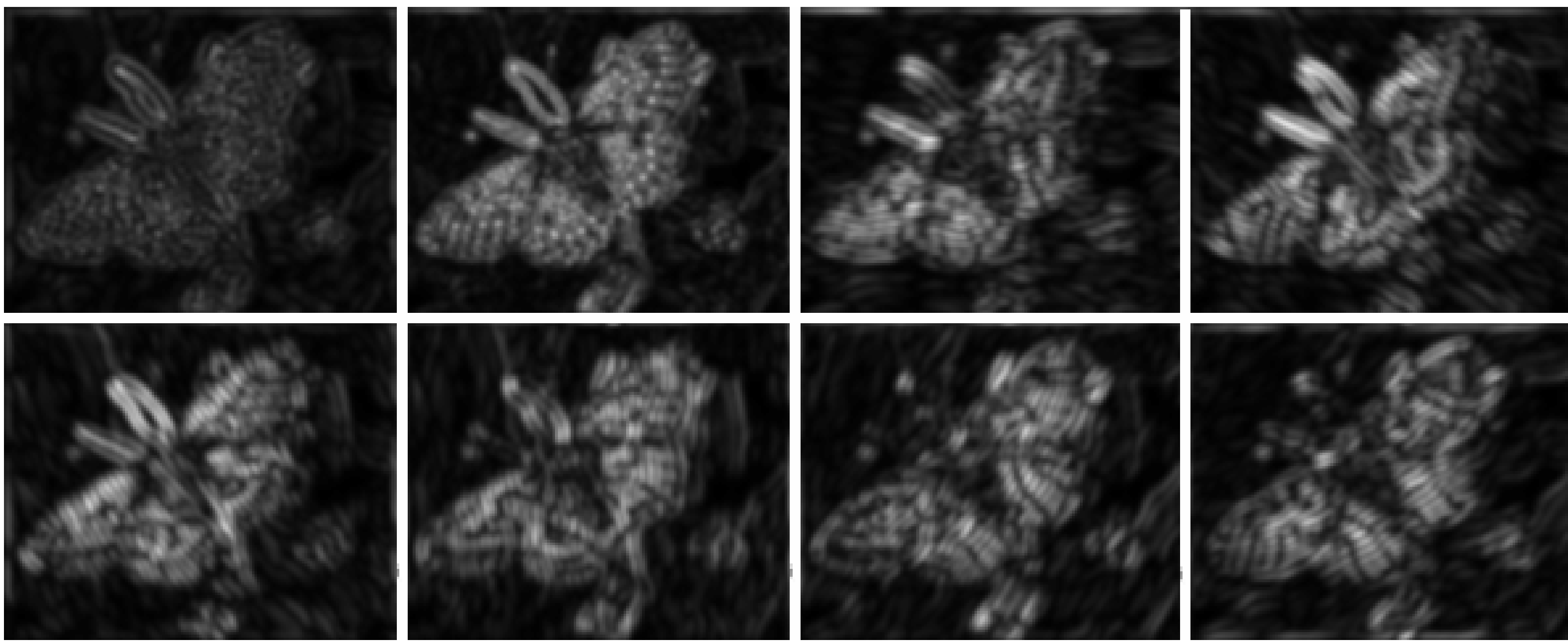
Representing textures

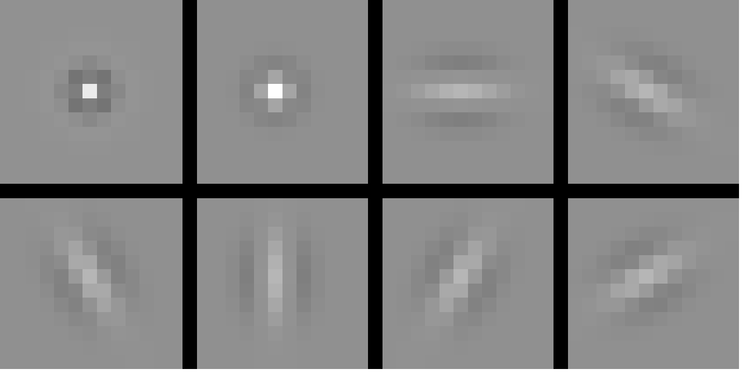
- Textures are made up of quite stylised subelements, repeated in meaningful ways
- Representation:
  - find the subelements, and represent their statistics
- But what are the subelements, and how do we find them?
  - recall normalized correlation
  - find subelements by applying filters, looking at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

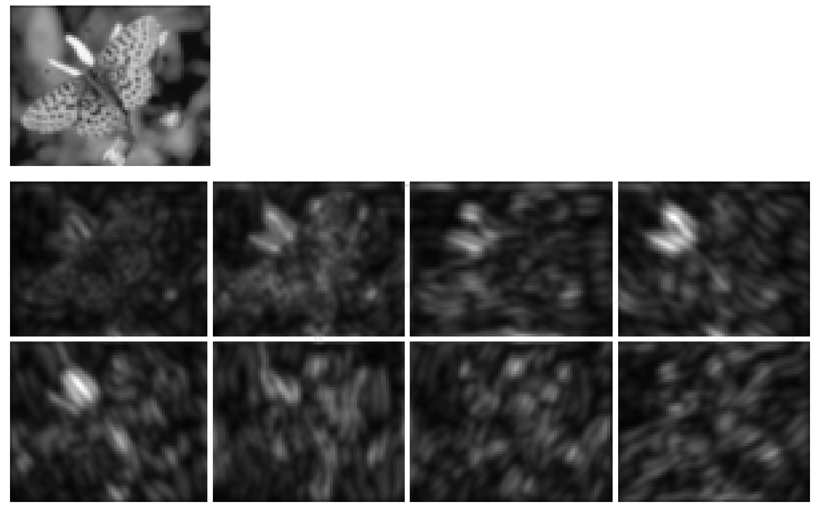


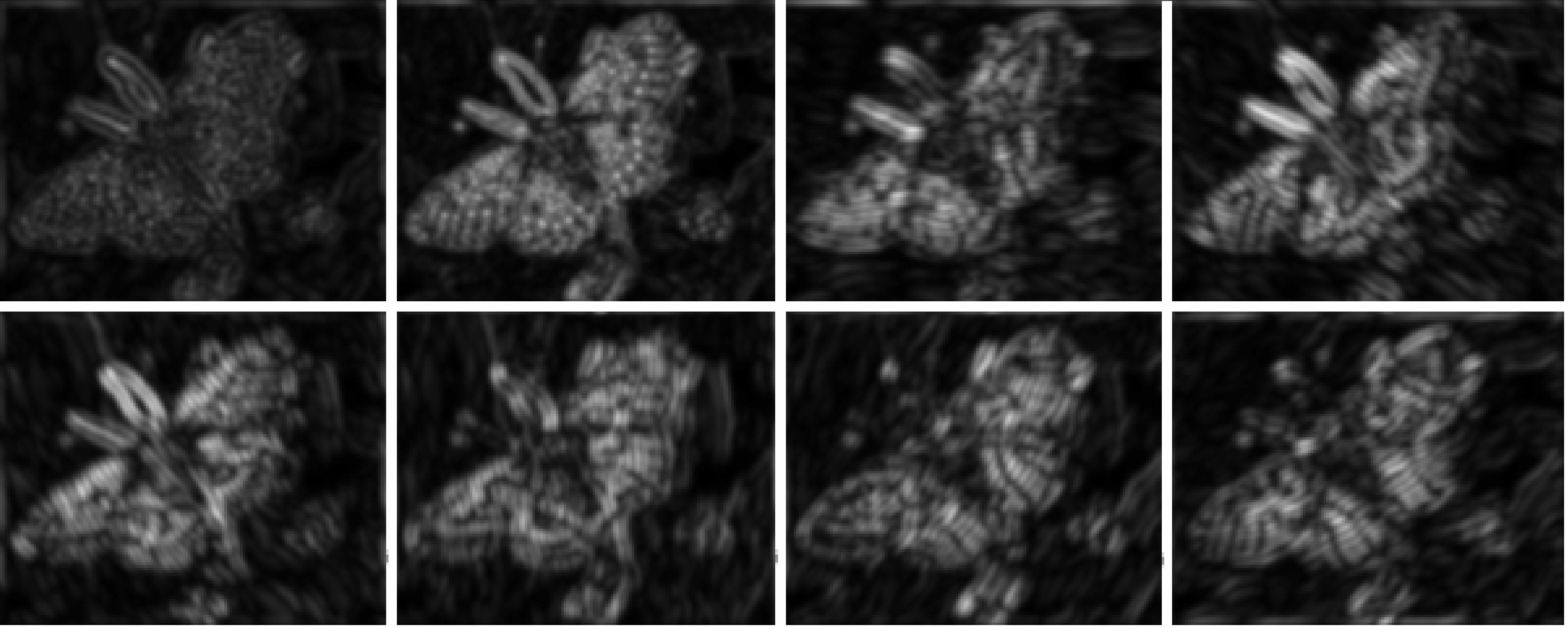
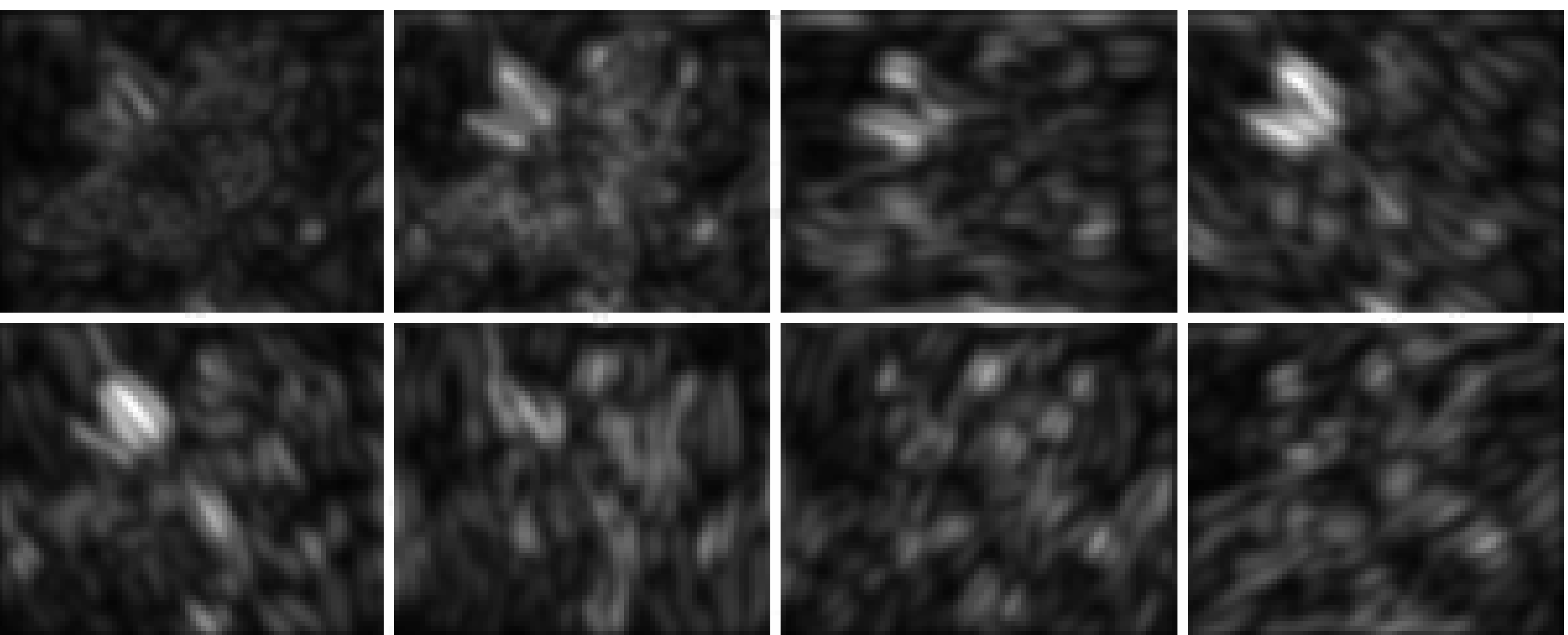

Squared responses
Spatially blurred
vertical filter
image
horizontal filter
Threshold squared, blurred responses, then categorize texture based on those two bits









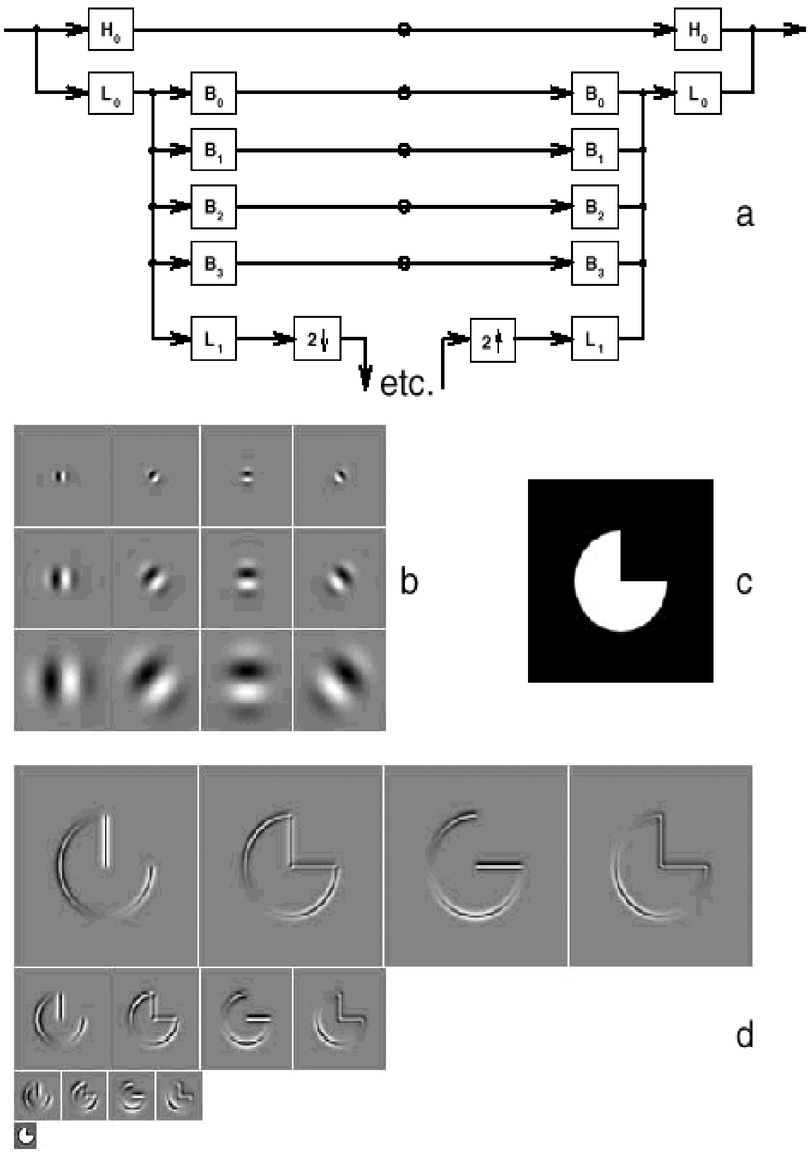
If matching the averaged squared filter values is a good way to match a given texture, then maybe matching the entire marginal distribution (eg, the histogram) of a filter's response would be even better.

Jim Bergen proposed this…

Pyramid-Based Texture Analysis/Synthesis

David J. Heeger[‡]
Stanford University

James R. Bergen[†]
SRI David Sarnoff Research Center



SIGGRAPH 1994

Histogram matching algorithm

```
Match-histogram (im1,im2)
  im1-cdf = Make-cdf(im1)
  im2-cdf = Make-cdf(im2)
  inv-im2-cdf = Make-inverse-lookup-table(im2-cdf)
  Loop for each pixel do
       im1[pixel] =
         Lookup(inv-im2-cdf,
                Lookup(im1-cdf,im1[pixel]))
```

“At this im1 pixel value, 10% of the im1 values are lower. What im2 pixel value has 10% of the im2 values below it?”

Heeger-Bergen texture synthesis algorithm

```
Match-texture(noise,texture)
  Match-Histogram (noise,texture)
  analysis-pyr = Make-Pyramid (texture)
  Loop for several iterations do
     synthesis-pyr = Make-Pyramid (noise)
     Loop for a-band in subbands of analysis-pyr
          for s-band in subbands of synthesis-pyr
          do
          Match-Histogram (s-band,a-band)
     noise = Collapse-Pyramid (synthesis-pyr)
     Match-Histogram (noise,texture)
```

Alternate matching the histograms of all the subbands and matching the histograms of the reconstructed images.

Learn: use filter marginal statistics.

Bergen and Heeger

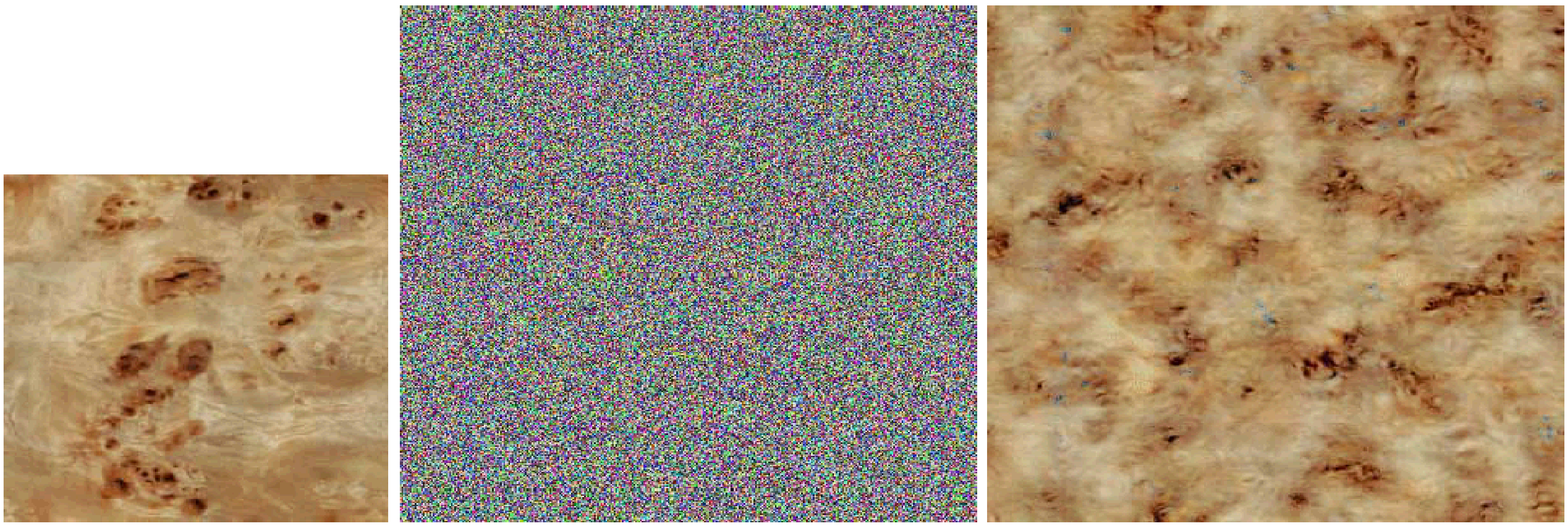


Figure 2: (Left) Input digitized sample texture: burled mappa wood. (Middle) Input noise. (Right) Output synthetic texture that matches the appearance of the digitized sample. Note that the synthesized texture is larger than the digitized sample; our approach allows generation of as much texture as desired. In addition, the synthetic textures tile seamlessly.

Heeger/Bergen, Siggraph 1994

Bergen and Heeger results

Figure 3: In each pair left image is original and right image is synthetic: stucco, iridescent ribbon, green marble, panda fur, slag stone, figured yew wood.

Heeger/Bergen, Siggraph 1994

Bergen and Heeger failures

Figure 8: Examples of failures: wood grain and red coral.

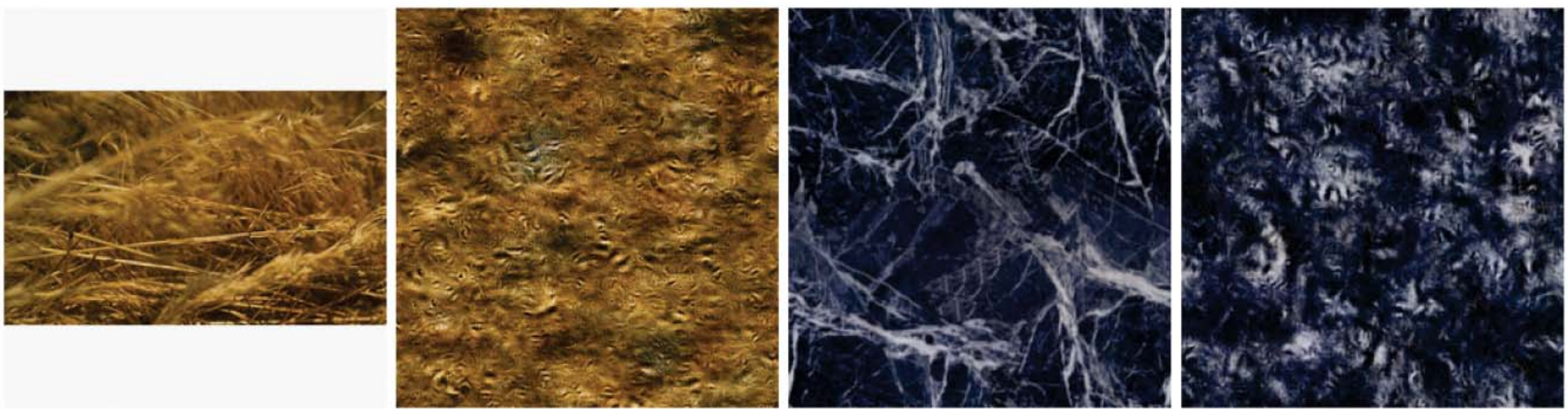
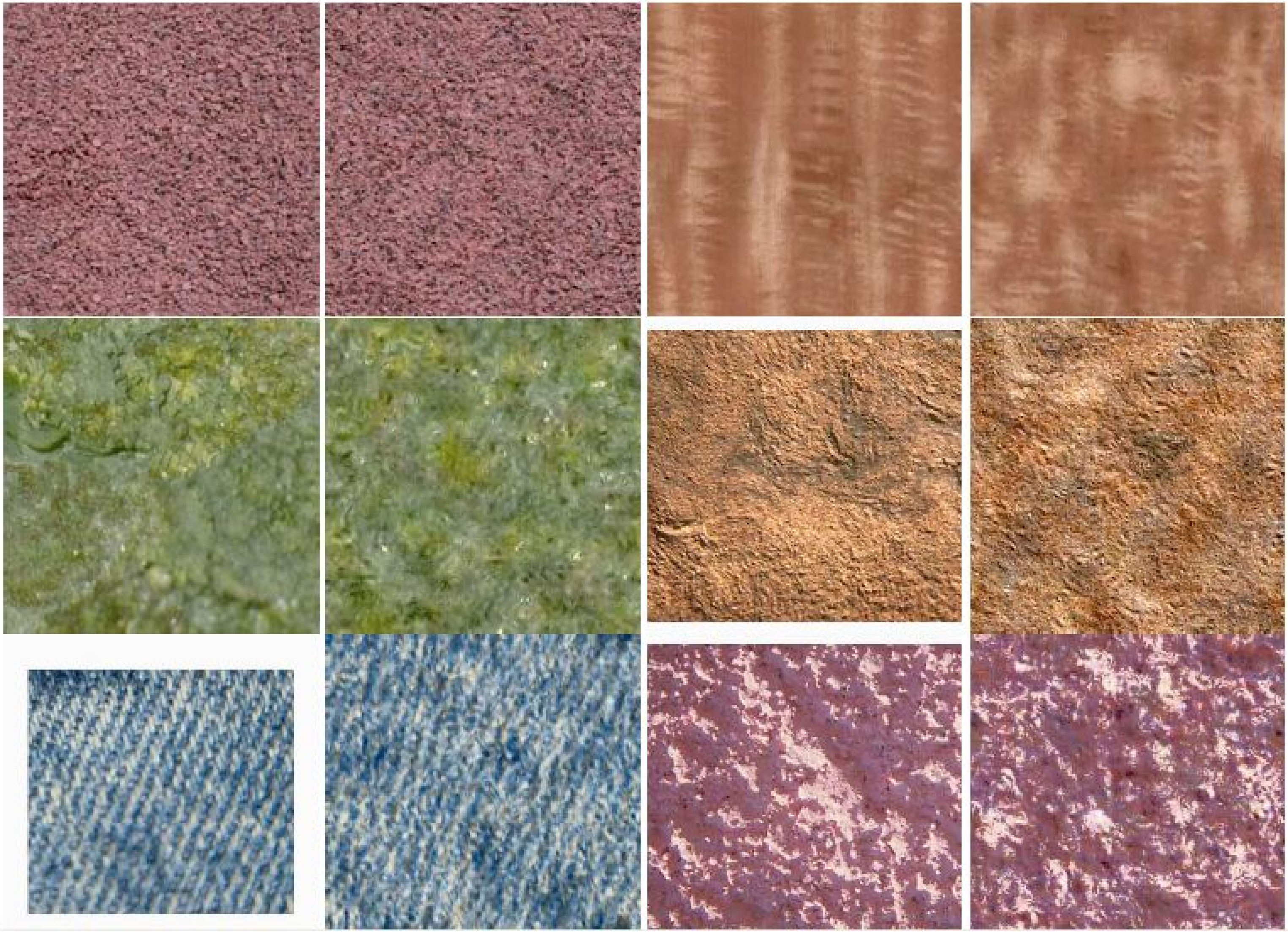


Figure 9: More failures: hay and marble.

Heeger/Bergen, Siggraph 1994

More examples



Heeger/Bergen, Siggraph 1994

More examples

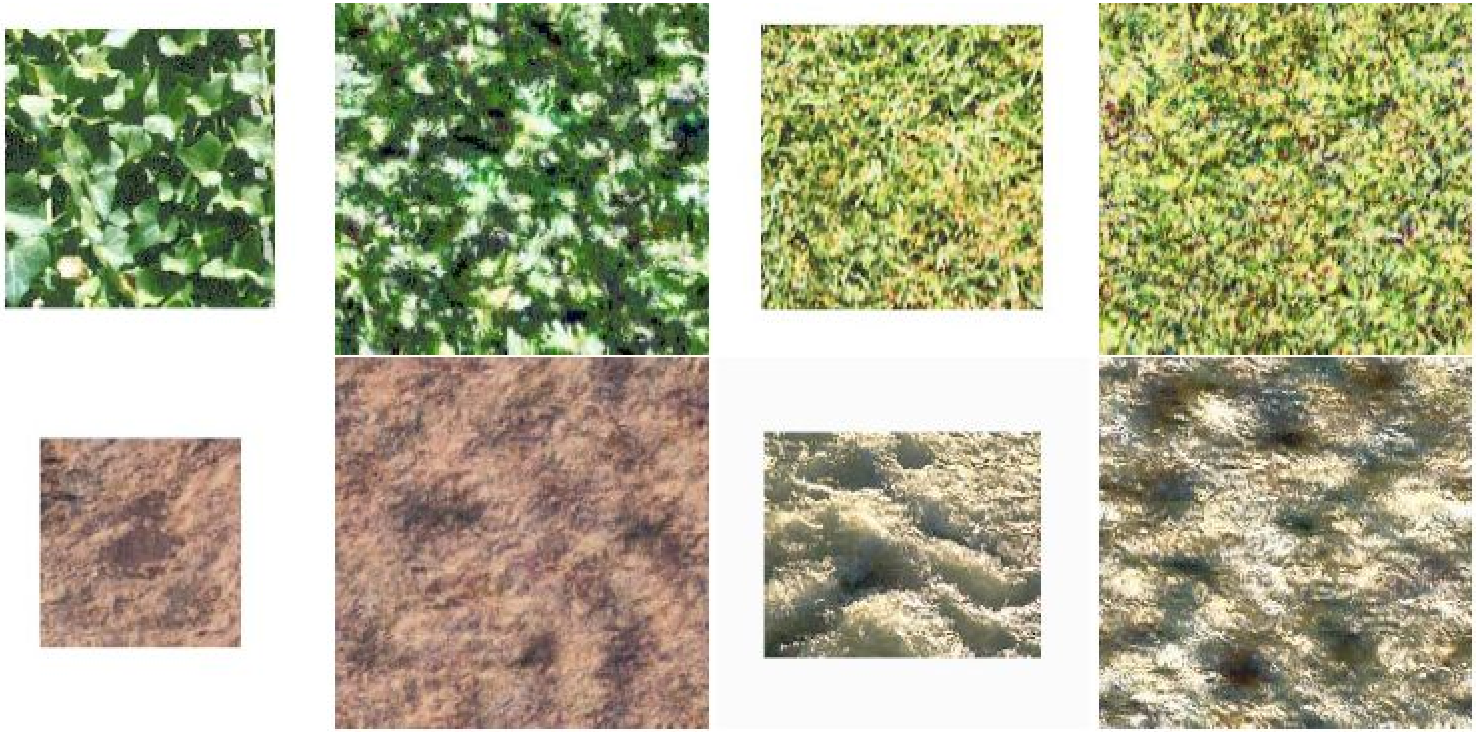


Figure 4: In each pair left image is original and right image is synthetic: red gravel, figured sepele wood, brocolli, bark paper, denim, pink wall, ivy, grass, sand, surf.

Heeger/Bergen, Siggraph 1994

Synthetic surfaces

Heeger/Bergen, Siggraph 1994

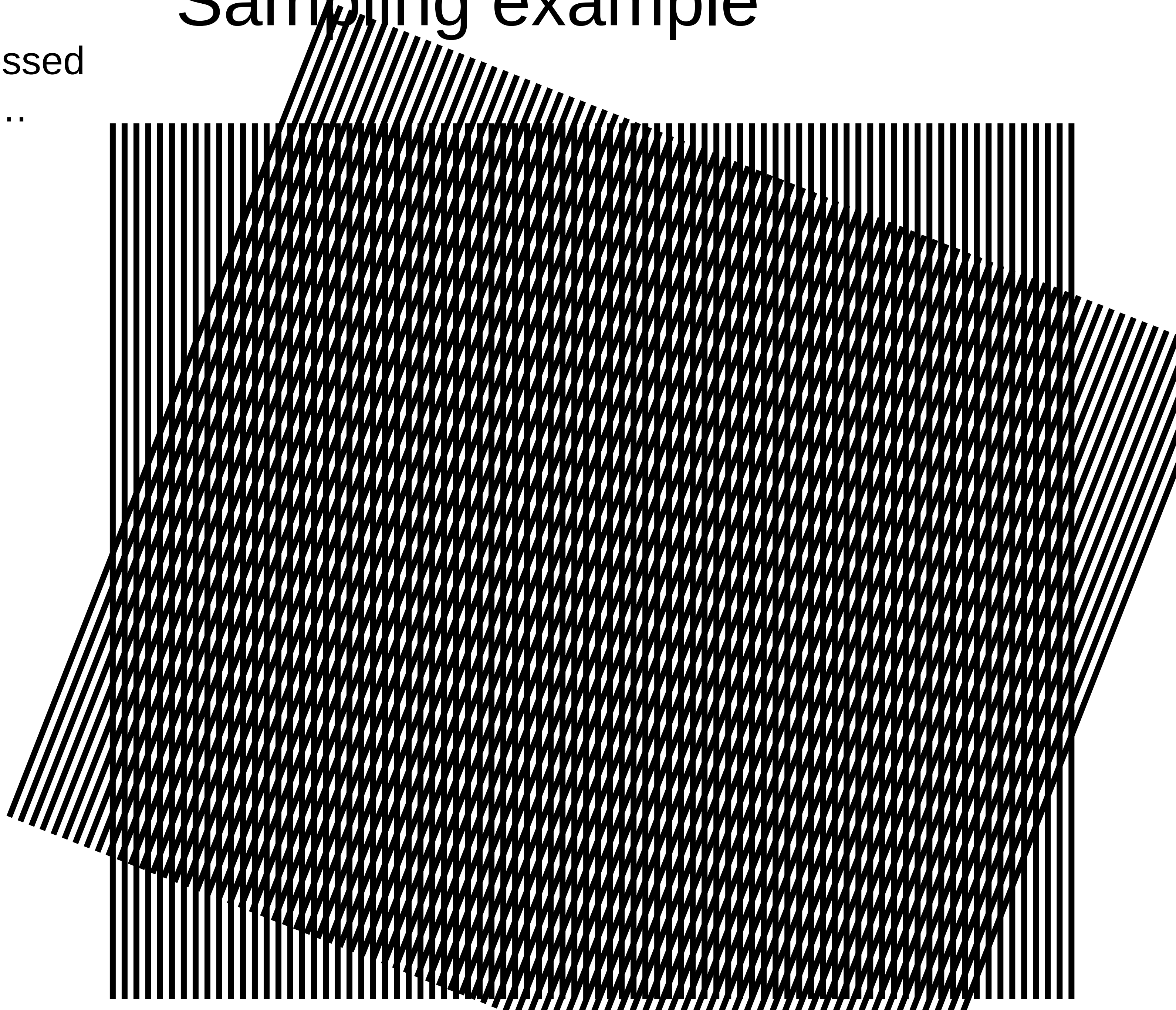
Synthetic surfaces

Figure 5: (Top Row) Original digitized sample textures: red granite, berry bush, figured maple, yellow coral. (Bottom Rows) Synthetic solid textured teapots.

Heeger/Bergen, Siggraph 1994

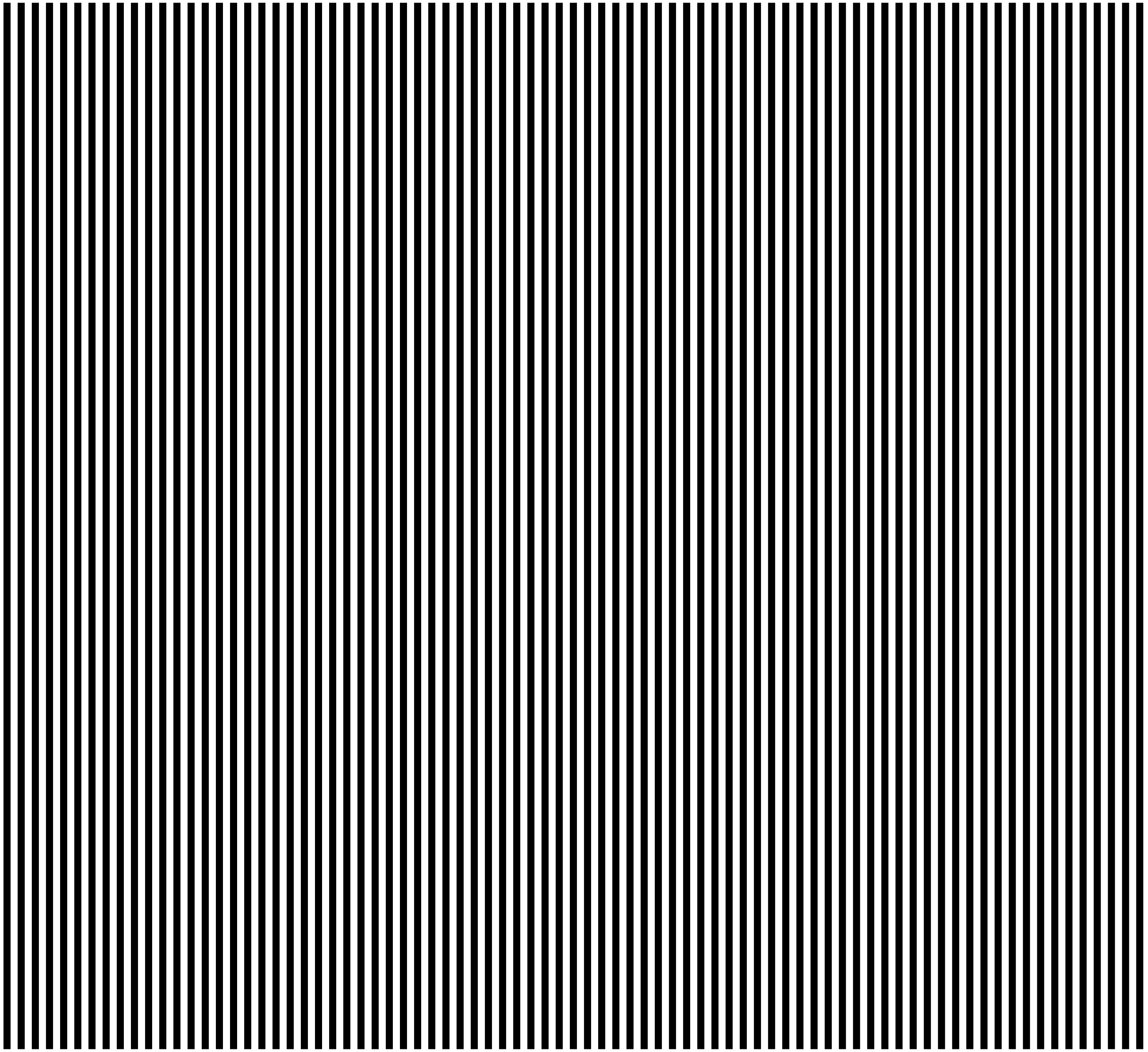
Sampling example

Analyze crossed gratings…



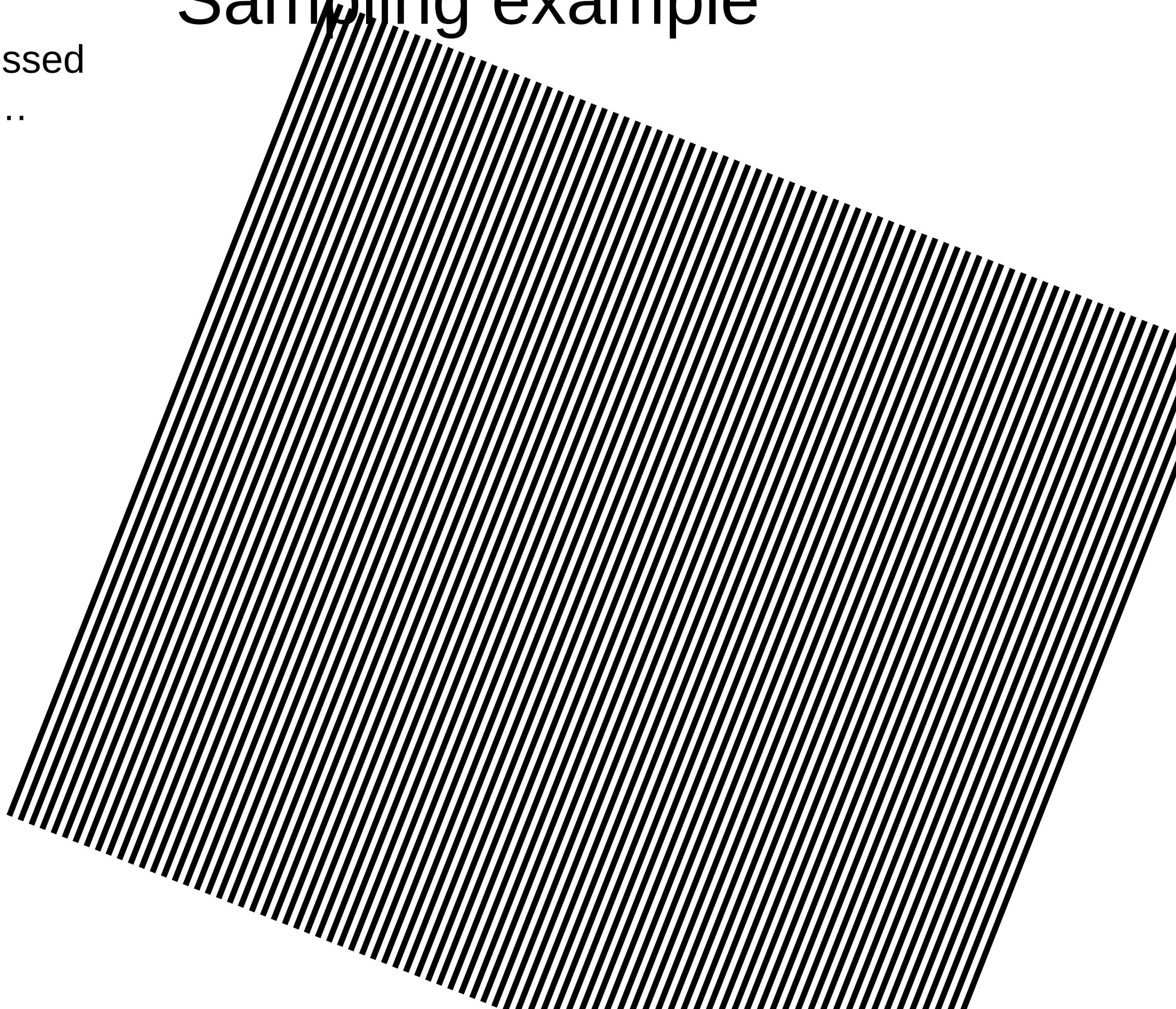
Sampling example

Analyze crossed gratings…



Sampling example

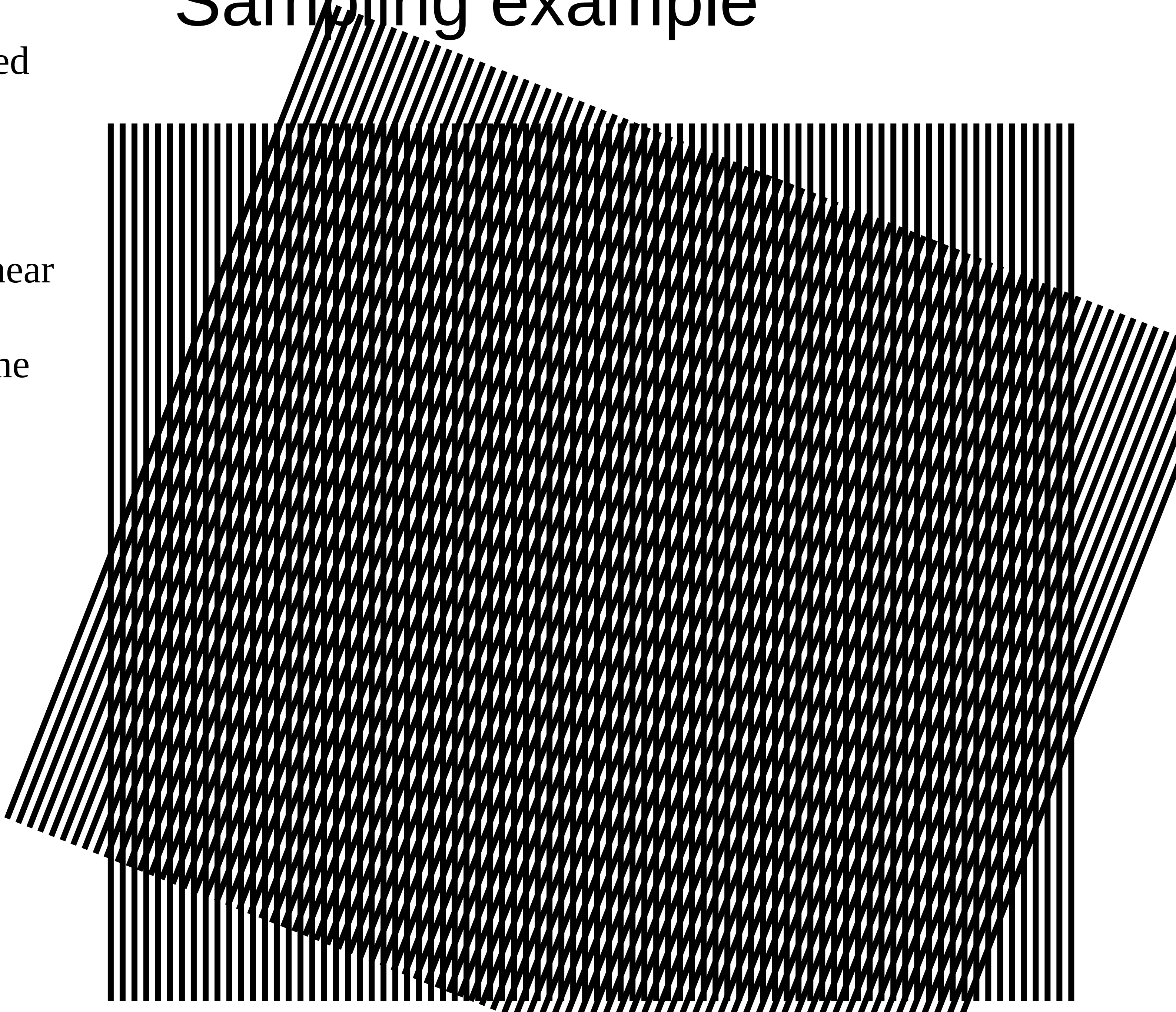
Analyze crossed gratings…

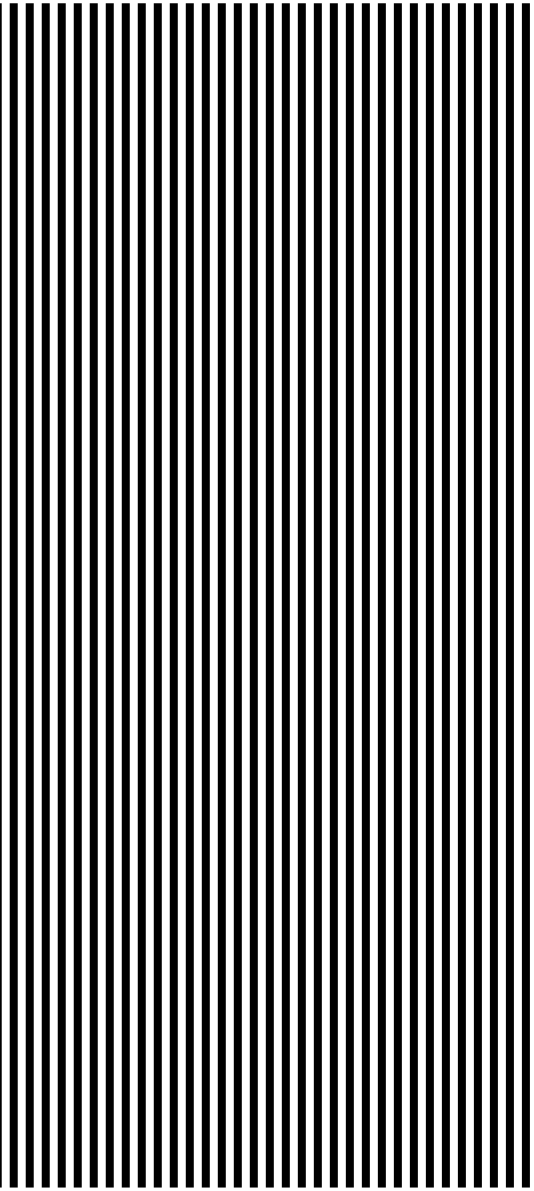


Sampling example

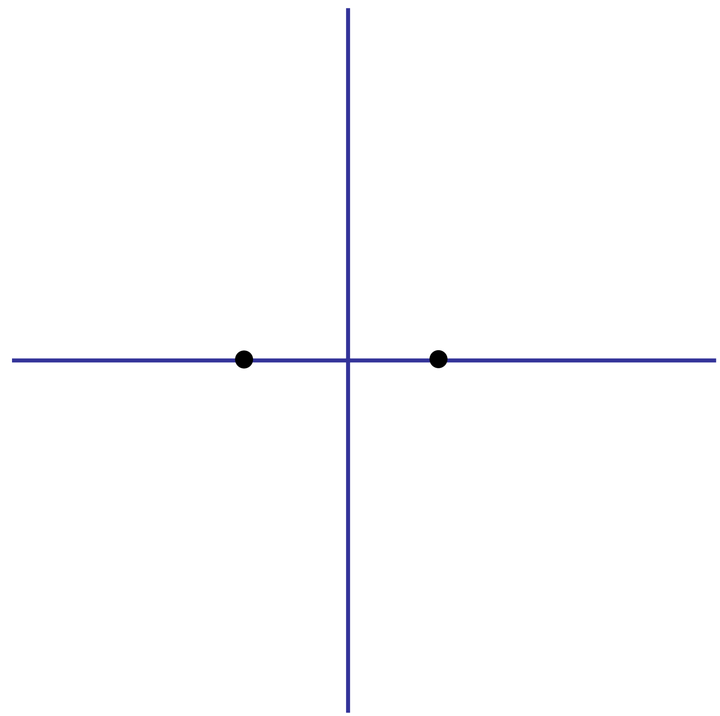
Analyze crossed gratings…

Where does perceived near horizontal grating come from?

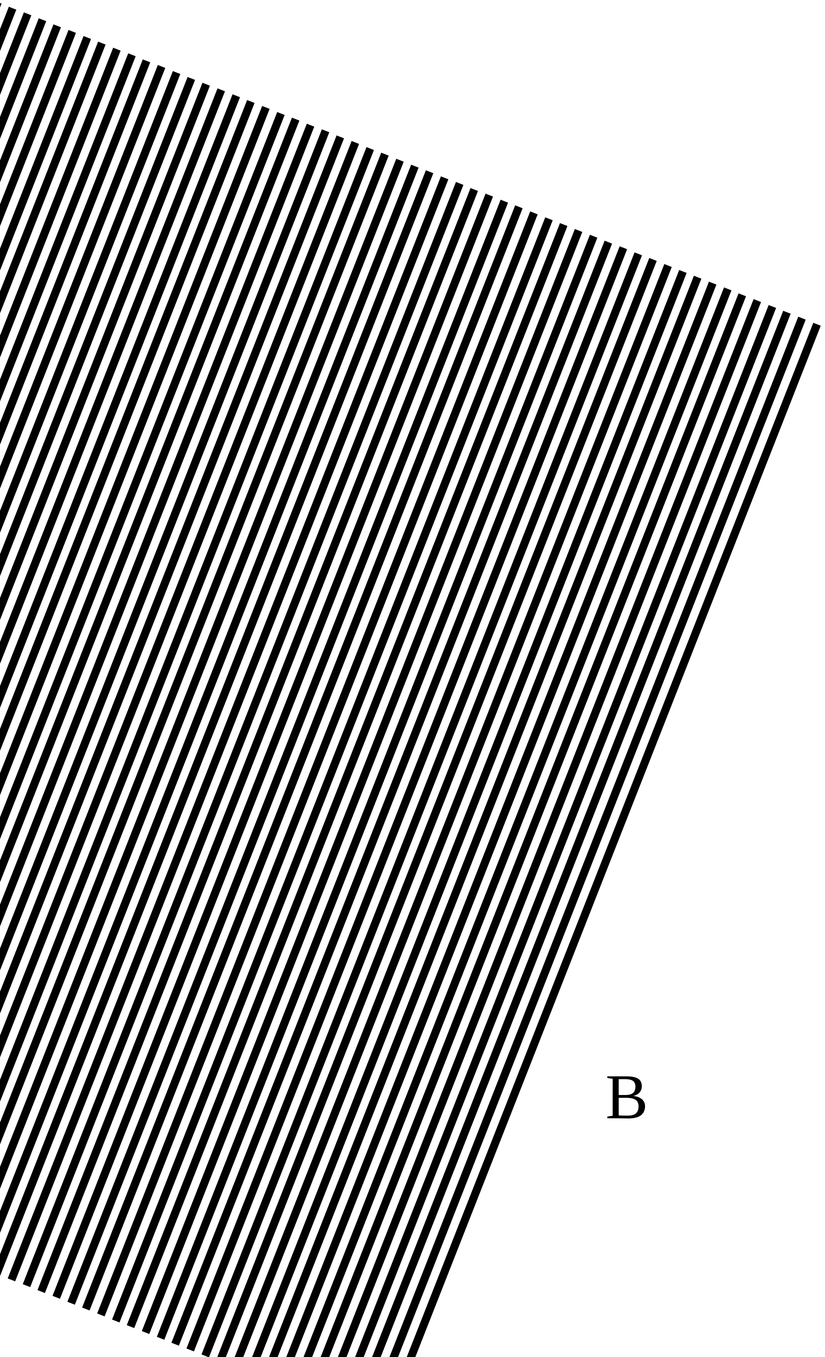




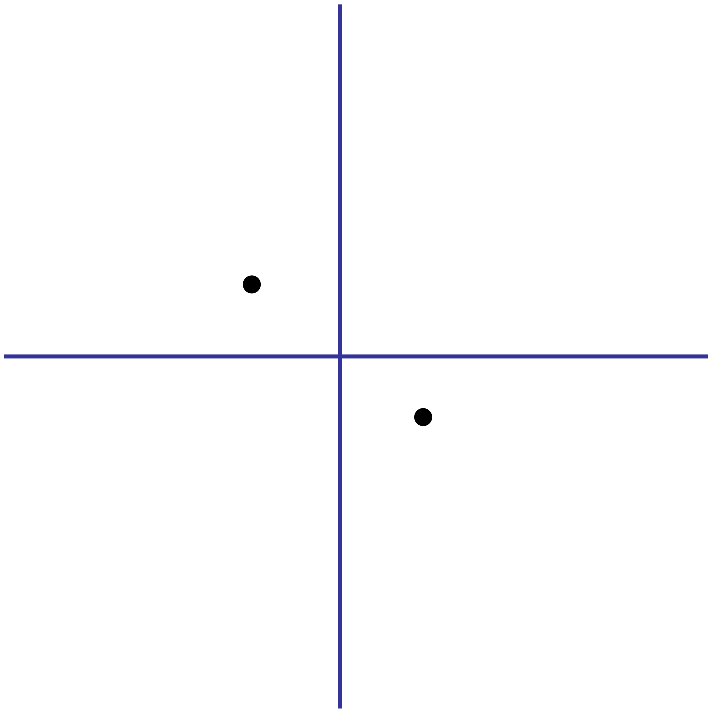
A



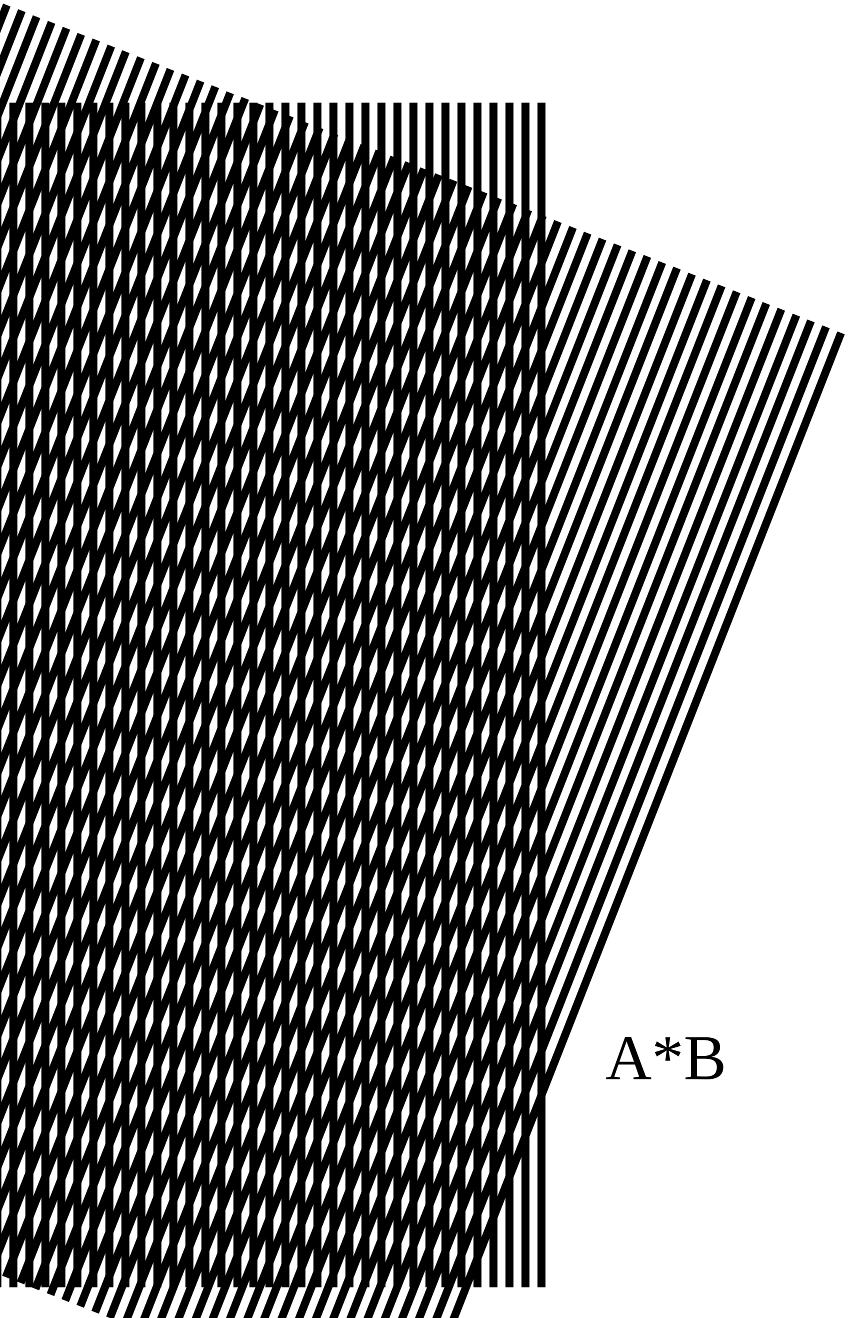
F(A)



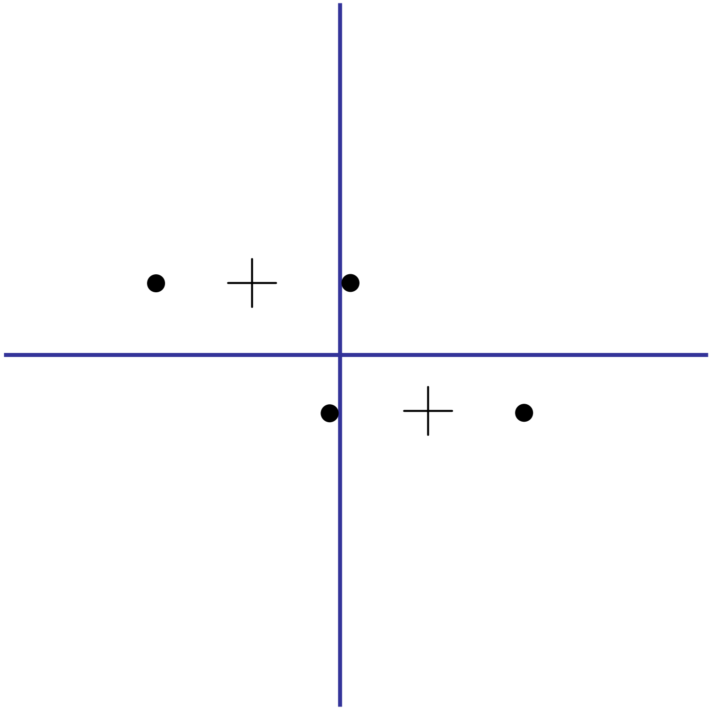
B



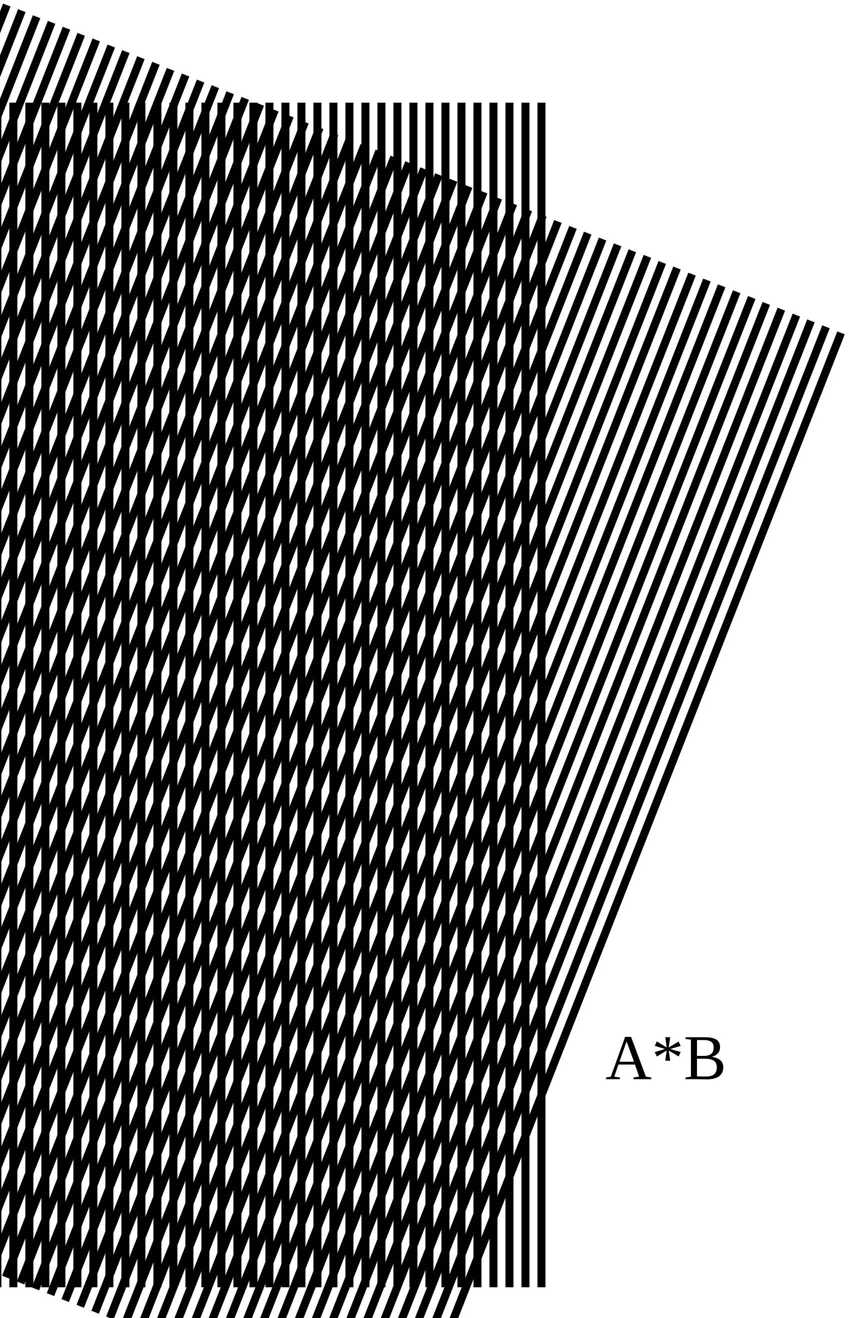
F(B)



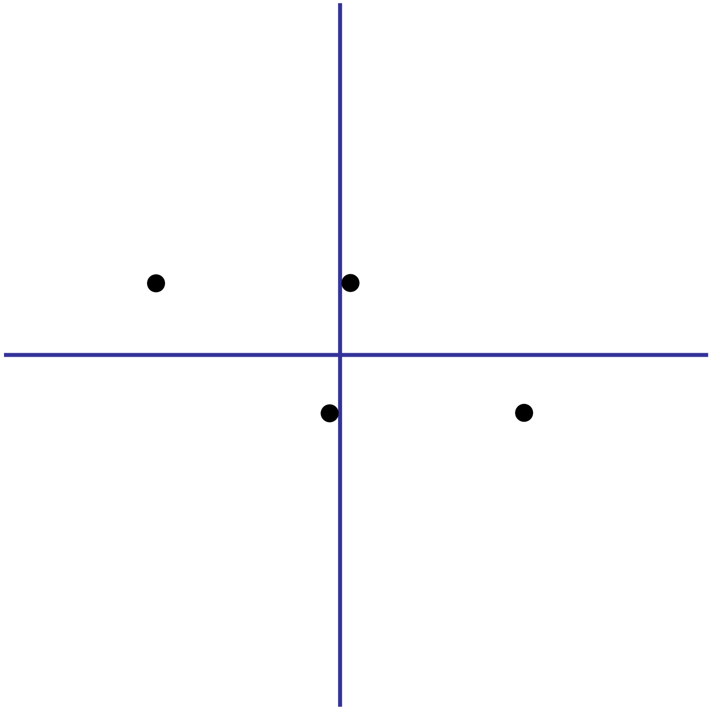
A*B



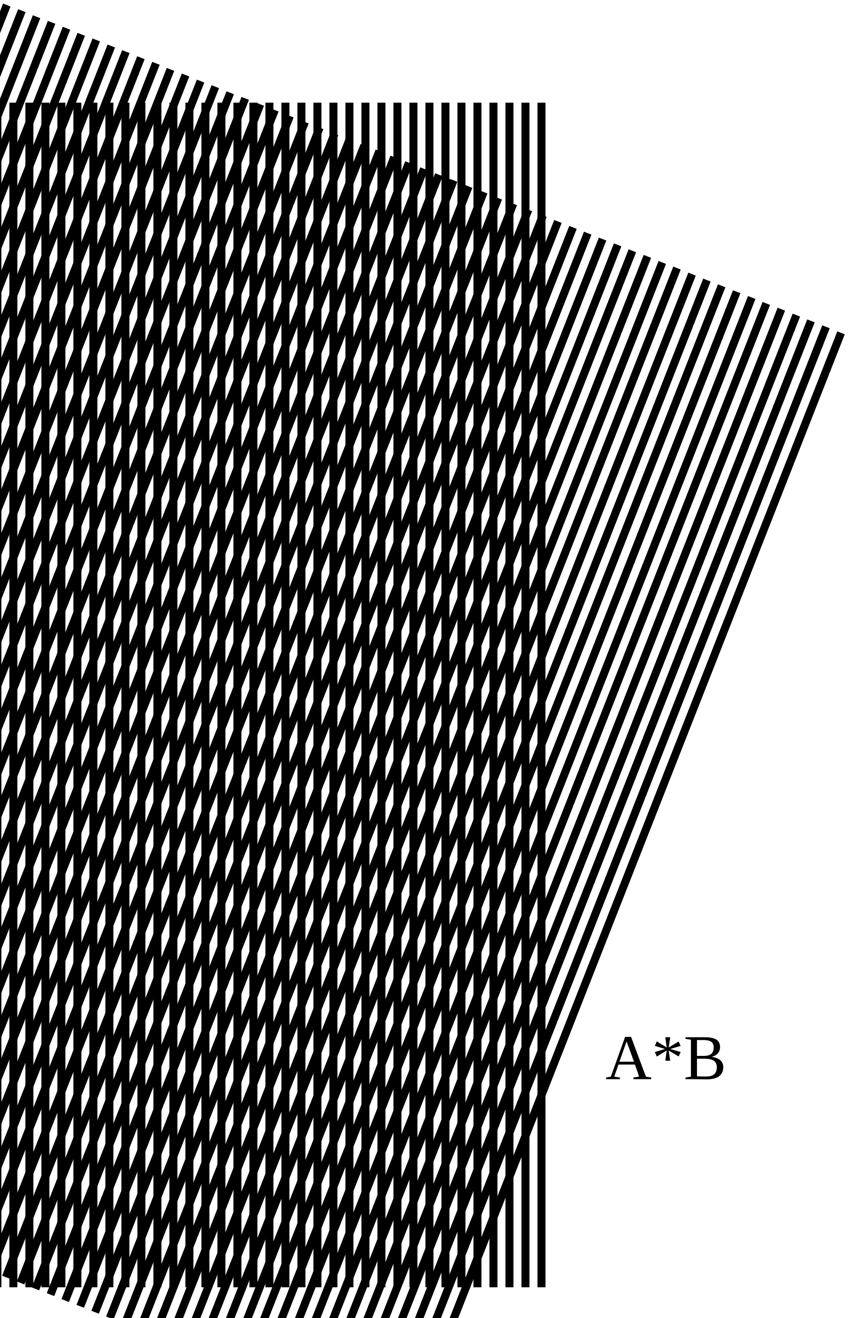
F(A)**F(B)



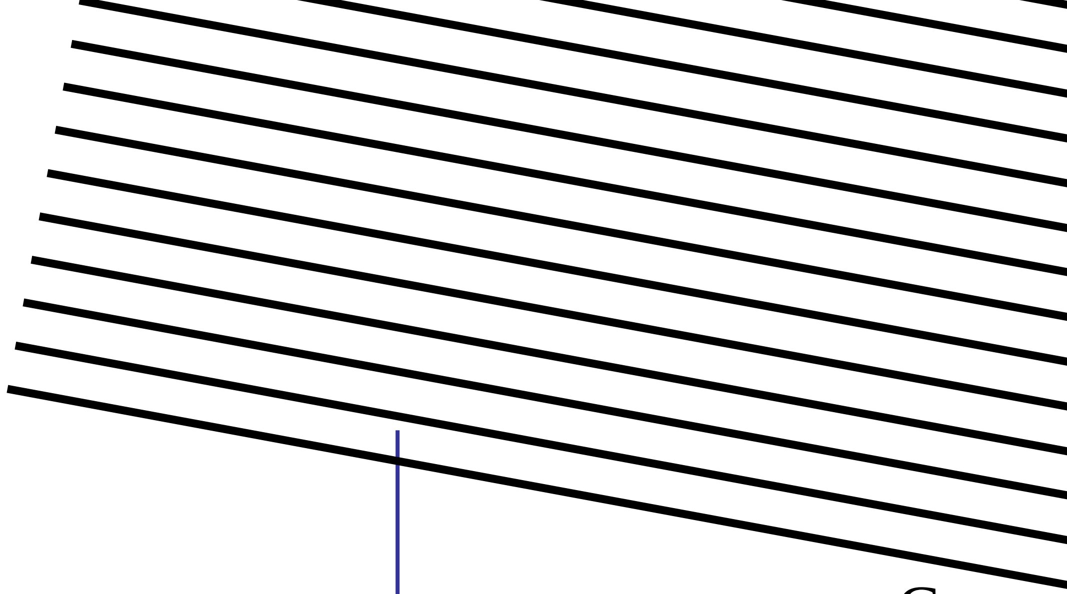
A*B



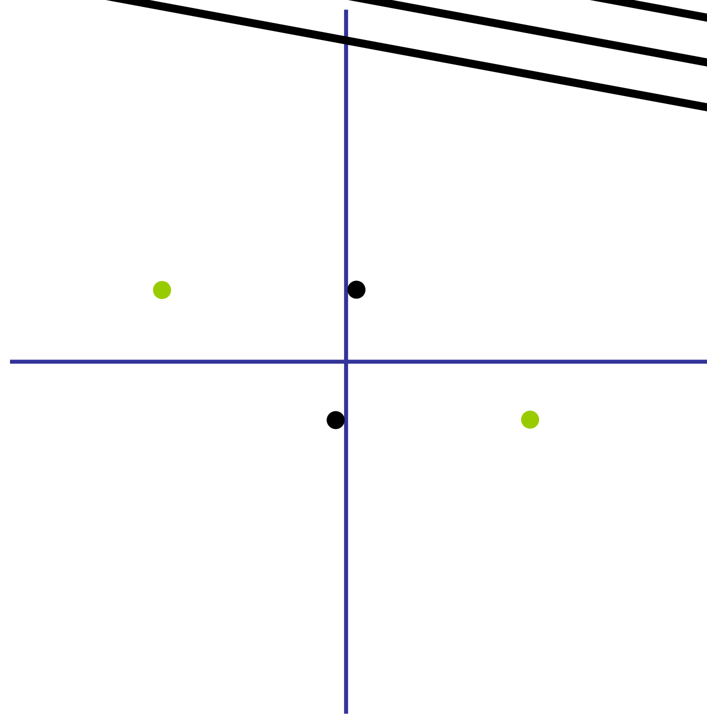
F(A)**F(B)



A*B



C



Lowpass(F(A)**F(B))
~=F(C)

end

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

Application to image compression

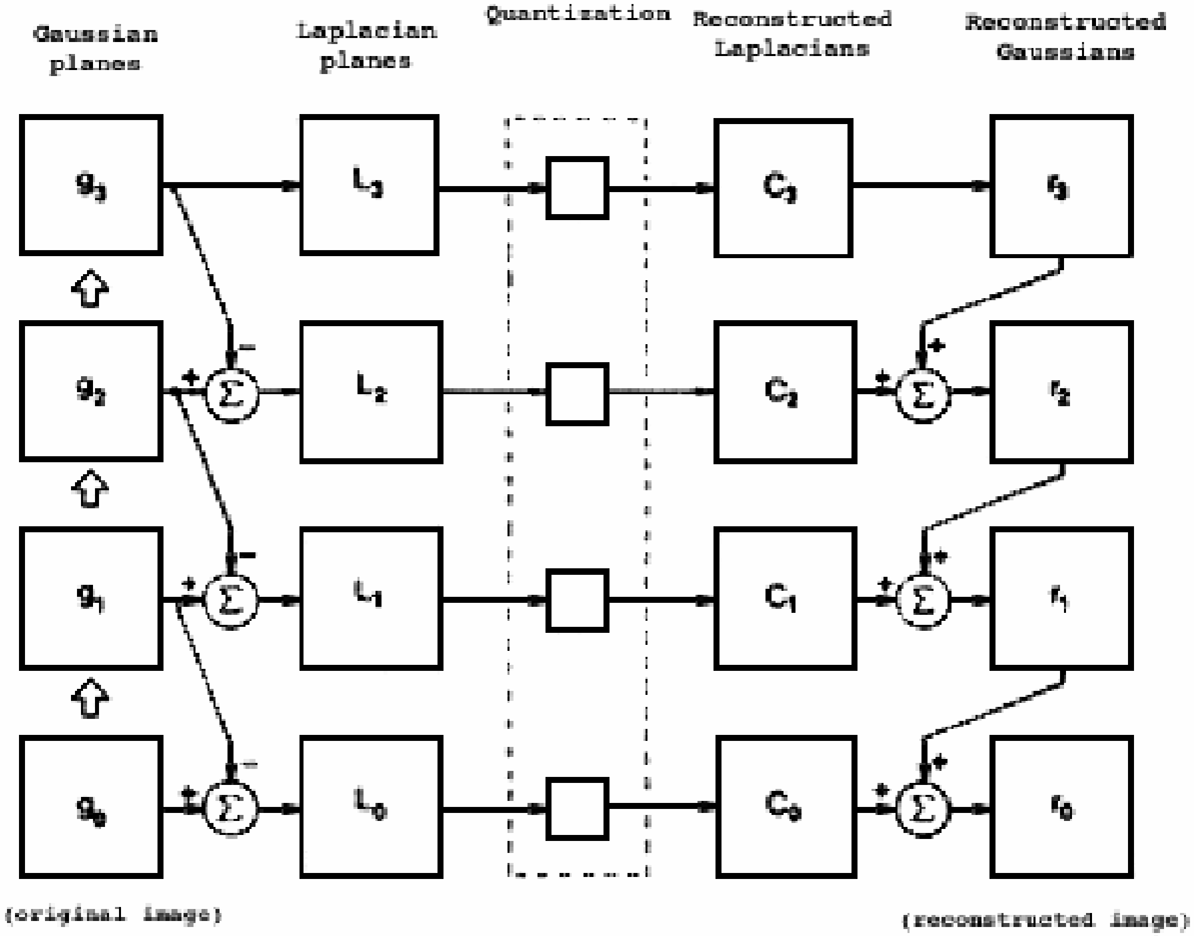


Fig. 10. A summary of the steps in Laplacian pyramid coding and decoding. First, the original image g_0 (lower left) is used to generate Gaussian pyramid levels g_1, g_2, … through repeated local averaging. Levels of the Laplacian pyramid L_0, L_1, … are then computed as the differences between adjacent Gaussian levels. Laplacian pyramid elements are quantized to yield the Laplacian pyramid code C_0, C_1, C_2, …. Finally, a reconstructed image r_0 is generated by summing levels of the code pyramid.

http://www-bcs.mit.edu/people/adelson/pub_pdfs/pyramid83.pdf

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983